Super Spaces: A Middleware for Large-Scale Pervasive Computing Environments

Jalal Al-Muhtadi, Shiva Chetan, Anand Ranganathan and Roy Campbell
University of Illinois at Urbana-Champaign
{almuhtad, chetan, ranganat, roy}@cs.uiuc.edu

Abstract

In previous work [1], we introduced a distributed middleware infrastructure for managing physically-bounded, standalone pervasive computing environments, which we refer to as Active Spaces. In this paper, we extend this middleware to support the organization, management and interactions of large-scale pervasive computing systems built from multiple Active Spaces or other pervasive computing components.

1. Introduction

Pervasive computing promises to revolutionize the way humans interact with machines, physical spaces, services, and other humans. Pervasive computing envisions a world in which a plethora of embedded devices, sensors, and digital communication components are woven into every day's physical spaces. These physical spaces are then transformed into context-aware information-rich "pervasive environments", which can take on the responsibility of serving users by tailoring themselves to users' preferences and performing tasks and group activities according to the nature of the spaces and their contents and purposes. As a result, users can focus on the tasks at hand, minimizing distractions, and allowing machines to learn, adapt, and facilitate users' interactions with their surroundings.

In our current project, we have built a distributed middleware to enable the construction and management of pervasive environments. In this paper, we refer to these pervasive environments as "Active Spaces" and view them as the basic elements of large-scale pervasive computing. An Active Space is a physical space coordinated by a responsive context-based software infrastructure that merges the computational infrastructure with the physical infrastructure and allows users to interact seamlessly with the surrounding environment. Our infrastructure enables the construction, management and development of standalone, physically-bounded Active Spaces. Our infrastructure is the result of three years of work and use, during which we were successful in deploying it in several different spaces and scenarios including meetings, seminars, classrooms, and home environments [1, 2]. However, in its current form, an Active Space is standalone and its interaction as a whole with other Active Spaces is limited. In the near future, we expect that processing power and communication networks will become inexpensive commodities that are available in large quantities everywhere. This will allow pervasive computing to triumph, opening the door to wide-scale deployment. In such a scenario, pervasive computing will no longer be restricted to several "prototype" rooms. Instead, a whole building with all of its rooms, facilities, and corridors or a campus with all of its buildings and courtyards will become pervasive. As a result of our experiences in using Active Spaces, we find that the promise of large-scale, wide deployment of pervasive computing cannot be realized without cost-effective and efficient mechanisms and policies to organize, manage, operate and repair large-scale systems built from Active Spaces or other pervasive computing components. This forms the basis for arguing in favor of introducing a higher-level of abstraction for managing and maintaining groups of pervasive computing elements, and extending our middleware infrastructure to support this view. For this purpose, we introduce "Super Spaces." A Super Space is a collection of reflective and recursive Spaces (Active Spaces or other Super Spaces) that enable large-scale management, operation, and maintenance of pervasive computing environments.

In this paper we argue about the importance of managing and orchestrating groups of Active Spaces. We give a high-level description of how we are extending our middleware to support Super Spaces.

The remainder of the paper is divided as follows. Section 2 motivates the need for Super Spaces by describing an usage scenario. Section 3 talks about the challenges. Section 4 describes our middleware architecture. Section 5 discusses how the proposed architecture can support some Super Space applications. Section 6 mentions some related work briefly. Finally, Section 7 concludes.

2. Scenario

Kevin is a cardiac surgeon who works for a city hospital. The hospital is equipped with various digital devices interconnected to form Active Spaces and Super Spaces. Each patient room is implemented as an Active Space. Each room has a wall mounted screen as well as interactive displays such as the Everywhere Displays[3]. A camera fitted to the ceiling is used to detect the patient's condition and state (e.g. lying on the bed, lying on the ground or walking). The room is fitted with a few wireless microphones and speech recognition applications. RF badges are used for building-wide location tracking. Corridors and other rooms are also implemented as Active Spaces. Each Active Space is part of a floor Super Space. Each floor Super Space is a child of the building Super Space.

When Kevin enters his office in the morning, the office Active Space detects him, greets him and displays his agenda on his wall-mounted display device. Kevin wants to visit some of his patients in the morning. His scheduler application receives patient information from the hospital database, which has been updated overnight based on the patient's conditions, and schedules his visits. The application gets information about the layout of the hospital and the locations of various patient rooms from a building-layout service. It then optimizes Kevin's path and also schedules the time he needs to spend with each patient. This information is loaded into Kevin's PDA that he carries with him all the time. The application then notifies all patient room Active Spaces about his visit time. The Active Spaces decide on the best means of notification and inform patients on their display devices or speakers. If a patient is not in her room, the Active Space locates the patient through the building location service and informs the appropriate Active Space to notify the patient. When Kevin enters a patient's room, the devices get configured for his requirements. Applications such as music or movie players are suspended and the patient's case history is displayed on the wall-mounted display device and a speech-to-text application pops up and configures a microphone to take notes. When Kevin finishes checking the patient, he updates the patient history using the interactive display. When he runs out of time, his PDA buzzes to inform him of the next patient to visit. After Kevin leaves the room, suspended applications are resumed.

3. Challenges

In order to achieve the above scenario, several significant challenges have to be overcome. Some of them are:

3.1. Scalability

It is clear that it is not possible to have just a single centralized Active Space to manage an entire building. For example, it would be difficult to have just one centralized repository or registry that keeps track of all devices and services present in the whole hospital building and their locations and properties. Such a repository would not scale. It is thus essential to employ a distributed architecture, for example, a federation or a hierarchy of repositories. In other words, we need to have several Active Spaces that manage separate parts of the building (such as rooms and corridors) and then organize these spaces in some manner.

3.2. Discovery and Inter-operability

The problem with having multiple Active Spaces is that they (and the services in them) have to interoperate in various ways. Services and applications in one space may need to discover the presence of other spaces and their services, contexts, and applications. Different components of the same application may be spread across different spaces. Different services across different spaces may have to exchange information of various kinds. For example, the scheduler application running on Kevin's handheld needs to communicate information about the schedule to displays in the different patient rooms.

3.3. Organization and Operation of Services

Each space may have several services running in it such as location, naming, event and context services. These services would typically have information about or provide services within the space in which they are running. However, it is necessary for similar services across spaces to communicate and collaborate for performing tasks that are spread across a wide area. For example, naming or discovery services running in different spaces would need ways of exchanging information in order to allow entities to discover entities in other spaces. This requires mechanisms for discovering similar services in other spaces, organizing the various services in some manner and having well established protocols for the way these services communicate and collaborate.

3.4. Location Modeling and Tracking

In this scenario, it is essential to have reliable methods for tracking the location of people. We also need geometric models of the building for location tracking. There are multiple mechanisms that can be used for tracking - 802.11 access points, GPS, Bluetooth, logins, RF badges, bats, fingerprint recognizers and other authentication devices. We need ways of aggregating location information obtained using different mechanisms and across different spaces.

3.5. Mobility

Mobile devices could enter and leave different Active Spaces. This requires mechanisms by which mobile devices could enter a space, become part of it (i.e. allow other applications and services in the space to use the device as well as allow applications running on the device to access other resources in the space) and finally exit the space. The other aspect of mobility is having mobile applications that follow a user around. This means that there needs to be mechanisms for migrating applications from one space to another.

3.6. Context Aggregation

Different spaces may have their own mechanisms to gather and infer context information. We need ways of aggregating context information from different spaces in order to get a more global picture of the context of the building and make cross-space inferences.

3.7. Security

Security and privacy are problematic in any pervasive computing environment, more so in Super Spaces. A notion of single sign-on for entities needs to be maintained at the Super Space level. Unobtrusiveness is a key issue in pervasive computing, so it is impractical to re-authenticate or renegotiate trust every time an entity moves from an Active Space to another. Now that a Super Space can provide a global view of the whole pervasive building or campus, users' privacy is threatened even more. Security policies that govern the relationships between different Active Spaces within a Super Space, as well as the relationship between a Super Space and its subspaces need to be crafted and studied carefully.

4. Architecture

In our architecture, an Active Space is the basic element in pervasive computing. We have implemented an Active Space in a room with several displays, desktops, handhelds and other sensors [2]. Each entity is registered in a database called Space Repository. A Super Space is defined and constructed in a recursive manner. A Super Space consists of one or more logically related subspaces as well as supporting infrastructure to manage services and applications that operate across the constituent subspaces. At the basic level, a Super Space is an Active Space.

There are a number of possible logical relations which motivate the grouping of Active Spaces (or Super Spaces) into bigger Super Spaces. The most obvious is physical proximity. For example, various rooms in a building may be running Active Spaces. All the Active Spaces in a floor could be combined into a Super Space for that floor. All the Super Spaces in the different floors of a building could be combined into a Super Space for the building. Other logical relations include common activity and common users. For example, if a particular class is being taught simultaneously in many rooms (using video-conferencing or distance-learning facilities), the Active Spaces in all the classrooms can be combined into one Super Space. A user can combine Active Spaces running in his home, his office and his lab into one Super Space as well. The main reason for combining Active Spaces (or Super Spaces) into a Super Space is to allow the management of multiple spaces using a single interface. This improves scalability, allowing applications and services to perform actions in or gather information from multiple spaces.

A Super Space has a set of core services - naming, discovery, space repository, event management, file system and context services. Services and applications in a Super Space operate at a more global level than those in Active Spaces. They make use of similar services in the constituent subspaces. They present a more consolidated view of the Super Space to users and other applications and services. They also perform operations in multiple subspaces and synchronize data and activities across different subspaces.

4.1. Structure of services and applications

The framework allows two models of services and applications in Super Spaces: 1) A hierarchical recursive model. 2) A peer-to-peer model.

Recursive model of services. Recursive services in Super Spaces are based on the reflective nature of these Super Spaces. Super Spaces consist of sub-spaces, which in turn consist of other sub-spaces. Each space is represented as an object and a parent space knows about its children spaces and can perform operations on them. Super Spaces can, thus, be represented as a DAG (Directed Acyclic Graph) of objects. The leaves of this DAG are individual, Active Spaces. Recursive services in Super Spaces rely on the notion that an operation on a Super Space object can be broken down into operations on its children objects. For example, the operation of switching off all lights in a building Super Space can be broken down into the operations of switching off all lights in all its children spaces (which may be individual rooms and corridors in the building).

Peer-to-peer operation of services. Another way of structuring services and applications in Super Spaces is by composing entities across different Spaces. One example of this is an application that is spread across multiple spaces. For instance, we may have a Jukebox application where the application logic and music soundtracks are at the user's home, the music output element (player) can be running in his office and an input element controller (song selector) with his son, who is in school, but wants his father to listen to a particular album. Other entities in different spaces can be similarly composed. In order to facilitate the use of components across spaces, we utilize inter-space "bridges," which are special components that foster communication between remote components or applications [4].

4.2. The Interaction Layer

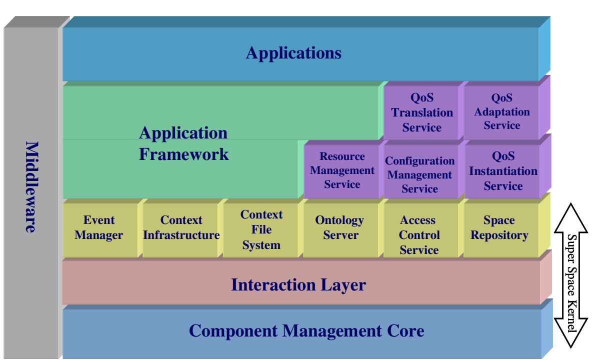

Fig. 1 Architecture of a Super Space

Figure 1 shows the architecture of a Super Space. It is very similar to the architecture of an Active Space. However, there is an additional Interaction Layer, which provides mechanisms to allow interaction among services and applications across the different spaces. The interaction layer has libraries to facilitate both recursive as well as peer-to-peer interactions between services. This allows developers of applications and services to not worry too much about how their services and applications interact with other applications, and develop applications almost as if they were just developing for one space. Multiple copies of their applications can be deployed in different spaces and they can all interact with one another seamlessly.

The interaction layer supports various patterns of interactions between services in different spaces. This allows different services and applications to reuse the same patterns and not have to redevelop interaction mechanisms. One such pattern of interaction is based on the Divide-and-Conquer paradigm. In this pattern, a service performs a task by dividing it up into smaller pieces and giving these smaller pieces to corresponding services in the child subspaces. The service then collects the results from the child services and combines them. For example, if a light-controller application operating at the building Super Space level sends the command to turn off all the lights in the building, it sends "switch off" commands to light controller applications running in each floor subspace, which in turn sends "switch off" commands to each Active Space running in rooms and corridors.

4.3. Mobility

The architecture allows mobile devices to become part of a Super Space. The key challenge is in letting mobile devices know where they are and which space(s) they can be part of. We use various techniques to give location and space information to mobile devices including infra-red beacons (broadcasts the space(s) that a device can join), and Bluetooth access points. Depending on which access point is being used by the device, it is allowed to be part of certain spaces. Another way for mobile devices to be part of a space is by having the device send its location information to one of the "Space Resolver" services, which gives it the reference to the spaces it can be part of. Once the device has a reference to the space, it authenticates itself to the space and is then allowed to access resources in the space. A reference to the mobile device is also inserted in the Space Repository so that other entities in the Space can discover and communicate with the device.

5. Super Space Applications

Aggregation of Active Spaces provides a platform for deploying various applications. Some applications are meaningful only in a Super Space while others are Super Space counterparts of Active Space applications. In this section, we discuss a few representative Super Space applications, their significance and details of their deployment in a Super Space.

5.1. Interactive Presentations

Distributed presentation sessions such as distance learning, multi-classroom lectures and interactive wide-area presentations require infrastructures that facilitate two-way communication between the presenter and the listeners. The present day systems require the attendees to be assembled in a room that provides two-way communication. Our architecture provides an infrastructure for such applications. A presenter defines a Super Space and starts a presentation session in that space. Each listener belongs to an Active Space that gets registered into the presentation Super Space. The listener can access applications of the Super Space and so can register as a listener of the presentation application. The presentation application provides some facilities to the listener such as listening to the lecture, obtaining a list of other listeners and interacting with the presenter or other listeners.

5.2. Guide Applications

Guide applications in museums, zoos or shopping malls require a global view of the state of the system. Guide applications in shopping malls help people locate required articles. These applications can also plot routes to lead a person to a room, a display item or to any entity in the physical space.

A museum guide application runs in the Super Space that covers the whole building. This application gets information from its children Active and Super Spaces about exhibits to be seen, current conditions (such as crowds, closed exhibits, etc.). The user can also give it information about what kinds of exhibits he wants to see. The application plots a route for the user through different rooms. When the user enters a room, the Active Space running there guides the user around the exhibits in the room. This guiding process can take place using handhelds, touch screens in the room or other interfaces. The Super Space application also ensures that different groups do not land up at the same place at the same time.

6. Related Work

Several research efforts [6-8] introduced a middleware layer to support the construction of prototype pervasive computing environments; however, they do not tackle the issues of managing and orchestrating large collections of these environments. Roman et al. [9] introduced a framework that provides "inter-space mobility" to support moving applications across different pervasive computing spaces. This approach assumes the existence of a method to map resources in the source to compatible devices at the destination. Applications are suspended while in transit and resumed once the user reaches the destination. In addition to supporting application mobility, we are aiming to support applications with components running over multiple pervasive environments, while allowing more interactions between the different spaces to take place.

7. Conclusion

In this paper we argue in favor of extending pervasive computing middleware to support the management of large-scale pervasive environments or Super Spaces and to orchestrate the various pervasive computing elements that make up these spaces. We also provide some motivating applications and scenarios.

8. References

[1] M.Roman et. al. ,"Gaia: A Middleware Infrastructure to Enable Active Spaces", IEEE Pervasive Computing, 2002.

[2] M.Roman et. al., "Gaia: Enabling Active Spaces", 9th ACM SIGOPS European Workshop, 2000.

[3] C.S.Pinhanez, "The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces," UbiComp, 2001.

[4] M.Roman et. al., "Dynamic Application Composition: Customizing the Behavior of an Active Space", IEEE PerCom 2003, March 2003.

[5] D.Snoonian, "Smart Buildings," IEEE Spectrum, vol. 40, 2003.

[6] B. Johanson et.al., "Experiences with Ubiquitous Computing Rooms," Pervasive Computing Magazine, 2002.

[7] P. Tandler, "Software Infrastructure for Ubiquitous Computing Environments: Supporting Synchronous Collaboration with Heterogeneous Devices," Ubicomp 2001.

[8] C. Becker et.al., "BASE - A Micro-broker-based Middleware for Pervasive Computing," IEEE PerCom 2003.

[9] M. Roman et.al., "Application Mobility in Active Spaces," First International Conference on Mobile and Ubiquitous Multimedia, Oulu, Finland, 2002.