Self-Sensing Spaces: Smart Plugs For Smart Environments

Hicham Elzabadani, Abdelsalam (Sumi) Helal, Bessam Abdulrazak and Erwin Jansen

Computer and Information Science and Engineering Department
University of Florida, Gainesville, FL-32611, USA

Abstract: In the past few years the field of pervasive computing has expanded and infiltrated down to everyday consumer activities. Smartness of devices is constantly increasing. Smart products are widespread; not just limited to appliances and electronics. With the availability of such products, there becomes an increased need for a scalable smart environment. The need for real world modeling of such environments is becoming increasingly important. It is very useful to be able to discover the location of a certain device, reason about it, and finally be able to control it remotely. Such self-sensing and delegation of control is needed by many applications. In this paper, we discuss the idea of self-sensing spaces - spaces that can sense themselves in terms of locating and controlling its contents. In addition, we discuss the implementation of smart plugs; a novel way to locate and control smart devices within an real smart house.

Key words: Pervasive computing, Smart Environment, Smart Plugs, Self-Sensing Spaces, Real World modeling, and Remote monitoring.

1. Introduction

Pervasive computing research provides users with the ability to perform computations that are available at any time, any means, and any place. A pervasive space consists of a collection of devices and objects. Within this space there are: "sensors," devices that sense the environment, "actuators," devices that can change the environment and static objects that never change (like furniture, walls, etc.). Many of the devices and applications in the pervasive space require some kind of remote monitoring and intervention. Objects of importance like humans, patients, bombs, and disasters need to be monitored from time to time. To this end, we envision a smart space, such as a home, that is capable of sensing the residents and itself. The space has a model of the world that can be used by remote monitoring and intervention services. The model of the world must at all times reflect the real world as closely as possible.

With current technology, this is however not feasible. The obstacles we face are:

- **Scalability**: This is a major problem when it comes to smart spaces. It is still impossible to upgrade a smart space easily. Introducing a new device requires a significant amount of engineering. A smart home resident for example should not call a technician every time he/she wants to install a new lamp. The smart home

should be smart enough to know that there is a lamp being installed and it should know exactly how to control it.

- **Lack of Interface:** Most of the smart appliances available in the market today do not contain an interface that allows a smart space to control the appliance.
- **Protocols:** Different protocols used for interaction with devices. Many devices that do have some form of interface use their own proprietary protocol of interaction. X10 for example is an easy, affordable way to control simple appliances. It becomes difficult if you bring a smart device that is not X10 enabled. Regardless of the technology used, a smart space should be able to communicate with any new smart device.

To overcome those obstacles, we try to incorporate computational devices in our physical environment to assist the user accomplish his/her daily tasks. Our end goals are self-integration of new devices, locating and identifying objects of interest to the monitoring/intervention application, and creating privacy-preserving real world models. The idea is similar to the PC's Plug-and-play system (PnP) which matches up device drivers with the various devices that are available in a computer. After discovering these devices, it initializes the devices to allocate resources. Once this has been taken care of, it is up to the operating system to load the appropriate device drivers.

In this paper, we present our smart plug concept, an intelligent way to sense new devices installed in a smart space. The example of power outlets and electrical devices is given. The rest of the paper is organized as follows. The next section talks about pervasive remote monitoring and intervention. Section 4 gives an overview of the technology we used. Section 5 explains how the Smart-Plug is implemented. In section 6, we discuss the implementation in the "Gator Tech Smart house" and finally we give a conclusion and future work.

2. Self-Sensing Spaces

Nowadays, the emergence of new technologies in the field of computer science and engineering give us the opportunity to achieve our goals in the field of self-sensing spaces. Pervasive systems are becoming part of our daily life, people are dealing with technologies such as wearable computers, smart homes (that can control temperature gauges, control lighting, or program a home theatre system), speech and gesture recognition sensors, optical switching devices and embedded sensors networks [1, 2]; A new platform such as OSGi (**O**pen **S**ervices **G**ateway **I**nitiative) facilitates the installation and operation of multiple services on a single device or gateway (an in-vehicle information system, smart handheld, set-top box, entertainment platform, multimedia or residential gateway). Finally, combining new technologies, like real world modeling and location tracking changes the way remote monitoring is being used. For example, caregivers will be able to locate a senior resident and check any suspected problems in the case of emergency.

Real world modeling refers to the capture, reasoning, and organization of objects of interest. It consists of two major components: location and context awareness [3]. Any object in the space could be modeled if we know exactly what this object represents and where it is located.

Despite the progress made so far with respect to real world modeling, many challenges, like managing complexity and scalability, transient environments and aggregation of sensor data, and privacy and security, must be overcome [4].

Our primary goal in the field of self-sensing spaces is to be able to sense any changes made in the space. For example, the smart space should be able to recognize that a new lamp is being installed; moreover, it should know the lamp's location and finally it should be able to control it. Since most of the appliances need electricity to operate, we use the power plug of the appliance to locate it. We call it the smart plug and it works as follows: First we locate all power outlets in the space, then we sense when a new device is plugged in any of the registered outlets. We recognize a plugged device using RFID (**R**adio **F**requency **Id**entification). Finally we use OSGi to register these devices in the framework [5-7]. In the next sections we discuss the methods and the techniques used to implement the smart plug idea.

3. Techniques / Technologies Used

3.1. Sensing New Devices

Every space (house, office, etc.) contains 2 types of objects. The first type consists of static (powerless) objects like tables, chairs, entertainment center, etc. We call them static because their status never changes. The other type consists of dynamic (powered) objects like lamps, radios, TVs, etc. We call them dynamic because their status changes frequently. The only property that is shared between the two types is location. Locating those objects is one of the major goals that we are trying to achieve in the self-sensing spaces project. Although we are working on both types, we will only deal with dynamic objects in this paper.

3.2. Locating Power Outlets

We assume, in this paper, that the smart space knows about all existing power outlets and their exact location. This can be done either manually (by measuring the distance between each power outlet) or automatically using laser measuring techniques. This technique will be used once to draw the floor plan of a space, then to locate each power outlet. This project is one among the few steps that we are working on to achieve the goal of a complete self-sensing space project.

3.3. Discovery of New Devices Using OSGi.

The current version of the OSGi specification that we are using is 3.0. Beside the framework, the specification includes several other basic services such as HTTP service, log service, configuration management, permission administration, preferences, user management, and device manager. The specifications of services do not cover all scenarios in a real home network. Consider the example of connecting a new device to the home network. The device manager is normally responsible for finding and downloading a driver for the new device, but the exact procedure depends strongly on the protocol used.

When a new device is installed in the home network, it must be detected. There are two ways to handle detection of a new device:

- A discovery protocol

- A tool for manual configuration of the system

The second approach is not of our interest because self-sensing spaces should run with zero-configuration, meaning that we cannot ask an engineer to come and upgrade the system every time we install a new device.

If a protocol with discovery features is used (which is the case in our research), the device will announce its presence after being connected to the home network. The system detects the device, assigns a device ID, and through the device manager downloads the driver from the device [8].

4. Implementation

4.1. Smart Plugs for Self-Sensing Spaces

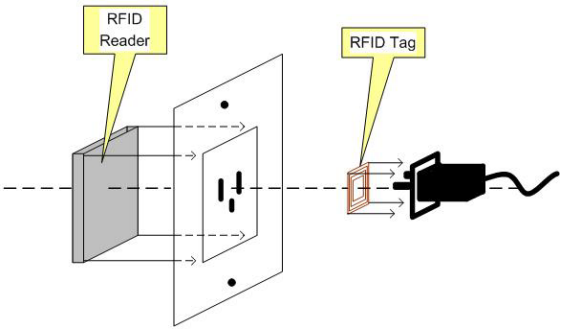

Figure 1. Outlet equipped with a low cost RFID reader and plug with a RFID tag.

Power outlets in self-sensing spaces are equipped with low cost RFID readers connected to a main computer. Electrical devices (equipped with power cords) like lamps or radios have an RFID tag on the plug. This tag contains information about the device (Figure 1). Whenever plugged into the outlet, the reader reads the tag, gets its contents, and sends it to the main computer which is an OSGi platform. Using this information, the main computer can directly identify this device.

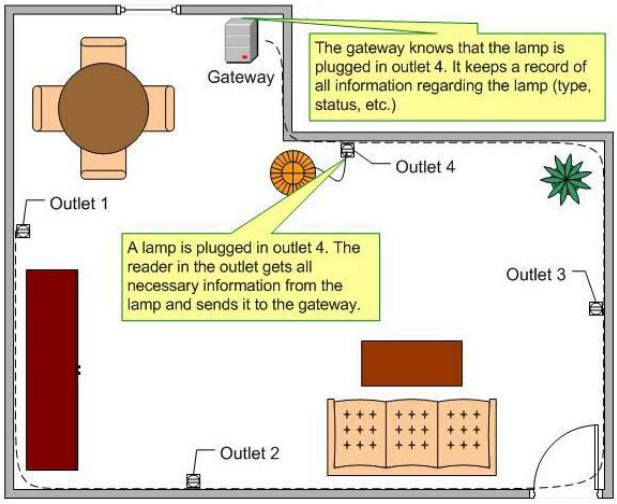

Figure 2**.** Power outlets locations in a room, detection and localization of a new device is plugged by getting the ID of the power outlet where it was plugged.

Knowing the location of each power outlet, the computer will know approximately where the new device is plugged in. Moreover, the computer will be able to control the new device using the information stored on the tag (Figure 2).

As we presented, the smart plug concept consists of two major parts: OSGi and RFID.

4.2. OSGI (Middleware)

We have developed a middleware layer (based on OSGi) that consists of several bundles that provide a framework for pervasive computing. The OSGi framework allows us to represent every physical device (sensor as well as actuator) as an individual bundle. Apart from bundles for physical devices, we have the following additional bundles:

- Floor plan: This bundle contains all the location information of the house. It contains a floor plan of all the individual rooms and position information of every device available. It is able to map coordinates to rooms and vice versa. This allows us to location based service discovery. For example, we can request every device in the kitchen.
- Sensor: This bundle contains a framework for automated installation of sensors. We make extensive use of the wire admin API. Every sensor is a producer that can produce a value.
- Context: The context bundle interprets sensory data and turns this into a high level representation. The context framework can also (de)activate behavior upon a change of observed state. For example, if the temperature changes from warm to cold we can turn on the heater.
- X10Controller: This bundle gives us basic access to X10 controlled devices.
- Resolution: This bundle takes care of installing a bundle associated with a smart plug. It also registers the new bundle with the floor plan.

New devices installed, in the smart space, are represented by OSGi bundles. A bundle is simply a Java archive file (jar) containing interfaces, implementations for those interfaces, and a special Activator class. The manifest file contained within the jar file includes special OSGi specific headers which control how the bundle will be used within the framework.

4.3. RFID

The core of any RFID system is the 'Tag' or ‘Transponder’ that can be attached to or embedded within objects. A RFID reader sends out a radio frequency wave to the 'Tag' and the 'Tag' broadcasts back its stored data to the reader. The system works basically as two separate antennas, one on the 'Tag' and the other on the reader. The data collected from the 'Tag' can either be sent directly to a host computer through standard interfaces, or it can be stored in a portable reader and later uploaded to the computer for data processing [9].

In self-sensing spaces, whenever an RFID reader, which represents a proxy between the device and the gateway, associated with the smart plug reads a tag, the context framework activates the resolution bundle. The resolution bundle will then inform the OSGi framework to load the bundle and activate it. When the bundle is activated it is the bundle’s job to register a service that allows access to the device that has been installed.

Depending on the size of the RFID tag’s memory, the bundle representing the new device to be installed in the smart space could be stored on the tag itself. If the OSGi bundle is too large, it is possible to just store a referral URL to where the gateway software can download it from. The referral URL can use any protocol, which the OSGi framework server has access to, such as http and ftp. Size might also not be the only reason why the referral concept would be used.

By having the software on a web server, upgrading the bundle is as easy as replacing the software. All the required downloading and installation of the gateway software, for the individual bundles, are performed by the gateway bundles installed in the framework. In case a URL is supplied on the tag, the framework can access bundle data using the bundle

5. Installation in the Gator Tech Smart House

Converting regular houses into smart ones is still not an easy task that any resident can do. People need to hire engineers to do the job. That is why we are working on creating a “smart house in a box” where anyone can buy a smart house kit and either installs it or hires a technician to do the job.

In order to use the smart plug in the Gator Tech Smart House[1], we had to create a middleware device that will transform a regular plug into a smart one as shown in Figure 3. Each power outlet in the Gator Tech Smart House is equipped with a Phidget RFID reader [10], all connected through USB to the main computer [11].

[1] Gator Tech Smart House is a real house in Gainesville, Florida.

Figure 3. The smart plug used in the Gator Tech Smart House.

The main computer contains the OSGi framework, which sits on top of a Java virtual machine, and is the execution environment for services. This computer could be an OSGi set-top box that plays the role of a gateway between the smart space and the external world. Each bundle will be downloaded and registered in the OSGi framework when the new device is installed in the smart space. The framework will be able to report the list of installed devices upon request. Any device within that list could be controlled via methods available in the bundle. Suppose that a new table lamp is brought to the house, all what need to be done is to attach a smart plug into the lamp's plug and plug it into any outlet in the house. The RFID reader of that specific outlet will read the content of the RFID tag and send it to the main computer.

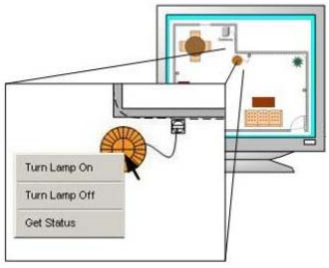

Figure 4. Information will be downloaded to the remote application on the fly. A click on the lamp will import all the available methods from the gateway.

The user can control the lamp remotely by a simple interaction with a graphical interface (Figure 4). A click on the lamp will download all available methods associated with this lamp. If the user clicks on a method, the remote application will send a request to the gateway to execute the action. On his side, the gateway will translate the request and execute the corresponding method.

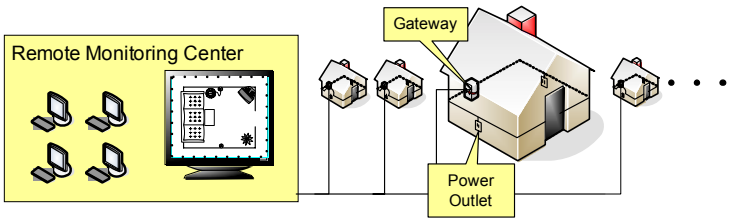

Figure 5. Self-Sensing Spaces in control center concept

One of our goals is to enable self-sensing spaces to connect to a remote center in case of emergency (Figure 5). A gateway will act as a proxy between the smart space and the center. This center will monitor and intervene in case of an emergency. Operators at the center will be able to better differentiate between false alarms and real emergencies which will reduce the number of unnecessary visits.

6. Conclusion

In this paper we presented smart plugs for self-sensing spaces. The use of RFID technology in smart spaces makes installing new devices a simple, automatic and scalable task. There is no need to reprogram a smart space anymore every time we need to install a new device. A smart space will be able to automatically integrate the new device upon installation.

Currently we are using RFID tags and readers to identify new devices. We could use the same idea with different, possibly cheaper, technology. Every plug could for example send a specific code using the X10 protocol when the device is plugged in. Another option would be to make use of Near Field Communication (NFC) [12, 13]. NFC is a short-range wireless connectivity standard that uses magnetic field induction to enable communication between devices when they're attached together, or brought within a few centimeters of each other

The next step in our research is locating and identifying static objects like furniture.

When completed, a smart space will be able to identify, locate and control most of its contents. At the same time we will be working on enhancing the remote monitoring and intervention application to support the new additions.

7. References

[1] H.-W. G. A. Schmidt and M. Beigl, "Adding Some Smartness to Devices and Everyday Things," presented at 2000 Third IEEE Workshop on Mobile Computing Systems and Applications, 2000.

[2] H. Gellersen, G. Kortuem, A. Schmidt, and M. Beigl, "Physical Prototyping with Smart-Its," *Pervasive Computing, IEEE*, vol. 3, 2004.

[3] M. Bauer, C. Becker, J. Hahner, and G. Schiele, "ContextCube - providing context information ubiquitously," presented at 23rd International Conference on Distributed Computing Systems, 2003.

[4] A. Ferscha and J. Johnson, "Distributed Interaction in Virtual Spaces," presented at 3rd IEEE International Wokshop on Distributed Interactive Simulation and Real-Time Applications, 1999.

[5] R. Want, "Enabling Ubiquitous Sensing with RFID," *Computer, IEEE*, vol. 37, pp. 84-86, 2004.

[6] D. Marples and P. Kriens, "Open Services Gateway Initiative: an introductory overview," *Communications Magazine, IEEE*, vol. 39, pp. 110-114, 2001.

[7] OSGi, *Osgi Service Platform, Release 2*: Ios Pr Inc, 2002.

[8] C. Lee, D. Nordstedt, and S. Helal, "Enabling Smart Spaces with OSGi," *Pervasive Computing, IEEE*, vol. 2, pp. 89-94, 2003.

[9] K. Romer, T. Schoch, F. Mattern, and T. Dubendorfer, "Smart Identification Frameworks for Ubiquitous Computing Applications," presented at First IEEE International Conference on Pervasive Computing and Communications, 2003.

[10] Phidgets, "http://www.phidgetsusa.com."

[11] A. Helal, W. Mann, H. Elzabadani, J. King, Y. Kaddoura, and E. Jansen, "Gator Tech Smart House: A Programmable Pervasive Space," *IEEE Computer Magazine*, pp. 64-74, 2005.

[12] S. ECMA-340, "Near Field Communication - Interface and Protocol (NFCIP-1)," *ECMA*, December 2002.

[13] Near Field Communication, "http://www.nfc-forum.org."