A Service Framework for Mobile Ubiquitous Sensor Networks and RFID

Tomás Sánchez López, Daeyoung Kim, Taesoo Park

***Abstract*— Recent research in mobile computing envisions a future of "smart environments", where user and computing merge to realize the so called "ubiquitous society". As an exponent of mobile information units, RFID (Radio Frequency IDentification) and Wireless Sensor Networks (WSN) will play a key roll in this future. In every real-world object, RFID tag information will be combined with sensor data to embody "smart" entities. These entities will move freely, providing their sensor values and identity and getting in return personalized services anywhere, anytime. Entities will interact, creating grouped objects that will provide unified and meaningful information, widely extending the available services. But how to control this ubiquitous flow of information? How will users really benefit from it? In this paper we will introduce WISSE, a framework that defines the interaction between mobile RFID/WSN entities and the services they receive.**

***Index Terms*— Sensor Networks, RFID, Pervasive, Ubiquitous, Smart Environments**

I. Introduction

THE vision of a future populated with intelligent objects to react and adapt to the environment for better serving the human being has been on the air for over a decade. Many new concepts have arisen from this idea, including terms such as "smart environments", "ubiquitous computing" and "pervasive computing". However, the underlying concept is simple: If computing and sensing units become small enough, they might be integrated in every place providing a rich flow of anytime-anywhere services.

Together with the maturing of the concept, technology has evolved to produce new and more feasible pervasive computing units. Amongst all, RFID (Radio Frequency IDentification) and WSN (Wireless Sensor Networks) are gaining special interest for their promising applications; on one hand, RFID tags are finally becoming mature enough for their massive deployment. On the other hand, WSN are now on the verge of producing practical application, not only for industrial use, but also to support people's daily life.

RFID technology is already a de-facto standard in wireless electronic identification which main applications reside in integrating RFID chips in real-world objects, including those carried daily by people (i.e clothes). The same way RFID technology aims to provide object information (simple identification) transparent to the user, in a close future wireless sensor devices installed in those same objects could *augment* their information in a similar transparent way. The resulting "smart" entities, equipped at the same time with RFID tags for global identification and WSN for real-time sensor information, will additionally group and interact providing a richer flow of aggregated and unique identified information. Finally, this information will be relied and interpreted at a *service provider* layer, which in turn will offer different services according to the information every object can provide. Unlike current research in the area, our framework supports dynamic and spontaneous formation of "smart" entities. Therefore, any fortuitous temporal union of "smart" objects will enable new combinations of sensor data and hence new services.

Projects such as [1] or [2] exist that address the ubiquitous or pervasive environments, although superficially considering mobile SN (Sensor Networks). Context aware research (such as [3] or [4]) is mostly focused on "augmenting" determined objects to offer specific services. Other projects (such as [5]) also imagine WSN attached to users, although mainly for monitoring activities rather than offering services. Finally, service frameworks for ubiquitous computing have also been defined [6], although they normally focus just on how to locate the elements that will provide the information for the service. However, to our knowledge none of them consider mobile SN in combination with RFID information, and those who embrace together both RFID and SN technologies, just reflect static SN such as those inside buildings or "smart homes" [7].

WISSE: Wireless Sensors and RFID for ubiquitous Smart Environments defines a framework for service delivery to mobile entities carrying RFID/WSN. Aside from the introduction of the framework, in this paper we investigate the interaction of the popular *passive* RFID tags (class 0 and 1) and the WSN, imagining scenarios where the manufacturers integrate also wireless sensor devices together with the already used passive RFID tags. Nonetheless, future work will also consider *active* RFID tags and their expected close integration with sensor devices.

The rest of this paper is organized as follows. Section II introduces WISSE framework. Section III presents the concept of entity, while Sections IV and V detail entity grouping and the virtual entity concept. In Section VI we describe some middleware, while in Section VII we define WISSE services. Section VIII introduces some applications and possible implementations, and section X concludes the paper.

The authors are with the School of Engineering, Information and Communications University (ICU), Daejeon, South Korea (e-mail: tomas@icu.ac.kr, kimd@icu.ac.kr, iamtaesoo@icu.ac.kr)

II. WISSE FRAMEWORK

WISSE envisions a future of smart environments in which mobile objects and users carrying RFID and WSN will get ubiquitous services according to its identity and real-time sensor information. Consider the following example: Anne finishes her work and goes back home on a hot summer day. Her combination of today's clothes carries RFID tags and temperature sensors. Moreover, her RFID enabled digital ID-card and her GPS cell-phone complete Anne's usual outfit. On her way home, her shirt sensor detects unusual high temperature due to today's atypical hot weather. Fortunately for Anne, her profile in the WISSE network has noticed this fact as well, and realizing that she is walking home (GPS) sends an actuation command on Anne's behalf to her home server, turning on the AC just one time. Moreover, Anne's cell phone receives a map with the location of a new ice-cream shop on her way home, just in case she feels like making a refreshing stop.

To make this example possible, WISSE defines a framework from the service receivers to the service providers and the logic for automatic, optimum service delivery. Fig. 1 depicts the WISSE framework and its main components. Throughout the following section we will detail each one of the blocks and the interaction between them.

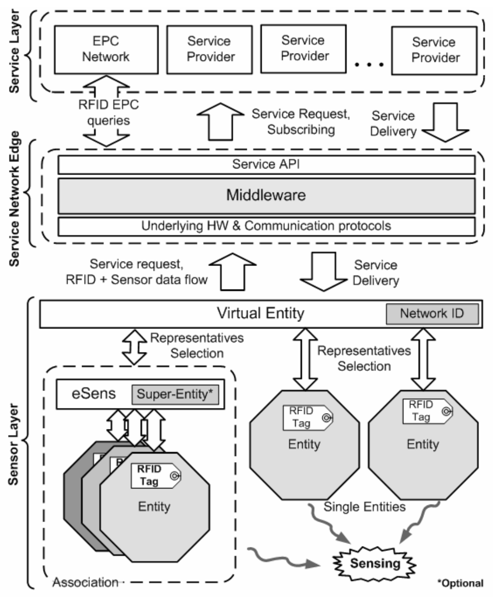


Fig. 1. WISSE Framework

Following the separation between service providers and receivers, WISSE defines two layers and the border that separates them:

- Sensor Layer (SSL): contains the real-world objects carrying RFID/WSN. These objects, called entities, may associate in *eSens* forming groups of meaningful information [III]. Examples of entities could be clothes, vehicle parts, tableware, cell-phones, etc. Examples of *eSens* could be users, vehicles and so on. From now on we will use the term *entity* to refer equally grouped or ungrouped objects (depending on the context), whereas *eSens* and *single entity* will be used to specifically name grouped and ungrouped objects respectively.
- Service Layer (SRL): contains those devices that process sensor and RFID information and offer services to the sensor layer. The EPC network service represents queries to obtain RFID information. Examples of other services could be from databases and software repositories to health monitoring or bridges to home actuation.
- Service Network Edge (SNE): although the presence of powerful gateway devices in mobile objects (i.e. PDA) is possible, it shouldn't be considered as a norm. This assumption would be unreasonable for inexpensive entities (i.e a cup with a temperature sensor and RFID tag) and restrictive for other entities and eSens (i.e. a user). WISSE doesn't make any assumptions in the capabilities of the SSL devices and defines a SNE holding edge devices that serve as temporal gateways for the SSL objects. Together with them, efficient entity representative selection and edge device control provide efficient communication.

III. ENTITIES

In this paper, WISSE entities are defined as SSL indivisible objects carrying a single (passive) RFID tag and one or more wireless sensor nodes. Also only-RFID entities are allowed, but with the restriction that there should be a RFID reader present to enable communication with the sensor nodes from its surroundings (i.e. its group). Finally, WISSE defines a special case of entities called super-entities. These entities represent higher level logical entities (i.e. users or vehicles). Examples could be digital ID-cards such as passports or driving licenses.

Entity sensor nodes are assumed to know the RFID code of the entity they belong to. For only-RFID objects, the RFID reader device will serve as an intermediary, although will also play its role as an independent sensor.

IV. ENTITY GROUPING

Single entity data is normally poor in the sense that can not offer enough context information for getting relevant services. Providing as much sensor data as possible for the same conceptual real-world object may extend its context information and hence the possible available services.

Grouping is the collaboration of entities to offer joint data and get joint services. As a first step of grouping, one or more representatives should be elected in order to communicate with higher levels (SNE). WISSE defines a two level clustering in which both single entities and *eSens* select their representatives. First, each single entity will choose one of its nodes as *Entity Cluster Head* (ECH). Second, one of the selected ECH will be chosen as *eSens correspondent*, and will be the one to communicate directly with the edge devices. Both ECH and correspondent roles will be assigned temporally and in a distributed manner according to the nodes capabilities and remaining energy. The election process is similar to the one specified in [8]. Correspondents will be later also used for grouping purposes. The clustering algorithms and the actual grouping are briefly explained in following subsections.

A. Entity Cluster Head Election

"ECH capable" nodes broadcast proposals for being ECH for a time T_{entity}, which is a function of the node's remaining energy. Only nodes which compute higher T_{entity} will respond to the proposal. In order to avoid collisions when more than one node tries to answer, a delay in the response is introduce, also function of the remaining power plus a random component. ECH election process start when (1) Time T_{entity} expires, (2) No cluster head is selected in current entity or (3) a node can not communicate with its ECH before T_{entity} expires.

Once the process is finished, ECH election result will be broadcasted to the rest of the entity nodes, which from that moment will use the ECH as rely node for extra-entity communication.

Unlike *correspondents*, whose entity might be ungrouped without prior notice, EHCs are unlikely to die without warning as they remain attached to its entities. In fact, apart from some rare HW failure, the major reason for a ECH node to disappear would be due to battery exhaustion. Following this reasoning, while correspondent nodes should not store irreplaceable databases, ECH might store some data to ease the sensor data collection process. WISSE defines that each ECH shall store a table with a list of its entity's nodes and the sensors they hold (*Node Table*). Note that one entity might have one or more sensor nodes and each sensor node can hold one ore more sensors. Moreover, each ECH will hold the entity ID of every one of the entities in its group (*Entity Table*). When a ECH finishes its representation period, its entity table and node table will be transferred to the next elected ECH.

B. Correspondent Election

Once the process of selecting ECHs is finished, a correspondent among them will be chosen as representative of the group.

It is possible for certain nodes of an entity to be in range with the SNE while some other remain "hidden" or out of range. Apart from an efficient energy use, the correspondent election procedure should not choose a correspondent which is hidden while some other ECH from the group is in range with the SNE. To address this issue, edge devices will send ADV packets (holding their Edge Device ID and Subsystem ID - Section V -) to advertise their presence. Only ECHs that receive an ADV will start the correspondent selection process. However, to prevent the no selection of a correspondent (as we need it for the grouping process), if no ADV is received by any ECH, a temporal "not connected" correspondent will be selected.

Correspondent election procedure follows the same algorithm exposed before for the ECH election, using $T_{corresp}$ as the proposed time. Correspondent election procedure starts when (1) no correspondent is currently selected in the group, (2) when $T_{corresp}$ expires, (3) when current correspondent looses range with the SNE, (4) an ECH who didn't participate in the previous election and has more remaining power than current correspondent now receives an ADV packet, (5) an ECH can not communicate with current correspondent before $T_{corresp}$ expires or (6) a new entity is added to the correspondent's *eSens*.

C. Grouping Procedure

Once a correspondent is selected, the entity it represents might group with other entities. WISSE provides automatic and intelligent grouping of entities in *eSens*. In order to provide dynamic *eSens* formation (grouping - ungrouping) and entity addressing for any entity in the real-world without causing conflicts, WISSE SSL networks should hold unique IDs (NetID). RFID EPCs (Electronic Product Codes) appear as the perfect candidates as they are inherently universal identifiers, becoming then part of the header of any packet sent by that entity. Whenever two entities group, one of the NetIDs is decided as representative for the group, and becomes the group's (network) NetID. The process of grouping and selecting a group's NetID is following briefly explained.

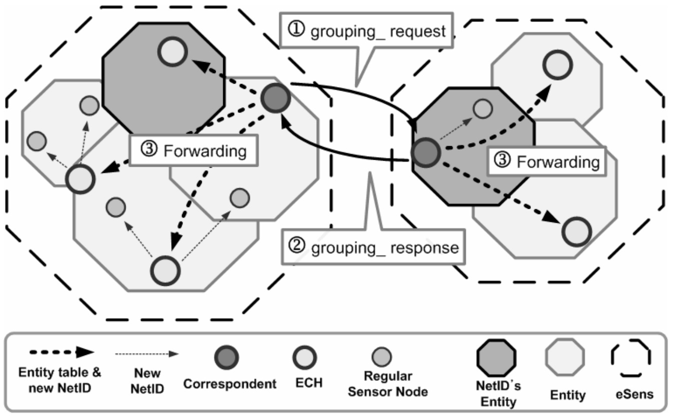


Fig. 2. Example of the grouping process

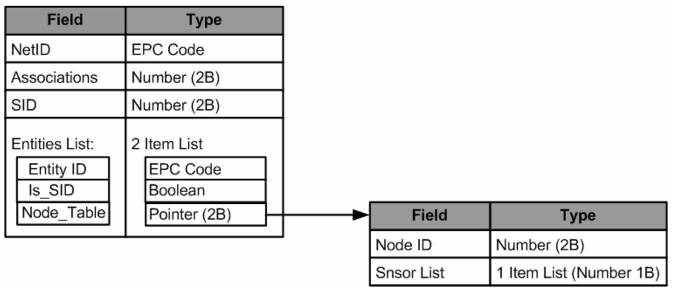
In order to look for possible nearby entities to group with, correspondents regularly broadcast a *grouping_request* packet advertising their presence. Any correspondent receiving a *grouping_request* packet will reply with a *grouping_response* if both entities remain in contact for a certain time *T*. The contents of the response packet will depend on which NetID is selected for the group, the nature of those NetIDs (if they belong to super-entities or not) and if grouping entities are *eSens* or single entities. In general, entities will exchange the selected NetID and entity tables, and keep count of how many entities are associated in their group. Fig. 2 shows an example of entity grouping and how the entity information is forwarded to all the entities in the group. The result of the grouping will be a single *eSens* with one representative NetID.

In the case of grouping only-RFID entities, RFID reader devices present in any of the grouping entities will be used as mediators in the process. Readers will issue *grouping_request* packets on behalf of the new only-RFID entities they detect but will ignore *grouping_responses* as no information needs to be stored.

V. Virtual Entity

In order to provide efficient service delivery, entity information should be stored in a single location. Keeping this information decentralized in entity tables and node tables would dramatically increase the communication overhead each time the information is required by the SSL. To solve this problem, WISSE introduces the concept of *virtual entities* and *edge device controllers*.

Virtual entities (VE) represent an *eSens* or single entity for the rest of the WISSE architecture. A virtual entity is a table which stores information of the entities associated with a certain NetID, including how many entities are grouped, their entity's ID, which sensors are carrying, etc. Fig. 3 presents the Virtual Entity Table (VET). The fields *NetID* and *Associations* represent the NetID of the registered entity and number of grouped entities respectively. *SID* specifies how many super-entities, if any, this VE has. Finally the *Entities List* holds, for each one of the entities in the group, its *Entity ID* and its *Node Table*. Super-entities will show TRUE in the *Is_SID* field, while only-RFID entities will hold a NULL value in their *Node_Table* pointer.



Field	Type
NetID	EPC Code
Associations	Number (2B)
SID	Number (2B)
Entities List:	2 Item List
Entity ID	EPC Code
Is_SID	Boolean
Node_Table	Pointer (2B)

Field	Type
Node ID	Number (2B)
Snsor List	1 Item List (Number 1B)

Fig. 3. Virtual Entity Table

Having discarded correspondents to store the entity's VET (IV.A), the question is which part of the upper layers should hold the database. Edge devices are not a good choice since this would also incur in traffic overhead each time an entity roams between them. Moreover, handoff between edge devices suggests the idea of some centralized controller to manage the process. Edge Device Controllers (EDC) perform the role of managers of all the edge devices in their *subsystem*. The amount of edge devices in one subsystem will depend in several factors such as the average traffic load (average number of VETs to manage) or geographical reasons (i.e. a shopping mall area might want to control their own subsystem for localized services). EDCs will thus hold a database with the VETs of the entities present in their subsystem, so entities roaming between edge devices belonging to the same subsystem won't require any VET related data transmission. Inter-subsystem handoff, however, will require EDCs to perform a VET transfer in order to avoid entities to carry out VE registration again. Fig 4 summarizes the process.

The process for an entity to upload for the first time its information to the EDC VET is called *Virtual Entity registration*. VE registration will take place when an entity is first provided with an ECH and correspondent. From that moment, any grouping process will trigger a *Virtual Entity update*. When a grouping occurs and, thus the VE update, the grouping party (single entity or *eSens*) whose NetID was discarded requests the removal of its VET and the transfer of its contents to the VET of the chosen NetID's entity. This process involves the copy of the Node Table and the updates of the fields *Associations* and *SID*. When the update is finished, the VETs of the grouping entities in the EDC will have merged into one VET representing the new *eSens*.

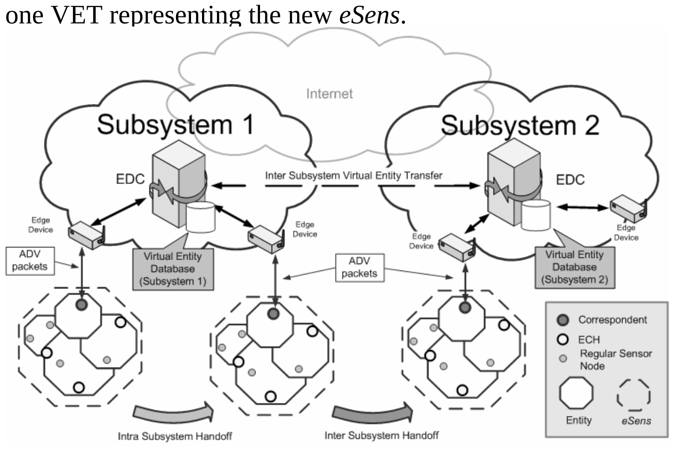


Fig. 4. WISSE Subsystem Infrastructure

VI. EDC Middleware

Apart from the tasks of managing the edge devices and store the database of Virtual Entities, the EDCs have the responsibility of act as mediators between the SSL and the SRL and build the logic for meaningful service delivery. The part of the EDC that performs this process is called *EDC Middleware*. The middleware of the edge devices controllers will thus manage the available services for the VEs and will be in charge of processing the requests from the SSL and contact the SRL on behalf the entities. A block diagram of the EDC middleware is shown in Fig. 5.

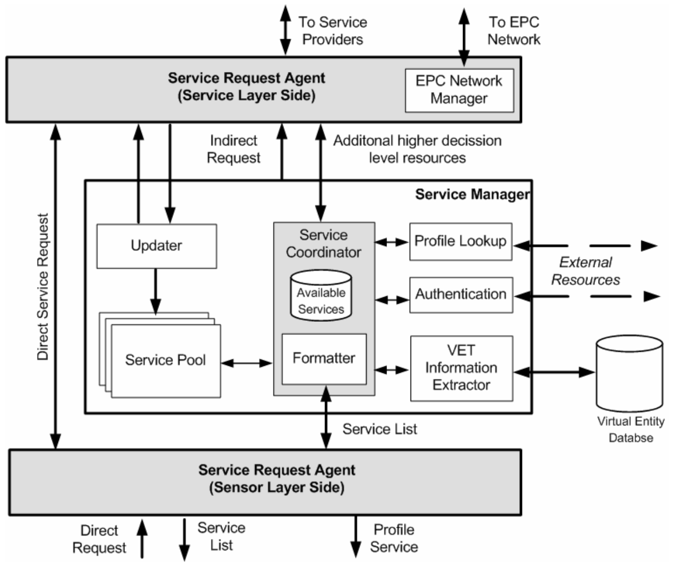


Fig. 5. EDC Middleware

A. Service Manager

The main part of the EDC middleware logic is implemented in the *Service Manager*. The service manager maintains a *service pool* of available services and which requirements are needed for them (i.e. which sensors. For the example in II, temperature and location sensors are necessary). The *updater* regularly searches for new services in the SSL and updates the service pool. The *VET Information Extractor* gathers the available sensors and ID information and transfers it to the *Service Coordinator*. To compute the available services for a certain VE, the service coordinator performs a cross-checking between the VET information extractor and the service pool, storing the matches.

Once the available services for each VE is computed (process that will be rerun periodically depending on new VE registrations and updates, as well as the appearance of new services), actual service request and delivery for SSL entities may start. There are two ways in which a service may be requested:

- Direct request: A formatted list of available services is delivered to the SSL waiting for manual service selection. For example, a user carrying a PDA is displayed a list of matching services from which she will choose if needed.
- Profile based request: Some part of the SSL or SNE contains a list of profiles that have been registered in advance. A *Profile Lookup* module will check available profile repositories and inform the Service Coordinator if some match with a managed VE is found. For the example in II, a profile similar to the following could have been used: "If Anne is going home and her temperature is over X, send 'Turn On AC' command to Anne's home server". Note that *temperature* and *location* refer to temperature and GPS sensors, while *Anne* refers to Anne's signature (i.e. her digital ID RFID EPC).

B. *Service Request Agents*

The *Service Request Agents* (SRA) will serve as intermediaries between the SSL, the Service Manager and the SRL. Once a user performs a *direct request* fruit of her choice from a previously delivered service list, the SRA-SSL will received the request, check it for consistency and hand it over the SRA-SRL for actual service request. After the service request is processed in the SRL and a response is received, the SRA-SRL will transfer it to the SRA-SSL and then the response will be delivered to the requester entity. If the service request comes from a profile match rather than from a user direct choice, an *indirect request* will be delivered from the Service Manager directly to the SRA-SRL. The service response will follow the same path as the direct request.

VII. Service Definitions

The different services that the SSL may provide are innumerable and depend on the service providers. In general, service providers will make available server equipment attached to the WISSE network following a standardized interface. This interface should include directives such as handlers for EDC service request and service subscription or issuing commands such as service response or service announce.

WISSE classifies the type of services in a 3-2 organization. On one hand, *Home Services*, *Personal Services* and *Object Services* are categorized according to the service receiver. On the other hand, *Public Services* and *Subscriber Services* classify them by their access policy. The first three types of services keep a close relation between them. Fig. 6 shows a diagram of this relationship and a list of examples.

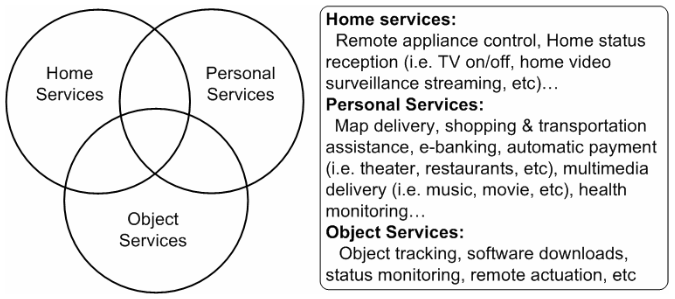

Home Services
Personal Services
Object Services
Home services:
Remote appliance control, Home status reception (i.e. TV on/off, home video surveillance streaming, etc)…
Personal Services:
Map delivery, shopping & transportation assistance, e-banking, automatic payment (i.e. theater, restaurants, etc), multimedia delivery (i.e. music, movie, etc), health monitoring…
Object Services:
Object tracking, software downloads, status monitoring, remote actuation, etc

Fig. 6. Service types by its receiver

A. *Service Types*

Personal services are those services requested by a user directly or inderectly (i.e. by manual selection or served through some profile). Home services refer to any service requested by an entity and delivered to some intermediate server for further processing (i.e. a Home Server). Object services are those which refer to inanimate entities and that can not be manually requested and delivered to the same entity. The overlapping in the classification obeys to combinations of the service types.

Public services refer to those services that are publicly available in the network and need no registration or subscription to the provider. Subscriber services, on the contrary, are those services which require an agreement with the provider (i.e. subscription and fee payment) to be used. Both service types in this classification are mutually exclusive and can be applied to the services in any of the previous classifications.

B. *The EPC Network Service*

The *EPC Network* provides a standard infrastructure to discover and share information about the RFID EPCs [9]. One of the main advantages of using RFID tag information as the NetID for WISSE entities, appart from its unique nature, is that those entities can benefit from the use of the EPC Network infrastructure. When the information about a virtual entity is retrieved from its VET, its NetID (or any of the Entity's ID in its Entities List) may be encapsulated inside an EPC Network packet and sent to the EPC ONS (*Object Name Service*). The ONS will eventually return the location of the the manufacturers EPC *Information Service* (IS), which contains information about the concrete tag RFID code of the entity associated.

The exposed infrastructure coupling provides a significant service enhancement. On one hand, real-world objects and users are able to get ubiquitous services according to the real-time sensor information and its abstract ID. On the other hand, providing EPC Network connectivity converts those abstract identifications into real universal identifiers, managed by their product manufactures and even able to take part on the supply chain. This could allow, for example, automatically adjust the service delivery to certain models of objects (i.e. cell phones) which specification can be retrieved from their manufacturer's IS.

Finally, the possibility of implementing local EPC IS and ONS and register them with the EPC *Discovery Service* broadens even more the oportunities. WISSE subsystems IS could keep track of entities and the edge device they used, and even history of sensor data fits into the data models proposed by EPC white papers [10]

VIII. Applications and Possible Implementations

Through this paper, many examples and applications of the WISSE service framework have been exposed. In concrete, example in section II has been frequently named and discussed. Fig. 7 represents this example graphically, including several of the concepts explained in the previous sections.

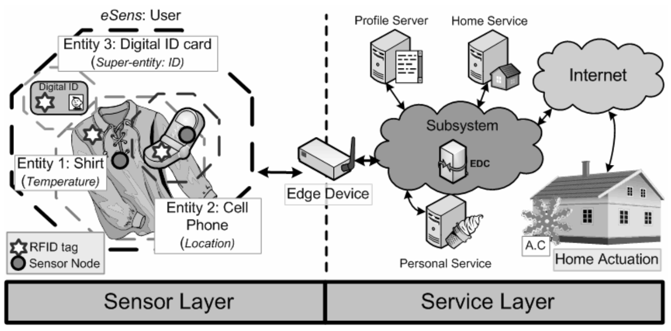


Fig. 7. WISSE example for automatic home actuation

In a practical point of view, developing from scratch the infrastructure to support WISSE would be a rather complex and costly task. Although the Sensor Layer part can scale steadily with the production of new objects carrying RFID tags and sensor devices, the Service Layer part and its Service Network Edge would need a bigger investment in equipments and network infrastructure before to start offering services. To alleviate this problem, as WISSE underlying communication protocols are not restricted, it is possible to integrate part of the WISSE infrastructure with existing telecommunication networks. For example, in the case of mobile telecom. networks, edge devices could be deployed that connect to the operator's network through its existing cellular infrastructure. Edge device controllers would be located in the Base Station Subsystems (BSS), connected to the Mobile Switching Centers (MSC) and Base Station Controllers (BSC) of the cellular networks. The reuse of part of the existing network would not only ease common procedures such as handoff between EDCs, but also provide an affordable way for the telecommunication operators to expand their services to mobile sensor networks.

IX. Implementation

To test the WISSE concept, we have implemented a simple scenario using real wireless sensor nodes [11]. Fig. 8 presents the elements of our scenario and the interaction between them. A sensor relays user's body temperature through its correspondent (PDA) to a *water supply* service on the service layer. The user receives a list of available services on its PDA and, once he selects the *water supply* service, water supply points (i.e. fountain) will appear on the screen if his temperature gets too high. GPS is used for locating the user and the closest water supply points. All algorithms explained in this paper were implemented both in the sensor node and the PDA. Also the VET and middleware concepts where partially implemented on the server side.

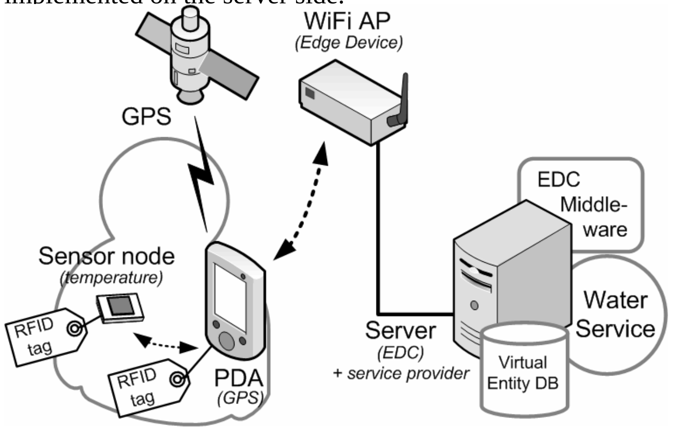


Fig. 8. WISSE implementation scenario

X. Conclusion

WISSE framework describes a future in which mobile entities will actively participate in ubiquitous services both providing and receiving information from the network. Our architecture defines "smart" entities that provide sensor data stamped with their unique ID (RFID), and that can spontaneously and dynamically interact enabling an infinite number of possible services. Additionally, our architecture might be merged into existing telecommunication infrastructure for fast deployment and low cost, offering a realistic approach to the present infrastructures' status.

References

[1] MIT Project Oxygen. 2004 Available: http://oxygen.lcs.mit.edu/.
[2] Manuel Román et al., "Gaia: A Middleware Infrastructure to Enable Activ e Spaces." IEEE Pervasive Computing, , Oct-Dec 2002, pp. 74-83.
[3] F. Kawsar et al., "Experiences with Developing Context-Aware Applicati ons with augmented artifacts", in *Proc. Ubicomp 2005*, Tokyo, Japan
[4] H.W. Gellersen et al., "Adding some smartness to devices and everyday t hings", WMCSA'00, 2000
[5] Emil Jovanov et al., "A wireless body area network of intelligent motion s ensors for computer assisted physical rehabilitation", Journal of NeuroEng ineering and Rehabilitation, Volume II, 2005
[6] M. Takemoto et al. "A service-composition and service-emergence framework for ubiquitous-computing environments," in *Proc. SAINTW'04*, 2004, pp. 313-318.
[7] Yoshinori ISODA et al., "Ubiquitous Sensors based Human Behavior Mo deling and Recognition using a Spatio-Temporal Representation of User S tates", in *Proc. AINA'04*, 2004, Valume 1, p. 512,
[8] Srikanth Kandula, Jennifer Hou, Lui Sha, "A case for Resource Heterogen eity in Large Sensor Networks", in *Proc.* MilCom 2004
[9] Verisign, "The EPC Network: Enhancing the Supply Chain", *Whitepaper*, 2004
[10] Mark Harrison, "EPC Information Service- Data Model and Queries", *Whi tepaper*, October 2003
[11] Daeyoung Kim, Tomas Sanchez, Seongeun Yoo, Jongwoo Sung, Jaeeon Kim, Youngsoo Kim, Yoonmee Doh, "ANTS: An evolvable Net work of Tiny Sensors", in Proc. EUC'05, 2005, Nagashaki, Japan.