Ad Hoc & Sensor Wireless Networks, Vol. 1, pp. 000-000
Reprints available directly from the publisher
Photocopying permitted by license only

Towards Commercial Wireless Sensor Networks: Business and Technology Architecture

VASSILEIOS TSETSOS, GEORGE ALYFANTIS*, TILEMAHOS HASIOTIS, ODYSSEAS SEKKAS, AND STATHES HADJIEFTHYMIADES

Pervasive Computing Research Group of Communication Networks Laboratory, University of Athens, Dept. of Informatics and Telecommunications, Panepistimiopolis, Ilissia, GR-15784, Athens, Greece
{b.tsetsos, g.alyfantis, thasiot, o.sekkas, shadj}@di.uoa.gr

(Received December 1, 2004; In final form February 21, 2005)

In this paper we study the opportunities of commercial exploitation of applications based on sensor networks. Such applications are quite famous nowadays in many different domains of everyday life (e.g., health monitoring, traffic monitoring). We introduce a new business model for sensor based services. Such business model involves entities like the mobile network operator and capitalizes on existing standards for application service providers (e.g., the OSA/Parlay standard). We also present the required technical framework that would facilitate the introduction and rapid adoption of the proposed business schemes.

Keywords: wireless sensor networks, middleware design, business models, pervasive computing, commercial applications, Parlay/OSA, OSGi.

1. INTRODUCTION

In the last few years, we have noticed a significant evolution of the mobile telecommunication networks and services. Nowadays, 2.5G mobile

Corresponding author: E-mail: g.alyfantis@di.uoa.gr

networks are the standard in most developed countries, whereas the penetration of 3G networks is steadily increasing (although slower than expected). This evolution was followed by an evolution of the provided services. New and innovative data services have shown up, which enhance the application environment of the end users. Probably the most promising of them are Location Based Services (LBS). LBS adapt the content provided to the users according to their physical location.

During these years, we have also witnessed an increasing interest of the research community in the wireless sensor networks (WSN) [3]. Such networks are the descendants of Micro Electro-Mechanical Systems (MEMS) and interconnect sensor nodes in order to provide a flexible topology for the dissemination of the sensed contextual information (e.g., temperature, motion, vibration). Since their invention, the WSNs have been mainly used in specific application domains such as habitat monitoring [4] (animal tracking, microclimate studies) and military surveillance applications.

By observing the above evolutions, one can predict that the ubiquitous environment of the future will comprise publicly and privately deployed sensor networks, which will enable the deployment of "smart services", accessible through advanced infrastructures (e.g., the capability-rich 3G mobile networks or open services gateways). We strongly believe that this is a one-way scenario, judging from:

- the current user demand for context-aware services [10],
- the mobile market economics (market players seek for innovation through new services and killer applications) and,
- the vision for pervasive environments, which are highly interactive and responsive [5] (e.g., smart spaces, intelligent classrooms).

In the following sections we discuss how such a Sensor-Based Services (SBS) model, which combines (mobile) telecommunication technologies and WSNs, can be realized. In Section 2, we study the proposed Sensor Based Services (SBS) provisioning system. In Section 3, we study technical issues related to the proposed network architecture. In Section 4, we study the economic aspects (revenue flows) of this new service model. Our current technology research towards the realization of this model is described in Section 5. The major problems that still prevent its wide and successful adoption are described in Section 6. The paper concludes with related work and future research directions.

2. SENSOR-BASED SERVICES

Two typical scenarios that illustrate our view on the integration of sensor infrastructures (both private and public) in public mobile networks are the following:

Scenario A - *Agy, a 3G mobile user, wants to monitor the healthcare (physical and biological state) of her grandmother, Sophie. For that purpose she orders her Mobile Network Operator (MNO) to install a full health-sensing system in Sophie's environment (house and body). Such system monitors her heart-pulse rate, body temperature, periodically takes photographs from the rooms and detects falls. Thus, Agy can periodically check Sophie's state and ensure that she faces no problem. In case of an emergency situation she will be notified through the Multimedia Messaging Service (MMS) push functionality provided by the MNO.*

Scenario B - *The vehicles of a country carry sensors according to the regulatory safety mandates. These sensors are powered by the main battery and monitor some operational parameters including speed (e.g., digital tachographs) and engine status. Some of these parameters are used only by the internal vehicle automation system (e.g., engine status) while others (e.g., speed) are also transmitted to adjacent nodes/cars through an RF module. Thus, all the cars in a specific region form an ad hoc wireless sensor network. Local authorities or the central government have installed gateway nodes in selected fixed positions (e.g., outside schools) that aggregate the sensed data and transmit them to a central system, along with the vehicle IDs, through the public mobile networks (alternatively, the police vehicles could carry such gateways). In this way, dangerous drivers could be timely detected and penalized accordingly.*

The above scenarios identify a major taxonomy of the possible WSN deployments: sensor networks can be either *private* or *public*. A private sensor network is deployed after a specific customer order to address her specific needs and, thus, it *can* and *should* be used only by this customer. In this case, the mobile network operator simply provides a secure bearer for the communication between the user endpoints. A public sensor network is deployed after a third-party's initiative and can be either of public unconstrained use or can be used by specific authorities for the public welfare. An example of a public WSN is the ad hoc sensor network of Scenario B, which monitors the traffic status so as to cater for public road

safety. Other examples of both private and public deployments are sensor-enabled rescue teams, requiring ad hoc cooperation of their members, homeland security and medical applications.

Scenario A bears some resemblance to the Location-Based Services (LBS) provisioning scenarios of the contemporary mobile networks. Firstly, they both deal with context-aware applications. In the case of LBS, the context is the user location, while for SBS, the context consists of the sensed environmental parameters (e.g., temperature). Secondly, the user registers for a health monitoring service similarly to the way she would register for a "traditional" data service, and receives the corresponding information either through a "pull" or a "push" model. In addition, the bearer of the service data in Scenario A is MMS, a widely used service today. However, significant differences do exist. For instance, the aforementioned WSN is a network deployed exclusively for the users directly involved in the scenario (i.e., the grandmother and its relatives) and, thus, constitutes a private infrastructure. On the other hand, the positioning infrastructure of the mobile networks is at public disposal (obviously this assumption holds only in case of a network-based positioning method, since GPS-based methods rely on the user terminal [12]). Another difference is that the contextual information (sensor data) does not have the role of the content differentiator but it is the content itself, as opposed to location information that is usually used as a content selection criterion.

Although the WSN-related scenarios seem more innovative and harder to realize than the LBS scenarios, this is not exactly the case. Firstly, the contextual information provided by a WSN is more accurate than the estimation of the user's location by a LBS system. This is mainly due to the imprecise positioning methods available today. Indeed, the lack of location accuracy is one of the main barriers that hinders the usability and penetration of the current location-aware services. Another promising characteristic of SBS is that, in general, they address more vital user needs than LBS. For example, environmental monitoring, healthcare monitoring, home security or remote industrial control, are regarded as more important and economically cost effective than a friend tracking service or a "find nearest restaurant" service. In other words, the added value for the user is much higher in the case of SBS.

The selection of a wireless sensor network instead of a (wired) static sensing infrastructure is a key decision in the service provisioning process. The WSN is highly flexible as it can be attached to mobile objects (e.g., wearable temperature sensors in Scenario A, car-mounted sensors in

Scenario B) and can be easily adapted to the considered problem. This adaptation is achieved by means of topological change, ad hoc network interoperation, and failure recovery (self-healing). This flexibility can be further translated into higher-level flexibility in the form of service differentiation, quality of service, ease of management, etc. One possible question is the involvement of a mobile operator in the SBS provisioning. Firstly, we believe that the ubiquitous connectivity and the universal access provided by modern and forthcoming mobile networks, along with the high usability of the modern and next generation handsets, are unique features of flexibility that cannot be provided to the end users in other ways. Moreover, as the costs of use for these networks are constantly decreasing and the available bandwidth increases, they can be considered comparable to many of the already deployed wired networks.

3. NETWORK ARCHITECTURE OF THE SBS MODEL

The foundational technical similarities between the LBS and the SBS systems indicate that only minimal changes should be performed in an already deployed service provisioning network that aspires to provide truly context-aware services. Figure 1 depicts the general architecture of a SBS solution based on a 2.5G or 3G network.

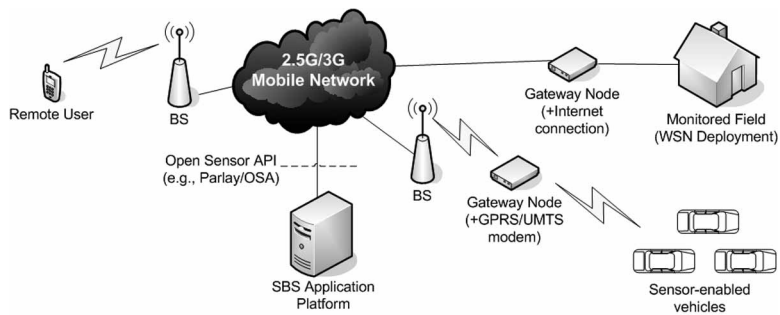

FIGURE 1
Enabling Sensor-Based Services in mobile networks.

The main modifications to the traditional network architecture will be:

A. The deployment of sensor networks in the monitored fields.

A typical modern sensor network consists of sensor nodes and a gateway node. The gateway node has more processing, energy and communication capabilities than the other nodes and is the connecting link between the sensor network and a WAN (i.e., mobile network or Internet). Hence, this node should be capable of communicating with the mobile network infrastructure, either directly (by incorporating a GSM/GPRS modem) or indirectly (Internet modem and Gateway GPRS Support Node - GGSN).

B. An application platform for the lifecycle management of the services and the handling of the remote sensor networks.

This platform is analogous to the LBS provisioning platforms [11], [14] and is responsible for the creation, deployment, and management of the SBS and the WSN. Furthermore, this platform, as a central component of the overall architecture, can handle all the relevant charging and payment issues. Such platform should be open and support the widest possible spectrum of underlying sensor network technologies.

C. An open Sensor API for the communication between the WSN and the platform.

The accumulated experience from the wireless networking applications dictates adherence to open public APIs for the interaction between the applications and the network elements. Since the WSN can be regarded as a network element, the support of de facto standards for the WSN handling (i.e., sensor data retrieval) by the platform seems both crucial and feasible. Such an API could be the Parlay/OSA (Open Services Access) [1]. The only major extension that would be required is the addition of a Sensor SCS (Service Capability Server) to the Parlay/OSA specification. Surely, the ongoing WSN research activity and the diversity in the WSN implementations introduce problems on such SCS standardization, but we can expect that the specific domain will be more clear and stable in a few years.

The aforementioned enhancements to the network are not so extensive and have limited impact to the already deployed telecommunication infrastructures, because they do not make any unrealistic assumptions on the capabilities of the existing core network and do not imply any alterations to the existing mobile terminal equipment. Thus, the value added services based on such a solution, could experience a fast market penetration with minimal infrastructure investment. However, some other technical issues

emerge, which could potentially prevent their introduction. Among them are standardization, security and privacy, which will be discussed in more detail in Section 6.

4. BUSINESS MODELING ISSUES

4.1 New business roles

The network architecture in Figure 1 implies some changes in the value chain of this new service model. We propose that the greater part of the LBS value chain [15], [16] remains unchanged, but two new business roles should be introduced: the WSN Provider (WSNP) and the SBS Operator (SBSO).

A WSNP possesses the technical expertise and capabilities for the physical deployment of the WSN. A WSNP may be a manufacturer of WSN hardware (and software) or may be a technical company, which has the required know-how for the deployment and operation of WSNs. Hence, the WSNP will typically be agnostic of the final applications. Among the responsibilities of the WSNP, apart from providing the necessary sensor and gateway equipment, is the provision of a "WSN driver" that conforms to the guidelines of the open Sensor API (see Figure 1). Such organizations can deploy WSNs either for public or private use.

The second introduced role (SBSO) owns the SBS Application Platform and is responsible for the deployment of the Sensor-Based Services. For this purpose, it indirectly uses the "WSN driver" provided by the WSNP (through the Sensor API). The Service Providers, in turn, design and implement the context-aware services on this platform. In general, the role of the SBSO is similar to that of the LBS Operator that hosts the LBS provisioning platform (SBSO can be regarded as an Application Service Provider, ASP). SBSO provides a service creation environment and all the software facilities necessary for the management of the services' lifecycle. It may also establish contracts with the Mobile Network Operator (MNO) for the use of other network features, such as location servers. Finally, SBSO generates Charging Data Records (CDR) that can be used by the MNO for the final user charging and the accounting between the involved partners. There can be many different SBSOs operating through a single mobile network, just as there are many web hosting companies in the Internet. This ensures that the SBS market will be open and will boost the desired competitiveness.

4.2 A SBS business model

The introduction of these new players affects the revenue flows of the traditional service provisioning "ecosystems" as shown in Figure 2. With the term "traditional" we refer to the common telecommunications business models: a "strong" MNO, which outsources the creation of value added services to third-party Service Providers (SPs). In these business models there was no player similar to WSNP and there were no private deployments.

By tracing the revenue flows of the business model in Figure 2 we see that the SBS subscriber pays *directly* only the Mobile Network Operator (MNO) (flow 1). This is very convenient for the user, who does not have to deal with many different billing/charging parties. At the same time, the role of the MNO is enhanced, as it can become a central payment handler. The MNO shares the user revenue with the SP(s) (flow 2), which in turn pays the SBSO for the service hosting in the SBS platform (flow 5). The SP also pays the WSNP for the network deployment and other administrative activities (flow 3). Finally, the MNO charges directly the SBSO (flow 4) for the use of its communication facilities (e.g., sensor SCS, location servers) in the deployed services.

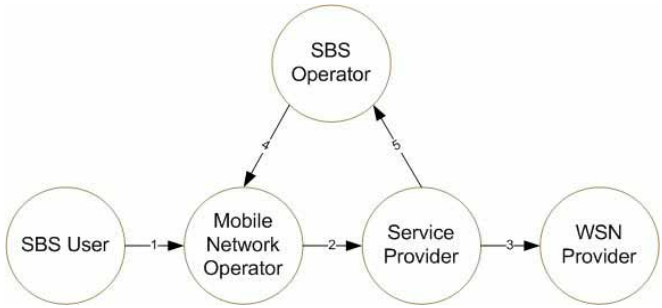

FIGURE 2
The revenue flows of the SBS model.

From the above model description we can observe that all the players incur an added value (AV) from the SBS provisioning. For the SPs, this AV comes from their closer relationship with the end-user, as they are *typically*

(i.e., legally) responsible for the WSN deployment as well as the proper service operation (WSNPs have the actual responsibility for the WSN deployment and good performance). A SP may have contracts with a large number of WSNPs, specialized in different application domains (e.g., security, healthcare, etc.). In general, these WSNPs can be considered as "employees" of the SP, which in turn can be considered as an "employee"-partner of the MNO. The AV for the WSNPs is evident, as they only receive income for their services/products. The MNO also strengthens its position in the value chain as it receives revenue from the additional provided services it provides to the subscribers and the possible payment services it provides to the other "players". Moreover, history teaches us that models that undermine the MNO's position in the value chain fail over time with increased probability. The mobile network is the central infrastructure in a SBS provisioning framework and MNOs will definitely rely on that in order to draw more profits and enhace their position in the market.

5. THE SOFTWARE ARCHITECTURE OF THE SBS PROVISIONING FRAMEWORK

One major drawback of the current sensor networks is the lack of standards for both the lower network layers (i.e., physical, MAC, etc.) and the higher application layers. The heterogeneity of the WSN technologies prevents their seamless use by ubiquitous services. We believe that, until the proper standards are established, a mediation layer (middleware) providing unified handling of these networks should bridge the gap.

In the case of telecommunication services, a good candidate for this mediation layer would be a combination of Parlay/OSA and OSGi (Open Services Gateway Initiative) [2]. Parlay/OSA is a well-established industry standard, which, currently, is in its early adoption stage. The APIs specified by Parlay/OSA expose (almost) all the network functionality (user interaction, mobility management, user terminal capabilities, location, etc.) to application developers without compromising the overall network security and exposing the technical peculiarities of the network to service providers. OSGi, on the other hand, was established in 1999 to define open specifications for the delivery and provisioning of multiple services over wide area networks to local networks and devices in homes, vehicles and other environments. The OSGi specifications try to standardize the secure and reliable service delivery and provisioning for remote life cycle

management of services, as well as for bridging between different networking standards. Until now, there have been several efforts in adopting OSGi as an open and standard platform for telematics services (see [13] for more references).

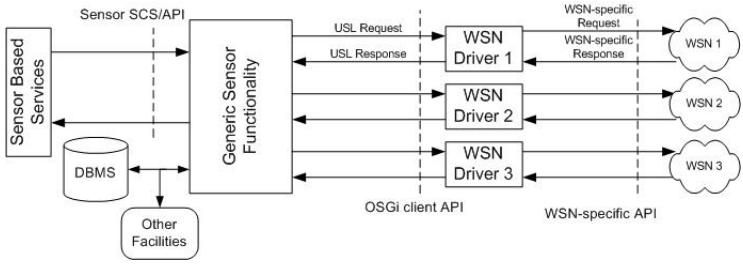

FIGURE 3
The distributed architecture of the overall SBS system.

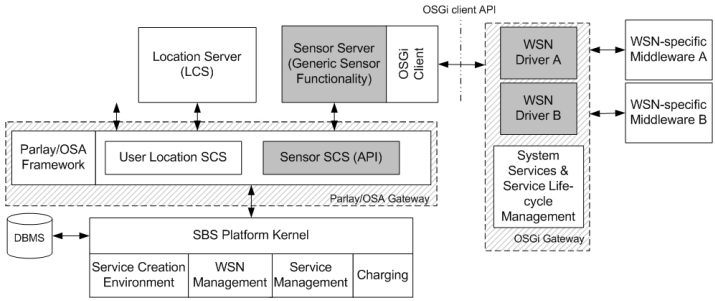

FIGURE 4
The proposed software architecture for SBS provisioning.

The general architecture of a SBS solution based on these specifications is depicted in Figure 3. Figure 3 shows how the various components of the architecture are distributed logically (their physical distribution can be

inferred in conjunction with Figure 1). The main advantages of this solution are the independence of the application from the underlying WSN technology, the simple integration of sensor information in real-world applications (regardless of the application domain) and the adherence to standard specifications. Specifically, the Parlay/OSA interfaces, appropriately extended with a Sensor Service Capability Server (SCS)/API, can provide developers with a common way for the interaction between the service logic and the actual sensor network. The mentioned Sensor SCS will also exploit other features of the Parlay/OSA Framework (e.g., charging, QoS, discovery). Furthermore, the OSGi Service Platform conceals the heterogeneity of the WSN technologies and enables their unified handling from the Parlay/OSA SCS through an OSGi Client API. Such API is not yet standardized but the MNOs have the potential to expedite the standardization process. In our opinion, such middleware brings the R&D community one step closer to the "Ambient Intelligence" [8], which envisages ubiquitous context-aware services. We are currently working on the implementation of such a middleware [30] and, especially, on the components that appear in gray in Figure 3. Figure 4 shows in more detail the proposed middleware (integration) architecture, focusing only on these components, which are described in more detail in the following tree subsections.

5.1 A developer-friendly Application Programming Interface (API)

Such API facilitates the development of context-aware services. This API is actually the Sensor SCS interface that was mentioned earlier and supports both event-driven (asynchronous) and query-based (synchronous) interaction with the underlying WSNs. It also enables developers to store the sensor data in a Database Management System (DBMS) and perform off-line and/or statistical post-processing on such information. The API takes into account the constraints of the WSNs, such as the need for composite queries that reduce the transmitted data (and, thus, energy consumption), and promotes their use. The API, which is an interface to the Generic Sensor Functionality (GSF), is object-oriented and it is built on top of a set of location modeling classes in order to facilitate the programming of context-aware services. For example, if we want to retrieve the (spatial) average temperature and the light intensity in office 'I9', we would create an object representing the region of the office and would invoke the method:

Office_I9.getSensorValue({AVG,NULL},{TEMPERATURE,LIGHT})

where NULL means that we do not want the light intensity measurements to be aggregated (e.g., minimum, average, etc). The AVG aggregate in this example denotes a spatial average. In case of a temporal average we would also need older data stored in the DBMS.

In fact, we designed two different aspects of the APIs, which serve different purposes: one that is based on the concepts of location and device of interest (high-level API), and one that is based on the concept of queries (low-level API). Application developers are capable of using either of them, according to their needs. Some indicative methods of the APIs are presented in Table 1. The first aspect of the API is more developer-friendly and intuitive, while the second is a lower-level, yet more flexible, interface. Its flexibility lies in the fact that users can construct more complex requests, decoupled from the location and device concepts that can, sometimes, prove a restrictive factor. Notice that the location-centric API incorporates certain location management methods, but these do not fall within the scope of the present paper.

5.2 The Generic Sensor Functionality (GSF)

GSF handles the developer's requests towards the WSN. This functionality maps the Parlay/OSA Sensor API to a general-purpose language, USL (Unified Sensor Language), that resulted from an extensive requirements analysis. This analysis aimed to address the widest possible variety of end-user requests, while taking into consideration the actual capabilities of existing WSN middleware [6][7]. The following requirements were specified for such language:

- Support for synchronous requests (queries) that retrieve the requested data, either from the WSN or from a DBMS store, and return the corresponding responses in real-time.
- Support for event-driven programming. Many context-aware applications need to trigger some actions after some events have been generated from the WSN. The users should be able to register listeners and handlers of such events (e.g., in case the temperature is higher than 42 ^{o}C, and indoor luminance is high, the fire alarm should be activated).
- Support for periodic monitoring of the sensor values (e.g., sensing of the temperature every 5 minutes, starting from Monday 2 July, 15:00 p.m. and stopping after a week).
- Easy and dynamic change of the supported sensor types and sensor functions. As new WSNs or sensors are deployed, the programmer

TABLE 1
Indicative API methods.

	Method
High-Level API	**Location.getSensorValue(SensorType, Function)** Returns the requested (function of the) value of the given sensor type, in the given location. The function can be *max, avg, etc.* *
	Location.getDeviceWhere (SensorConditions) Returns the device ID that satisfies the given sensor conditions.
	Location.getTimeWhen (SensorConditions)** Returns a timestamp that satisfies the given sensor conditions.
	Location.addListener(EventFilter) Registers a new listener for a given location. It is triggered when the conditions specified in the EventFilter are satisfied.
	Device.getSensorValue (SensorType, Function) Similar to the first method of this table.
	Device.getLocation () Returns the location of the given device
Low-Level API	**Query.setSensorTypes (SensorTypes)** Sets the requested sensor types (e.g., temperature, humidity, etc.)
	Query.setFunctions (Functions) Sets the functions (*min ,max, etc.*). These should correspond one-by-one to the sensor types of the previous method.
	Query.setMonitor(Monitor) Indicates that the query is monitor. The monitor attribute is a Monitor object that contains the time parameters.
	Query.abort() Cancels the query represented by the query object.
	Query.send() Sends the constructed query to the RR Proxy.
Metadata Methods	**getSupportedSensorTypes(Location)** Returns the sensor types supported by the system in a given location.
	getSupportedTemporalFunctions() Returns the temporal functions supported (e.g., min, avg, count, etc.)

* the functions that can be applied on a sensor type are categorized in spatial and temporal functions, ** the presence of "time" in a method (either in its name or the attribute list) denotes involvement of offline processing by a DBMS

should be able to include the new functionality in the program logic. Moreover, in case of sensor failures, which is a quite common case, adequate error handling should be enabled by descriptive error indications.

The USL specification, an XML Schema we have designed [27], describes two types of entities: *requests* for direct sensor retrieval or registration on sensor events, and *responses*, containing the actual sensor data or error messages returned from the network. GSF also integrates other facilities such as databases for sensor data post-processing and registries for discovery of available sensor networks.

5.2.1 The USL Request

The USL Request represents the requests (synchronous, event-driven and periodic) of the user towards the WSN or a DBMS with stored sensor data. The root element Request contains a unique ID attribute. This ID is generated by the API (see Section 5.1) and uniquely identifies a client request. In case of a synchronous query, the client specifies the contextual information it is interested in within the RequestedInfo element. This information may be one or more sensor readings, the location where a condition holds, the time instance or the duration of a specific phenomenon, the device ID that satisfies some criteria or a combination of the above. Of course, not all combinations are considered valid, so in the WSN Driver a semantic checking procedure is performed (see Section 5.3). The constraints of the query are described in the QueryFilter element. In this element, one can declare time conditions (TimerExpr element), sensor conditions (SensorExpr element) and limit the query to a specified location or sensor-enabled device (Location and Device elements respectively). The SensorExprType (see Fig. 5) is the XML representation of the following Extended BNF [28] grammar:

SensorExpr = [Function,] SensorType, Conditional, Value;
Function = 'tempAverage' | 'tempMinimum' | 'tempCount'
|'tempMaximum' | 'spatialAverage' | 'spatialSum';
SensorType = 'Temperature' | 'Humidity' | 'Acceleration';
Conditional = 'Greater' | 'Less' | 'Equals' | 'WithinRange';
Value = Alphanumeric | RealNumber | Integer;

Of course, the definitions of Function and SensorType are not complete. Their potential values are defined in a separate XML Schema described at the end of Section 5.2.2. The TimerExpr element has a similar syntax.

Additionally, there is an optional GroupBy sub-element that can group the results similarly to the known SQL functionality and has almost the same syntax as RequestedInfo. The Monitor element, if present, denotes that the query should be executed periodically as described by the StartTime, StopTime and Period elements. From now on, we will refer to these periodic

queries as monitors. Similarly to the events, they require that the developer register listeners accordingly.

On the other hand, in case of event-driven programming, the only sub-element of Request is the Event. This element contains a set of conditions (EventFilter element) that, when satisfied, trigger some events to the upper layers of the middleware framework. These upper layers have already registered listeners for these events and upon receipt of an event (through the USL Response entity) they perform some predefined (by the developers) actions. The EventFilter is very similar to the QueryFilter except for the fact that it does not contain time conditions. This seems quite reasonable for this first version of USL as events generated from time conditions can only be implemented with the aid of an offline data storage and require complex information processing techniques. As there is ongoing research in these areas [29], a future version of USL may also support time-based events (e.g., if the temperature change rate in a computer room is +3 ^{o}C/h, inform the building caretaker so as to check the air conditioner).

The USL Request, apart from enabling the registration of event-listeners and the description of queries, can also dispose the already registered event-listeners or monitors. For that purpose the Event and Monitor elements contain the boolean attribute abort. When a user disposes an event or monitor, its known ID is passed in the ID attribute of the Request element and the corresponding abort attribute is set to true, while all the other elements are absent or blank.

A sample first version of the USL Request Schema is depicted in Figure 5, and a sample USL Request is presented in Listing 1. The listing demonstrates a request for a temperature sensor reading, from a sensor mounted on the device with ID 543.

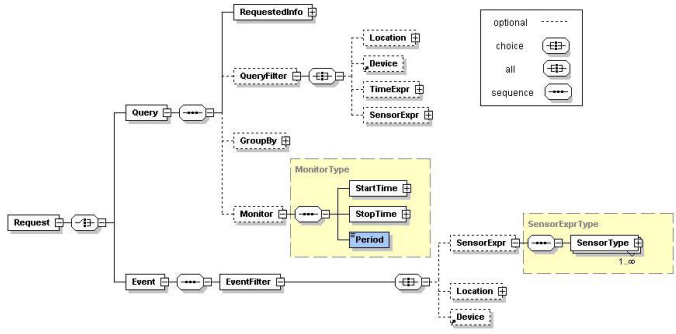

FIGURE 5
The XML Schema of USL Request.

```
<Request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation=" USLRequest.xsd"
ID="123456">
   <Query>
      <RequestedInfo location="false" time-instance="false"
       device="true" sensor="true" time-duration="false">
         <SensorList>
            <SensorType function="NoFunction"
              type="temperature"/>
         </SensorList>
      </RequestedInfo>
      <QueryFilter>
         <Device ID="543"/>
      </QueryFilter>
   </Query>
</Request>
```

LISTING 1
A sample USL Request.

5.2.2 The USL Response

The USL Response is much simpler than the Request entity. The root element Response has also an ID attribute, and always contains the ReturnStatus element. If the error attribute of this element is set to true, then an error has occurred within the WSN or the platform and its type (ErrorType element) is returned to the API in order to raise an application exception. If no error occurred and the request defined a query (or a monitor), the requested data (in RequestedInfo) is returned to the requestor. Alternatively, if the request registered an event, then all the elements except for the ReturnStatus are absent (i.e., the response is equivalent to a flag indicating that the event has taken place).

Some of the parameters in the aforementioned USL elements and attributes may vary occasionally (e.g., due to deployment of new sensors). These are the sensor types, the (unit transformation) functions that can be applied on them and the types of the error indications. All these are described as enumerations in a separate XML Schema document and are included in the above Schemata in order to impose some constraints during the XML validation of the USL request/responses. This separate XML Schema can be regarded as part of a GSF configuration registry, because it is updated whenever the configuration of the WSNs is modified.

A sample version of the USL Response Schema is depicted in Figure 6, and an example USL Request, corresponding to Listing 1, is presented in Listing 2.

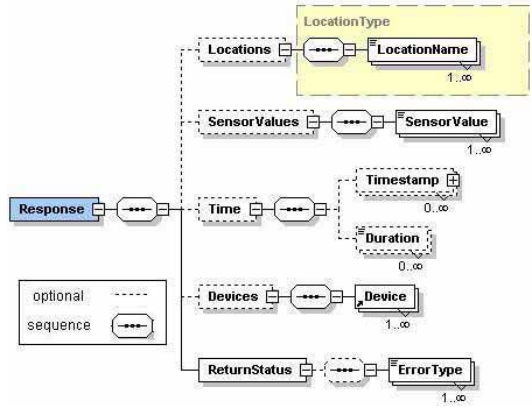

FIGURE 6
The XML Schema of USL Response.

```
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="
WSNResponse.xsd" ID="123456">
   <SensorValues>
      <SensorValue function="NoFunction" type="temperature">37</SensorValue>
   </SensorValues >
   <ReturnStatus error="false">
   </ReturnStatus>
</Response>
```

LISTING 2
A sample USL Response to the Request of Listing 1.

5.3 WSN driver

This driver parses the generated USL requests and translates them to WSN-specific code. For example, if we have a sensor network of TinyDB-enabled motes, the WSN driver would translate the USL requests to TinyDB [7] queries, would collect the TinyDB response, and would finally pass a corresponding USL response to the upper layers of the Sensor SCS. The deployment of the WSN driver in an OSGi Gateway located in the monitored field enables the aggregation of the sensor data and thus is more cost- and bandwidth-efficient. Additionally, OSGi fully supports the dynamic management of the WSN driver (e.g., remote installation of updates) and has been adopted by major industries (e.g., for vehicular telematics and home automation solutions).

The WSN driver processes a request received by the GSF by involving the following steps:

1. *USL Request Parsing (Syntactic Analysis)*: The WSN driver interprets the data elements of the XML-based USL request.
2. *Semantic Analysis*:
   a. *Type Checking*: After parsing the USL request, the WSN driver has to examine, whether the underlying WSN, for which it is responsible, is capable of supporting the request. Firstly, it extracts the list of sensor types (e.g., temperature) to be queried from the RequestedInfo element of the USL Request, and the sensor types (attributes) that appear in the filters from the QueryFilter (or EventFilter) of the USL request. Then, it consults the GSF configuration registry to discover the sensor types supported by the underlying WSN, and in case that any of the attributes of the QueryFilter (or EventFilter) is not supported, the processing ends and an error indication (e.g., Not Supported Request) is returned to the GSF. Otherwise, if all the attributes of the QueryFilter (or EventFilter) and at least one of the requested attributes are supported then the processing goes on.
   b. *Semantic Checking*: During this step the WSN driver checks the semantic validity of the request. If the semantic check fails, an error indication (e.g., Not Valid Request) is returned to the GSF and the whole process ends.
   c. *Request Routing*: If a time expression is among the requested data (e.g., "when was the temperature over 50°C?"), the processing pertains to offline data, and it is served by a DBMS facility with older sensor readings. Otherwise, the request will be served by the WSN.
3. *Request Code Generation*: This step relies on the decision made on the previous step. If the request has to be processed offline, then the USL request will be translated to an SQL query, else, if the request has to be processed in real time by the sensor network, the USL request will be translated to WSN-specific code.
4. *Request Injection*: If all previous steps were completed successfully, the resulting request is injected into the underlying WSN or the DBMS facility and the WSN driver waits for a reply.

6. SBS PROVISIONING ISSUES

The proposed commercialization plan for WSNs, although not very difficult to realize, cannot be materialized in the near future. The main reasons for this are the lack of standards, regulation and the security

holes of the WSNs, which enable privacy threats. The standardization of the WSN network stacks and the application-level interfaces has not even started yet, due to the ongoing research on these fields. Some efforts have been made with the IEEE 802.15.4 specification [9] (part of the ZigBee Alliance stack [19]), but they are covering only a narrow range of issues (i.e., physical and data link layer protocols). In addition, the introduction of third parties (i.e., WSNPs), which have direct access to the user's personal information, can potentially constitute a problem. The users may not be willing to cooperate with such parties and let them in their houses or vehicles, unless there is legal assurance that their sensitive data will not be disclosed for malicious purposes. Such assurance involves the regulation of these issues on a governmental, or better international, level. Moreover, as the recent experience from the LBS world has shown, the users will not adopt easily services with potential privacy threats. However, the situation for SBS is worse since the security threats in the physical, network and application layers are still poorly investigated [17], [18]. For example, eavesdropping is an easy task in current WSNs because most radio-frequency (RF) modules use simple modulations (e.g., FM), broadcast their messages and, in general, there is not cryptography support due to the physical limitations of the sensor nodes (energy, processing power and bandwidth). The insufficient support of the user's privacy increases the risk of the potential investments on these new markets and will postpone considerably the penetration of the SBS services.

7. RELATED WORK

Although the vision for smart appliances and context-aware network services is not new, the industry and research institutes have not yet started to investigate the commercial coupling of sensor and telecommunication networks. This could be attributed both to the limited penetration of 3G services and to the sensor network open problems described in Section 6. An approach, which uses the OSGi Service Platform for the provision of context-aware automotive applications, is presented in [13]. In [20] a context-aware service provisioning platform for 3G networks is described. Some economic and business aspects of sensor networks are being studied in recently published commercial reports [21].

7.1 Existing Sensor Solutions

Some companies already provide commercial sensor nodes and software applications. Sensoria Corp. [22], for example, provides fault-tolerant sensor networking technologies, sensor nodes and open software platforms for sensor-based applications. Moreover, Crossbow Technology Inc. [23] has a great variety of WSN solutions (mainly hardware). The company is also cooperating with other end-to-end solution providers for the development of integrated systems. Some other relevant companies are Oracle Corp. [24], Intel Corp [25], and Sensicast Systems Inc. [26].

These companies/industries could potentially assume the roles of WSNP, SBSO and/or SP. However, this is not the situation today for five reasons:

- The main products of all these companies address environmental monitoring, military applications, building/industrial automation and supply chain management solutions. Hence, their target markets are industries and public sector agencies, and they lack support for customized end-user solutions.
- They have not exploited WSN deployment scenarios in cooperation with mobile network operators yet.
- There are no standard interfaces and platforms for the development of sensor-based services. The existence of such facilities would probably aid the introduction of SBS in wide area mobile networks.
- There is no regulation and legal framework for the provisioning of such services.
- The potential business models have not designed yet, since many technological barriers still exist.

8. CONCLUSIONS AND FURTHER WORK

In this paper, we propose a model for the commercialization of the wireless sensor network through the provision of Sensor-Based Services for mobile subscribers. This architecture is based on existing open standards, (Parlay/OSA and OSGi), and introduces new middleware functionality, which enables the provision of the services and the handling of the sensor networks. In addition, we described a potential business model for such services. Finally, we discussed some problems, which hinder the realization of this new service model and could form future research directions. We are currently implementing the described middleware functionality and developing a prototype system for further

experimentation. Such system could be further exploited for testing the business arrangements discussed in the paper.

Moreover, we plan to extend the proposed business model, in order to provide for service roaming; that is give the ability to customers to be free to "roam" from one MNO to another, without losing the sensor-based services they have subscribed for. Towards this direction we intend to explore whether well-established telecommunications business models and regulations (i.e., facilities similar to number portability) can be also applied for SBS.

9. REFERENCES

[1] Moerdijk, A.J. and Klostermann, L., (2003), "Opening the Networks with Parlay/OSA: Standards and Aspects Behind the APIs", *IEEE Network*, pp. 58-64.

[2] OSGi Alliance Specification Overview, URL: http://www.osgi.org/resources/spec_overview.asp

[3] Akyildiz, I. et al., (2002), "Wireless sensor networks: a survey", *Computer Networks*, Elsevier, pp.393-422.

[4] Szewczyk, R.et al., (2004), "Habitat monitoring with sensor networks", *Communications of the ACM*, 47(6), pp. 34-40

[5] Essa, I., (2000) "Ubiquitous Sensing for Smart and Aware Environments", *IEEE Personal Communications, Special Issue on Networking the Physical World*, pp. 47-49.

[6] Shen, C-C., Srisathapornphat, C. and Jaikaeo, C., (2001), "Sensor Information Networking Architecture and Applications," *IEEE Personal Communications*, pp. 52-59.

[7] Levis, P. et al., (2004), "The Emergence of Networking Abstractions and Techniques in TinyOS", *First USENIX/ACM Symposium on Networked Systems Design and Implementation*, San Francisco.

[8] ISTAG, "Scenarios for Ambient Intelligence in 2010", Final Report, Feb 2001, EC 2001 URL: http://www.cordis.lu/ist/istag.htm

[9] Gutierrez, J.A., et al., (2003), *Low-Rate Wireless Personal Area Networks: Enabling Wireless Sensors with IEEE 802.15.4*, IEEE Press.

[10] Barkhuus, L., Dey, A., (2003), "Is Context-Aware Computing Taking Control Away from the User? Three Levels of Interactivity Examined", *Proceedings of UBICOMP 2003*, Seattle, pp. 149-156.

[11] Telenity's Canvas™ Location Enabling Server, http://www.telenity.com/location_services.php

[12] Swedberg, G., (1999), "Ericsson's mobile location solution" *Ericsson Review No. 04*.

[13] Zhang, D., Wang, X., et al., (2004), "OSGi Based Service Infrastructure for Context Aware Automotive Telematics". *IEEE Vehicular Technology Conference*, Milan, Italy.

[14] Ioannidis, A., et al., (2003), "PoLoS: Integrated Platform for Location-Based Services", *IST Mobile & Wireless Communications Summit*, Portugal.

[15] Paavilainen, J., (2002), *Mobile Business Strategies,* Addison Wesley.

[16] Timmers, P., (1998), "Business Models for Electronic Markets", *Journal on Electronic Markets*, Vol 8, pp. 3-8.

[17] Perrig, A., et al., (2001), "SPINS: Security Protocols for Sensor Networks", *MOBICOM 2001*, Rome, Italy.

[18] Slijepcevic, S., Tsiatsis, V., Zimbeck, S., (2002), "On Communication Security in Wireless Ad-Hoc Sensor Networks", *WETICE'02*, Pittsburgh.

[19] ZigBee Alliance, http://www.zigbee.org

[20] Costa, P., et al., (2004), "Towards a Services Platform for Mobile Context-Aware Applications", *First International Workshop on Ubiquitous Computing (IWUC 2004)*, Portugal.

[21] ON World Inc., (2004), Wireless Sensor Networks: Mass market Opportunities.

[22] Sensoria Corp., http://www.sensoria.com

[23] Crossbow Technology Inc., http://www.xbow.com

[24] Oracle Sensor-Based Services http://www.oracle.com/technologies/rfid/index.html

[25] Intel Corp., http://www.intel.com

[26] Sensicast Systems Corp., http://www.sensicast.com

[27] Unified Sensor Language XML Schema, http://p-comp.di.uoa.gr/projects/sensation/

[28] Crocker, D.H., (1982), "*Standard for the format of ARPA Internet text messages*", STD11, RFC 822, UDEL.

[29] Zhao, F., Guibas, L., (2004) "*Wireless Sensor Networks: An Information Processing Approach (Morgan Kaufmann Series in Networking)*", Morgan Kaufmann.

[30] Hasiotis, T., et al., (2005) Sensation: A Middleware Integration Platform for Pervasive Applications in Wireless Sensor Networks, *European Workshop on Wireless Sensor Networks 2005*, Istanbul, Turkey.