ICU

AUTO-ID LABS

Wireless Sensor Networks and RFID integration for Context Aware Services

Tomás Sánchez López and Daeyoung Kim

Abstract: Recently, the debate over the integration of Wireless Sensor Networks (WSN) and Radio Frequency Identification (RFID) has garnered an increasing amount of attention despite their appearing to be disparate technologies. On one hand, the RFID community realizes how adding sensor data to their infrastructures could provide added value to the RFID-based services. On the other hand, the use of unique identifiers in WSN would improve their manageability in a future where millions of sensor nodes could be scattered across the globe. However, not only is it not clear which specific applications such an integration could bring, there is also no consensus regarding the best approach to achieve it. In this article, we propose a framework in which WSN and RFID coexist to provide context information about users and objects. WSN information is coupled with unique identifiers and participates in the RFID core services. Additionally, context aware services interact with the RFID infrastructure and provide real-time service to the users participating in the framework.

Information and Communications University
119 Yuseong-gu, 305-714, Daejeon, South Korea
e-mail: {tomas, kimd}@icu.ac.kr

AUTOIDLABS-WP-SWNET-026

Information and Communications University · 1997 · ICU 한국정보통신대학교

1. Introduction

The evolution of WSN and RFID research have followed separate paths and have traditionally considered distinct technologies. WSN are networks of small, cost-effective devices that can cooperate to gather environment information via simple integrated sensors. Due to the limited resources of the sensor nodes, most research efforts of WSN fall into developing efficient protocols which collect data and forward it to some local base station, neglecting considerations towards infrastructure networks or standardization. RFID and the EPC Network [EPCnetwork(2005)], however, were envisioned for the tracking of assets in the global supply chain. In order to link the physical object identification data with its related information on the RFID infrastructure, an Internet based architecture was proposed and developed as an international standard through associations among the industry and global standardization bodies. If these technologies are so different in nature, why is the discussion about their integration gaining so much momentum among the industry and research institutions? The answer may vary across the WSN and RFID communities, but yet appears to be a common understanding: RFID and WSN can complement each other adding value to the services they already provide.

Among many other applications, WSN are used to assist the so called Smart Spaces. In these type of applications, sensors gather users' context as circumstances or conditions that surround their behaviour. Services are then offered to the context source, involving automatic actions that a user would desire in a particular situation. We believe that the integration of RFID and WSN can be extremely advantageous in these type of applications. The same way RFID provides object information (simple identification) transparent to the user, wireless sensor devices installed in those same objects could augment their information in a similar transparent way. Merging RFID and sensor networks into the same objects would result in truly smart objects, moving around while providing their sensed context with a unique identification number. In our research we analyze the challenges that arise from integrating RFID and WSN for their use in Smart Spaces. We design a framework that considers all the processes involving users receiving services according to their context, from RFID tag memory design, to WSN protocols and distributed middleware and services. We also consider the best ways for integrating our ideas with the existing RFID infrastructures, the EPC Network, and provide evaluation tools and implementation scenarios that aim at proving our concepts.

This article is structured as follows. Section 2 describes the system design and identify the different players in our framework picture. Sections 4 and 5 detail the most important roles of the architecture, taking special care of the way the distinct parts match to produce the global system results. Finally, in Section 6 we provide our experience implementing a real scenario and a set of simulation and monitoring tools that help us to evaluate our results

2. System Architecture

In our framework we design an integration scenario where mobile objects and users carrying RFID tags and WSN receive ubiquitous services according to their identity and real-time sensor/actuator information. We call this framework WISSE (Wireless Sensors and RFID for Smart Environments). The cornerstone of WISSE is a managed merge of RFID and sensor/actuator information at the context level, as well as a distributed infrastructure that allows the tracking of the context in a real-time manner. In this way we aim at addressing two typical problems of RFID-WSN integration and Smart Spaces: First, since the RFID and sensor/actuator information are integrated at the lowest level of the architecture, we ease the ambiguity problem where WSN information and identity data meet in upper layers but lack enough context to produce accurate matching. Second, by providing a distributed infrastructure and middleware, we solve the locality problem associated with WSN, where service receivers are confined to the local area of the WSN operation.

The WISSE framework is a layered architecture that discriminates service providers (context consumers), service receivers (context producers) and the border that separates these two layers:

- *Context Layer*: contains the objects carrying the RFID/WSN and the users they belong to. These objects, which we call entities, are the sources of the context and may associate forming groups of meaningful information. Examples of entities are clothes, identification tokens, electronic devices, vehicle parts, tableware, etc.
- *Service Layer:* Contains the service providers registered to the WISSE network. As it will be detailed later, the service providers specify a series of requirements in terms of sensors and actuators that clients must meet in order to be eligible to receive their services. Additionally, service providers hold the logic and processing power needed for executing the services they offer, as well as a standard way of specifying how the clients should access their services.
- Service Network Edge: It provides the means for matching the context of the entities in the Context Layer and the service requirements from the Service Layer. Since the entities are mobile, their associations are spontaneous and temporal, which results in a very dynamic context. Furthermore, those entities may be anywhere and anytime, demonstrating the need for a distributed infrastructure that will connect clients and servers while tracking their context and requirements respectively. The Service Network Edge holds the tools for receiving, storing and matching user and service information in a distributed, real-time manner.

Figure 1 depicts the WISSE framework through an example. Starting from the bottom, the Context Layer holds several entities such as clothes, furniture and personal identification cards. Three entities associate around a user creating the grouped entity “businessman”, which will immediately begin to produce joint context information. When associating, entities (a smart phone, a shirt and a digital ID card) choose a common identification (the ID card) and a representative (the smart phone) for transmitting their context to the WISSE Service

Network Edge. As an example scenario, consider the following situation: While going back home after work, the user's shirt temperature and humidity sensor detects an increasing body heat resulting from an unusual warm day. Together with the location provided by his phone's GPS and identity from his ID card, this information is transmitted to the system by a CDMA link. The Service Network Edge recalls that this user also hired a home-network actuation service, and together with the user's context information, triggers a remote actuation command to turn on the businessman's home A.C. before he arrives home.

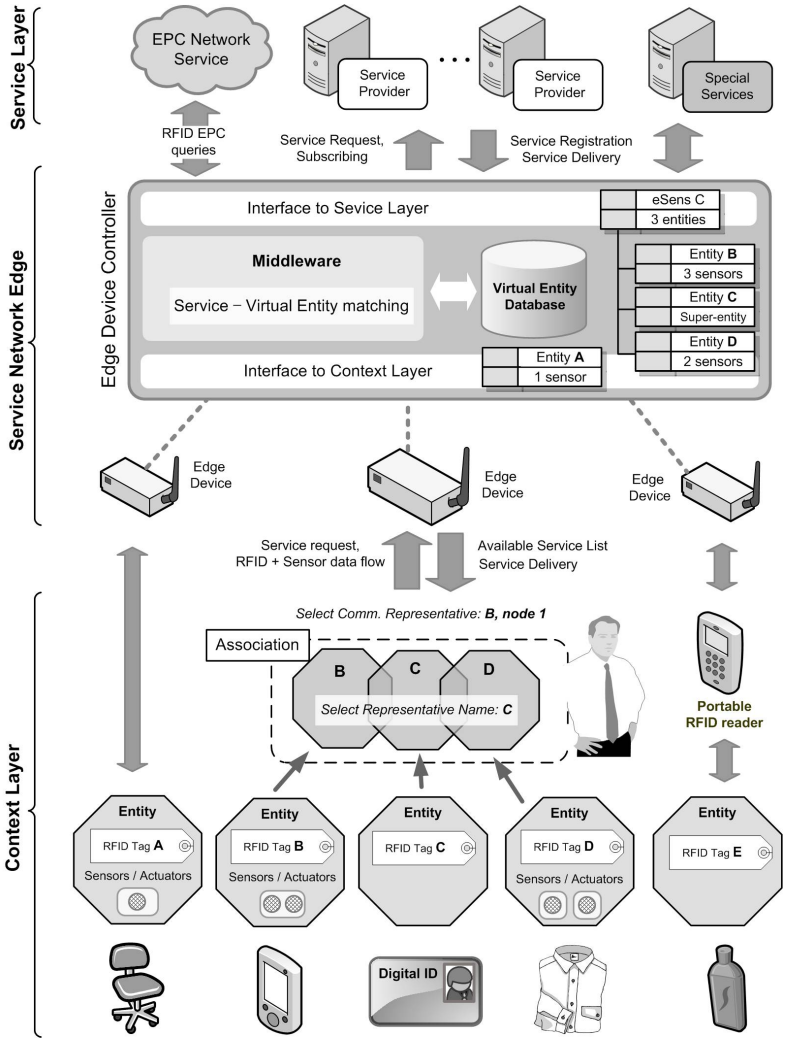

Figure 1: WISSE Framework example

3. Related Work

Although the integration of RFID and sensor data is considered, in general, useful and desirable by the community, there have been very limited attempts to its realization. In the realm of Context Aware systems, for instance, RFID is merely considered as a supplement to the context with no integration with sensor data. Isoda *et al.* [SpatioTemporal(2004)], for example, deduce the user state according to RFID readings of tags located into himself and objects that he carries or encounters. Nonetheless, RFID information is never integrated with other sensor readings of the user space. In [eXspot(2005)], where a Smart Museum concept is implemented, WSN are integrated with RFID readers. However, the WSN nodes are only used to relay RFID codes of users that are navigating through the museum space. Later, those codes will only be used to infer the museum areas where the user has been. In a similar way, Ho *et al.* [Ho(2005)] use WSN and RFID readers to read and relay RFID codes, this time belonging to medicine cases in an elder healthcare application.

There are few works which actually aim at integrating RFID and sensor data. Projects from companies such as HP [SmartRack(2004)] and BP Oil [BP(2006)] use made-on-measure designs to associate RFID codes and simple sensor information. Whilst these strategies do provide a certain integration, their architectures are application specific and are not applicable for the pervasiveness that WSN are envisioned for. On another related area, works such as [Opasjumruskit(2006)], [Cho(2005)] and [Mitsugi (2006)] provide designs for RFID tags including sensors, either passive or battery assisted passive tags. However, their designs stop at the hardware level, without describing how the proposed design could be used to empower a RFID-sensor integrated infrastructure.

The strategies that we follow in order to merge RFID and sensor information have also a big impact in the way the context is gathered and used. Traditionally, in Smart Spaces the user and the context are decoupled in the sense that the main source of information is not from the user's point of view but from the system's point of view. User's context is built entirely in the upper layers (i.e. middleware) from independent sensor readings distributed along the edge of the system. Context information is then interpreted to extract an overall meaning, and actions (services) according to that meaning are taken. This approach has clearly some limitations because the context about a mobile entity will always be limited to its external sensing and to make adequate decisions with this limited information. In WISSE, the context information is obtained from the user's point of view, transferring part of the context reasoning to the user's level by allowing conditional associations of entities before the context is transmitted. We believe that this framework balances the context processing tasks, providing an optimum combination of pre-processing and post-processing of information.

General Context Aware systems such as [Oxygen(2001)] [Gaia(2003)] and [Autonomous(2005)] provide general architectures for context aware services, but they suffer from the limitation of external sensing that we mentioned before. Projects such as [Augemented(2005)] and [Smartness(2005)] also focus on augmenting everyday objects, although they lack a general infrastructure to provide spontaneous services. Furthermore, both sets of related work do not specifically consider RFID information. Finally, to the best of

our knowledge, there exists no previous work in extending the EPC Network infrastructure to provide context aware services.

Our work, thus, departs from the previous work in several areas and has the following main contributions. First, it provides a global infrastructure for the merging of RFID and sensor data. By using this infrastructure, entities provide their context data anywhere - anytime, and the fused data can be accessed in a global manner by the system uses and third parties in a standard, transparent way. Second, the mentioned integration is achieved by a dynamic association of smart entities, as opposed to the static sensor readers from current techniques. The result of these contributions is an integration framework in which RFID and sensor data are collected, stored and mined in a global way, and in which the data and its internal relationships can be used as an input for Context Aware systems.

4. Context Layer

The context layer is the part of the architecture where the context originates. In order to produce this context information, Context Layer's entities are equipped with a single RFID tag and may also carry several sensor nodes with one or more sensors and actuators each. For the RFID tag, we chose the EPCglobal's class 1 Generation 2 (also known as Gen2, a type of passive tags) [Gen2(2005)] and related standards. Gen2's globally unique number, the Electronic Product Code (EPC) [Tag(2005)], is an identification scheme designed to support the needs of various industries by accommodating both existing and new coding schemes. The Gen2 specification also supports an extended in-tag user memory that we will use to store a minimum set of logical information associated with the tag's identity. As we will see later, this information will prove very useful to create an initial integration at the lowest level.

WISSE entities are no physically different from any object that may carry a RFID tag. The difference is functional; while normal RFID tagged objects are just individually read by some RFID reader, WISSE entities may associate and then transmit their information jointly. In order to achieve this, entities use the wireless communication capabilities of their sensor nodes. Grouping is the process by which two or more entities decide to collaborate by sharing their context information. Entities periodically advertise their presence by sending advertisement packets and listening for other entities in periodic, unsynchronized intervals. The grouping process is divided in two phases. The first phase involves the information stored in the RFID tags' user memory space to make filtering decisions and prioritize the association process. The second phase involves choosing both communication and naming representatives, invoking the addressing process and distributing the results to all the group members. When the grouping process is finished, a new entity, composed of all the entities participating in the group, is born. Only then the entities of the group will be aware of their new membership and the results will be communicated to the upper layers.

4.1. Phase 1: Filtering and Classification

In general, the main purpose of the RFID tags is to provide unique identification. In the EPCglobal Class-1 Gen2, that unique identification is given by the tag Electronic Product Code (EPC). The standard also specifies other kind of information that may be stored in the tag's memory. There are four logically separated memory banks in a Gen2 tag. Bank 1 contains the EPC information, while banks 0 and 2 contain other data for security and compatibility reasons. Gen2 also specifies a fourth memory bank called the "user" memory bank. Its organization, size and purpose is said to be user-specific. By using this additional memory to store a small set of logical data, the first phase of the grouping procedure can 1) provide a simple context interpretation from the very beginning, 2) provide priorities when associating multiple entities at the same time and 3) provide a minimum security to allow private groups and prevent unauthorized associations.

We divide the user memory bank in three parts. The first part contains two 16-bit identifiers, a "User ID" and a "Group ID", which unlike the tag's EPC, refer not to the physical product but to its logical use. The group ID is compulsory and shall contain a non zero value for any RFID tag compatible with the WISSE architecture. Its use is to define general object classes such as furniture, human, vehicle, food, book, clothes, etc. The user ID is optional and is intended for user-defined object classes. The second part contains the "restriction bits", a set of bits that dictate the grouping interests of this entity . By using these bits, an entity could, for example, limit its interest to those entities belonging to the same group or the same user. This part also contains a 8-bit level identifier, which represents an estimation of the entity's status in a hierarchical entity structure. For example, those objects which directly represent user's identity, such as ID cards or passports, will be assigned a higher level in the hierarchy. Finally, the third part specifies a user password for preventing unauthorized association to user-defined groups.

WISSE entities broadcasts their EPC and logical information in *grouping_request* packets. Received requests from other entities are organized in classes according to their userID and groupID, representing priorities on the association process. Also depending on the ID values and the restriction bits, some of the requests might be dropped and ignored. Only one class will be processed in one grouping period, while the rest of requests will be queued in a FCFS manner for next processing time.

4.2. Phase 2: Naming and Update

In order to properly deal with dynamic wireless networks, WISSE organizes any combination of associated entities in a double clustered architecture. On one hand, each individual entity chooses a cluster head, which will communicate with other cluster heads from other entities. On the other hand, a group of associated entities chooses a *correspondent* cluster head to communicate with the service network. Both election processes follow a similar algorithm based on [Heterogeneity(2004)] . In the second case, for example, *correspondents* will broadcast a proposed time $T_{corresp}$ based on its remaining energy. Only those entity cluster

heads from its same network that compute a higher time will answer the proposal. The election process is not only triggered by *correspondent* time expiration, but also other events such as addition or removing of entities. By following this procedure, we assure that 1) the *correspondent* node will always have the highest remaining energy in the network and 2) that the lifetime of the network is extended since the overhead of being *correspondent* is shared among all the entities.

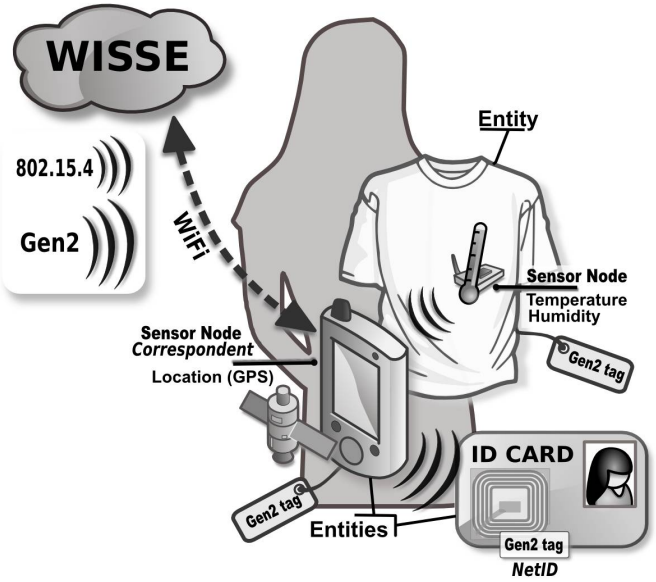

Figure 2: Context Network Example

Entity's *correspondent* is used to broadcast the *grouping_request* packets in the first phase. Correspondents also receive other entity's requests, that they process in order to decide how to associate. In the second phase, the *correspondent* node will analyse the group of highest class requests, ignoring those that were filtered out by the first phase. The main goals of phase 2 are 1) to chose a representative EPC among all the RFID tags of the entities involved in this association, 2) to compute the new network address of the *correspondent* nodes that are associating and 3) to communicate all the individual entities of the final group the results of the association process. When the second phase ends, the representative EPC will become the new network identification (or netID) of this group of entities. The netID will be the number that will identify this group of entities in the system. Figure 3 depicts the algorithms involved in the second phase of the grouping. A detailed explanation of the process can be found in [ISWPC(2007)]

Figure 2 shows an example of a context network, where three entities composed of one node each associate using various communication technologies such as gen2 air protocol, 802.15.4 and 802.11. In this example, the representative election protocol yields the PDA entity as the *correspondent* due to its higher battery power and connectivity with upper layers. The netID EPC, however, will be provided by the ID card's RFID tag, which holds a

higher level than the other entities. Together, these entities will provide location, temperature, humidity and identity information.

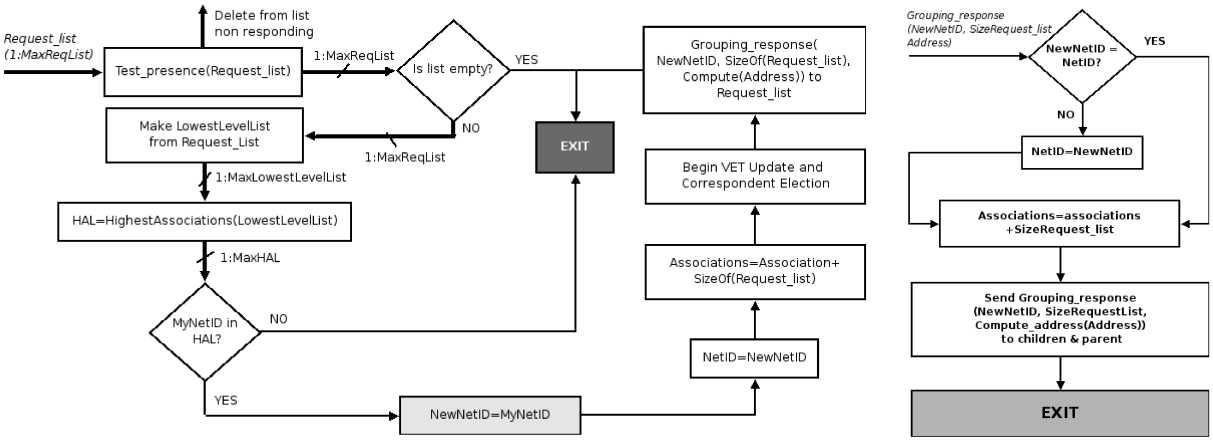

Figure 3: Block diagrams related to the grouping_request and grouping_response packets. The algorithm in (a) will be executed when a list with accepted requests is received from Phase 1. The algorithm in (b) will be executed when the a grouping response packet is received

5. Middleware and Service Layer

After each association process finishes, an updated WSN with new identification, sensor and actuator capabilities is born. This WSN holds dynamically elected gateways, the entity *correspondents*, to communicate with other WSN and upper layers. The second phase of the association process also involves to transfer the new entity's capabilities towards the WISSE network infrastructure. The information transferred by each entity is stored in a database for further processing. This information is organized around the entities' netID, an EPC, which assures a global data search key directly linked with the real entities. The piece of information associated with a particular EPC representing an entity is called Virtual Entity. Each Virtual Entity, thus, includes information of one or more associated entities, such as their RFID tags' EPCs, their sensor nodes or the particular sensors/actuators each node holds.

The main logic of the Service Network Edge is the WISSE middleware (Figure 4). The middleware has direct access to both the Virtual Entity database (VED) and the Service Layer, where the service providers register their services to the WISSE infrastructure. The process involving the first time recording in the database of an EPC and its related

information is called Virtual Entity Registration. Any subsequent variation of that information (such as disassociation of entities, dying of sensor nodes, etc) is called Virtual Entity Update.

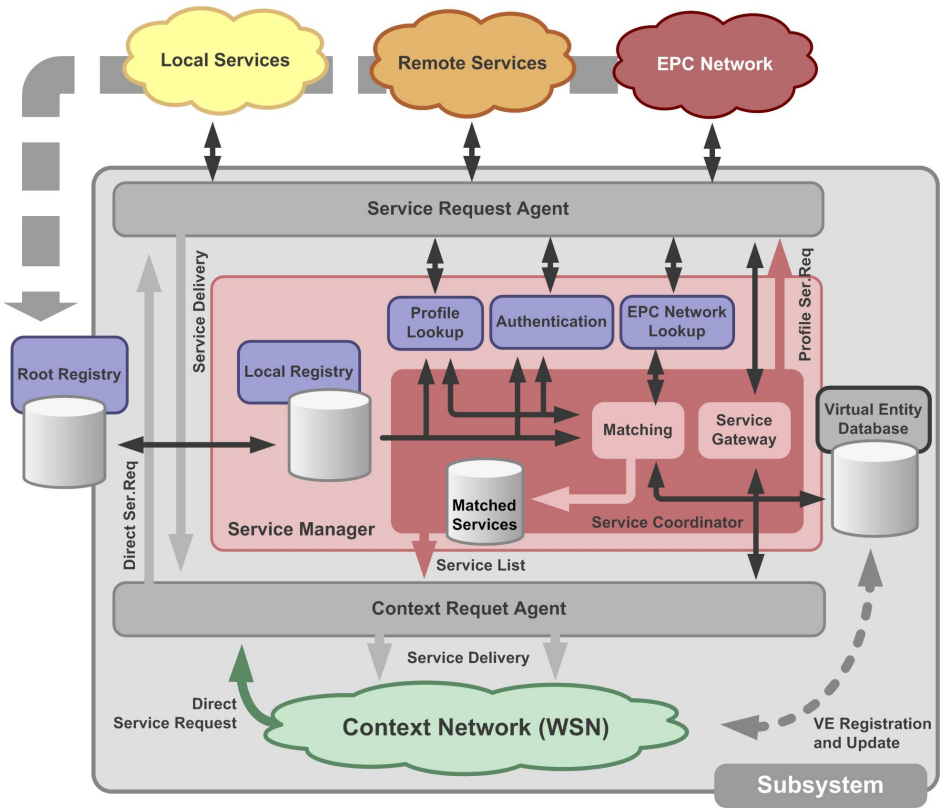

Figure 4: WISSE Middleware

The service providers that register their service in WISSE must provide a standard interface that will be accessed by the middleware. By accessing this interface, the middeware must retrieve the requirements of the service in terms of identity and sensor and actuator types. With the service requirements on one hand and the capabilities of the entities on the other, the middleware performs a matching process to find out which Virtual Entities satisfy the requirements of which registered service. The output of this process is a list of matched services, containing all the services whose requirements are fulfilled by all the Virtual Entities. Any addition of new services or any update in the Virtual Entity Database will trigger a new matching process. This assures a filtered and updated database of which possible services to offer to the clients.

Finally, the entities need to decide which of all the eligible services they are particularly interested in. There are many possible ways for notification, although the simplest way might be to provide a list of possibilities through a PDA or cell phone. In WISSE it also possible the definition of profile sets where users define beforehand which behaviour the entities they own should have when services become available. Profile records are linked to the VED, maintaining references to which EPCs (entities) and which services the profile refers to.

Profile parameters may be combinations of sensors, actuators and identities, first order logic comparisons of their values, and the way a certain service should be executed.

The actual communication between the service provider and the entities (clients) is independent of the WISSE framework. Some examples include 1) downloading the service through a binary file, plug-in script or similar, 2) using RPC (Remote Procedure Call) such as web services or 3) just forward their context information to the service provider servers. For those entities without a direct connection to the service providers, the middleware also provides a gateway module in order to route the information. Other things that the clients may want to define with the service provider are the execution of a specific part of the service (for example, providing some actuation) when a certain condition is met, such as reaching a defined sensor threshold, receiving some sensor event or arriving at a given location.

6. Implementation and experimental results

Besides the gen2 tags, our framework does not make any specific assumption on the technology necessary for its implementation. This rather abstract approach is meant to allow various combinations of technologies that can produce a similar result. In this section, we present one of such combinations that we believe is particularly suitable for the framework.

6.1. Context Layer

Traditional data-centric WSN topologies are relatively easy to debug since they Base Station provides a sink where all the network information is collected. In order to debug the WISSE protocols, where topology and information exchange is dynamic, we built a graphical monitoring software that draws the nodes and the network traffic. The traffic information if collected by a sniffer node attached to the USB/serial port, and it is also logged for future replay. We also used the same GUI to build a context network simulator, aiming for further debugging and performance analysis. In the “simulator” mode, the software can create, schedule and monitor independent WISSE entities and sensor nodes. A battery emulator inside each one of the nodes provides approximate battery values to the representative election protocols. Entities can be also added and removed, triggering the appropriate procedures inside the network. Furthermore, the simulator can be connected to the middleware to effectively provide information as if it would be a real WSN in the Context Layer.

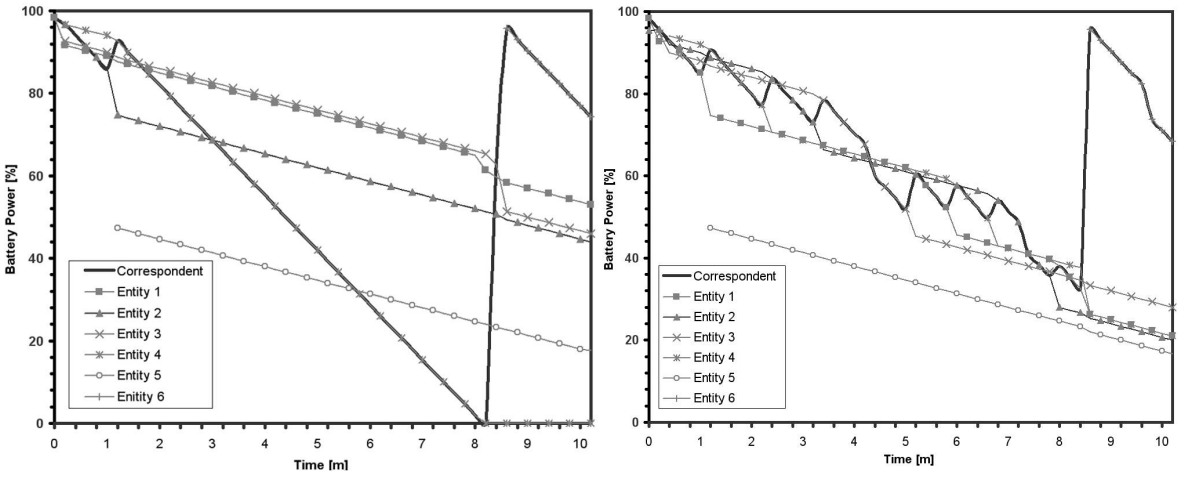

Figure 5: Remaining entity energy with (right) or without (left) activating the dynamic correspondent election

Figure 5 shows the results of simulating a WISSE context network with and without activating the dynamic *correspondent* election protocols explained in section 4. The drain of the battery power was increased in order to keep the simulation time shorter. In this test, four entities are deployed randomly and their start times are also set at random. Two more entities are added one and nine minutes after the start, with 50% and 100% of their remaining battery power respectively. The line marked as “Correspondent” represents the power level of the entity that is *correspondent* at each point in time, and so it moves from one entity line to the other when a *correspondent* election occurs.

Results show that, under the same circumstances, the life of a network (defined as the time elapsed until the first of its nodes exhausts its battery) that implements the dynamic election protocol is considerably longer than a network that doesn't. In general, WISSE context networks using the dynamic election at the *correspondent* level live up to a 53% more than those who doesn't. We also performed further simulations altering several variables such as the number of nodes per entity and the representation period. Our conclusions are that the representation period of a node can be tuned dynamically according to the number of nodes of the entity. This property is specially suited for the WSN introduced in our work, since the number of entities per network can vary with time, and the representation period for a given *correspondent* must be adapted to these circumstances. After this evaluation, we consider that the presented protocols show promising results for the dynamic association of entities discussed in this paper.

6.2. Middleware and Service Layer

Although the process of providing a real integration inside the EPC Network goes beyond a database redesign, it is also true that the storage of information is one of the key steps towards data sharing. To this extent, we decided to implement the WISSE's VED in the form of a modified EPC Network repository (EPCIS repository [EPCIS(2005)]), where the relationship among entities is expressed in the same terms as the containment relationship among pallets, boxes or individual products.

Web services [WebServices(2004)] are the paradigm of interoperable client-server interaction over the Internet, and are the core technology of many Internet based businesses and infrastructures such as the EPC Network itself. In our implementation, we use Web services to register and discover WISSE services (Universal Description Discovery and Integration - UDDI), to specify their interfaces (Web Service Description Language - WSDL) and to communicate clients and servers (Simple Object Access Protocol - SOAP). Since the services must specify not only their interfaces but also the requirements for the Virtual Entities, we introduce an additional specification, called the Service Definition File (SDF), binded to the UDDI registry. The SDF uses XML to define which requirements in term of sensors, actuators and identity a client must meet. For the identity requirements, the middleware's matching module may need to contact the EPC Network and ask the manufacturer's repository about the details of a particular EPC.

Figure 6 shows a screenshot of the simulator (left) and the middleware monitor (right) running a specific simulation. In the Simulator, only the *correspondent* of each entity and the serial part of its EPC are directly shown to keep the visualization cleaner. However, the other nodes of that entity are also accessible via a context menu. In this screenshot, entity 2 is both *correspondent* and netID of the group, while node's 3 information is shown on the right side. In the middleware monitor, the group hierarchy inside the VED is shown both in tree and graphical manner, while the services available for this particular group are shown at the bottom.

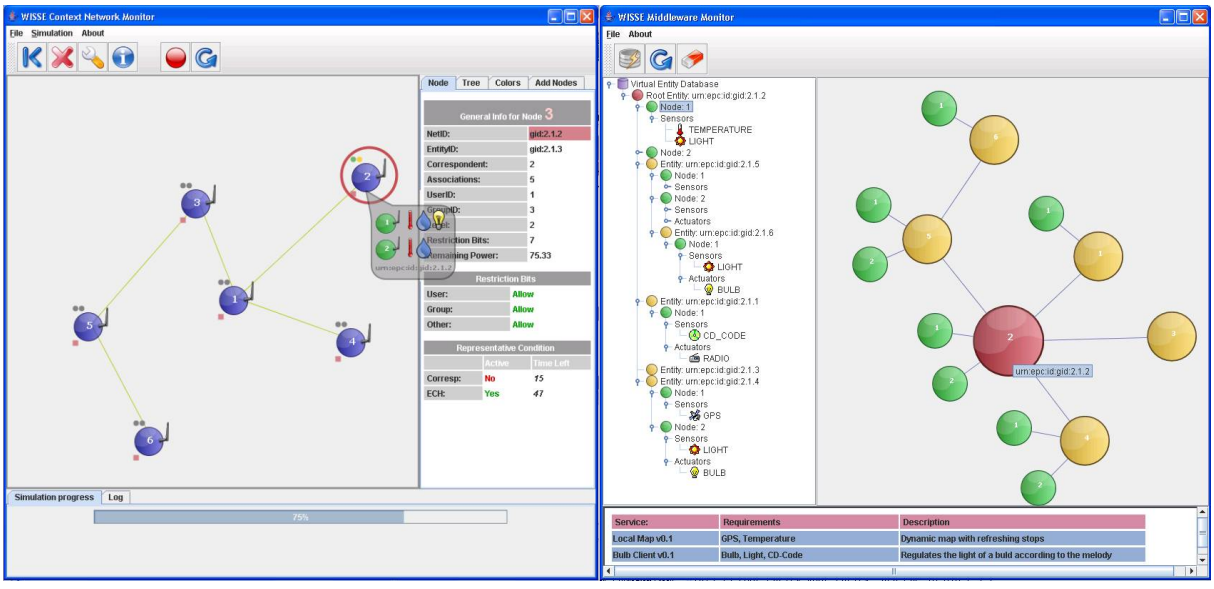

Figure 6: Context Network simulator and monitor (left) and Middleware monitor (right)

6.3. Real-world experiment

Based on the implementation details outlined previously, we built a scenario that offers three different services according to the dynamic capabilities of several context layer entities. For this scenario, we use a sensor node with a temperature and humidity sensor and a laptop with a PCMCIA RFID reader. The sensor node is our development [ANTS(2005)] , and its main features are an Atmega 128 micro-controller (128KB of Flash, 4 KB of RAM), CC2420 as RF transceiver (2.4 Ghz, 802.15.4 compatible) and various sensors included in a separate pluggable board.. The laptop is connected through the USB/Serial to another sensor node to allow WSN communication. Additionally, we also use the simulator to artificially add several entities to the Context Layer. The implementation scenario is depicted in Figure 7. At first, only the laptop, with no additional sensors, is used. This results in a service offering simple product information according to the RFID tag that is read. Secondly, the laptop and the sensor node are grouped. This time, a new service is matched which warns the user if the tag that is being read has temperature constraints and a too-low or too-high temperature is detected. In this case, the laptop will be selected *correspondent* due to its higher power. Finally, a first level RFID tag, belonging to a user's digital identification card, is added, which results in a new service that was previously specified as a profile. In this case, the result is a remote actuation service that simulates the starting of the A.C at the user's smart home.

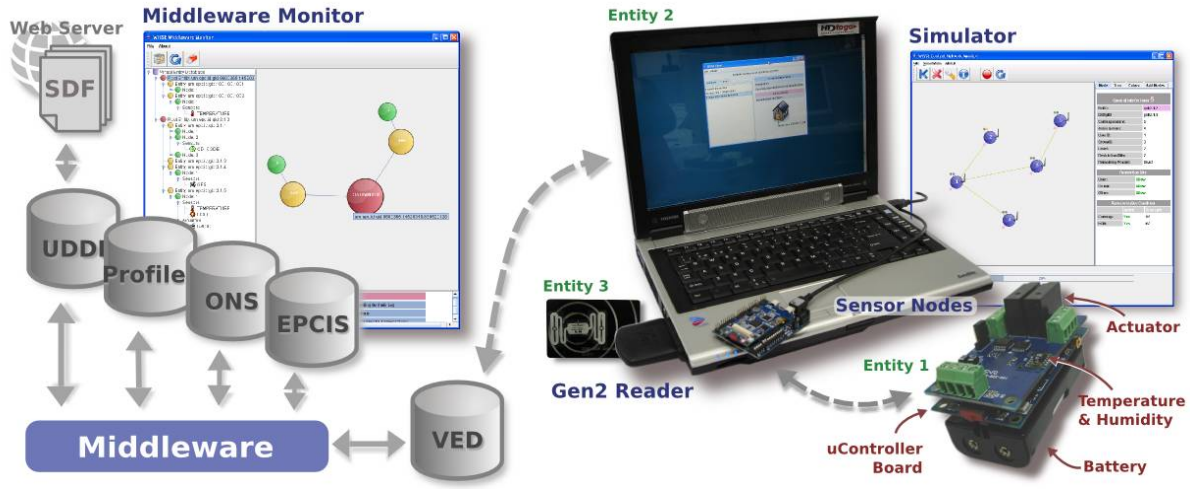

Figure 7: Implementation Scenario of the WISSE framework

The implementation presented here gave us the opportunity to make an initial assessment of each one of the components of the WISSE framework. On the hardware side, the main challenges arise with the sensor nodes' and active tags' constrains. As exposed earlier, in our implementation we use a 4KB RAM micro-controller. This reduced memory size is not enough to store the entire implementation of our Context Layer protocols, since buffers, timers, and drivers (such as sensor, actuator and RF) have to be accommodated. However,

we believe that these constraints can be easily overcome with the evolution of the technology. We are already considering the use of other micro-controllers, such as the MSP430, with more than double of RAM memory. On the software side, we believe that existing technologies can be used to a great extent to satisfy the requirements of our framework. However, we also think that special care should be taken into developing clear and efficient interfaces to link the different components. Another key factor on the efficiency of the infrastructure is the organization of the Virtual Entity Database. In the presented implementation, we focused on the software functionality in order to emphasize the viability of our proposal. In the future, however, we would like to focus on efficient communication, interoperability and performance.

7. Conclusion

In this article, we propose a framework for the integration of RFID and WSN in order to offer context aware services to users and objects. To the best of our knowledge, our contribution constitutes the first work in which sensor and RFID data merge to build dynamic context, and in which all the architecture around this context is designed to be compatible with the current EPC Network infrastructure. To prove the viability of our design we propose an implementation scenario using both real and simulated WSN, Web services and EPCIS-like repositories. Finally, we also provide software tools for monitoring and evaluating our algorithms. In concrete, we show that the WSN algorithms used in our work have an excellent performance and are particularly suitable for the framework.

References:

ANTS(2005): D. Kim, T. Sanchez Lopez, S. Yoo, J. Sung, “ANTS platform for Evolvable Wireless Sensor Networks”, The 2005 IFIP International Conference on Embedded And Ubiquitous Computing (EUC'2005), LNCS, Nagasaki, Japan, December 2005.

Augemented(2005): F. Kawsar, Kaori Fujinami, Tatsuo Nakajima, “Experiences with Developing Context-Aware Applications with augmented artefacts”, Ubicomp 2005, Tokyo, Japan

Autonomous(2005): Kiran Kumar, Salim Hariri, Nader V. Chalfoun, “Autonomous Middleware Framework for Sensor Networks”, ICPS 2005, 11-14 July 2005

BP(2006): Jonathan Collins, “BP Tests RFID Sensor Network at U.K. Plant”, RFID Journal Online, June 21, 2006

Cho(2005): Narrijun Cho; Seong-Jun Song; Jae-Youl Lee; Sunyoung Kim; Shiho Kim; Hoi-Jun Yoo, “A 8-/spl mu/W, 0.3-mm/sup 2/ RF-powered transponder with temperature sensor for wireless environmental monitoring”, Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on 23-26 May 2005 Page(s):4763 - 4766 Vol. 5

EPCIS(2005): EPC Global Working Draft, “EPC Information Services (EPCIS)”, Version 1.0, September 2005

EPCnetwork(2005): Ken Traub et all, “The EPC Global Architecture Framework”, EPCglobal, July 2005

eXsport(2005): Waylon Brunette, Jonathan Lester, Adam Rea and Gaetano Borriello, “Some sensor network elements for ubiquitous computing”, IPSN 2005, 388-392, April 2005

Gaia(2003): Anand Ranganathan and Roy H. Campbell, “A Middleware for Context-Aware Agents in Ubiquitous Computing Environments”, Middleware 2003, LNCS 2672, pp. 143/161, 2003.

Gen2(2005): EPCglobal Inc, “Class 1 Generation 2 UHF Air Interface Protocol Standard Version 1.0.9: “Gen 2””, Ratified Standard, January 2005.

Heterogeneity(2004): Srikanth Kandula, Jennifer Hou, Lui Sha, “A case for Resource Heterogeneity in Large Sensor Networks”, in Proceedings of MilCom 2004

Ho(2005): Loc Ho, Melody Moh, Zachary Walker, Takeo Hamada and Ching-Fong Su, “A prototype on RFID and sensor networks for elder healthcare: progress report”, Proceeding of the 2005 ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis, August 2005

ISWPC(2007): T. Sanchez, D. Kim, K. Min, J. Lee, “Dynamic Context Networks of Wireles Sensors and RFID tags”, ISWPC 2007, San Juan, Puerto Rico, 5-7 February, 2007

Mitsugi(2006): Jin MITSUGI, “Multipurpose sensor RFID tag”, APMC 2006 workshop on Emerging Technologies and Applications of RFID, WS04-4, 2006., pp.143-148

Opasjumruskit(2006): Opasjumruskit, K.; Thanthipwan, T.; Sathusen, O.; Sirinamarattana, P.; Gadmanee, P.; Pootarapan, E.; Wongkomet, N.; Thanachayanont, A.; Thamsirianunt, M. “Self-powered wireless temperature sensors exploit RFID technology”, Pervasive Computing, IEEE Volume 5, Issue 1, Jan.-March 2006 Page(s): 54 – 61

Oxygen(2001): Larry Rudolph, “Project Oxygen: Pervasive, Human-Centric Computing - An Initial Experience”, LNCS Volume 2068/2001, Springer

Smartness(2005): H.W. Gellersen, Albrecht Schmidt and Michael Beigl, “Adding some smartness to devices and everyday things”, WMCSA'00, 2000

SmartRack(2004): Aloire Gilbert, “HP developing 'smart rack' to ease data center work”, News.com, October 29, 2004

SpatioTemporal(2004): Yoshinori ISODA et al., “Ubiquitous Sensors based Human Behavior Modeling and Recognition using a Spatio-Temporal Representation of User States”, AINA’04, Valume 1, p. 512, 2004

Tag(2005): EPCglobal Inc, “EPC™ Tag Data Standards Version 1.3”, Standard Specification, July 2005.

WebServices(2004): Gustavo Alonso, Fabio Casati, Harumi Kuno, Vijay Machiraju, “Web Services: Concepts, Architectures and Applications”, 2004, ISBN: 978-3-540-44008-6