EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

EDITORIAL Open Access

Combining Cloud and sensors in a smart city environment

Nathalie Mitton[1], Symeon Papavassiliou[2], Antonio Puliafito[3*] and Kishor S Trivedi[4]

Abstract

In the current worldwide ICT scenario, a constantly growing number of ever more powerful devices (smartphones, sensors, household appliances, RFID devices, etc.) join the Internet, significantly impacting the global traffic volume (data sharing, voice, multimedia, etc.) and foreshadowing a world of (more or less) smart devices, or "things" in the Internet of Things (IoT) perspective. Heterogeneous resources can be aggregated and abstracted according to tailored thing-like semantics, thus enabling Things as a Service paradigm, or better a "Cloud of Things". In the Future Internet initiatives, sensor networks will assume even more of a crucial role, especially for making smarter cities. Smarter sensors will be the peripheral elements of a complex future ICT world. However, due to differences in the "appliances" being sensed, smart sensors are very heterogeneous in terms of communication technologies, sensing features and elaboration capabilities. This article intends to contribute to the design of a pervasive infrastructure where new generation services interact with the surrounding environment, thus creating new opportunities for contextualization and geo-awareness. The architecture proposal is based on Sensor Web Enablement standard specifications and makes use of the Contiki Operating System for accomplishing the IoT. Smart cities are assumed as the reference scenario.

Keywords: Internet of Things, sensor networks, Cloud computing, smart cities

Introduction

As suggested by current ICT trends (Future Internet), sensing and actuation resources can be involved in the Cloud, in our view not exclusively as simple endpoints, but they should be dealt with in the same way as computing and storage resources usually are in more traditional Cloud stacks: abstracted, virtualized, and grouped in Clouds. Moreover, by adding sensors and actuators into the mix, new opportunities arise for contextualization and geo-awareness. Following the naming conventions for (virtualized) computing resources ("Infrastructure as a Service"—IaaS) and storage resources ("Data as a Service"), we may define such an approach by the phrase "Sensing and Actuation as a Service" (SAaaS). Beyond enabling fixed infrastructure, the resulting scenario is highly dynamic since it may also involve volatile mobile devices. Thus, a workable plan to address such issues suitably is to resort to the volunteer contribution model as an underlying approach.

A remarkable point of contact for both sensing environments and Clouds is the Internet of Things (IoT), where underlying physical items can be further abstracted according to thing-like semantics. Indeed, the outlined infrastructure could be the workbench on top of which such an abstraction would be implemented, where "things" handlers, pointing to physical items (e.g., documents, cars, products, parts, etc.), can be discovered, selected, and allocated. Things/objects become communicant and can also store information on and in their surrounding environment. They also become a gate to interact with our environment. According to a recent Gartner report there will be 30 billion devices connected by 2020. In this way, we can assume such a scenario as a plethora, an ecosystem, a constellation of generic devices and sensor networks (SNs) that are interconnected on the Internet. It is therefore natural to think about possible ways and solutions to face an all-encompassing challenge, where such an ecosystem of geographically distributed sensors and actuators may be discovered,

* Correspondence: apuliafito@unime.it
[3]University of Messina, Messina, Italy
Full list of author information is available at the end of the article

selected according to the functionalities they provide, interacted with, and may even cooperate for pursuing a specific goal.

This scenario has been envisaged, from many different perspectives, along several research trends (Future Internet, IoT), on which institutions and governments are spending huge efforts, having already identified such topics as strategic ones. This is also in line with the technological trend, i.e., identifying personal and mobile Clouds as the hottest Cloud topics of 2012 [1]. Computing, storage, and sensing therefore become complementary aspects in the big picture, and a comprehensive approach from the sensing/actuation perspective is needed to optimally coordinate their interactions, thus creating a pervasive infrastructure interacting with the surrounding environment.

An emerging category of devices at the edge of the Internet are consumer-centric mobile sensing and computing devices, such as smart phones and in-vehicle sensors. These devices will fuel the evolution of the IoT as they feed sensor data to the Internet at a societal scale. Individuals with sensing and computing devices collectively share data and extract information to measure and map phenomena of common interest.

Today, people are increasingly capable of creating and sharing written and recorded content via the Internet. Through the use of sensors (e.g., cameras, motion sensors, and GPS) built into mobile devices and web services to aggregate and interpret the assembled information, a new collective capacity is emerging—one in which people participate in sensing and analyze aspects of their lives that were previously invisible. This trend, often named Participatory Sensing and/or Mobile Crowdsensing, is primarily concerned with data collection, processing, and interpretation. This essentially emphasizes the involvement of users and community groups in social networks, documenting different aspects of their lives.

In such a context, mechanisms and tools for discovery and selection of virtual sensors and actuators according to both functional and nonfunctional properties expressed in terms of specific (QoS/SLA) constraints, actions while taking into account sustainability and energy efficiency issues of energy-constrained (battery powered) devices and SNs, are required.

Other issues to be addressed are related to the heterogeneous resource mashups, i.e., how to orchestrate assorted sensing, actuation, computing, storage resources of volunteer-based sensing Clouds with those of existing public/private computing and storage Clouds. The aforementioned objectives lead to what are to be considered as two independent solutions, as an SAaaS Cloud may provide its own, standalone service, that can be either mashed up or not, and a mashup provider may as well mash up resources without necessarily involving volunteer SAaaS Clouds.

Vision and paradigm

In this way, our perspective moves towards the Cloud of Things (CoT) as compared to the IoT and Web of Things paradigms. A CoT implies much more than just interconnecting and hyperlinking things. A CoT provides services by abstracting, virtualizing, and managing things according to the needs and the requirements specified by users, negotiated and agreed to by the parties through specific SLA agreements/procedures. The purpose is to implement services to provide indexing and querying methods applied to things, i.e., heterogeneous (sensing, actuation, computing, storage, and energy sources) resources aggregated according to a given thing-like semantics and provided to final users, developers, SaaS providers, etc., as a service, thus named TaaS.

In this context, needed background and enabling technologies to implement this stack are: resources and things abstraction and virtualization, with proper semantics in relation to the domain under consideration (primarily sensors, actuators, and IoT); volunteer techniques and mechanisms for autonomous enrolment and distributed coordination; Cloud-like, service-oriented interfaces, and fruition (on-demand adaptive “elastic” tools); interoperability and federation techniques, standards and tools to enable heterogeneous resource/Cloud mashups; business logic; security and trustworthiness policies.

Sensed information is generally acquired by independent administrations deploying their own monitoring infrastructure and software architecture. Sharing such information can be strategic, not only offering advanced services in Smart Cities, but also processing them for making correlations on data can be very complex. The idea of such a massive scale data sharing is leading towards the concept of system of systems, which aims to achieve task-oriented integration of different “systems” provided by independent public and private organizations, offering new levels of effectiveness and efficiency. An example of its applications is the World-Wide Smart Cities initiatives that involves many Administrations and is a concrete reality [2].

According to the systems integration idea, the IoT allows a high level of interoperability and a certain degree of flexibility. It enables seamless communication flows between heterogeneous devices, hiding the complexity of the end-to-end heterogeneity from the communication services. However, the complexity of technologies and the plethora of heterogeneous interconnected networks limit integration strategies.

Therefore, much research by the scientific community is still necessary. For example, IBM India has recently funded a new research activity, for considering Sensor

Web technologies in the context of smart cities (SENSIT [3]). The project is specifically aimed at the low-cost sensor-based solution to assist India with rainfall monitoring and flood forecasting.

In this article, we present a new architecture that provides to Internet users the capability to obtain any type of data acquired from different heterogeneous sensing infrastructures (SIs), exposed in a uniform way. The data provisioning is very flexible and it meets the user requirements. This result is achieved by accomplishing a high level of abstraction of sensing technologies and sensed data. The architecture has been designed considering the following main purposes:

- The provisioning of data has to be performed with high reactivity and high level of scalability.
- The system has to provide a rapid setup of deployed sensors and an easy integration of new sensors in the sensing environment.

To earn these requirements, specific design strategies have been developed. In particular, a hierarchical organization of the architectural components allowing to separately manage a high-level intelligence, achieving the abstraction of data and the fulfillment of clients requests. Furthermore, a strong interaction of the system with sensors has been accomplished through a peripheral decision-maker who is able to analyze, filter and aggregate sensed information. The data abstraction layer of our architecture has been developed according to the Sensor Web Enablement (SWE) standard defined by the Open Geospatial Consortium [4]. Nevertheless, our solution overcomes the limitations of SWE, only conceived for the Web use of sensors. The layer for the interaction with the SIs makes use of Contiki [5], an Operating System designed for sensors and embedded systems. It gives a uniform platform for communicating with heterogeneous sensors.

The remaining of the article is organized as follows. We first provide a background information of related ideas in the following section. After that, the whole proposed framework and its components are explained in Section “Reference scenario and proposed architecture”. Implementation details are given in Section “Implementation issues”, while Section “Case study: smart cities” discusses a case study related to services development in smart cities. Finally, this study is concluded with the suggestions on future works.

Related work

In the sensor technology domain, virtualization has been proposed with the goal of enabling seamless interoperability and scalability of sensor node platforms from different vendors via uniform management, with the interposition of an abstraction layer between the application logic and the sensor driver [6] (also in the IoT context [7,8]). Virtualization can also be performed by forming virtual sensor networks, enabling multi-purpose, collaborative, and resource-efficient exploitation of the physical infrastructure that may involve dynamically varying subset of sensors. Software abstraction layers are used to address interoperability and manageability issues [9] and to allow the dynamic reconfiguration of sensor nodes within the WSN, for whichever purpose [10], and the combination of sensor data [11].

Regarding the description and implementation of frameworks for efficient representation, annotation and processing of sensor data, the goal of the OGC SWE [12] initiative is the definition of Web service interfaces and data encodings to make sensors discoverable and accessible on the WWW, able to receive requests. On the other hand, the W3C Semantic Sensor Network Incubator Group aims at extending this syntactic level interoperability to a semantic level (CSIRO [13,14]).

Significant research on sensing, actuation, and IoT is directed towards the efficient semantic annotation of sensor data. In [15], an approach is proposed to make sensor data and metadata publicly accessible by storing it in the Linked Open Data Cloud. Similarly, in [16] an infrastructure called SensorMasher provides the ability for non-technical users to access and manipulate sensor data on the Web, while in [17-19] different ontologies and semantic models are presented for sensor data representation, such as SUMO, Ontosensor, and LENS. A detailed survey of existing sensor ontologies is available in [20]. Also a European FP7 project, SENSEI[a], was launched from 2008 to 2010 to deal with these aspects. Some great industrials such as Ericsson[b] position themselves on SmartCities as well showing it as the next challenge.

A promising research field is the IoT [21,22] that aims at meshing a networked environment, where the nodes may also semantically be tagged as things from physical world items. Although the resources in the Cloud could be useful to overcome certain constraints of smart devices in IoT scenarios, absence of context-awareness in the Cloud widens the gap between elastic resources and mobile devices. Several bridging approaches exist [23] but bindings are required to handle mappings between physical environments in IoT and virtual environments in the Cloud [24], as those described in [7]. Forming Clouds of sensors and other mobile devices shows similarities to existing technologies developed in the area of dynamic services [25]. Service registries are to act as repositories for metadata concerning services. They can be architecturally centralized or distributed and, for information retrieval, keyword-based, signature-based, semantic based, context-based and quality based

[26]. Service monitoring and tracking facilities are devised in order to deal with the inherently unreliable nature of services, that cannot be assumed "always on", as mobile-powered ones may go offline in one location and turn up again somewhere else, and the availability of some services may swing steadily in a unpredictable way.

In literature, some works deal extensively with issues related to Smart Cities. The authors of [27] highlight how the cities of the future will need to collect data form a lot of urban sensors, such as smart water, electric meters, GPS devices, building sensors, weather sensors, and so on. Many of them are low-cost sensors with a high level of noise and unreliable communication equipments. The key idea for getting high-quality services from such cheap sensors is the cross-correlation of sensed data from several sensors and their analysis with sophisticated algorithms. In South Korea, there are several initiatives to move from Ubiquitous City (U-City) to UEco-City, that is a city designed with consideration of environmental impacts. For example, the authors of [28] present a platform for managing urban services that include Convenience, Health, Safety, and Comfort. Also, the differences between "smart city" and "digital city" are detailed in [29].

To understand how Smart Cities may benefit from Sensor Web technologies, Hernandez-Munoz et al. [30] presented an extension of their framework, called Ubiquitous Sensor Networks [31] that leverages the SWE along with the SIP protocol. One of the main problems of the SIP protocol is related to the network constraints. Usually, network administrators limit the Internet communications using firewalling policies, and the SIP heavily suffers from this limitation [even more if a Network Address Translator (NAT) is present].

The authors of [32] propose a solution that, exploiting the XMPP protocol to deploy SWE solutions, is immune to such a problem, in fact it is able to overcome Firewalls and NATs locking. Interesting research papers are related to Contiki a tiny Operating System for Sensors. The designers of Contiki are looking at how to make interoperable SNs using IPv6 [33] and how to efficiently store data inside sensors [34]. In particular, the authors of [34] have introduced a tiny database, called Antelope. The article shows how sensors are becoming even smaller parts of the current Internet (in terms of used protocols and added capabilities). Nevertheless in our view the abstraction should be unrelated with the underlying sensing technology.

The problem to define an abstraction of sensed data representation was also identified in [35]. The focus of this article is on the mechanisms for evaluating contextualizing rules. For example, in processing of spatial objects, the authors analyze the concepts of proximity, adjacency, and containment. They even introduced the contexts of data representation with different dynamics. Furthermore, a global model is introduced with dynamic interoperability without taking into consideration how the global view should be accomplished. The decision maker should evaluate several incoming data, but it is not clear how to address such a problem (i.e., scalability problems).

Reference scenario and proposed architecture

Our main objective is to design an innovative architecture, able of adding new sensing capabilities with Zero-Conf approaches, abstracting sensing data, conferring to world wide systems a high reactivity and high level of scalability. To realize this, we need to manage each system by using a cluster-based approach, as it is shown in Figure 1, and originally discussed in [36]. We assume that a system is mapped on sites that could be of different scales. In the scenario of Smart Cities, a site can be a building, a factory, city roads, or also a whole city. Each site is an autonomous system (AS), in which Clients and Services interact with each other. The AS has its own SIs (data producers) and Clients interested in sensed data (data consumers). SI could be a single sensor or possibly a full multi-hop wireless SN. They are possibly mobile and thus connect at different points of the cloud along time. Data gathered by SIs into a site (for example, SIAi in the site A) are stored through a Database (DB) Manager in a local relational database, in order to offer an efficient retrieval of data to internal clients (in the example, ClientAi). This database could be delocalized and distributed in order to share and store information in a smart way and in a transparent manner for the users [37]. It could even include sensors themselves. However, the same data can be useful for Client in other sites, such as sites B and C (see Figure 1). To also offer such a service to external clients without wasting services performance of a site, the DB Manager publishes sensed data on a global distributed column-based database (such as Cassandra [38]) accessible from all sites.

Whenever a client requests data, according to authorization rules and existing agreements among sites, the DB Manager checks both, either if the client requests can be satisfied within a site or if it needs external information. In the latter case, it enables the DB Service Manager to perform a query on the distributed database. This approach allows many sites to cooperate with each other, sharing data and services, at the cost of a higher complexity architecture.

In order to build such an ambitious architecture, i.e., a Cloud of Sensors based on the SAaaS paradigm, in Figure 2 we introduce the whole stack and a high-level schema of the architectural modules, identifying three main components, from the bottom up: Hypervisor, Autonomic Enforcer, and VolunteerCloud Manager. As

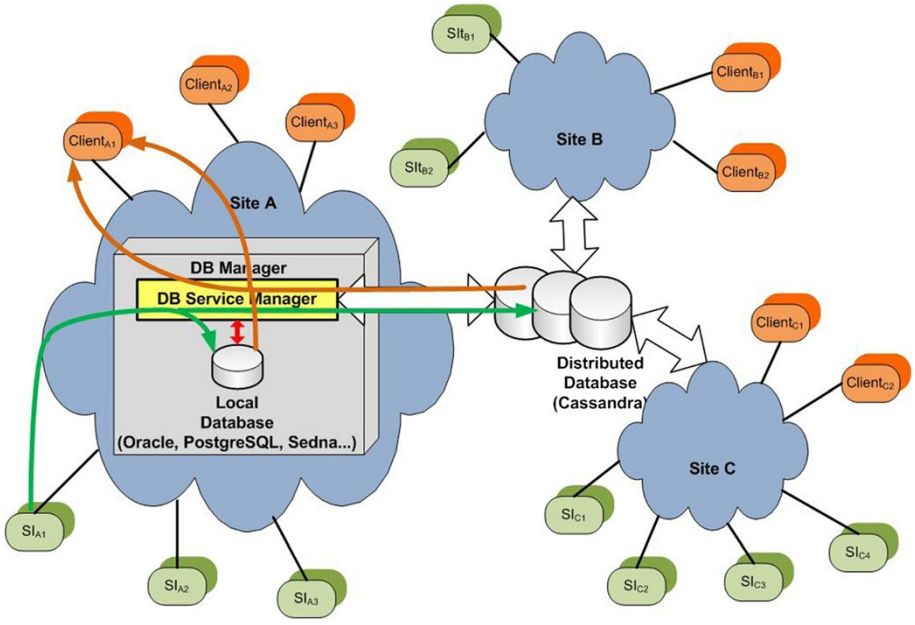

Figure 1 Logical organization of the system in a world wide scenario.

discussed in [39], the lowest block, the Hypervisor, works at the level of a single node, where to abstract away either embedded sensors available on a personal device (e.g., smartphone) or standalone sensors, smart or otherwise, belonging to a network (WSNs). Among its duties are relaying commands and data retrieval, abstraction of devices and capabilities, virtualization of abstracted resources, semantic labeling and thing-enabled services. The Adapter enables the communication directly with sensing/actuation devices and keeps track of resources connectivity. It translates application commands and forwards them to the underlying physical resources, using the native communication protocol of the resource.

The Cloud modules, under the guise of an Autonomic Enforcer and a VolunteerCloud Manager, deal with issues related to the interaction among nodes, belonging to a single Cloud, for generating a Cloud of Sensors: the former is tasked with enforcement of policies, local versus global (i.e., relayed) policy tie-breaking, subscription management, cooperation on overlay instantiation, where possible through autonomic approaches, while the latter is in charge of exposing the generated Cloud it hides by means of Web Service interfaces, framing reward mechanisms and policies in synergy with SLA matching, to be mediated by QoS metrics and monitoring, as well as indexing duties to allow for efficient discovery of resources.

One envisioned paradigm is to enable every traditional entity (node, user, and provider) to be exposed and consumed as a Service that provides and requests content information. Applications use user-generated content from fixed and/or mobile devices gathered in collaboration with its owner/operator. Such a model requires fundamentally novel algorithms for the data collection, aggregation, analysis, and composition of different services. Moreover, it entails novel application-level mechanisms in order to enable those who request or provide services to share data, while respecting the privacy of those involved.

Cloud-Based Services (CBServices) can be any "heavy" type of services that needs more resources and infrastructure in order to function properly. Streaming video, music on-the-go, social networks, web browsing, are among most popular applications in cloud environments. From the server side, all these services have several and usually intensive requirements in resources, middleware software, and infrastructure. CBServices can use traditional publish-subscribe model into Cloud environment in order to be used by other users or services, however use advanced features/properties of Cloud environment/platform to allow elastic services scalability and global delivery on-demand.

Mobile-Based Services on the other hand include mobile nodes that are moving in a non-structural way and provide any type of services and information from their current location. A user with a mobile phone or a tablet device can provide various types of location-based information depending on the application. The advantage of these kinds of services is that they exchange content that is user-generated and often very dynamic. A service asking for traffic information along a route can receive

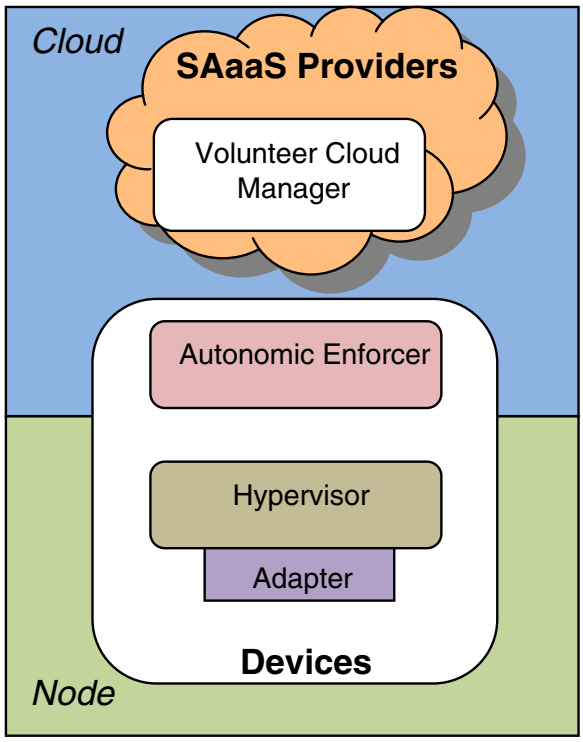

Figure 2 Architectural schema and modules.

dynamic and updated information from other users/services along the same route without necessarily requiring the support of a heavy centralized system. Such an architecture provides information to the user in a flexible and fast way.

Implementation issues

In this section, we report and describe the status of the current implementation of the SAaaS Cloud framework. Although this is still a work in progress, here we hope to provide proof of feasibility for the architecture depicted in Figure 2 and present available underlying solutions.

Hypervisor

The Hypervisor block implements the abstraction of sensing and actuation resources, providing functionality at the level of a single node, explicitly defined as a management domain, either an SN controlled by a specific gateway, or a standalone/set of sensors within a device. In SNs, a node may be less easily identifiable with respect to the one-to-one relationship we have for smartphones and other personal smart devices: more specifically we may have an SN made up of thousands of sensors, yet only exposing few sinks, where only part of the stack (i.e., nodal components) can be deployed. A modular architecture of the Hypervisor identifies the following three components: Adapter, Node Manager, Abstraction, and Virtualization Unit.

The lowest component, the Adapter, was developed by means of modifications to CLEVER, an IaaS stack with a flexible framework for internode/cloud communication and event notification [40]. This fork, CLEVERSens [41], works over a common baseline environment, having chosen Contiki [5] as open source platform to deploy on gateways and other sensing hardware for development and field testing, leveraging, in line with the OGC-mandated SWE framework, the set of XML-based languages and Web service interface specifications they defined.

Among many, the following SWE standards have been implemented in the Adapter to ease the discovery, access and search over sensor data:

- SensorML—models and XML schemas for describing sensors systems and processes; it provides information needed for discovery of sensors, location of sensor observations, processing of low-level sensor observations and task-oriented listing of properties;
- O&M (Observation and Measurements)—models and XML schemas for encoding observations and measurements from an SN;
- SOS (sensor observation service)—interface for requesting, filtering, and retrieving observations and sensor system information.

The internals consist of the following three layers, from the top down:

- REST APIs as interface, which allow on-demand interactions with clients, applications and/or other services;
- an SOS Agent, which faces up the abstraction of sensed data according to SOS specifications, supporting all mechanisms for describing sensors and observations, setting new observations and gathering measurements from SNs. It makes use of SensorML, for describing sensor systems and sensed data, and the O&M standard, for modeling sensor observations;
- a Sensor Manager (SM), able to interact with sensors, coordinates their activities and collect data for the upper layers. It provides a uniform management of heterogeneous sensors.

Moreover, we are planning to extend it further to cover Node Manager capabilities, for instance in relation to power consumption self-tuning, to be implemented with hooks also at runtime and OS layer under Contiki. We also intend to exploit Abstraction & Virtualization capabilities, engaging the OGC actively to expand on existing standards and propose new specifications for

composition of advanced virtual sensors, unbundling of resources from complex devices, and instantiation of abstracted resources with proper reliability and sandboxing mechanisms, in line with typical (IaaS) hypervisor-driven capabilities.

Autonomic enforcer

The bridge between virtualized nodes and SAaaS Clouds is the Autonomic Enforcer. It is a module that, first and foremost, allows the node to join a Cloud, thus exposing its resources as services through the Internet. Furthermore, the Autonomic Enforcer locally manages the node resources considering both higher level Cloud policies and local requirements and needs, e.g., power management on mobiles. This is therefore implemented in a collaborative and decentralized way, making decisions by interacting with neighboring nodes, and adopting autonomic approaches. The Autonomic Enforcer is to be deployed into each node of the SAaaS infrastructure in order to apply the policies of the VolunteerCloud Management module, self-adaptively.

In combination with the Hypervisor Node Manager, the Autonomic Enforcer makes up a hierarchical, decoupled, two-level autonomic management system entirely deployed and working on the node. The former operates at device level, more specifically within an SN domain, while the latter enforces higher level Cloud targets. To this purpose, four main blocks have been identified in the Autonomic Enforcer functional schema: a Policy Actuator below, Policy Manager and Subscription Manager above it, and Cloud Overlayer on top of them.

The heart of the Autonomic behavior for the Enforcer lies in its ability to leverage an architectural model and a runtime infrastructure where cooperating agents, the SelfLets [42,43], can provide services, and consume those offered by other SelfLets as well, being able to make decisions based on local knowledge of the surrounding environment.

A SelfLet can easily be tuned in terms of both default behaviors and autonomic policies. The idea is that, by keeping tabs on local resources, each SelfLet settles on whether to carry out certain global optimization actions, such as redirecting requests, teaching policies (and the implementations of related mechanisms, if the need arises) to other SelfLets, or learning from others as well.

In our ongoing efforts, we are taking into account revenues and costs, to be relayed to the Reward System in the Volunteer-Cloud Manager, generated as a result of demand for service in a SelfLet-driven environment (i.e., the Enforcer Managers) according to concurrent, eventually contrasting, requests, i.e., when originating from subscriptions of a single node to several Clouds. After evaluating a set of candidate optimization policies, inclusive of (eventual) subscription tuning, each SelfLet can pass its choices to the Actuator and inform its neighbors, following a greedy strategy, or a non-greedy one, depending on the state of surrounding SelfLets.

VolunteerCloud manager

In the development of the relevant modules, we strongly based our study on the results provided by the Cloud@-Home project [44]. The VolunteerCloud manager aims to consolidate volatile, ad hoc, dynamic resources and services, such as volunteer-contributed sensors, in a Cloud environment. The main focus is on methods alleviating the effects of resource churn, where their performance is largely dynamic, their lifespan is short, nodes are mobile and heterogeneous, and information on their status is partial and typically out of date. While this layer operates on largely unreliable and unpredictable resources, it provides services featuring increased dependability either to the Cloud layer or to other peer Cloud systems. The VolunteerCloud Manager defines and imposes management strategies at the Cloud level, through a continuous interaction with each single device belonging to the constituted sensing Cloud. Such policies have to be therefore acted upon at node level by the corresponding Autonomic Enforcer. The VolunteerCloud Management builds upon nodes, through the Autonomic Enforcer, a volunteer-based sensing Cloud, and implements services for interacting with it. The functionalities have been grouped into five components: Indexing & Discovery service, Reward System, SLA Manager, QoS Manager, WS Frontend.

With regards to the Indexing & Discovery Service, for the time being, such component is designed and implemented through a register service which receives and manages the requests for registration from node owners, collecting the corresponding description files into a database, under a steady flow of updates and, optionally, distributed, for increased fault tolerance. An alternative design could be hinged on DHTbased algorithms for P2P establishment, tracking the providers' statuses to spot those that may offer better support to fault tolerance, and a simpler way to keep the status about the chosen provider up-to-date.

The Reward System implementation is based on the solutions provided by BOINC [45] and EDGI [46]. Credit-reward systems are used here to reward cooperative and fair behaviors and to motivate resource providers, or donors. BOINC for instance employs a credit system where volunteers are awarded credits based on donated CPU and GPU time.

More specifically we are working on a hierarchical solution that implements an overlay credit system on top of volunteer credit systems (e.g., BOINC) adapted to sensing and actuation resource metrics, i.e., primarily the contribution time. The higher layer in the overlay

assigns (further) QoS credits rewarded for donated resource time. These credits can be reused and spent by the contributor into the SAaaS infrastructure, for allocating sensing and actuation resources.

The QoS Manager can be considered the counterpart to the Resource & QoS Manager (RQM) in the Cloud@Home architecture. It is in charge of tracking resources, logging all the requests and their status, and is composed of a core system (RQMcore) together with interfaces to all the other components.

Similarly, the SLA Manager corresponds to the Cloud@Home SLA Management module. It is in charge of the negotiation, monitoring, and enforcement of SLAs, and cooperates with the RQM component for QoS aspects, specifying and applying the policies related to whole Cloud Management.

For the time being our work consists of converting and adapting the current Cloud@Home implementation (RQM and SLA Management module) into the corresponding components of the SAaaS-VolunteerCloud Manager framework (QoS and SLA Managers).

Case study: smart cities

We guess that in Smart Cities, smart sensors with high processing power and multi-tier/IP capabilities will be deployed. Sensors are deployed everywhere, in street to measure the traffic, in gaz or water pipes for monitoring and management, for pollution detection purposes, etc. In this scenario, we assume that sensors will be equipped with a lightweight operative system for SN nodes. Two major operating systems lead the way on firmware development for motes: Contiki and TinyOS [47]. Contiki is an open source, highly portable, multitasking operating system for memory-efficient networked embedded systems and SNs. Contiki is designed for microcontrollers with small amounts of memory (a typical Contiki configuration is 2 kB of RAM and 40 kB of ROM). Contiki has been used in many projects, such as road tunnel fire monitoring, intrusion detection, wildlife monitoring, and in surveillance networks. One of the biggest features of Contiki is the very light implementation of the IP stack, called uIP, with 6LoWPAN support. This implementation was awarded the IPv6 ready silver seal from the IPv6 Ready Logo Program. For this reason and because Contiki uses C-like programming (versus the nesC used by TinyOS), in our architecture we selected ContikiOS against TinyOS. In particular, the Hypervisor Module makes use of Contiki commands to manage sensors and perform their specific functionalities. In coordination with the Autonomic Enforcer, it tracks sensors as nodes, detecting if they are moving, entering or leaving the system. It periodically runs an Initialization process to detect changes in the nodes configuration (e.g., their position) and in their availability. It is responsible for extracting data from packets sent by sensors, and makes them available to the SOS Agent.

In a site, to guarantee the scalability of the architecture, multiple instances of SMs for different SIs are connected to a single SOS Agent, as shown in Figure 3. They are independent processes, which can be executed even in different hosts. Inside the SIs, different types of nodes organization can be considered: only one SN managed by an administration, a set of several independent devices, or both.

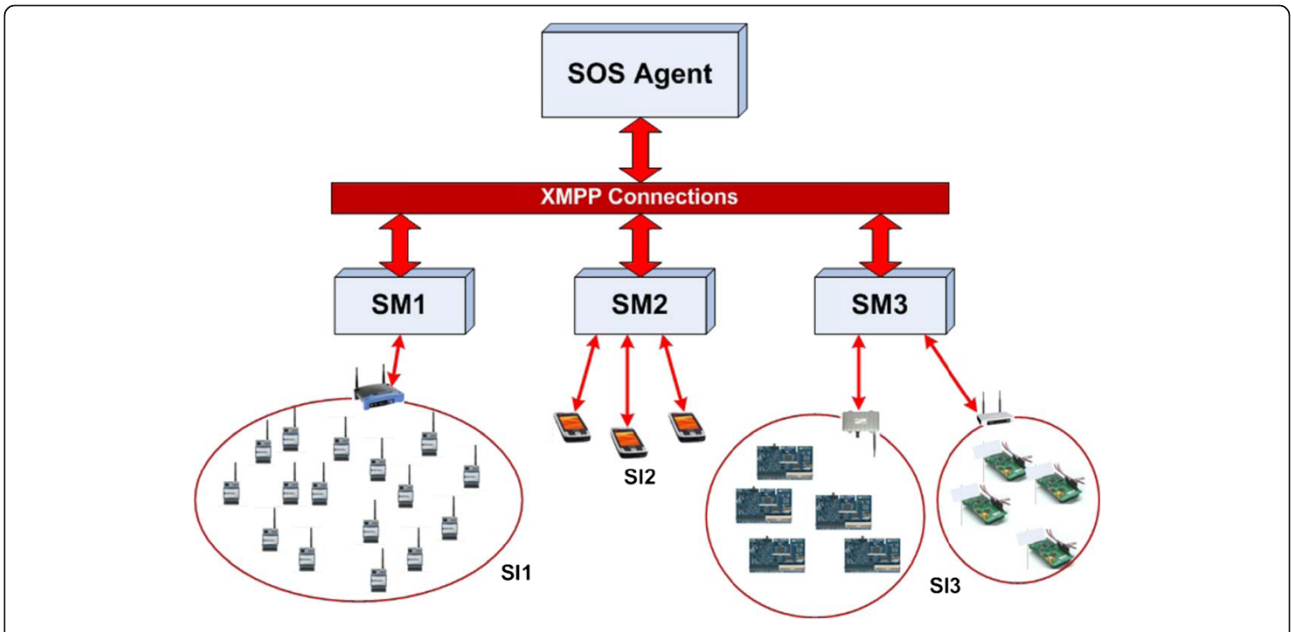

Figure 3 Hierarchical relationship between a SOS Agent and one or more SMs.

In fact, the SM abstracts the hardware features of sensing devices, the communication technologies, and the topologies for communications among sensors. The communication between SMs and the SOS Agent is based on the XMPP communication protocol. XMPP is widely used (see GTalk chatting protocol of Google) and very flexible (contrary to other messaging/signalling protocols, e.g., SIP) since it offers:

- decentralization of the communication system (i.e., no central server exists);
- flexibility to maintain system interoperability;
- fault-tolerance and scalability in the management of connected entities;
- native security features based on the use of channel encryption and/or XML encryption;
- NAT and Firewall pass-through capabilities.

Conclusions

This article intends to shift the boundaries towards a Cloud of sensors and the like, where sensors and actuators not only can be discovered and aggregated, but also dynamically provide as a service, applying the Cloud provisioning model. Having in mind the (agreed) user requirements, it is thus possible to establish Sensors and Actuators as Service providers. The SAaaS envisages new scenarios and innovative, ubiquitous, value-added applications, disclosing the sensing and actuation world to any user, a customer and at the same time a potential provider as well, thus enabling an open marketplace of sensors and actuators.

This requires an ad hoc infrastructure that has to deal with the management of sensing and actuation resources provided by both mobiles and SNs, addressing the volatility of mobiles through volunteer-based techniques, in a SAaaS perspective.

A possible area of application of such idea could be the IoT. To this purpose, it is necessary to deal with things, exploiting the well-known ontologies and semantic approaches shared and adopted by users, customers, and providers to detect, identify, map, and transform sensing resources. In this article, we identify and outline the roadmap to implement this challenging vision. A high-level modular architecture has been defined, identifying blocks to deal with all the issues herein discussed. Such architecture offers data gathered from many heterogeneous SIs to Internet clients in a uniform way, by using an abstraction layer designed according to the specification of the SWE standard. To support different types of sensors, the interaction with heterogeneous sensors has been accomplished using the Contiki Operating System.

Many topics are still open problems and challenges, thus material for future work. We specifically aim to develop advanced services for data filtering and aggregation, in order to apply them to a specific Smart City use case.

Endnotes

[a]www.ict-sensei.org/. [b]http://www.slideshare.net/skripnikov/ericsson-smart-city-vision.

Competing interests

The authors declare that they have no competing interests

Author details

[1]Inria, Paris, France. [2]National Technical University of Athens, Athens, Greece. [3]University of Messina, Messina, Italy. [4]Duke University, Durham, USA.

Received: 13 July 2012 Accepted: 13 July 2012
Published: 8 August 2012

References

1. http://www.datamation.com/cloud-computing/7-hot-cloud-computingtrends-for-2012-1.html
2. *The Top 10 Smart Cities On The Planet*, http://www.fastcoexist.com/1679127/the-top-10-smart-citieson-the-planet
3. *Sensing troubled waters in a Smarter City*, http://www.ibm.com/ibm100/us/en/service/ grants/centennial grant indian inst tech 2011.html
4. *Sensor Web Enablement, Sensor Web Enablement*, 2011. Available: http://www.opengeospatial.org/standards
5. A. Dunkels, B. Grnvall, T. Voigt, *Contiki—a lightweight and flexible operating system for tiny networked sensors, in Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN '04)* (, Tampa, Florida (USA), 2004), pp. 455–462. *2004*
6. M. Iqbal, D. Yang, T. Obaid, T.J. Ng, H.B. Lim, *Demo abstract: a service-oriented application programming interface for sensor network virtualization, in Information Processing in Sensor Networks (IPSN),10th International Conference* (, Chicago, Illinois (USA), 2011), pp. 143–144. 2011
7. S. Alam, M. Chowdhury, J. Noll, *Senaas: an event-driven sensor virtualization approach for internet of things cloud, in IEEE International Conference on Networked Embedded Systems for Enterprise Applications (NESEA)* (, Suzhou (China), 2010), pp. 1–6. *2010*
8. H.B. Lim, M. Iqbal, T.J. Ng, *A virtualization framework for heterogeneous sensor network platforms, in Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems* (SenSys'09) Berkeley (USA, 2009), pp. 319–320. doi:10.1145/1644038.1644080
9. M. Yuriyama, T. Kushida, *Sensor-cloud infrastructure—physical sensor management with virtualized sensors on cloud computing, in 13th International Conference on Network-Based Information Systems (NBiS)* (, Takayama (Japan), 2010), pp. 1–8. *2010*
10. A.P. Jayasumana, Q. Han, T.H. Illangasekare, *Virtual sensor networks—a resource efficient approach for concurrent applications, in Fourth International Conference on Information Technology. ITNG '07* (, Las Vegas, Nevada (USA), 2007), pp. 111–115. 2007
11. K. Aberer, M. Hauswirth, A. Salehi, *Infrastructure for data processing in large-scale interconnected sensor networks, in 2007 International Conference on Mobile Data Management* (, Mannheim (Germany), 2007), pp. 198–205
12. C. Reed, M. Botts, J. Davidson, G. Percivall, *Ogc(r) sensor web enablement: overview and high level architecture, in IEEE Autotestcon* (, Baltimore, Maryland (USA), 2007), pp. 372–380. 2007
13. H. Neuhaus, M. Compton, *The semantic sensor network ontology: a generic language to describe sensor assets, in AGILE Workshop Challenges in Geospatial Data Harmonisation* (, Hannover (Germany), 2009), p. 112117
14. M. Compton, H. Neuhaus, K. Taylor, K.N. Tran, *Reasoning about sensors and compositions, in Proceedings of the Semantic Sensor Networks* (, Washington DC (USA), 2009), pp. 33–48. 2009
15. H. Patni, C. Henson, A. Sheth, *Linked sensor data, in International Symposium on Collaborative Technologies and Systems (CTS)* (Illinois (USA), Chicago, 2010), pp. 362–370
16. D.L. Phuoc, M. Hauswirth, *Proceedings of the 2nd International Workshop on Semantic Sensor Networks (SSN09) in Conjunction with ISWC*, in *Linked open*

data in sensor data mashups, ed. by D.D. Kerry Taylor (, vol. 522 (CEUR) 2009)
17. C. Henson, J. Pschorr, A. Sheth, K. Thirunarayan, *Semsos: semantic sensor observation service, in International Symposium on Collaborative Technologies and Systems. CTS '09* (, Baltimore, Maryland (USA), 2009), pp. 44–53. 2009
18. M. Eid, R. Liscano, A. El Saddik, *A universal ontology for sensor networks data, in IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, CIMSA 2007* (, Ostuni (Italy), 2007), pp. 59–62
19. C. Goodwin, D. Russomanno, *An ontology-based sensor network prototype environment, in Proceedings of the 5th International Conf. on Information Processing in Sensor Networks* (, Nashville, Tennessee (USA), 2006), pp. 1–2
20. M. Compton, C. Henson, L. Lefort, H. Neuhaus, A. Sheth, A survey of the semantic specification of sensors. Networks **522**, 17 (2009). http://ceur-ws.org/Vol-522/p6.pdf
21. K. Kumar, Y.-H. Lu, Cloud computing for mobile users: can offloading computation save energy? Computer **43**(4), 51–56 (2010)
22. J. Bourcier, A. Diaconescu, P. Lalanda, J.A. McCann, Autohome: an autonomic management framework for pervasive home applications. ACM Trans. Auton. Adapt. Syst **6**(1), 8:1–8:10 (2011). http://doi.acm.org/1921641.1921649
23. T.F. Bissyande, L. R'eveill'ere, Y.-D. Bromberg, J.L. Lawall, G. Muller, *Bridging the gap between legacy services and web services, in Proceedings of the ACM/IFIP/USENIX 11th International Conference on Middleware* (ser. Middleware '10 (Springer-Verlag, 2010), Bangalore (India), 2010), pp. 273–292
24. R. Golchay, F.L. Mouel, S. Fr´enot, J. Ponge, Towards bridging IOT and cloud services: proposing smartphones as mobile and autonomic service gateways. CoRR abs/1107.4786 (2011)
25. F. Hao, T.V. Lakshman, S. Mukherjee, H. Song, Enhancing dynamic cloud-based services using network virtualization. SIGCOMM Comput. Commun. Rev **40**(1), 67–74 (2010). doi:10.1145/1672308.1672322
26. M.S. Lew, N. Sebe, C. Djeraba, R. Jain, Content-based multimedia information retrieval: state of the art and challenges. ACM Trans. Multimed. Comput. Commun. Appl. **2**(1), 1–19 (2006). doi:10.1145/1126004.1126005
27. M. Naphade, G. Banavar, C. Harrison, J. Paraszczak, R. Morris, Smarter cities and their innovation challenges. Computer **44**(6), 32–39 (2011)
28. J. Lee, S. Baik, C. Lee, Building an integrated service management platform for ubiquitous cities. Computer **44**(6), 56–63 (2011)
29. K. Su, J. Li, H. Fu, *Smart city and the applications, in International Conference on Electronics, Communications and Control (ICECC)* (, Ningbo (China), 2011), pp. 1028–1031. 2011
30. J.M. Hernandez-Munoz, J.B. Vercher, L. Munoz, J.A. Galache, M. Presser, L.A.H. Gomez, J. Pettersson, in *The future internet, in Smart Cities at the Forefront of the Future Internet*, ed. by J. Domingue, A. Galis, A. Gavras, T. Zahariadis, D. Lambert (Springer-Verlag, Berlin, 2011,), pp. 447–462. http://dl.acm.org/citation.cfm?id=1983741.1983773
31. J.M. Hernandez-Munoz, J.B. Vercher, L. Munoz, J.A. Galache, Presser, *Ubiquitous sensor networks in ims: an ambient intelligence telco platform* (ICT Mobile Summit, Stockholm, 2008), pp. 10–12
32. M. Fazio, M. Paone, A. Puliafito, M. Villari, *Heterogeneous sensors become homogeneous things in smart cities, in International Workshop on Extending Seamlessly to the Internet of Things (esIOT12)* (Palermo, Italy, 2012)
33. J. Ko, J. Eriksson, N. Tsiftes, S. Dawson-haggerty, J. Vasseur, M. Durvy, A. Terzis, A. Dunkels, D. Culler, Beyond interoperability pushing the performance of sensor network ip stacks. Networks **19**(2), 112–113 (2011). http://www.ncbi.nlm.nih.gov/pubmed/15664783
34. N. Tsiftes, A. Dunkels, *A database in every sensor, in Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems, ser. SenSys '11* (, New York (USA), 2011), pp. 316–332. 10.1145/2070942.2070974
35. D. Ballari, M. Wachowicz, M.A. Manso, Metadata behind the interoperability of wireless sensor network. Sensors **9**, 3635–3651 (2009)
36. M. Fazio, M. Paone, A. Puliafito, M. Villari, *Heterogeneous Sensors Become Homogeneous Things in Smart Cities, in International Workshop on Extending Seamlessly to the Internet of Things* (, Palermo, Italy, esloT 2012), p. 315319. 46 –July 2012
37. A.C. Viana, N. Mitton, L. Schmidt, M. Vecchio, *A k-layer self-organizing structure for product management in stock-based networks, in 7th IEEE International Conference on e-Business Engineering* (, Shanghai, China, ICEBE 2010), p. 263269. 2010
38. *The Apache Cassandra Project Develops a Highly Scalable Second-Generation Distributed Database*, http://cassandra.apache.org
39. S. Distefano, G. Merlino, A. Puliafito, *Sensing and Actuation as a Service: a new development for Clouds, in 11th IEEE International Symposium on Network Computing and Applications (IEEE NCA12)* (, Boston, Massachussets (USA), Aug 2012), pp. 153157–2324
40. F. Tusa, M. Paone, A. Celesti, M. Villari, *An innovative open source middleware for managing virtual resources in federated clouds* (Published on the BOOK of Open Source Cloud Computing Systems: Practices and Paradigms IGI Global, 2011), p. 6189. http://cloudtechnologies.morfeo-project.org/archives/open-source-cloud-systems-call-for-chapters
41. M. Fazio, M. Paone, A. Puliafito, M. Villari, HSCLOUD: cloud architecture for supporting homeland security. Int. J. Smart Sensing Intell. Syst. **5**(1), 246–276 (2012)
42. B. Caprarescu, N.M. Calcavecchia, E.D. Nitto, D.J. Dubois, *SOS cloud: self-organizing services in the cloud, in Bionetics'10: Bio-Inspired Models of Network, Information and Computing Systems* (, Boston (USA), 2010), pp. 151–158
43. N.M. Calcavecchia, D. Ardagna, E. Nitto, *Run-time Models for Self-managing Systems and Applications, ser. Autonomic Systems*, in *The emergence of load balancing in distributed systems: the selflet approach*, ed. by D. Ardagna, L. Zhang (Springer, Basel, 2010,), pp. 97–124. doi:10.1007/978-3-0346-0433-8/_5
44. S. Distefano, A. Puliafito, M. Rak, S. Venticinque, U. Villano, A. Cuomo, G.D. Modica, O. Tomarchio, *Qos management in cloud@home infrastructures, in International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery* (CyberC, Beijing (China), 2011), p. 190196
45. T. Anderson, *The BOINC credit system, Tech. Report* (, University of California, 2012). http://boinc.berkeley.edu/trac/wiki/CreditNew
46. P. Kacsuk, J. Kovacs, Z. Farkas, A.C. Marosi, Z. Balaton, Towards a powerful european dci based on desktop grids. J. Grid Comput. **9**(2), 219–239 (2011). doi:10.1007/s10723-011-9186-z
47. P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, D. Culler, *Ambient Intelligence*, in *TinyOS: an operating system for sensor networks ambient intelligence*, ed. by W. Weber, J. M. Rabaey, E. Aarts (Springer, Berlin, 2005,), pp. 115–148

doi:10.1186/1687-1499-2012-247
Cite this article as: Mitton *et al.*: Combining Cloud and sensors in a smart city environment. *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:247.