Pers Ubiquit Comput (2012) 16:291–308
DOI 10.1007/s00779-011-0399-8

ORIGINAL ARTICLE

Adding sense to the Internet of Things

An architecture framework for Smart Object systems

Tomás Sánchez López · Damith C. Ranasinghe · Mark Harrison · Duncan McFarlane

Received: 15 October 2010 / Accepted: 4 April 2011 / Published online: 3 June 2011

Abstract The Internet of Things (IoT) concept is being widely presented as the next revolution toward massively distributed information, where any real-world object can automatically participate in the Internet and thus be globally discovered and queried. Despite the consensus on the great potential of the concept and the significant progress in a number of enabling technologies, there is a general lack of an integrated vision on how to realize it. This paper examines the technologies that will be fundamental for realizing the IoT and proposes an architecture that integrates them into a single platform. The architecture introduces the use of the Smart Object framework to encapsulate radio-frequency identification (RFID), sensor technologies, embedded object logic, object ad-hoc networking, and Internet-based information infrastructure. We evaluate the architecture against a number of energy-based performance measures, and also show that it outperforms existing industry standards in metrics such as network throughput, delivery ratio, or routing distance. Finally, we demonstrate the feasibility and flexibility of the architecture by detailing an implementation using Wireless Sensor Networks and Web Services, and describe a prototype for the real-time monitoring of goods flowing through a supply chain.

Keywords Smart Objects · Sensors · RFID · Internet of things

T. Sánchez López (✉) · M. Harrison · D. McFarlane
Auto-ID Lab, Engineering Department,
University of Cambridge, 17 Chales Babbage Road,
Cambridge CB3 0FS, UK
e-mail: tomas.sanchez@autoidlabs.org.uk

M. Harrison
e-mail: mark.harrison@cantab.net

D. McFarlane
e-mail: dcm@eng.cam.ac.uk

T. Sánchez López
Innovation Works, EADS UK, Quadrant House,
Celtic Springs, Coedkernew, Newport NP10 8FZ, SW, UK
e-mail: tomas.sanchezlopez@eads.com

D. C. Ranasinghe
Auto-ID Lab, The Schoold of Computer Science,
University of Adelaide, Adelaide, SA 5005, Australia
e-mail: damith@cs.adelaide.edu.au

1 Introduction

The *Internet of Things* is a concept that encompasses a variety of technologies and research areas that aim to extend the existing Internet to real-world objects [1]. Advances in fields such as automatic identification, wireless communications, integrated sensing, or distributed data processing have narrowed the gap between the notion of ubiquitous computing set 15 years ago [2] and a world of networked, sensing, and intelligent 'things'. The potential benefits from the IoT realization are many, both for individuals and businesses. Some of the most promising applications include: improved management of global supply chain logistics, product counterfeit detection, manufacturing automation, smart homes and appliances, e-government (electronic official documents and currency), improved integrated vehicle health management, and e-health (patient monitoring and patient records).

Automatic identification technologies such as Radio Frequency Identification (RFID) are fundamental to the realization of the IoT because they enable "things" to be linked with their virtual identity on the Internet. RFID tags attached to objects expose unique identification numbers that can be read wirelessly by interrogating devices and

used to obtain information related to individual instances of objects managed by networked back-end systems. Miniaturised sensors now provide the ability to monitor the condition of objects and consequently make it possible to dynamically act upon changes to the status of objects such as that derived from their temperature, humidity, and chemical composition. Furthermore, historical records including both identification and sensor data can be used off-line to trace the evolution of the objects' location and status throughout their life. Low-power radio communication technologies and the availability of increasingly powerful low-cost embedded processing increase the autonomy of objects by providing them with networking capabilities and local intelligence. Finally, distributed information infrastructures using Internet protocols for communication serve as connection hubs for all the 'things', together with other resources such as databases, data mining tools, and computer networks.

Both the vastly different and increasing number of technologies involved in the IoT concept suggest that its success must inevitably require the effective integration of various technologies. Recent efforts in the standardization of RFID technologies have produced a networked infrastructure where the identities coming from wireless tags attached to real-world objects can be filtered, stored, queried, and linked to on-line object information [3, 4]. Standardization bodies such as the IEEE [5] or the Open Geospational Consortium (OGC) [6] have developed sensor standards that may use Internet protocols to access sensor descriptions and values. EU-funded projects such as PROMISE [7] or SENSEI [8] describe the use of tags with embedded processing capabilities in order to equip the objects with certain intelligence. Although such works have produced important advances in technology and standards that could empower the IoT revolution, none provides a complete integration framework. Consequently, an architecture that is capable of integrating various functionalities necessary to realize the IoT is still lacking.

Many people associate the so-called Internet of Things (IOT) with Auto-ID technologies such as barcodes, matrix codes, and low-cost passive RFID tags. While passive RFID is looking increasingly promising for automating the tracking of physical objects, the energy available from power harvested via the antenna of the tag is insufficient for powering sophisticated sensing and logging capabilities, given their current power requirements. There are many situations where it is necessary to monitor not only the location and movement of objects but also their condition. In this paper, we describe and implement an architecture for Smart Objects that extends established concepts for networked RFID with new functionalities that can support co-operative Smart Object implementations using complementary technologies, such as wireless sensor networks. This approach enables us to integrate the most significant functionalities described earlier to realize the capabilities of the IoT vision: unique and automatic identification, the sensed condition of objects, embedded processing for local intelligence and autonomy, object-to-object networking, and an Internet-based information infrastructure. The proposed framework introduces the use of the Smart Objects (SO) conceptual model as a cornerstone; Smart Objects model the capabilities of objects participating in the IoT, from the basic capacity to provide a unique identification number to more complex abilities such as the capacity to perform ad-hoc networking and object-centric complex decision-making. The SO framework is completed with an information infrastructure that leverages the capabilities of Smart Objects to provide services to end-users such as identification and condition information through a well-defined set of interfaces. Our objective in this paper is to demonstrate both the usefulness and the capability of this framework to address many of the challenges and applications of the Internet of Things, through its ability to integrate a heterogeneous set of devices and the functionalities of various technologies.

The rest of this paper is organized as follows. Section 2 introduces related technology and architectural work. Section 3 represents the core of our contribution by introducing the Smart Object framework architecture and its key design features. Section 4 evaluates the proposed architecture and Sect. 5 goes further by detailing a generic implementation. Section 6 presents an example scenario built on top of the implementation, and Sect. 7 lists the most important barriers to the adoption of our work. Section 8 finishes the paper by presenting our conclusions and lessons learned.

2 Technology background and related work

RFID has become the leading technology for automatic identification due to its advantages over other technologies such as barcodes. An RFID system consists of a transponder tag attached to an object, a reader that interrogates the tag using the wireless medium and a back-end system to organize the captured data [9]. In this way, RFID tags containing unique object IDs can be read automatically without requiring line-of-sight. Passive RFID has traditionally provided automatic object identification and the location at which the objects have been identified. Recent developments in active and semi-passive RFID provide enhanced object identification functionalities with various degrees of autonomy [10]. Rapidly evolving new applications, such as food safety and vehicle health management, are based on knowing the condition of objects (e.g., temperature, stress, strain, shock). Although RFID is an

important technology in the realization of a seamless link between individual physical objects and their digital representations, it can not provide the condition information that the next generation of real-world applications require.

Sensor technologies have attracted relatively recent popularity due to their ability to gather the real-world conditions used by modern computing applications [11]. Sensor transducers are normally incorporated as part of computing systems in order to assist or complement data collection. In this regard, recent research efforts pursue the transmission of sensor data through radio links aiming to facilitate sensor deployment. Wireless Sensor Networks (WSN) are a popular research area in this direction, where miniaturized, energy-efficient battery-powered wireless devices serve as a platform for transmitting the sensor data [12]. A recent trend is to incorporate sensors into RFID tags [13, 14].

Both RFID and sensor technologies are key enablers of the IoT because they provide the means to identify objects and to obtain their condition. However, there is a need for a common underlying infrastructure that links the physical objects to the digital domain and helps to manage their information. The definition of these links and how they bring together all the enabling technologies forms the integration architecture that might potentially realize the IoT concept. Research in such architectures is in general missing, although there are some noticeable efforts that, either on their own or in combination with others, show a great potential for filling this gap. Table 1 summarizes the most relevant-related work, either on architectures that do integrate RFID and sensors or on standardization efforts that we consider key toward the realization of the IoT. As Table 1 highlights, although the technologies for gathering, processing and distributing information about objects either already exist or are well advanced, there is little or no integration among them, leading to a lack of a comprehensive platform for heterogeneous data processing and sharing.

The term "Smart Object" is often mentioned in the literature, together with other similar concepts such as "intelligent products" [22] or "smart parts" [23]. The realization of these concepts is often as varied as the number of terms used to describe them, although they seldom link RFID and sensor technologies, as well as other fundamental features of the IoT such as ad-hoc networking or embedded object logic. A school of thought that is growing in popularity associates the Internet of Things and the Smart Object terminologies with IP-based protocols, specially with a modified version of IPv6 for low-power embedded devices called 6LoWPAN [24]. The IP for Smart Objects alliance [25], for example, promotes the use of the 6LoWPAN and ROLL [26] for networking Smart Objects. However, the IPSO alliance does not give a comprehensive definition of what Smart Objects are, nor does it provide a design document of SOs beyond the reference to IETF, IEEE, or ISO/IEC standards. Other advocates of the use of 6LoWPAN for the realization of the Internet of Things also tend to focus on adapting existing Internet protocols for constrained embedded devices, but do not concern themselves with the building blocks of true object integration [27]. For example, IP addresses are poor for describing object identity, unlike RFID-based identification systems such as the Electronic Product Code (EPC) which are structured to enable manufacturers to manage the creation of identifiers for their products and to help consumers access serial and class-level product information from those identifiers [28]. Finally, efforts such as the Web of Things [29] extend the IP-based Smart Objects idea and propose the use of a RESTful approach [30] to access sensor and actuator information. RESTful approaches leverage the way the World Wide Web works by using HTTP standard methods such as *GET* and *PUT* to access and manipulate information from networked Internet repositories. Although some work is being done to extend the REST architecture to constrained devices, such as that of the Constrained RESTful Environments (CoRE) working group from the IETF [31], the Web of Things is still just an architectural conceptualization and not a design for an integration architecture.

In the rest of the paper, we aim to introduce an architectural framework that not only supports the most important foreseen features of the IoT, but that also provides all the details for creating efficient and complete implementations. Our proposed architecture does not assume the use of any underlying communication technology and includes new features such as context-based ad-hoc network and clustering of objects, which can potentially improve the quality and meaning of the collected data.

3 The Smart Object architecture framework

Based on the main shortcomings of the related work in the area, we devise an integration framework that incorporates all the main features involved in the Internet of Things concept. Our framework will thus be able to identify and monitor the condition of objects, using existing standards where possible. The integration of ID and sensor data in relation to a particular object will be unambiguous. Objects will be capable of establishing networks, and the membership and structure of these networks will depend on the context of the situation in which the objects are involved. Objects inside the networks will cooperate to manage their resources and to take complex decisions on data routing, inter-network relationships, event generation, and others. Object data will be made available to users of the IoT via well-defined interfaces, and this data will be organized in such a way as to allow user access with various degrees of granularity. These features are summarized in Fig. 2.

Table 1 Summary of RFID and sensor integration related work

	Description	Main shortcomings	Potential improvements
PROMISE [7]	EU project using Product Embedded Information Devices (PEID) for monitoring ID and condition of objects during their life cycle	Little alignment with standards, no networking of PEIDs	Adoption of ID standards. Consider sensor (object) networking
OGC SWE [6]	Extensive set of protocols and interfaces to share sensor information in a standard way over the Internet.	IDs not globally unique. Sensors are not considered to monitor objects or products	
SENSEI [8]	EU project designing an architecture for the connectivity of global & heterogeneous sensor and actuator networks via the specification of open service interfaces	Under development. No ID standardization	Adoption of ID standards. The project needs to reach a mature state before its results can be evaluated
EPC Network [3]	Emerging industrial RFID standards architecture based on unique item identification via the Electronic Product Code (EPC)	Does not yet handle sensor data	Extend current standards with sensor data
BRIDGE [15]	EU project for developing new technologies within the EPC Network	Work with sensors does not extend the EPC Network standards	
EPC SN [16]	Auto-ID lab project to extend the EPC Network with sensors	Too complex to allow a full architecture extension	Compromise in developing a simpler functional part of the extension
ISO 18000-6 [4], 24753 [17], IEEE 1451-7 [18]	Set of standards dealing with the integration of RFID with sensors	Under development. Cooperation among standardization bodies is complex and slow	Standards need to reach a mature state before they can be used
GSN [19]	Middleware to collect and share information from RFID and WSN over the Internet	RFID and WSN data are not integrated at object level	Provisions for collecting both RFID and sensor data from an object
CoBIs [20]	EU project defining reusable services which use WSN and RFID to represent physical entities involved in business processes		
SARIF [21]	Middleware for designing applications requiring RFID and WSN information	Integration is only by spatial comparison, no mention of Internet scalability	Consider more integration methods and an explicit connection with Internet protocols

The rest of this section describes the design of an architecture that implements the integration framework features described above. Later sections will demonstrate our key contributions and evaluate the central conceptual components of the architecture framework design.

3.1 Smart Object definition

The integration of object identification and sensor data streams may be realized in multiple ways. For example, it is possible to merge ambient sensory data provided by a sensor-rich space to identities of objects entering that space by detecting when objects enter the area. However, this scenario might prove difficult to implement since detection systems must be placed on the boundaries of the space. Moreover, architectural logic must be put in place for merging both independent streams of data (sensor data and automatic identification data). Furthermore, sensor data could be deemed inappropriate since the transducers could be located at a considerable distance from the monitored objects.

The previous scenario highlights one of the main drivers for the use of sensor data at the object level. Studies have shown that even in closed 14-m refrigerated containers, the variation of temperature across pallets can reach up to 35 percent [32] and that in chilled delivery vans, the temperature from one package to another can vary up to 4°, resulting in the potential growth of bacteria among perishables [33]. Other studies have also shown how the distribution of cargo in confined spaces can influence the detection of safety-related events such as fires when environment sensors are used [34].

With the new developments on integrated circuitry, micro-electro-mechanical systems (MEMS), wireless communications, and embedded technologies in general, the vision of an integration that occurs at the hardware level is more plausible and logical than ever. As the amount and type of information that can be embedded into assets increases, we witness an evolution toward object-centric systems, where manufactured items take control over the flow of information which was traditionally retrieved manually by human operators. As a result of the

augmented capabilities and "intelligence" that the object-centric paradigm supports, this new generation of assets has been called "smart products" or, more generally, "Smart Objects" [35]. Our framework for building Smart Objects is based on five fundamental properties: Smart Objects are those objects that

- possess a unique identity.
- Are able to sense and store measurements made by sensor transducers associated with them.
- Are able to make their identification, sensor measurements, and other attributes available to external entities such as other objects or systems.
- Can communicate with other Smart Objects.
- Can make decisions about themselves and their interactions with external entities.

The Smart Object properties directly contribute to the realization of the framework features described throughout this section. Figure 1 provides an overview of the architecture design, and its conceptual components are described in the following subsections. Figure 2 summarizes the architecture design components that address each of the Smart Object framework features and their relationship to the SO properties.

3.2 Object identification and sensor integration

Objects such as consumer goods, product parts, assembly machinery, logistics and transportation items (e.g., pallets, containers, vehicles), warehouses, retailers' facilities, or end-user assets may be subject to condition monitoring and therefore provide valuable information for themselves or other objects in their vicinity. In order to monitor their condition, our architecture design makes use of embedded devices with wireless communication capabilities. These devices would be attached to the objects, becoming a part of them, the same way a bar-code is integrated into the vast majority of today's products.

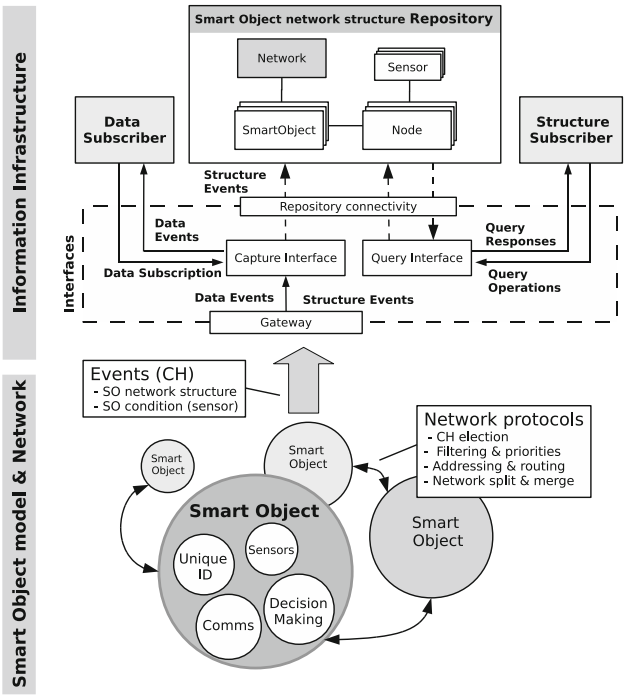


Fig. 1 Smart Object system architecture

The devices attached to the objects are meant to provide a global identification, to sense the status of the object and/or their surroundings and to provide a wireless interface for data communication. These devices are thus aligned with the developments in sensor-enabled active tags, with the fundamental difference that our devices (which we call simply "nodes") hold the logic to not only communicate with each other but to form ad-hoc networks for the better management of resources and data capture. Our nodes are basically WSN devices with a standardized unique identification. However, unlike traditional WSN, nodes are expected to move together with the objects they monitor and to continuously interact with other nodes. This dynamism is not a common characteristic of such restricted devices and coping with it constitutes a major part of our work in this area. Algorithms and strategies for dynamic object networking and administration are described in Sects. 3.3 and 3.4

The use of the same device for capturing both ID and sensor data provides integration at the hardware level with no ambiguity. Sensor and ID data are matched the instant they are captured and travel together in communication packets and events. Consequently, they are stored and discovered as one.

3.3 Object networking

RFID and sensor technologies that capture data about specific objects have traditionally used point-to-point communication between the device representing the object (e.g, RFID tag) and the data reader (e.g., RFID reader). This form of communication is sufficient for simple applications, but might be insufficient when large numbers of devices coexist or a certain degree of intelligence on the data gathering is required.

Smart Object networks utilize ad-hoc networking for communications and clustering for energy management. Clustering extends the network lifetime by electing a representative network member, or Cluster Head (CH), which collects all the communication within the network and forward it to the infrastructure. Cluster heads are beneficial because they handle the burden of communicating with external entities, a task that is usually more power-intensive than intra-network communication. Furthermore, CHs

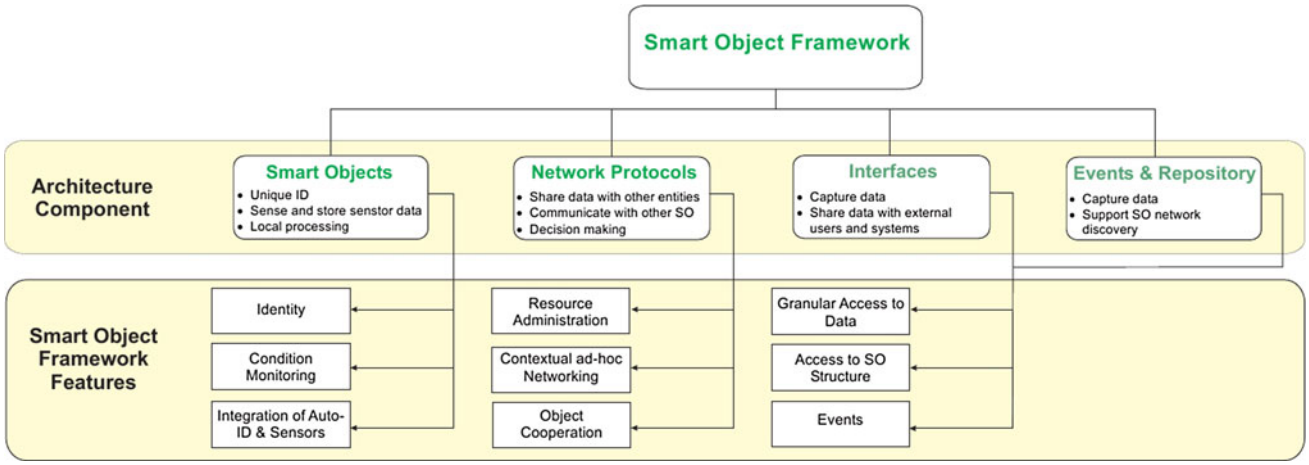


Fig. 2 Mapping between properties, features and architecture components of the Smart Object framework

can provide a centralized control over network functionalities when needed. They may also be elected according to their particular capabilities, such as radio ranges, and processing power. On the downside, CHs consume more energy that the other network members and their role must be periodically rotated in order to avoid the premature exhaustion of their battery power.

Clustering is a relatively popular technique for balancing the communication load and the energy expenditure in wireless networks. What makes the approach used for Smart Objects different from existing approaches is the combination of a number of techniques aimed at serving mobile and independent groups of nodes:

- *Energy-based CH election* The residual energy of the network nodes (i.e, the amount of energy left in its battery to enable its operation) is used to calculate the best candidate when the CH role is rotated.
- *Variable CH period* Residual energy is also used to calculate the amount of time that a node will have the CH role until the next rotation occurs. This technique effectively controls the energy expenditure of CHs thereby maximizing the number of alive nodes.
- *Double clustering* Physically large Smart Objects (e.g., containers, logistic vehicles) might be equipped with more than one node to provide accurate status across the physical space. In the same way that a network of Smart Objects elects a CH (NetCH) to balance the communication burden of the SO network with other infrastructure components, a Smart Object with various nodes elects one of the nodes as a Smart Object CH (SoCH) to balance the load of communicating with other Smart Objects. As we will show in Sect. 4, this double clustering mechanism provides benefits in terms of energy consumption distribution and network lifetime.
- *No global cluster information* Nodes do not need to store information about all the members of a cluster. This property is vital for the scalability of the algorithms.
- *No need for synchronization* Cluster nodes do not need to be synchronized in order to elect the CH or to operate inside the cluster. Synchronization among network nodes is time-intensive and energy-consuming.
- *Mobility support* The algorithms support both the movement of nodes within the network as well as merging and splitting of networks. Section 3.4 describes in more detail how the SO architecture supports these features.
- *Low algorithm complexity* The number of nodes fully participating in a given CH election is always lower than the total number of nodes in the cluster, making our approach less complex than any other clustering algorithm.

The first two items on the list outline the algorithm for selecting new cluster heads. When a node ends its CH period, it sends an `Election` message to announce a new election procedure and request other nodes to bid for becoming a new CH. This message contains a computed time, which is the proposed period for staying as a CH. This proposal is based on the current residual energy of the CH node, so that the greater the residual energy the greater the proposed time period. When a cluster member receives an `Election` message, it sends back an `Election Response` message only if its own computed time period is higher than the CH proposal. After waiting for a fixed period of time for counterproposals from the rest of the cluster members, the CH compares the time periods of all the received responses and sends an `Election Set` message to all the cluster members setting the node with the highest proposed period as the new CH. The new CH

will remain in its role for the timer period stated in its proposal.

Networking of SO also differs from other wireless networking approaches in that it is additionally employed as a filter to data capture. Rather than forming networks with random objects, SO may only interact with one another if they share a common context (e.g., participate in the same shipment, are parts of the same composite object, are stored in the same place). To achieve this, a number of SO attributes and algorithms for classifying and prioritizing SO interactions are also included within the logic of the nodes. This processes effectively influences how the clusters are formed, transforming the network clustering into more than a balancing technique, and shaping the network structure into logically connected groups of objects.

3.4 Network administration

Smart Objects are mobile objects that may move independently of their network and therefore may cause unexpected network changes such as the departure from their current network (i.e, network split) or the arrival to a new network (i.e, network merge). Traditional WSN research has focused on data routing and resource management on static systems. However, the dynamic nature of SO interactions differs from existing WSN research and requires further considerations to be embedded into the network protocols. As a part of the research in Smart Object systems, mechanisms to handle spontaneous additions and departures of SO to/from the network were developed.

Just as computer networks require each network node to have an address in order to route communications between any two computers, SO networks also require each object to have an address. The address of an object should be different from its ID because object IDs are selected with a business context in mind (e.g., bar codes, EPCs), while object addresses are selected depending on the topology of the network (e.g IP addresses). As mentioned in Sect. 2, a number of initiatives have emerged aiming to introduce IP protocols in the Smart Object arena. Although implementation of these protocols such as 6LoWPAN [24] have achieved small network stacks suitable for some embedded devices, resource constrained hardware implementations (e.g with up to 4 kb of RAM) are still unable to hold both the IP stacks and the clustering algorithms presented in this paper. We believe that cheap (and thus feature-limited) WSN-like devices are the future in the mass production of Smart Objects, and therefore can not require the use of IP protocols for our architecture. For this reason, a new addressing and routing mechanism was developed specially tailored for the dynamic and distributed nature of the Smart Object networks.

The addressing and routing mechanism developed for the SO architecture, called Sequence Chain (SC) [36], have been integrated into the algorithms that handle the formation of clusters and are therefore within the processes that manage the splitting and merging of networks. Sequence Chain assigns addresses following a hierarchical structure and therefore the routing of packets between network nodes follows the resulting *tree* (i.e, tree routing). Sequence Chain has the following main properties:

- *Unique and reusable addresses* Addresses are unique and thus require no duplicate address detection. The address of a node that leaves the network can be reused for nodes joining the network at a later time.
- *Network merge and split support* A network can recover from the leaving or merging of both single nodes and networks.
- *Fully distributed* Each address is calculated locally without the need for any central authority. Network recovery after a split or merger is also managed locally by the affected portions of the network.
- *Low overhead and scalable* Addresses are not limited in size but grow with the addition of new nodes. The local decision-making results in low latencies and processing complexity that do not increase with the number of network nodes. Routing decisions are taken based only on address comparison and do not require route discovery.

3.5 Event generation and data sharing

A Smart Object itself is not enough to achieve the expected benefits of "smart products". There is also a need for flexible and efficient ways of managing the Smart Object information and making it available to end-users. This functionality is achieved by providing an *information infrastructure* with which Smart Objects can communicate.

The SO architecture presented here is event driven. Events are generated by changes in the SO network conditions and are forwarded from their point of origin toward the information infrastructure. Two family of events exist: those that are created to reflect changes of the network structure (e.g network split or merge) and those that are used to periodically report sensor data (i.e, Smart Object condition). The former set of events are used to create and update a virtual network structure on the infrastructure side, which we call the *Smart Object network structure repository*. This repository provides clients with the ability to discover the Smart Objects that form part of the same network, their particular relations with other Smart Objects and the sensors and output types supported by each SO.

As described in previous sections, the formation of networks is controlled by a mechanism that considers

contextual Smart Object relationships. In this way, clients accessing the SO network structure repository can determine which objects form part of the same contextual situation and decide which object or sets of objects would produce the sensor data most relevant to them. This design effectively provides a mechanism whereby a client has access to a vast variety of logically related data sources with a very narrow entry point (e.g., by only knowing the ID of one of them)

Events sent from the Smart Object networks to the Information Infrastructure will be received and transformed into messages understandable by the Information Infrastructure components. Traditional WSN use *base stations* to collect wireless sensor data and to connect wireless sensor nodes to other computing infrastructures. A base station would normally receive wireless messages, process them, and either perform only local computation (e.g display the result of analysing their contents) or send their information to other components over a network for further analysis and processing. Much in the same way, the *gateway* component of the SO architecture takes the role of physically receiving messages from the protocols used by the SO wireless nodes, decoding those messages and interpreting their contents, and finally putting those contents into infrastructure-native messages that can be forwarded toward the rest of the Information Infrastructure components. The need for a gateway node is the result of a generic framework description, which is not bound to any specific transport protocol, on both the SO network and the Information Infrastructure.

Infrastructure clients access the SO network structure repository through a *query interface* and may subscribe to sensor data events through a *capture interface*. The design of these interfaces is such that it allows access to information with various degrees of granularity. For example, a client might subscribe to the data from a single sensor, a Smart Object, or a whole network. Subscriptions might also be specified for any data from a particular sensor type. The flexibility of these interfaces, coupled with the logical relationships among the objects of a network, results in a powerful yet simple design.

4 Architecture evaluation

Both the Smart Object network algorithms and the infrastructure design were evaluated. Apart from confirming the design features described in previous sections, the network protocols were evaluated in terms of energy consumption and the variables that influence it. The addressing and routing mechanism was also compared with the de-facto industry standard ZigBee [37]. The information infrastructure design was evaluated in terms of its flexibility to adapt to various implementation technologies and its scalability in terms of supporting increasing loads of Smart Object data and concurrent clients.

A mathematical energy model of the SO nodes was developed and plugged into a purpose-built Smart Object network simulator. The reasons for developing a simulator tool were fourfold:

1. To be able to accurately control and simulate the particular attributes of Smart Objects and their networks, which differ in many ways from regular Wireless Sensor Networks.
2. To have a graphical representation of the complex and distributed SO networks, including each SO real-time status and the messages exchanged between them.
3. To serve as a monitor for our implementation trials, whereby a *sniffer* node would listen to the messages exchanged in an implemented SO network and transmit them to the simulator tool. In this *monitoring* mode, the tool would interpret the messages from the sniffer node and draw a graphical representation of the status of the network, in real-time.
4. To be able to extend an implemented network with simulated Smart Objects. In this way, a hybrid simulation/implementation scenario can be established, providing an effective way of testing the implemented network with communication loads that would not be possible due to the lack of hardware resources.

Simulations used a number of variables that influence the performance of the SO networks, such as the number of nodes per cluster, the CH rotation time, the SO priority within the cluster membership queues, or the radio range of the nodes. The simulation assumptions are detailed on Table 2. Other simulation details will be given as necessary when describing particular results.

The key results of our evaluation are discussed in the following subsections.

4.1 Improved network lifetime and balanced energy distribution

The double clustering mechanism and the variable rotation time of CHs results in a balanced energy distribution among network nodes and longer network lifetimes.

Figure 3 shows the average decrease in the residual energy of the Smart Object CHs when the number of nodes per SO is varied. The number of SOs in the network was fixed to four. The zigzagging patterns reflect the CH rotation mechanism, with decreasing residual energy as the simulation progresses. In Fig. 3a, the double clustering is used, and both the roles of NetCH and SoCH are rotated. As the number of nodes per SO increases, CHs will use

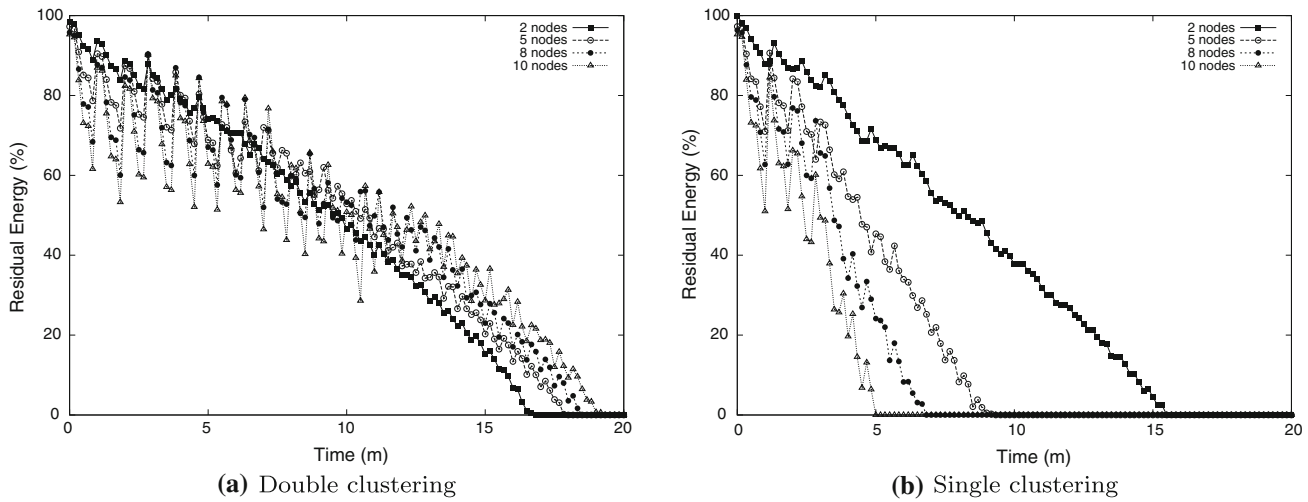
Table 2 Simulation assumptions

Assumption	Description
Simulation area	500 m^2
Initial node position	Randomized
Addressing tree construction	The root of the tree is assigned to the closest node to the center of the simulation area, and the rest of the nodes are joined to the network in an increasing distance from the center
Initial residual energy	100 % if not explicitly stated
Communication range	Enough to reach other network nodes, if not stated otherwise
Energy consumption for transmitting (e_{tx}) and receiving (e_{rx})	$e_{tx} = e_{circuitry} * k + e_{amplify} * k * d^c, e_{rx} = e_{circuitry} * k$, where e_{tx} is the energy used while transmitting data, e_{rx} is the energy used while receiving data, $e_{circuitry}$ is the energy needed to feed the tx/rx circuits, $e_{amplify}$ is the energy required to amplify the tx signal, k is the length of the packet to be transmitted, d is the distance between source and target and c is the path loss exponent [38].

more energy in each period in order to cope with the increasing packet forwarding needs. This is reflected in the graphs by steeper peaks. However, the greater number of nodes also contributes to an increase in the lifetime of the network by providing more candidates for the CH election. As the graph shows, the simulation with 10 nodes per SO continues for a longer time than the other simulations with fewer nodes. Figure 3b shows the same set of simulations but only activating the NetCH rotation mechanism. As the graph shows, the increase in the number of SO nodes now shortens the network lifetime.

Figure 4 shows the average decrease in the residual energy of the network when the time between CH rotations is changed. The short lines perpendicular to the *x*-axis represent the time that the first node of the network completely exhausted its battery. Two observations can be deduced from these graphs: Firstly, the network lifetime tends to increase as the period increases. Secondly, the distribution of energy among the network nodes worsens as the period increases. Long periods postpone the CH rotation excessively, provoking the *death* of the CH before the election takes place. This is a sign of poor energy distribution, and a balance should be found between long network lifetimes and ensuring that all the nodes of a network keep enough energy to operate.

Optimal balanced results can be obtained by incorporating the number of nodes of the cluster in the calculation of the CH rotation period in every CH election process. An energy model was developed in [39], and its solution produced a unified equation that network nodes might use to calculate the optimal CH rotation period as a function of the current number of nodes in the cluster. The number of nodes in a cluster is a known variable calculated in a distributed manner throughout the life of the network [36]. Figure 5 shows the average decrease in the residual energy of the network when the time between CH rotations is calculated dynamically using the aforementioned equation. The time of the death of the first node is `12.16` min from



(a) Double clustering **(b)** Single clustering

Fig. 3 Average energy consumption pattern of the network with 3 (**a**) and without 3 (**b**) the Smart Object cluster head (*SoCH*) rotation mechanism activated

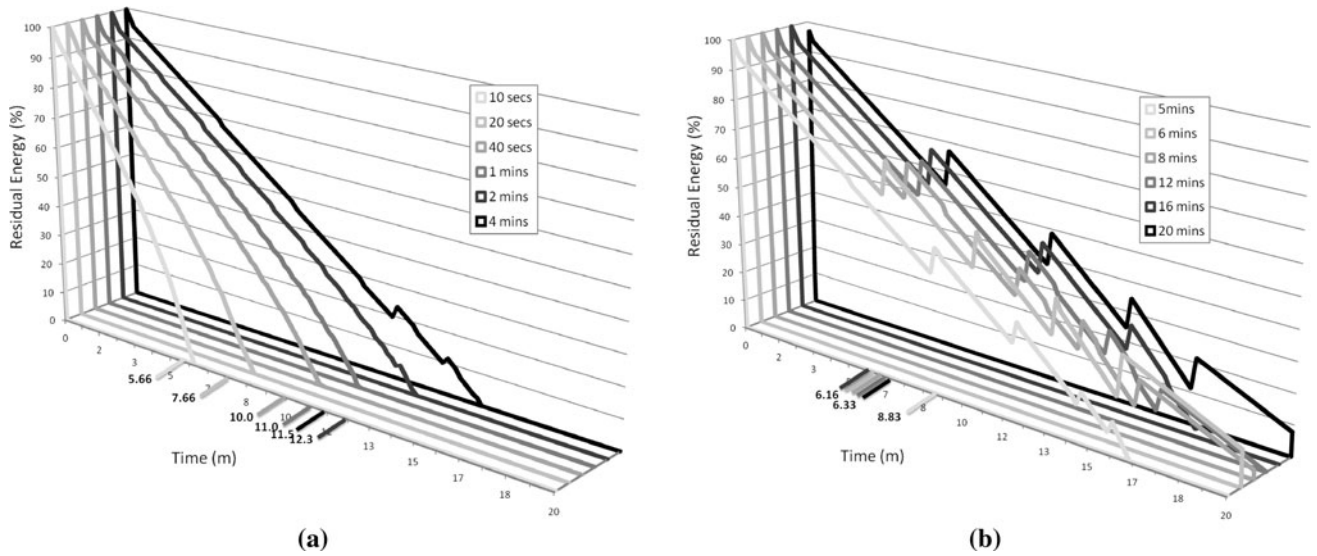


Fig. 4 Average energy consumption pattern of the network when increasing the CH period

the start of the simulation. This time is comparable with the highest time obtained from Figure 4, which is `12.3` min for a rotation period of 2 min. However, the new results produce a network lifetime of almost `20` min, which is far better than the static `2` min period, which barely reached the `13` min mark. We can therefore conclude that the dynamic calculation of the optimal CH rotation period depending on the number of nodes present in the cluster maximizes the network lifetime while balancing the energy consumption among the cluster members.

4.2 Lower overhead and greater scalability

The protocol for electing a new CH not only generates a small amount of messages but its benefits over the number of exchanged messages increase with the number of nodes within the cluster. Figure 6a shows the total number of messages exchanged in the CH election process in a 20 min simulation. The "# election procedures" line plots the number of election procedures that were undertaken for each variation of the number of nodes. Higher number of nodes results in longer CH periods due to the better distribution of the available energy and therefore in lower number of election procedures. A similar effect was observed in Fig. 5 and explained in Sect. 4.1 The "No protocol optimization" line plots the messages exchanged if a simple request/response protocol is used. In this protocol, a CH broadcasts an `Election` message within its cluster at the beginning of new rotation period. This message would be answered by all the cluster members, and a new CH would be elected according to their responses. This strategy results in a linear increase in the number of messages with the number of cluster members. The "With protocol optimization" line shows the number of exchanged messages using the Smart Object protocols. In this protocol, not all the cluster members answer the CH `Election` message, but only those that have a higher residual energy than the current CH. This strategy avoids the exchange of unnecessary messages, yielding increasing savings with increasing numbers of cluster members.

To emphasize the improvement of the proposed protocol, Fig. 6b shows the average number of messages exchanged in each CH election procedure, obtained by dividing the total number of messages by the number of CH election procedures in each simulation. Based on these results, we can conclude that the protocol optimization strategy delivers a highly scalable architecture.

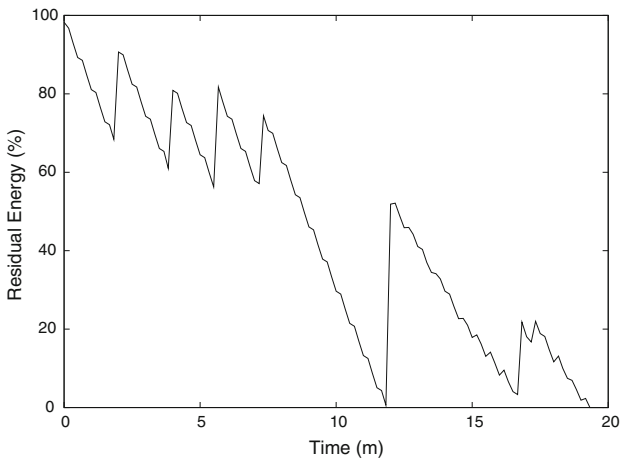


Fig. 5 Average energy consumption pattern of the network when dynamically calculating the CH rotation period according to the number of cluster nodes

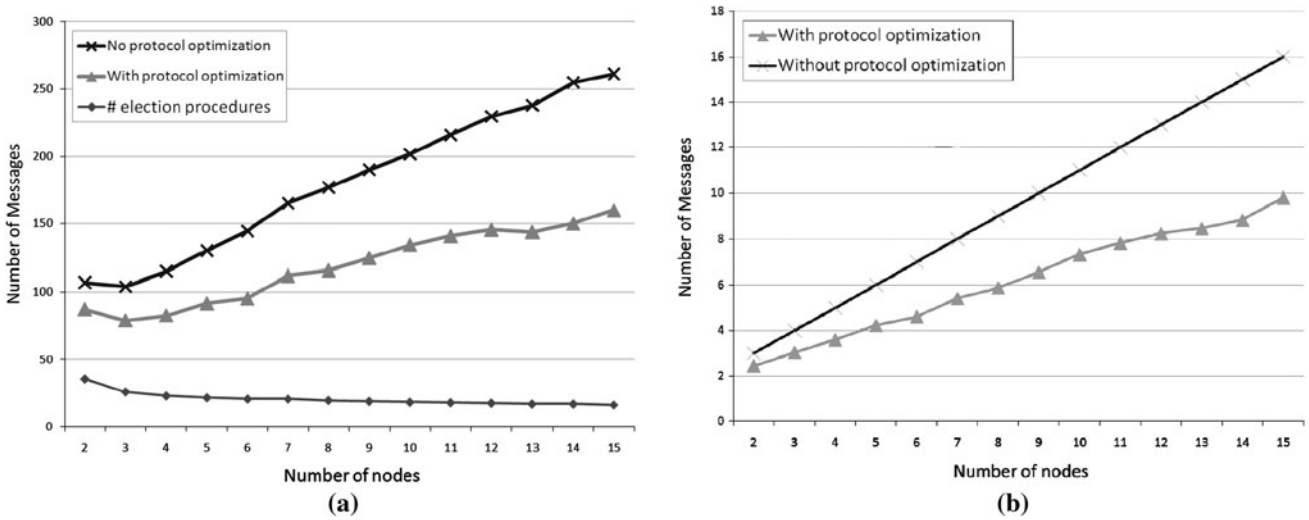


Fig. 6 Number of messages exchanged in the CH election process during a simulation time of 20 min

4.3 Low latencies

The time that a new Smart Object will have to wait from the moment it sends a request message until the Smart Object becomes part of a new network (i.e, merger latency) is short, even when several requests are queued. Figure 7a shows the merger latency with three priority levels. A SO will be classified within a priority level according to the filtering algorithms mentioned in Sect. 3.3 The cross-mark represents the average waiting time with processing cycles of 5 seconds, while the vertical lines represent the maximum and minimum values across 10 repetitions. Although the latency seems large for 2nd and 3rd priority levels, the probability that requests will have to wait several cycles is small. As an example, Fig. 7b shows the distribution of the probability that a SO classified in the 2nd priority level will have to wait several processing cycles.

The routing latencies inside a network, i.e, the time taken for a packet to navigate a route inside a network of Smart Objects (i.e, network latency), are also short. Since the packet latency is associated with its delivery ratio (i.e, the number of packets that reach their destination), Fig. 8a plots both performance measures averaged throughout 10 repetitions, with random origin and destination nodes and varying the number of nodes between 20 and 100. In order to gain an understanding of how well the Sequence Chain (SC) protocol performs, we measured the same performance indicators in the ZigBee tree routing protocol [37] within the same simulation environment. Figure 8b and c plot the results setting the `nwkMaxDepth` variable of ZigBee to "`2`" and to "`7`", respectively. While SC addressing trees are not restricted in their depth, ZigBee address trees need to fix their depth in order to calculate the addresses of their nodes. They achieve this by setting the `nwkMaxDepth` variable to the desired maximum tree depth [37]. "`2`" was chosen as the minimum value with which a tree can be formed with `20` nodes, and "`7`" was chosen as the value showing the best results in terms of tree

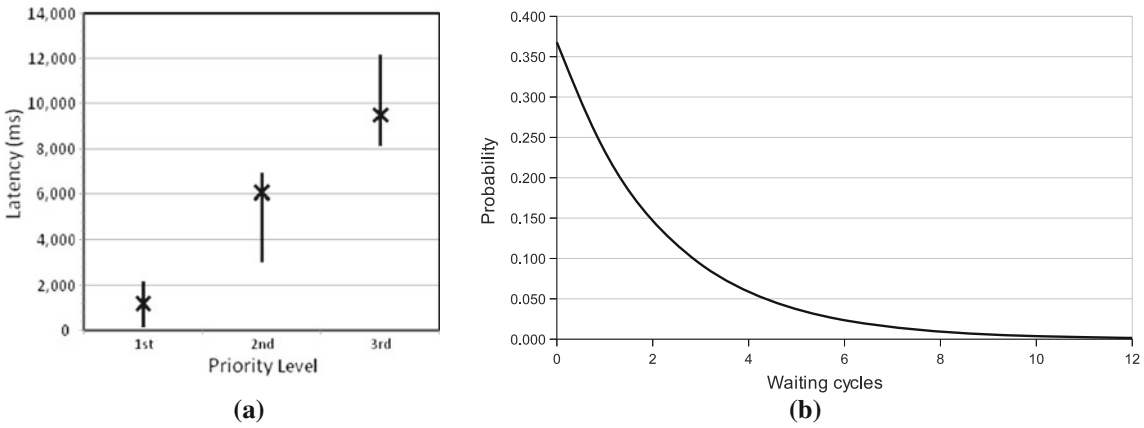


Fig. 7 Cluster membership processing latency

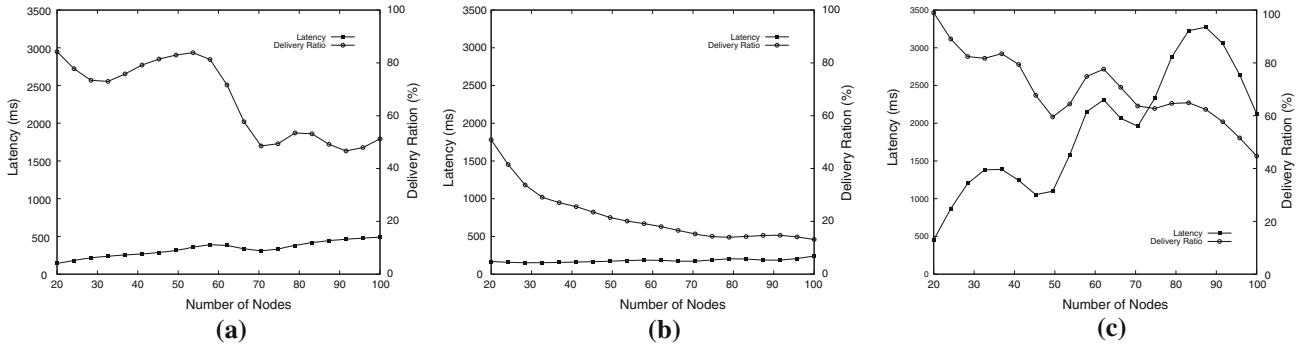


Fig. 8 Comparison of latency and delivery ratio inside SO and ZigBee networks **a** Smart Object networks, **b** ZigBee networks with `nwkMaxDepth=2` (**c**) ZigBee networks with `nwkMaxDepth=7`

connectivity (see Fig. 9a). As results show, ZigBee presents a very large trade-off between delivery ratio and latency, where delivery ratios comparable to those of SC produce large latencies, and where small latencies comparable to those of SC produce very low delivery ratios. Judging by these results, we can conclude that SO networks generate very low network latencies, outperforming the well-established routing protocol ZigBee for the same delivery ratios.

4.4 Stable topologies

The last subsection showed how ZigBee topologies can produce very disparate performance results depending on very specific deployment decisions. The value of the `nwkMaxDepth` variable is supposed to be selected according to the forecast topology for a particular deployment scenario. Since the mobility of Smart Objects makes it impossible to determine how the topology of SO networks would evolve, a stable performance, independent of deployment scenarios, becomes very important. Figure 9 shows additional indicators of the stability of both SC and ZigBee topologies, with a particular focus on how ZigBee performance is affected with the selection of the `nwkMaxDepth` variable. All the graphs in Fig. 9 show SC as horizontal lines since the measurements do not depend on the variation of any variable, and adapt to the random simulated deployment scenarios. Values shown here average the results of over 10,000 simulations, where the number of nodes was varied from 0 to 100 and the radio range of the nodes was varied from 0 to 300 m. For each combination, 100 repetitions with randomized node deployment locations were conducted.

Orphan nodes (Fig. 9a) are those nodes that could not connect to the tree during the tree formation phase either because no neighbor was found within its radio range or because the neighbors found are unable to accept more children. Fewer orphans result in better network connectivity. The *depth* of the tree (Fig. 9b) is an indicator of the tree efficiency, since increasing the number of links, a message has to traverse from its origin to the tree's root requires more time and energy. Similarly, the *number of hops* (Fig. 9c) is the number of links between random origin and destination nodes. Results in Fig. 9 highlight not only the variability of the performance results of ZigBee according to its topologies, but also the performance superiority of SC in tree connectivity, tree depth, and number of routing hops.

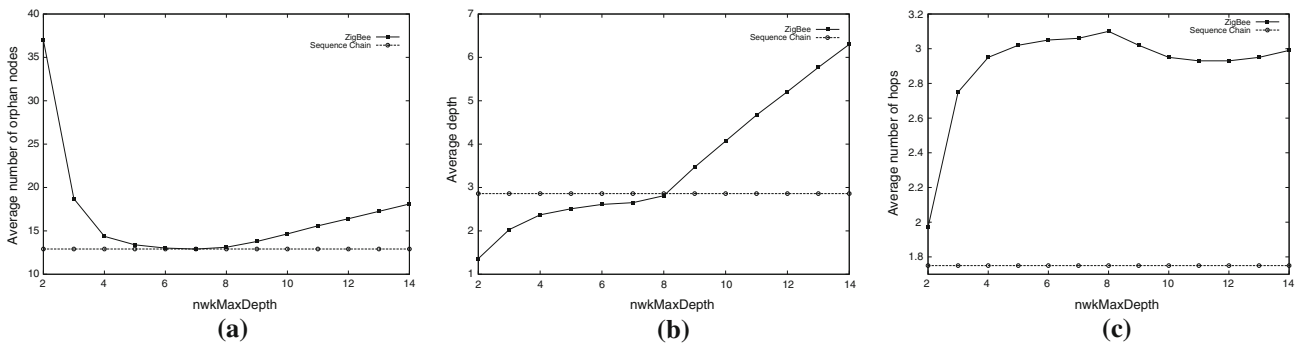


Fig. 9 Comparison of topology stability between SC and ZigBee tree networks **a** average orphan nodes, **b** average maximum tree depth, **c** average number of hops

4.5 Infrastructure flexibility and performance

The information infrastructure design was evaluated in terms of its flexibility for use as a reference model for implementation. The design was thus implemented with various popular technologies, including XML and JSON-based web services, REST and binary distributed object paradigms such as Java Remote Method Invocation (RMI). The various architecture interfaces and events were successfully converted into these technologies with no loss of functionality, and it was consequently concluded that the proposed design is flexible and adaptable for the Internet of Things concept.

The information infrastructure was also evaluated in terms of data access performance. In particular, the design of real-time push of sensor data toward the infrastructure subscribers was tested and its benefits quantified in terms of scalability and latency. Two strategies were trialed. Firstly, the sensor data were stored in the SO network structure repository along with the network structure data. Secondly, the sensor data were pushed in real-time from its reception in the capture interface to the subscribed clients. Figure 10 plots the latency of 100 sensor data events between their reception by the gateway until their delivery to the data subscriber, with the two strategies described earlier. In order to test the performance with different load conditions, several concurrent clients (i.e, data producers) were tested while maintaining the same amount of total data events. This test was performed using the implementation described in Sect. 5. Results show very clearly that whereas the latency increases linearly with the number of concurrent clients when the repository is used, it remains constant when real-time push is utilized. These results where retrofited into the design process to devise efficient strategies for delivering large amounts of sensor data to a large number of concurrent subscribers.

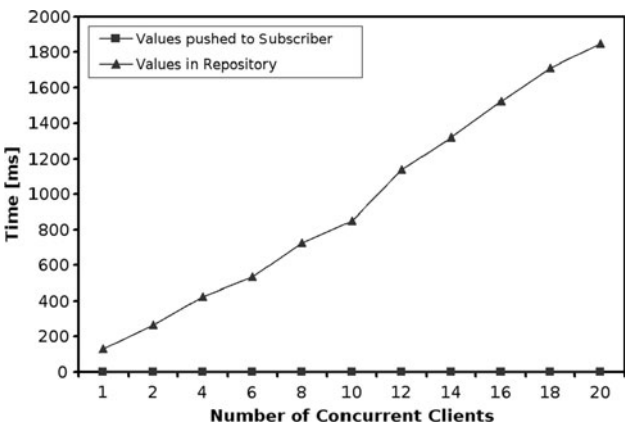


Fig. 10 Comparison of sensor data event latencies with and without using the Smart Object network structure repository

5 Architecture implementation

The described architecture features a flexible design which does not restrict the technology used for its implementation. In order to provide a prototype implementation that would allow us to test a variery of scenarios, the architecture was implemented using Wireless Sensor Networks, Web Services and relational databases (Fig. 11). One such scenario, based on a supply chain application, is presented in Sect. 6.

For empowering objects into Smart Objects presented in Sect. 3, we chose Wireless Sensor Networks as the logical solution given the current state of wireless, low-power embedded technologies. Smart Objects were implemented using the ANTS sensor network platform [40]. Each node featured an 8 bit μ-controller, a 2.4 Ghz transceiver, 128 kb of Flash memory and 4 kb of SRAM and a variety of sensors including pressure, humidity, temperature sensors and accelerometers. The implementation used the memory and processor of each node to execute the algorithms described earlier, and the radio module to communicate with other nodes and with the information infrastructure.

As shown in Fig. 11, the SO network (a) communicates with the infrastructure via a gateway node (b). In the implementation, the gateway node serializes the SO events and transmits them to a Network Translator (c), whose role is to convert those messages into HTTP client requests to a Java Servlet server that acts as the point of entry to the information infrastructure. From all the technologies evaluated for the infrastructure implementation,

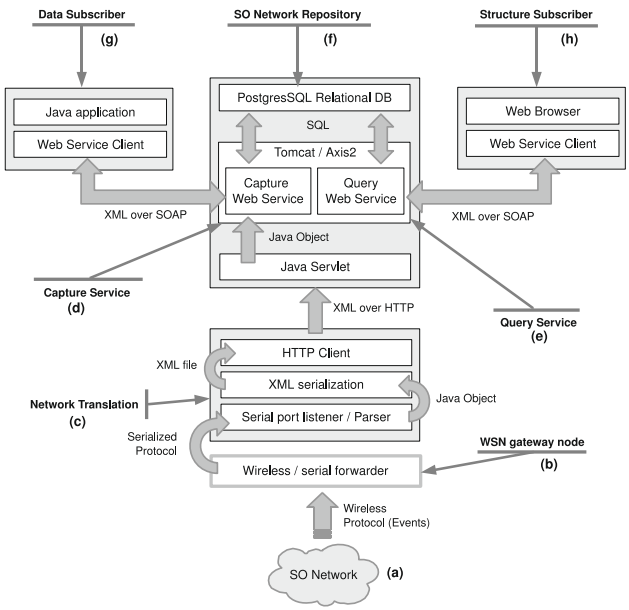


Fig. 11 Smart Object information systems architecture implementation

XML-based Web Services (WS) were chosen due to their ability to provide cross-platform and cross-language communications over a network. XML and XSD were used to encode the events and event data. WSDL was used to describe the capture (d) and query interfaces (e). SOAP and HTTP were used as messaging protocols to transfer the XML-encoded events between the different infrastructure components. A relational database was used as a repository (f) in order to store the structure of the Smart Object networks. Due to the flexible and interoperable nature of the XML-based WS, infrastructure clients could use a variety of methods for encoding requests sent to the Capture and Query Interfaces. In our implementation architecture, both Web Browser (h) and Java based clients (g) were built. A screeshot of one such client for the supply chain scenario presented in Sect. 6 is shown in Fig. 13.

6 Example scenario

While there are a large number of applications that could benefit from the proposed framework, ranging from preventing counterfeiting to defeating bio-terrorism, we consider an application in transport logistics to illustrate how users can detect, track, trace, and manage complex business problems using the Smart Object framework.

An increasing number of drivers, such as achieving greater supply efficiencies through the elimination of waste, legislative drivers such as 2001/95/EC and 178/2002 to both reduce waste and make available 'fresh' and 'safe' food for consumption by the general public [41], and investments by retailers to meet customer expectation of quality [42], have created the impetus to re-examine ways of managing supply chains. More specifically, those drivers associated with the transport of perishable goods, such as meat, poultry, fruits, and vegetables, which are sensitive to environmental conditions such as temperature, are particularly relevant for our discussion [43]. On top of these, the shift in the world economy toward a low carbon future through the reduction of carbon dioxide emissions, energy consumption, and wastage caused by the transport processes are also seen as related and significant goals [44].

Consequently, there is an imperative need for ensuring food quality across the supply chain. Achieving the objectives of food quality partly relies on physical traceability throughout the chain. As a result, the management of the supply chain, more significantly the cold chain related to the manufacture, distribution and sale of perishable, and condition-sensitive products, are seen as high priority applications.

Today, these goals are addressed to a limited extent through central planning and optimization. Large logistics networks consist of numerous destinations and vehicles. Due to the complexity and scale of transportation processes, an optimal solution for managing such a distributed and large scale supply network cannot be achieved centrally in a real-time manner to effect real change on the ground. Dynamic changes and unforeseen situations, such as traffic jams, machine failures (e.g., refrigeration units), and changes to delivery quantities pose a significant challenge to central planning and control facilities.

In this section, we argue that the integration of the Smart Object framework approach in transportation logistics to manage a supply network in a distributed manner can potentially reduce waste, handle dynamic situations autonomously, and ensure freshness of products as well as complying with legislative requirements. This is demonstrated by the following scenario, illustrated in Fig. 12. Although the following application scenario is based on sample data (instead of data from real-life RFID implementations), it is still a high impact and high priority application.

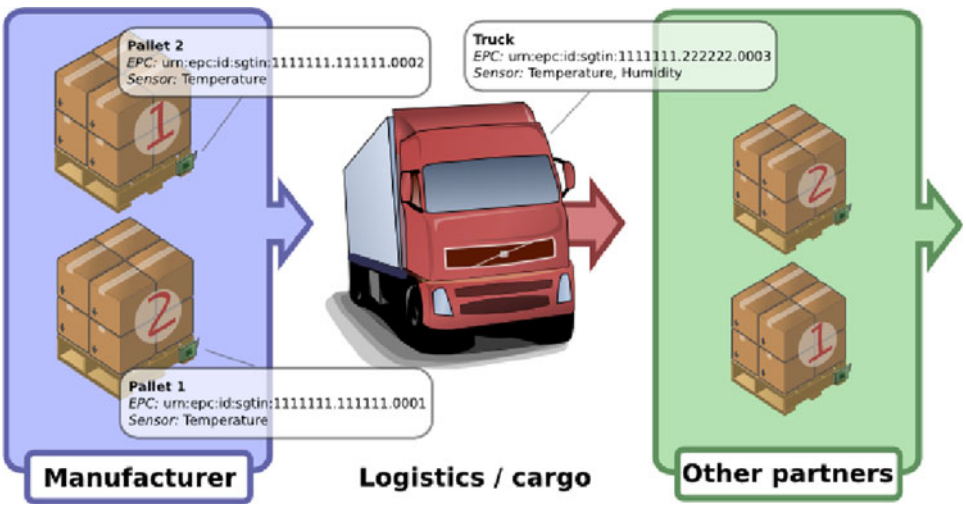


Fig. 12 Example scenario for condition monitoring of refrigerated products in the supply chain

Thousands of products such as seafood, milk, and fresh vegetables must be refrigerated upon manufacture. These products are packaged and transported on pallets. These pallets travel from the point of manufacturer to retailers through various intermediaries such as distribution and storage locations. As is often the case, these refrigerated goods are transported between locations using third party logistics (3PL) providers in refrigerated trucks or containers. In the prototype application scenario, illustrated in Fig. 12, each pallet is a Smart Object equipped with a node that can sense ambient temperature. Furthermore, the trucks employed by the logistics services provider incorporate temperature and humidity sensors. Although the prototype application illustrates pallets and a truck, the application supports more complex and multiple Smart Object networks (e.g where each box in each pallet is a Smart Object).

As indicated in Fig. 12, all the Smart Objects in the scenario such as pallets and the refrigerated truck are identified through a unique identifier, an Electronic Product Code (EPC), stored in the memory of embedded wireless sensor nodes. The sensors installed in the node of the truck measure the temperature and humidity of the climatic zone within the truck, and the node has access to Internet-based services through a mobile gateway (e.g., 3G cellular network). The enhanced infrastructure provided by the Smart Object framework is used in this application to achieve the real-time management of the product's supply network in a distributed manner, and to reduce waste, help ensuring freshness of the products as well as complying with the regulations in place.

6.1 Real-time condition track and trace

The condition of the pallets, resulting from the interpretation of sensor measurements taken by their integrated tags, can be tracked in real-time along the entire supply chain. Tracking is independent of the location of the client, the truck, and the pallets as long as the identity of the pallets are known and their network CH has access to a gateway connected to the Internet. Partners such as the manufacturers that released the product, retailers that are waiting for their shipment, or the logistic companies that are responsible for the transportation of the goods can enable tracking and tracing by simply subscribing to the capture interface with the unique identification numbers (the EPCs used in the application) of the pallets as illustrated in Fig. 13.

6.2 Dynamic service discovery

The SO networking design enables the discovery of additional sensor sources relevant to the context of a particular object and the automatic selection of the most appropriate data source from multiple data providers.

By querying the infrastructure repository with the EPC of a specific pallet on the truck, parties throughout the supply chain can retrieve the EPCs of all the Smart Objects participating in the current network. This is a dynamic discovery mechanism that allows parties to determine that the truck contains a humidity sensor as well as a temperature sensor. Where humidity is also relevant to the condition of the refrigerated products, the parties can also

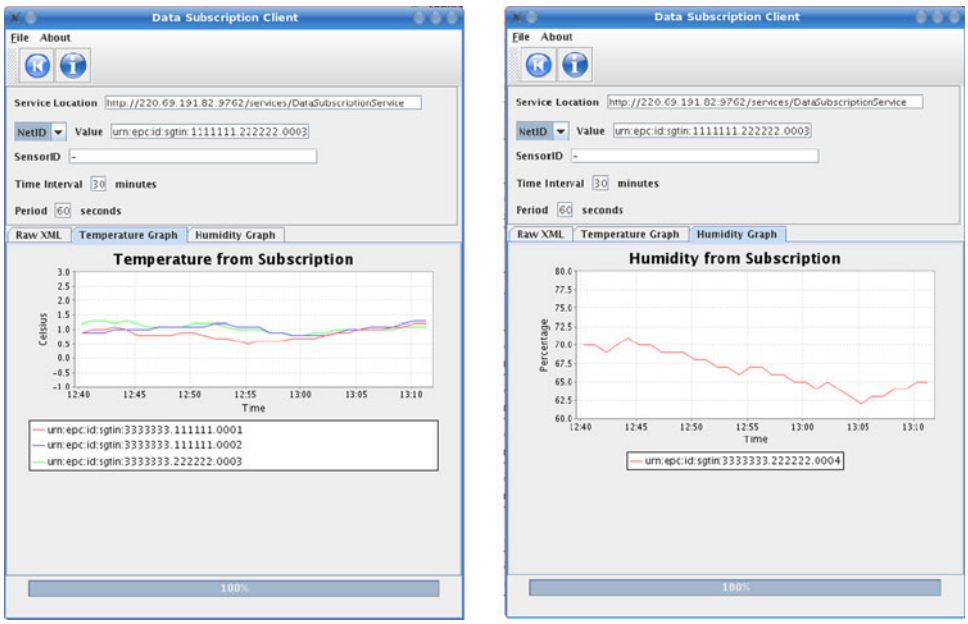


Fig. 13 Data subscription Web Service Java client software running the example scenario. The *service location* string points to the capture interface endpoint. The granularity of query can be selected using the drop-down list and *SensorID* fields. The duration of the capturing process and the polling period can also be specified. *Graphs* show the sensor data for the available transducers, on the requested Smart Object with the required granularity. Smart Objects are identified by EPCs. The raw XML files being exchanged with the server via SOAP can also be visualized for debugging purposes

subscribe to the humidity data stream (see Fig. 13). Furthermore, leveraging the EPC Network standards and infrastructure, it is possible to query on-line information services, such as EPC Information Services [45], to obtain additional information related to specific Smart Objects. For example, a trading partner in the prototype application can determine the accuracy or suitability of the temperature sensors of both the truck and the pallet using sensor hardware metadata made available though the EPC Information Services of the manufacturer or the logistics company to select the more accurate or suitable sensor data stream. In situations where each item on a pallet carries a node, pallet networks would be formed prior to the truck network and could thus build an effective hierarchical network structure aiding the discovery of meaningful contextual condition information.

6.3 Level of service provisions

The SO architecture design permits assignment of priorities and restrictions to control SO interactions. As a result, logistic companies can ensure that only products from selected suppliers, for instance gold service members, make use of the resources of the company (e.g only the node of pallet 1 in Fig. 12 stores the appropriate credentials and can therefore use the sensor and gateway of the truck). As a result, the framework not only supports advanced application scenarios but also provides the opportunities to develop value added services.

Figure 13 shows a screenshot of the Data Subscription client software, monitoring all the sensor sources from the SO network formed inside the truck from the example scenario, for a period of 30 min, in 60 s intervals.

Figure 14 shows two examples of messages exchanged between the various architectural components of the SO framework for the scenario presented in this section. Figure 14a shows the body of a SOAP message returned by the Query interface to an `IDquery` request for the SoID `urn:epc:id:stgin:1111111.1111111.0001`. The SOAP header has been removed for clarity. The response includes a list of the nodes that the SO contains, together with sensors and their characteristics, as well as the identifier of the network where the Smart Object is located. Figure 14b shows a single `DataEvent` as sent by the same Smart Object toward the Capture interface. The example illustrates how the sensor data are encoded within the `DataEvent` messages. Data events like the one shown in Fig. 14b are parsed by the client software pictured in Fig. 13 in order to plot the graphs displayed in the screenshot. The *Events* and *Sensor* XML schema describe the encoding of the events and the meaning of the sensor data, respectively.

```
<soapenv:Body>
  <ns:IDqueryResponse xmlns:ns="http://www.autoidlabs.org.uk/xsd/SO/QueryService">
  <ns:return> <xml version="1.0"encoding="UTF−8"?>
   <QueryResponse QueryType="IDQuery" IDType="SoID">
    <SoValues>
     <SoID>urn:epc:id:stgin:1111111.1111111.0001</SoID>
     <NetID>urn:epc:id:stgin:1111111.2222222.0003</NetID>
     <Node>
       <NodeID>urn:epc:id:stgin:1111111.1111111.0001</NodeID>
        <Sensor>
         <SensorID>urn:epc:id:stgin:3333333.1111111.0001</SensorID>
         <Type>Temperature</Type>
         <Output>Celsius</Output>
        </Sensor>
      </Node>
    </SoValues>
   </QueryResponse>
  </ns:return >
 </ns:IDqueryResponse>
</soapenv:Body>
```

(a)

```
<?xml version="1.0" encoding="UTF-8"?>
<Events:DataEvent
          xmlns:Events="http://www.autoidlabs.org.uk/xsd/SO/Events"
          xmlns:epcglobal="urn:epcglobal:xsd:1"
          xmlns:sens="http://www.autoidlabs.org.uk/xsd/SO/Sensors"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://www.autoidlabs.org.uk/xsd/SO/Events Events.xsd">
<Events:SoValues>
  <Events:SoID>urn:epc:id:stgin:1111111.1111111.0001</Events:SoID>
  <Events:NetID>urn:epc:id:stgin:1111111.2222222.0003</Events:NetID>
  <Events:Node>
    <Events:NodeID>urn:epc:id:stgin:1111111.1111111.0001 </Events:NodeID>
    <Events:SensorDataEvent>
        <Events:SensorID>urn:epc:id:stgin:3333333.1111111.0001 </Events:SensorID>
        <Events:Value>1.25</Events:Value>
    </Events:SensorDataEvent>
  </Events:Node>
</Events:SoValues>
</Events:DataEvent>
```

(b)

Fig. 14 Examples of messages exchanged between the architectural components **a** example of response by the Query interface, **b** example of a `DataEvent`

7 Challenges

7.1 Economic challenges

It is only in recent years that simple passive RFID tags have become available at sufficiently affordable prices (around 7*c* per tag in large volumes) that many industry sectors are considering widespread adoption of RFID. Sensor-enabled active tags are likely to cost considerably more than simple passive RFID tags because of the additional cost of the sensor, memory capacity, and batteries. For this reason, many of the early trials apply them to reusable assets (e.g food trays, pallets) rather than individual items in order to amortize the cost over a much longer period of service.

7.2 Security and trust issues

RFID usually requires the assignment of unique identifiers for each object. This results in fine-grained visibility and tracking information, but means that an individual object is no longer ‘anonymous’ as simply another instance of a particular product type. At the same time, complete sensor information for an object is realistically likely to be fragmented and distributed across its lifecycle, with each organization holding only the sensor information that was collected while the object was within their custody. In

order to gather complete sensor information from across multiple organizations, it may be necessary to provide authentication and business relationship credentials in order to justify the request, both for querying data from the Smart Objects and when they attempt to connect to the infrastructure gateways.

7.3 Scalability challenges

A further challenge to the collection of sensor data is the capacity available for storage of historical data. Consideration should be given to mechanisms that allow thresholds and also more complex exception criteria to be defined in a standardized manner. This may ultimately lead to a different paradigm for information sharing, in which alerts and details of exceptions are returned by queries, rather than large volumes of low-level location and sensor data. This strategy may allow compression of historical data for long-term storage and could also address some of the data sharing concerns since organizations may prefer to provide object information only when exceptions and alerts arise.

8 Conclusion

We have presented an architecture that uses Smart Objects to integrate technologies such as automatic identification, sensor systems, embedded processing, context-aware ad-hoc networking, and Internet-based services, which are identified as central to the realization of the IoT concept. Practical experience gained with the evaluation and implementation of the architecture demonstrates that it is both feasible and flexible to adapt to a variety of applications and off-the-shelf technologies. Key observations showed that the proposed architecture has good performance in terms of network lifetime, overhead, and scalability, as well as producing low latencies in the various processes of the network operation. We can thus conclude that not only does the proposed architecture succeed in incorporating a number of key technologies but it does so from a practical standpoint. Finally, a number of identified challenges suggest that the adoption of the IoT, in general, and our architecture, in particular, is not only limited by developments in technology but also by social and trust issues.

References

1. Uckelmann D, Harrison M, Michahelles F (2011) Architecting the Internet of Things. Springer, Berlin
2. Weisser M (1999) The computer for the 21st century. ACM SIGMOBILE Mobile Comput Commun Rev 3(3):3–11
3. EPCglobal Inc (2009) The EPCglobal architecture framework, v 1.3, standard specification, March 2009
4. ISO/IEC 18000-6 REV1 (2009) Information technology radio frequency identification for item management Part 6: parameters for air interface communications at 860 MHz to 960 MHz, August 2009
5. Song EY, Lee K (2008) Understanding IEEE 1451 networked smarttransducer interface standard. IEEE Instrumentation & Measurement Magazine, April 2008
6. Open Geospatial Consortium Inc (2007) OGC sensor web enablement: overview and high level architecture. OGC White Paper, OGC 07-165, v3
7. Kiritsis D et al (2008) Product lifecycle management and information tracking using smart embedded systems, final report, PROMISE FP6-507100, EU, June 2008
8. Carrez F et al. (2009) Deliverable D3.2, reference architecture. SENSEI FP7 215923, EU, January 2009
9. Ranasinghe DC, Leong KS, Ng ML, Engels DW, Cole PH (2005) A distributed architecture for a ubiquitous RFID sensing network. In: Proceedings of the ISSNIP'05, Dec 5–8
10. Finkenzeller K (2003) RFID handbook: fundamentals and applications in contactless smart cards and identification. Wiley, London
11. Beigl M, Krohn A, Zimmer T, Decker C (2004) Typical sensors needed in ubiquitous and pervasive computing. In: Proceesdings of INSS
12. Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E (2001) Wireless sensor networks: a survey. Computer networks. Elsevier, Amsterdam
13. Sample AP, Yeager DJ, Powledge PS, Smith JS (2007) Design of a passively-powered, programmable sensing platform for UHF RFID systems. In: Proceedings of IEEE international conference on RFID
14. Mitsugi M (2006) Multipurpose sensor RFID tag. In: Proceedings of the APMC 2006 workshop on emerging technologies and applications of RFID, Yokohama, Japan, 12–15 December, IEICE, Japan, pp 143–148
15. EU FP6 BRIDGE Project BRIDGE: building radio frequency identification solutions for the global environment, 2006–2009. http://www.bridge-project.eu
16. Sung J, Sanchez Lopez T, Kim D (2007) The EPC sensor network for RFID and WSN integration infrastructure. PerComW 2007, IEEE Computer Society, pp 618–621
17. ISO/IEC CD 24753.2 standard draft (2009) Information technology Radio frequency identification (RFID) for item management application protocol: encoding and processing rules for sensors and batteries, March 2009
18. IEEE P1451.7/D.07 (2009) Draft standard for smart transducer interface for sensors and actuators—transducers to radio frequency identification (RFID) systems communication protocols and transducer electronic data sheet formats. January 2009
19. Aberer K, Hauswirth M, Salehi A (2006) Global sensor networks. Technical report LSIR-REPORT-2006-001
20. The CoBIs Consortium (2007) Deliverable D104, final project report. CoBIs FP6-004270, EU, March 2007
21. Cho J, Shim Y, Kwon T, Choi Y (2007) SARIF: a novel framework for integrating wireless sensors and RFID networks. IEEE Wire Commun 14:50–56
22. Wong CY, McFarlane D, Zaharudin AA, Agarwal V (2002) The intelligent product driven supply chain.In: Proceedings of the IEEE international conference on systems, man and cybernetics, vol 4, p 6
23. Wycisk C, McKelvey BH (2008) ülsmann, M smart parts supply networks as complex adaptive systems: analysis and implications. Int J Phys Distrib Logist Manage 38(2):108-125
24. Montenegro G et al (2007) IPv6 over low-power wireless personal area networks (6LoWPANs): overview, assumptions,

problem statement, and goals. Network working group, IETF, August 2007
25. The IPSO Alliance. http://www.ipso-alliance.org. Accessed 25 October 2010
26. IETF ROLL Working Group. http://tools.ietf.org/wg/roll. Accessed 25 October 2010
27. Shelby Z, Bormann C (2009) 6LowPAN: the wireless embedded Internet. Wiley, London
28. EPCglobal Inc (2010) *EPC*™ Tag data standards version, v 1.5, standard specification, August 2010.
29. Trifa V, Guinard D (2009) Towards the web of things, Whitepaper v 1.0
30. Fielding RT (2000) Architectural styles and the design of network-based software architectures. Ph.D disseration, University of California, Irvine
31. Shelby Z et al. (2009) CoAP feature analysis. 6lowpan Internet draft, IETF
32. O'Connor MC (2006) Cold-chain project reveals temperature inconsistencies. RFID J. http://www.rfidjournal.com/article/view/2860/. Accessed 25 October 2010
33. Estrada-Flores S, Tanner D (2005) Temperature variability and food spoilage during delivery of online retail products. Acta Hortic (ISHS) 674:6369
34. Blake D (2009) Effects of cargo loading and active containers on aircraft cargo compartment smoke detection times. Technical report DOT/FAA/AR-09/52, U.S. Department of Transportation, Federal Aviation Administration
35. Sánchez López T, Ranasinghe DC, Patkai B, McFarlane D (2009) Taxonomy, technology and applications of smart objects. J Inform Syst Front 1–20. doi:10.1007/s10796-009-9218-4
36. Sanchez Lopez T, Huerta Canepa G (2009) Distributed and dynamic addressing mechanism for wireless sensor networks. Taylor & Francis (accepted to the Int J Distrib Sensor Netw)
37. *ZigBee Specification* (2008) Document number 053474r17, ZigBee Alliance, January 17 2008
38. Heinzelman WR, Chandrakasan A, Balakrishnan H (2000) Energy-efficient communication protocol for wireless microsensor networks. HICSS'00;January 47; Maui, USA. USA: IEEE Computer Society; 2000. p 10
39. Sanchez Lopez T, Kim D, Huerta Canepa G, Koumadi K (2008) Integrating wireless sensors and RFID tags into energy-efficient and dynamic context networks. Comput J. doi:10.1093/comjnl/bxn036
40. Kim D, Sanchez Lopez T, Yoo S, Sung J (2005) ANTS: an evolvable network of tiny sensors, embedded and ubiquitous computing, LNCS. Springer, 3824/2005, pp 142–151
41. Trienekens JH, Beulens AJM (2001) The implications of EU food safety legislation and consumer demands on supply chain information systems. In: 11th Annual world food and agribusiness forum, Sydney.
42. Deasy DJ (2002) Food safety and assurance: the role of information technology. Int J Dairy Technol 55(1):35–67
43. *The Chill-On project* (2005) EU DG Research OOD-CT-2005-016333. http://www.chill-on.com/
44. European Commission (2004) European CO_2 capture and storage projects, project synopses, sixth framework programme, European communities
45. EPCglobal Inc (2007) *EPC*™ Information services, v 1.0.1, standard specification, September 2007