

“The Components of a Smart Space Platform for Smart Service Deployment”

Darach Cawley
Telecommunications Software and Systems Group
Waterford Institute of Technology
25/04/2003

Abstract

A Smart Space Service is currently defined as a service offered to a person or computing entity in a Smart Space. The Smart Space Service may originate inside the Smart Space or from outside the Smart Space. A service such as this needs to be deployed in an environment where the service itself can avail of a number of fundamental services. Conceptually a platform providing such fundamental service components may be realised to fulfil the services needs. This paper will investigate the requirements of smart services and their users within Smart Space environments and from these requirements identify the core components needed within a platform to support the services and their users. It will also consider candidate technology platforms and their components and look at their applicability to smart space environments.

1. Introduction

The traditional environment is designed to foster a thriving community of people, and modernity has introduced many possibilities that until now have not been possible. Today has seen the introduction of a diverse range of ideas and technologies, all of which are proving to be beneficial to the development of the environment. The possibilities are endless and the technologies to implement them are emerging at a substantial rate. As the development of these new technologies continues, the environment that is to support these technologies needs to be identified and described.

This environment is currently defined as a physical space rich in devices and software services that is capable of interacting with people, the physical environment and external networked services. The aim of this environment or smart space is to orchestrate the use of integrated physical and computing environments to bring tangible benefits to people in support of their tasks. The smart space interacts with people in a natural and familiar way to help them engage in their tasks.

A task is currently described as a description of a user's intention that has a defined outcome. A single or a set of smart space services can support a task. A task might represent a short or long lived set of activities.

A smart space service is currently defined as a service offered to a person or computing entity in a smart space. The smart space service may originate inside the smart space or from outside the smart space. It may be a composition of internal or external services. Services include mechanisms for interacting with users and with the physical environment.

Most software developers are perfectly happy just to write software applications. You know, programs that do something or solve some particular problem. But the brave among us want to change the world more significantly, and they choose to work on platforms: big giant slabs of software that don't quite do anything out of the box, but which enable a whole world of new and interesting applications. So they write operating systems, or DBMSs, or language runtimes like Java, and they hope to attract independent software developers to create the great new applications that their platforms enable. (Joel Spolsky, 2002)

As a platform vendor, you would only be as successful as the people who build on you. You will want to appeal to the very best developers and therefore you will have to supply the right services in order to appeal to them.

One of the motivations behind the M-Zones project research is the lack of a suitable software infrastructure to assist in the development and deployment of applications for ubiquitous computing habitats or living spaces. This paper will evaluate current technology platforms and compare their components to those realised by the requirements of the users and third-party services in a smart space, evaluating their applicability to the environment.

2. Smart Service & User Requirements

As smart spaces become increasingly sophisticated there will be an increased demand for services, which will enable users of the smart space to operate the environment in a customisable way. The services should also be capable of managing communications between equipment of different levels of “intelligence” (sensors, computers, embedded processors).

A service would be deemed useless if it did not support one or more of the users' requirements. Furthermore, the components required in a platform depend on the requirements of the services the platform is going to support. There is no use for a platform that does not support the needs of third party services or a platform that supplies services that are not going to be used by the third party services. This flow of dependency shows us the importance of both the user and the service in the development of a platform therefore we must look at these areas in detail.

2.1 User Requirements

There are three general types of users of a platform, the normal user, who uses the third-party services, the administrator, who controls and maintains the platform, and the developer who develops the third-party services for deployment on the platform. I have come up with a number of requirements for these different users below.

2.1.1 Normal User Requirements

A normal user of a service would have some requirements that they would expect from not only from one service but also from all services. Here are some basic and generic user requirements that I as a user would expect in a smart space environment:

- The ability to use the services in any environment
- The ability to customise the services to their own needs
- The guarantee that there will be an acceptable level of quality of service
- The guarantee that their information has some level of security
- The ability to use a number of services simultaneously
- The ability to use the services on a range of devices
- The ability to use the devices and services easily
- The ability to afford the cost of using the services

2.1.2 Administrator Requirements

As a platform is a complex system with a lot of resources and services to control and maintain, it is safe to assume that some sort of administrator user would be required to control this. As an administrator, here are some requirements that I would expect in a smart space environment:

- The ability to configure the platform for different environments
- The ability to manage and maintain data stored on the platform
- The ability to manage and maintain the other resources of the platform
- The ability to control all users and services using the platform

2.1.3 Developer Requirements

For a platform to be used in any environment it needs to make itself attractive to developers. It should make it easy for developers to develop services for it otherwise they will have no interest. As a developer, here are some requirements that I would expect in a smart space environment:

- A development toolkit for the easy development of services for the platform
- An API for the integration of the service to the platform
- Documentation on the use of the toolkit and API

2.2 Service Requirements

It is hard to try and pinpoint the exact generic requirements of a service as each service varies with regards to how and what it provides a user but here are some examples I would suggest:

- Logging in/logging out a user to the platform
- Authenticating a user to the platform
- Subscribing/unsubscribing a user to the platform
- Saving user preferences to the platform
- The lookup and discovery of other third-party services

One way of obtaining the requirements of a service is to pick a user group and choose a service for that particular group. If we select a University Campus and pick the student and staff population as the user group, we can now choose a service for these users. A timetable service is an example of such a service. It could provide some of the following features:

- Produce timetables for the users
- Enable a staff member to locate another staff member based on their timetable
- Allow a lecturer to attach messages to lectures for the students to read
- Change the status of a particular lecturer
- Notification to the students via SMS or email of changes to their timetable

These are examples of the typical features that could be provided to the user. This service could be made ‘Smart’, in the sense that it could have the ability to locate people based on their timetable and habits. The timetable service would not provide all the features although, as some features like the notification feature could be required by many services. The need of a particular functionality, which all third-party services would use to support their users, is what allows us to realise some of the core services needed in a platform.

The services the users will be using are third-party services that are developed by developers externally to the platform. The components in the platform are services for the third-party services (an example would be a lookup service to find other third-party services). The platform would also allow third-party services to avail of the use of other third-party services.

3. Core Platform Components

We have looked at the requirements of a user and the service they may use and now we want to recognize the components that are required in a platform in order to support those requirements.

3.1 Platform Requirements

Some main requirements that should be provided by a platform for services and users in a ubiquitous computing environment would fall under the following headings:

Mobility

First of all, the platform has to deal with the mobility of the users and other physical objects. Therefore it must support mobile wireless communication and small mobile computing devices. It also must support the inevitability of frequent network disconnection and the formation of ad hoc networks.

Heterogeneity

The platform has to accommodate a large variety of application environments, ranging from areas covering whole nations to areas just covering an office. This leads to a very heterogeneous system, which has to support different network technologies and different tracking systems for example. Further aspects of heterogeneity come

from the different services that rely on the presence of this platform and the various input and output devices that have to be supported.

Scalability

The platform has to scale well for both a large number of cooperating services, which realize the application in each application context, and a large number of devices involved each time the application is used. Services represent the logical dimension of the application, while devices represent its physical dimension.

Configurable

The platform has to be able to configure the platform services to meet the different requirements of a given application domain, i.e. not all domains are the same and the platform should be adaptable to any given environment.

Self-configurable

It should also be self-configurable, be able to dynamically configure itself and its services when changes occur in its environment. It should be evolvable, able to adapt to any change that may occur in the underlying platform design such that the application is provided with a seamless access of services. (S.S. Yau, 2002)

Context Awareness

Ubiquitous computing devices also have to be aware of the types of services and devices in their environment i.e. a lookup and discovery service. Who owns what and what do they have access to, given their capabilities and their users profile. Knowing where the user and their device are located is also a key aspect of a smart space and should be supported by the platform.

Quality of Service

A control of Quality of Service (QoS) is desirable to control response time, availability, data accuracy, consistency, and security levels (Kurt Geihs, 2001).

Resource Management

The platform should have the capability to monitor local resource availability for reserving, negotiating and adapting resource usage.

Portability

The platform should have the ability to be provided on a number of technologies, to be portable on different servers. One smart space may be different to the next one, thus creating a need for portability from one technology to another.

3.2 Platform Components

Smart Services need to be deployed in an environment where the service itself can avail of a number of fundamental services. This is where the concept of the platform and the platform components is realised. Based on the requirements of the users, services and platform, the core components needed within a platform to support these requirements have been realised as the following:

Profile Service

This service allows access to the profile information of a user – the profile information accessed here is core or essential information like the user's name, address, phone number etc... Continuing on from the example of the timetable service from the previous section, the service would obtain the profile information of a student from the Profile Service and any other information like the timetable of the student would be stored in a separate area or with another service.

The profile service also stores information on devices in a Smart Space and the agents within it too.

Subscription Service

This service would allow the subscription of a user to a third-party service (e.g. the timetable service). It would also allow the registration and deployment of a new third-party service to the platform.

Database Service

All services running in the platform will have access to the database service. The database service has an interface to whatever database is being used by the platform (note: the platform could be using a number of different types of databases). This would control access to and maintain the database.

Presence Service

This service allows third party services to access to users' presence information – this information would store what services the user is currently using, the device they are using them on and where they are using them (i.e. their location).

Accounting Service

This service would be used to store the accounting information of the platform. The billing of the user for the use of the third party services and the billing of the third party services for the use of the core services within the platform.

Notification Service

This service would be used by the third party services for notifying their users of updates or news relevant to their service. This could notify the users in a number of ways (e.g. SMS or email)

Resource Management Service

This service would monitor all local resources, ensuring availability of resource reserving, negotiating and usage.

4. Candidate Platforms

In this section we will look at two of the biggest platforms that are currently available, J2EE and .NET. We will look at the capabilities of these platforms and investigate whether they support the platform requirements we realised in the last section thus checking for their applicability to the smart space environment.

Table 4.1 gives us a summary of what of the realised platform components do the two of the biggest platforms today provide and whether they are applicable to the smart space environment.

Ref.	Requirement	J2EE	.NET
1.	Mobility	YES	YES
2.	Heterogeneity	YES	YES
3.	Scalability	YES	YES
4.	Configurable	YES	YES
5.	Self-Configurable	NO	NO
6.	Context Awareness	NO	NO
7.	Quality of Service	YES	YES
8.	Resource Management	YES	YES
9.	Portability	YES	NO

Table 4.1

J2EE

1. Mobility is provided by the use of a combination of either J2SE or J2EE and J2ME (J2ME, 2003).
2. Heterogeneity is supported in a way, because J2EE is both scalable and configurable, and supports a number of different technologies.
3. J2EE supports scalability that range from 10s to 100s to 1000s (J2EE Overview, 2003). This is very important when it comes to mobility and when users are roaming in and out of smart spaces.
4. J2EE components and applications are configurable and customisable at packaging and deployment time, which is good when working with different domains.
5. J2EE does not have the ability to self-configure itself when changes occur in the environment.
6. J2EE does not currently handle the full requirements of context awareness as stated earlier but may do so in future evolving services (Service Delivery, 2003). The technology Jini is available for service and device discovery but when it comes to finding the location of a user or device in an environment, J2EE does not have that ability.
7. It has a certain level of Quality of Service when it comes to whether a message had been sent or not. A transaction will not take place if it is not sure whether the message arrived at its destination.
8. Resource Management (J2EE Management Statistics, 2003) is available.
9. J2EE is portable over a number of technologies and it is open source.

.NET

1. Mobility is supported by .NET by its extension .NET Mobile (2003).
2. Heterogeneity is supported in a way, because .NET is both scalable and configurable, and supports a number of different technologies.
3. .NET supports scalability for a range of different user numbers.
4. It is configurable to the environment it is being used in.
5. .NET does not have the ability to self-configure itself when changes occur in the environment.
6. .NET does not currently handle the full requirements of context awareness but may do so in the future.
7. Quality of Service is supported
8. .NET does have its own version of Resource Management
9. .NET is portable in a sense, but applications developed for .NET would not be portable to other platforms like J2EE.

5. Conclusion

In this paper we have introduced the concept of Smart Spaces and the services within them. We have looked at the requirements of a user in a ubiquitous environment. We have also looked at the requirements of a service in the same type of environment, keeping in mind that the service is there to support those requirements of the user. Based on these combined requirements, we have realised the requirements of a

platform that is needed in this environment. After establishing the components in this platform, we looked at two of the widest used platforms in today's environment, and compared them with the components that were realised by the requirements. After this we established that the J2EE or .NET platforms do support a number of the platform requirements but are still not currently ready to support the requirements of a ubiquitous smart space environment.

6. References

James Kao, *Developer's Guide to Building XML-based Web Services with the Java 2 Platform, Enterprise Edition (J2EE)*, (2001)

Alan Davy, *Middleware infrastructure for services*, (2002)

MDE Enterprise Java Team, *eMobile Application Demonstrates End-to-End J2EETM 2 Interoperability*, (2000)

Object Management Group, CORBA Services (2002): Available from <http://www.omg.org/technology/documents/formal/corbaservices.htm> [Accessed 24th April 2003]

S.S. Yau and F. Karim, "Reconfigurable Context-Sensitive Middleware for Pervasive Computing", IEEE Pervasive computing, 2002

Kurt Geihs, "Middleware Challenges Ahead", IEEE computer, 2001

Joel Spolsky, (2002): Available from <http://joelonsoftware.com/articles/Platforms.html> [Accessed 16th April 2003]

Sun Microsystems, J2EE Services (2002): Available from http://java.sun.com/blueprints/guidelines/designing_enterprise_applications/platform_technologies/platform_services/index.html [Accessed 22nd April 2003]

Jini (2002): Available from <http://www.jini.org/> [Accessed 16th May 2003]

Service Delivery (2003): Available from <http://www.sun.com/software/sunone/cover/2002-0325/> [Accessed 19th May 2003]

J2ME (2003): Available from <http://java.sun.com/j2me/> [Accessed 19th May 2003]

J2EE Overview (2003): Available from <http://java.sun.com/j2ee/overview.html>

[Accessed 19th May 2003]

J2EE Management Statistics (2003): Available from
<http://java.sun.com/j2ee/1.4/docs/api/javax/management/j2ee/statistics/package-summary.html>

[Accessed 19th May 2003]

.NET Mobile (2003): Available from
http://www.w3schools.com/dotnetmobile/mobile_intro.asp

[Accessed 19th May 2003]