Directory Enabled Smart Spaces

Enda Cahill

A dissertation submitted to the University of Dublin,
in partial fulfilment of the requirements for the degree of
Master of Science in Computer Science

2002

Declaration

I declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed: ____________________
Enda Cahill

Date: 16th September 2002

Permission to lend and/or copy

I agree that Trinity College Library may lend or copy this dissertation upon request.

Signed: ____________________
Enda Cahill

Date: 16th September 2002

Abstract

Significant research and development in the area of system management has been carried out over the last number of years. This research has led to a number of proven technologies that are currently being used to manage large-scale systems. This dissertation aims to build upon this research and development to examine how a smart space management system could be created.

The dissertation examines how an existing information model can be extended to adequately model the information that is required to manage a smart space. Policies, which can control the behaviour of the smart space, are also modelled using existing policy models. The information model and the policies are then mapped onto a directory service that is used as a repository.

A prototype application is described which can use the directory service to control and manipulate a smart space. This prototype application is used as a platform to demonstrate the design decisions that have been taken.

Conclusions will be drawn as to the potential benefits of using the Directory Enabled Networks Initiative and the Common Information model for the representation of smart space management information. The suitability of using policy based management techniques in the management of a smart space environment will also be discussed.

Acknowledgements

I would like to thank my supervisor, Mr. Vinny Wade, for his invaluable advice and guidance during the year. Thanks also to Caitríona, and my family for their patience and support. Finally thanks to my classmates and friends who made the year so enjoyable.

Table of Contents

Chapter One

1. Introduction

1.1 Overview

The development of smart spaces is an area into which considerable research effort is being directed as the number of such projects increases [12, 15, 16]. As the cost of embedding intelligence in a device decreases it is expected that billions of these devices will be interwoven and connected together forming a world-wide distributed system several orders of magnitude larger than today's Internet [1]. The management of this new, highly dynamic infrastructure and the development of the applications and services that will control it will be a challenging research topic during the early part of the twenty first century.

The vast majority of the current research into smart spaces is focussed on the development of a single smart environment. These projects examine the issues that are evident in that smart space environment in order to understand the challenges in developing smart space applications. In this dissertation an alternative approach will be taken which will look at the issues in managing a more flexible multi application smart space. While examining the management challenges in this multi application smart space the principles from current management technologies such as the Directory Enabled Networks [2] Initiative and policy based networking will be applied. The dissertation will examine the ability of proven enterprise network management technologies to play a role in the management of the new smart space environments. A smart community space will be used as the foundation for a prototype smart space management application. A community space was chosen because of the variety of different events that can take place in such an environment. In modelling this smart community space it is hoped to successfully answer the following questions:

1). Is the representational power of the directory service adequate to store the required management information

2). Is the directory service dynamic enough to support the update of the management information

3). Is the use of policies beneficial in the management of the smart space

4). Does the directory service support efficient retrieval of the information by the smart space management application.

While examining these questions it is hoped to achieve a greater understanding of the issues in smart space management and in the modelling of smart space environments for representation in a directory service. The applicability of existing DEN (Directory Enabled Networks) and policy based management technology to the management of these highly dynamic environments will also be evaluated.

1.2 Objective

The objective of this dissertation is to examine the issues in smart space management by modelling a sample multi application smart space. One of these issues is the representation of the data that is needed to manage the smart space. An accurate model needs to be created of the smart space devices, infrastructure and the occupants of the smart space as well as the tasks and events that occur in the smart space. These models then needed to be mapped onto an appropriate repository from where they can be processed by software applications that will manipulate the smart space. The potential benefit in using the CIM (Common Information Model) [3] and DEN technologies to model and store this information will be examined. Finally the issue of using policies to control the smart space environment will be examined. In order to do so the approaches used in policy-based networking will be reapplied to the management of the smart space.

1.3 Motivation

The motivation for this dissertation is the requirement for greater understanding of the issues in managing multi application smart spaces. This will aid in the creation of a framework for the development of smart space management applications. The need to examine appropriate frameworks from which smart space management applications can be developed is necessary to allow the development of large-scale multi application smart space management applications that function in an open and scalable manner.

1.4 Methodology

The methodology used will be to examine the existing representations used in management applications and determine if these representations can be successfully extended to appropriately model the smart space. Using this extended model a representation of the chosen smart space will be created in the directory service. An application to leverage the directory service in a simulation of the smart space in operation will then be developed. The results from the examination of the prototype will then be used to evaluate the usefulness of the information model and directory representation. Finally examples of how the popular communication technologies in use in the smart space arena could be integrated with the prototype will be given.

1.5 Dissertation Overview

Chapter two will first discuss the current state of the art in using directory systems as a basis for a management infrastructure. The current developments in CIM and DEN technologies will be detailed before proceeding to examine the current research in smart spaces. Finally four major projects, which are currently under development, will be discussed in relation to how the issues in managing those smart spaces have been approached.

In chapter three the issues in smart space management will be discussed and how those issues compare to the management of existing systems. An evaluation will be made of which components of existing management systems are appropriate for reuse in smart space management applications.

Chapter four will detail the design of the prototype system. Representation of the smart space infrastructure, policy, services, tasks and users will be discussed. Extension of the directory schemas to store the necessary information for the management system will be discussed. The approach used in the prototype for mapping context information onto the directory will also be examined.

Chapter five will discuss how the prototype was implemented with the particular applications and technologies that were chosen. The methods used to simulate the smart space environment will also be discussed. Finally examples of how the prototype could be integrated with the popular communication technologies that are currently underlying smart space research will be given.

In chapter six the prototype implementation will be evaluated and the findings from the dissertation discussed. The initial objectives will be revisited in order to determine if they have been achieved successfully. A review of the characteristics of directory services, which make them a suitable platform for management applications, will be made.

Chapter seven will outline the conclusions from the dissertation and suggest further research that could be carried out to build upon what has been achieved by carrying out this work.

Chapter Two

2. State of the Art in Directory Enabled Networks and Smart Spaces

2.1 Introduction

This chapter will briefly outline the state of the art in the technologies that provide the foundation for the dissertation. Firstly the Directory Enabled Networks [2] Initiative will be examined and in order to complete that task fully the Common Information Model [3] will also be introduced. The directory service technologies that provide a platform for the DEN initiative will then be discussed. Finally the current research in smart spaces and the management of smart space environments will be detailed along with information from four of the most prominent smart space research projects [12, 15, 16, FCE, MUSE].

2.2 Management Technologies

This section will introduce the state of the art in the management technologies that the dissertation will use as the foundation for the development of the smart space management prototype. The management technologies include the Directory Enabled Networks Initiative, the Common Information Model and Policy Based Networking.

2.2.1 Directory Enabled Networks

Cisco [CSCO] and Microsoft [MSFT] originally proposed the Directory Enabled Networking Specification in 1997 with the following objectives:

1). To outline the base schema extensions to any directory, so that network devices and configurations can be stored in a directory.

2). To enhance a directory to introduce "network services" and logically associate users with particular network configurations.

3). To gain industry-wide consensus for the proposed directory enhancements, and use this specification as a foundation for building future applications (Operations & Maintenance, Provisioning, Service Admin. & Management).

4). Create a new paradigm of network-aware applications that leverage network resources commensurate with the users identity.

The specification provides a schema and information model for representing network elements and services in the directory. When this specification is implemented it allows the services to be associated with users and applications that also exist in the directory. The directory can then be populated with data on actual network devices allowing these devices to be controlled and configured through the directory based on the needs of individual users or applications. The network can also be reconfigured dynamically according to policies defined in the directory, thereby facilitating policy based management of the network.

A mapping of the schema and information model to a directory using LDAP (Lightweight Directory Access Protocol) or an X.500 directory access protocol is also contained in the DEN model. This mapping is an extension of the Common Information Model (CIM), which is also managed by the DMTF (Distributed Management Task Force) [DMTF].

2.2.2 Policy Based Management

Policy based management is defined in [4] as "a viable technology to provide greater control and management of underlying networks via the creation and distribution of high-level policies (business rules), integrated with the enabling mechanisms of the network infrastructure." Policy based networks are often based upon a directory service as the policy repository while policy modelling and usage is an integral part of DEN.

Policy Based Management manages the network by focusing on the users and applications rather than the individual network devices that make up the network infrastructure. Policies are created that define the business goals and objectives of the organization. These policies and goals are then stored in a policy repository before being translated by the network and applied as configurations to each element in the network as appropriate. Numerous policies can be stored with different conditions defining when they should be applied to the network, this allows the network to be dynamically reconfigured as conditions change or events occur. In effect the network is now managed as a whole rather than as a set of individual unrelated network elements.

Policy based management makes it easier to apply specific rules to individuals, groups of users, network elements or applications. In this way the network can easily be configured to provide more bandwidth to important users or mission critical applications while also knowing which groups of users have priority at each stage of the working day, week or year. Less human intervention is required in the day-to-day operation of the network and the network is intelligent enough to respond to events as they occur.

There are four main components that make up the Policy Based Network Management system. These were defined by the IETF (Internet Engineering Task Force) [IETF] as a schema for the enforcement of a Policy Based Network Management system.
They components as defined in [5] are:

- Policy Management Tool
- Policy Repository
- Policy Target
- Policy Consumer

2.2.2.1 Policy Management Tool

The Policy Management tool consists of a Policy Editor that provides an interface for entering, viewing and authoring policies and the Policy Translator that translates general policy rules into device independent rules that specific policy consumers can use. The management tool also includes a Policy Validation function, which checks data types and semantics of the policy, and a Global Detection function, which checks for conflicts with other existing policies.

2.2.2.2 Policy Repository

The Policy Repository is essentially a store to keep the policies in after they have been translated and verified by the policy management tool. Policy Consumers access the policy repository to download the policies.

2.2.2.3 Policy Target

The Policy Target is the network component that must adhere to the Policy rules that are applied to it by the policy consumers. Policy targets can be any type of network component.

2.2.2.4 Policy Consumer

The Policy Consumer's main function is to retrieve policies from the Policy Repository and deploy these policies to Policy Targets. This may involve translating the policy into a form that the device can understand.

2.2.2.5 Policy Based Management Evaluation

The use of policy in the management of a network allows the network to be controlled in a declarative fashion. The benefits of this declarative methodology is that the services and service levels that are required from the network can be stored in policies independent of the underlying network technologies. This implementation independence can provide many benefits when the network consists of diverse technologies or when introducing new technologies to an existing network. However there are still many difficulties in using policy based technologies. These difficulties include the fact that the process of evaluating and determining conflicts in policies is very complex. The scalability of policy languages in a large network environment has not as yet been proven. A standard methodology to create abstractions of an environment to use when creating policies has not yet been defined.

2.2.3 CIM Common Information Model

CIM [3] is an object oriented information model that describes how a system and its components maybe managed. One of the primary aims of CIM is a consistent presentation of the managed environment despite the different protocols and data formats that the managed devices may support. CIM is a layered model with each lower layer further refining the layer above. All the layers are refinements of the core model. Following the core model there are seven common models, which are System, Device, Application, Network, Physical, User and Policy.

The CIM Physical Common Model has the following objectives [2]:

- Model the characteristics of physical entities, as well as the physical aspects of a managed entity
- Model the physical connectivity aspects of a managed environment
- Model the physical containment aspects of a managed environment

The common super class for all elements in this model is the *PhysicalElement* other classes defined in this model include: *Location, PhysicalCapacity, MemoryCapacity, ReplacementSet, PhysicalPackage, PhysicalFrame, Rack and Chassis*.

The purpose of the CIM Network Common Model is to model the logical characteristics, functions and capabilities of managed objects that are contained in, and form, a network. The model reinforces the distinction between physical and logical aspects of a managed object. The five main goals of the CIM Network Model are [2]:

- Model the logical characteristics of network devices
- Model the logical characteristics of network systems
- Model various networking protocols
- Model the connectivity aspects of a managed environment
- Model the administration of networks and network elements

The Network Common Model does not introduce any new top-level Classes but refines the classes from the core model into sub-classes such as: *LogicalElement, ServiceAccessPoint, Service StatisticalInformation, FilterList, FilterEntry, RoutingPolicy* and *Rotuemap* along with many others.

2.2.3.1 Evaluation of CIM

CIMs use of object oriented modelling technologies makes it a very powerful and flexible information-modelling tool. The core model of CIM concentrates on modelling managed systems as the related to a network environment, this focus can lead to difficulties when modelling services that are not traditionally thought of as computing services. The use of an implementation neutral schema for representing the information model can lead to difficulties and inconsistencies when mapping that schema onto the actual repositories. The information model is however capable of representing management information in a logical if complex manner.

2.2.4 Relationship of DEN and CIM

DEN extends CIM by providing information that is specific to networking and more refined than the data that CIM provides. The DEN LDAP mapping provides a directory schema that defines entries that can be added to an existing schema to represent network elements and services. It also provides entries that represent policy rules and related policy information. The DEN schema was originally composed of a Physical, Logical and Policy Model. The DEN Physical model became the CIM Physical Common Model, the Logical Model was integrated into the CIM Networks Common Model and the CIM Policy Common Model was developed on the DEN Policy model.

The DEN specification itself contains these models and appropriate portions of the core and other common models plus a mapping of the CIM schema for implementation in a directory that uses DAP (Directory Access Protocol) or LDAP as its access protocol.

The DEN model extends CIM in several ways [3]:

- DEN adds the modelling of network elements and services.
- DEN addresses the mapping of generalized information defined by CIM and DEN into a Lightweight Directory Access Protocol (LDAP) directory.
- DEN integrates concepts from X.500 directories with directories that use LDAP as their access protocol.
- DEN extends the information model to describe how network elements and services behave.

The purpose of the DEN Policy model was to provide an Information Model and an extensible class hierarchy that enabled application developers to represent policies that controlled access to and allocation of network resources. The Model defines the following sub-classes to enable that modelling: *Policy, NetworkingPolicy,* *DiffServPolicy, PolicyCondition, NetworkingPolicyCondition, PolicyAction, TimeOfFDayValidity* etc.

2.2.5 Motivation of DEN

Directory Enabled Networks provides two important benefits [2]:

- Network elements and services are represented in a standard way allowing diverse applications to share and reuse the same data.
- All objects of a managed environment are represented as objects. This allows the different types of entities that make up a managed system to be treated in the same way. This provides a unified way of representing information about different types of entities.

These benefits help to overcome the fact that many organisations are still using multiple directories and data stores for representing and managing the same information. Each management application represents the objects it manages in a different and sometimes non-compatible format, leading to a proliferation of directories storing the same data for different purposes. Modelling this data in a common format can allow different management applications to leverage the same underlying directory and allow the development of new types of management applications that can take advantage of the combined data stores.

Directory Enabled Networks also allows Policy Based Networking to progress to a new level where the network can be reconfigured dynamically based on predefined policies stored in a data store. Modelling users and policies in the directory is an important part of enabling this. Resolving these issues and making the management of enterprise networks more straightforward are the original motivators for the Directory Enabled Networks specification.

The DEN specification provides a common information representation of all the managed objects in a system. This facilitates greater compatibility between network devices and management applications from different vendors. This common representation is one of the greatest attractions to network managers because it offers them another alternative technology that allows them to use one information store and modelling schema to represent policies and profiles for differing devices. Industry acceptance has been gained by the specification because the model is vendor neutral and provides the same benefits for each equipment, application or directory vendor. Each vendor can use their product or service to manage, participate or store information for other DEN compliant devices and applications.

Directory Enabled Networking enables complex rules to be defined and implemented that allow the network to perform as intended, without defining specific rules for each device on the network, this helps to automate configuration management of the devices on the network. Implementing policy-based user support is another advantage of the DEN specification. Users and services can be associated with profiles that allow the network to be dynamically reconfigured according to the parameters contained in the profiles associated with the users or services.

However there are difficulties in the implementation of DEN for example directories systems are not suited to storing information that is updated frequently. Careful consideration needs to be given to where this type of data should be stored in the implementation. Local changes must be propagated throughout the network that can cause inconsistent views of the network while this propagation is occurring. Likewise replication to avoid data loss because of directory failures can lead to inconsistencies in the data stored in directories. Finally directories as a technology do not support transactional operations that can lead to discrepancies or inconsistencies in the overall view of the network as provided by the directory.

2.2.6 Technologies to support DEN

The protocols, which the DEN specification makes use of, are the Lightweight Directory Access Protocol and the X.500 standard. Common Information Model (CIM) is not a protocol but rather a data model. CIM presents an object-oriented methodology for describing not just an individual component of the system, but also the management of system information.

First released in 1988, the X.500 standards define a specification for a rich, global, distributed directory based on hierarchically named information objects that users can browse and search using arbitrary fields. The model encompasses the overall namespace and the protocol for querying and updating it. The protocol is known as "DAP" (Directory Access Protocol). DAP runs over the OSI network protocol stack -- that combined with its very rich data model and operation set makes it quite "heavyweight". A major part of X.500 is that it defines a global directory structure. It is essentially a directory web in much the same way that http and html are used to define and implement the global hypertext web. Anyone with an X.500 or LDAP client may browse the global directory just as they can use a web browser to browse the global Web.

Lightweight Directory Access Protocol or LDAP is accepted as the industry standard for developing Directory based applications. LDAP was developed as a replacement for the complex directory access protocol defined in the OSI X.500 standard. The initial protocol was designed at the University of Michigan [MICH] to operate over the TCP/IP stack, since then the protocol has been handed off to the IETF (Internet Engineering Task Force) [IETF], which now controls the specifications for LDAP. The protocol provides the ability to read, search and modify a directory and has been implemented in all the major directory systems such as Novell's e-Directory [NOVL], Microsoft's Active Directory [MSFT] and Sun [SUNW] / Netscape's [NTSC] I-Planet.

The LDAP protocol provides a mechanism for passing text-based queries from an LDAP client to an LDAP server over the network. The goal is to allow users quickly and easily create and query directories of people and information for example user names, email address, and telephone numbers.

Directory Services play a vital role in the management of most large networks providing access to a variety of information about users, machines and applications. Following Microsoft's release of Active Directory with the Windows 2000 operating system directory service products are now also being used in the management of smaller scale networks. Five of the leading Directory service products in mid 2002 are Novell's eDirectory, Microsoft's Active Directory, IBM's [IBM] SecureWay, Siemens [SMNS] DirX and iPlanet Directory.

Both SecureWay and DirX were developed to implement the X.500 directory standard while Netscape developed iPlanet from the LDAP protocol. Active Directory and eDirectory were developed with Microsoft Windows 2000 and Novell Netware respectively. Resource and relationship modelling and management vary between each of the five products as do access protocols. SecureWay and DirX support LDAP through translation gateways, iPlanet and eDirectory support LDAPv3 while Active Directory is LDAPv2 compliant with minimal support for LDAPv3. The recommended access protocol for Active Directory is ADSI (Active Directory Services Interface). iPlanet provides facilities for the management of user objects, groups and authentication and does not in general facilitate the management of other resources.

2.2.7 Analysis of Management Technologies

Directory Enabled Networks and the Common Information Model provide a consistent way to model management information before representing this information in a directory service. The information model accurately models the objects that are regularly encountered in a network environment. The standardization of the models aids the interoperability of products from different vendors. Directory services are becoming a more prominent technology in medium size network environments following the release of Microsoft Active Directory. Policy based management is a technology that has not yet matured but all factors indicate that it will have an important role to play in future management systems.

2.3 Smart Spaces

A Smart Space consists of users, embedded devices and traditional computing devices; a communication mechanism, typically wireless, between all of the devices is also necessary. "Smart Spaces are everyday environments, which are populated with traditional computing hardware as well as embedded computers, information appliances, and multi-modal sensors allowing people to perform tasks efficiently by offering unprecedented levels of access to information and assistance from computers" [9]. The Smart Space vision is closely related to that of Ambient Intelligence, which is essentially a combination of the Ubiquitous Computing, Ubiquitous Communication and Intelligent User Interface areas. "Ubiquitous Computing is the integration of microprocessors into everyday objects like furniture, clothing, white goods, toys and even paint" [10]. "Ubiquitous Communication enables these objects to communicate with each other and the user by means of ad-hoc and wireless networking" [10]. An Intelligent User Interface enables the inhabitants of the Ambient Intelligence environment to control and interact with the environment in a natural and personalised way.

In [11] Martin Herman describes a Smart Space as having the following characteristics:

1. It may perceive and identify users and their actions and goals.
2. It may use speech, natural language, computer vision, and other perceptual user interfaces.
3. It provides interaction with information- rich sources.
4. It enables devices carried or worn into the space to be easily integrated with the devices present in the space.
5. It provides extensive information presentation capabilities.
6. It understands and anticipates user needs during task performance.
7. It provides for distributed and local collaboration, including collaboration with field personnel and mobile workers.
8. It provides improved memory and summaries of activities and deliberations for later use.

The goal of smart spaces is to place people at the centre of the environment while the computing infrastructure should be almost invisible and people should also be able to interact with the environment in a natural way. The ability to adapt the computing infrastructure to the environment and individuals as opposed to adapting the individuals and the environment to the computing infrastructure is key. Advances in the areas of Human Computer Interfaces and wireless communication have been important to enabling this. Further improvements are expected in the next number of years especially with the cost of embedding intelligence into devices that will enable items such as the smart toothbrush etc.

Challenges that smart spaces, as a consumer technology will be facing are user acceptance and ensuring that issues such as privacy and security are addressed appropriately.

2.3.1 Services provided in Smart Spaces

Many of the research projects being undertaken are to provide services such as the Smart Home (EUNICA Project), Smart Living Room [SLR] and Smart House [inHaus]. These projects create functionality in the environment that can partially automate the management of the smart space and enable remote control from an Information Appliance such as a Personal Digital Assistant (PDA). Examples of such functionality are light management and heating management. This functionality can then be further aggregated to provide a service in the environment such as energy management over the entire smart space.

2.3.2 Physical Resources in a Smart Space

The generic smart space environment will consist of a large number of embedded sensors to monitor and gather information from the environment. These sensors may take the form of heat sensors, motion detectors or more advanced vision based systems. In the future it is expected that almost every device in the smart space will have at least an embedded sensor. The smart space will also contain more powerful computing devices with extra functionality over sensors and detectors. These computing devices may be embedded in control systems for everyday appliances or maybe attached to information consoles. In almost all of the current projects there is at least one very powerful computing device that controls a majority of the intelligence in the environment.

To facilitate the communication of the multitude of devices each smart space contains a network infrastructure. This infrastructure can consist of wired or wireless technologies. Static highly intelligent devices will usually be connected together using a wired infrastructure while more mobile or less intelligent devices may be connected to the environments network using wireless protocols such as Bluetooth [BLUE] or a protocol from the 802.11 family.

Many of the smart spaces e.g. inHaus [INHAUS] also make use of mobile telecommunications technologies to provide additional functionality in the environment. In the case of inHaus this extra functionality is the ability to control and monitor the house from a specially equipped car.

2.3.3 Use of Context in a Smart Space

For a smart space to be effective it must be able to react appropriately to the users changing requirements at every situation, i.e., the smart space must exhibit situational awareness. A key part of developing situational awareness in a smart space is to be able to determine the context of a situation. Context can consist of attributes such as Time, Location, Identity or Role and Resource and this information must be retrieved and analysed before any given situation can be reacted to. The time data may be analysed to determine for example if it is currently daytime or nighttime, if the subject is currently during scheduled work or leisure hours. The location information must be analysed to determine the subject's current location in physical space. The Identity or role data can be used to determine the subject's interests, preferences or objectives. That information can then be used to determine if the subject can play a specialist role in an activity that is currently in progress. The resource being accessed can be used to determine the subject's current task or objective and to adapt this resource if necessary.

2.3.4 Non-Physical Resources that support Context

Non-physical resources in the Smart Space are those that support context such as a calendar for time management, a task for role management or an event for event management. It is necessary that the information on each one of these resources is correctly modelled and stored in an appropriate repository so as to facilitate access from appropriate software applications.

2.3.5 Software Structure

In many of the research projects intelligent agents control the objects that are resident in the smart space. To facilitate the operation of Intelligent Agents it is necessary to disseminate the information collected in the environment to the appropriate agents. An appropriate format for the modelling of the information for transfer between agents is also necessary. Extensible Mark-Up Language (XML) is used in many of the research projects often combined with a Tuple Space for sharing information. MIT Intelligent Room project has developed an extension to the Java programming language called MetaGlue that provides the following additional capabilities [12]:

1. Configuration management
2. Establish and maintain the configuration each agent specifies
3. Establish communication channels between agents
4. Maintain agent state
5. Introduce and modify agents in a running system
6. Manage shared resources
7. Event broadcasting
8. Support for debugging

This dissertation is not be concerned with the inter agent communication but rather examining the directory service as an appropriate medium for storing data to aid in the management on the smart space.

2.3.6 Information Management

However it is necessary to ensure that any information collected and processed by the Smart Space management system is not accessible to agents who may abuse the information. It is also necessary to protect the privacy of the individuals who occupy the Smart Space, the occupants should not feel like they are under constant surveillance and should be comfortable that the data being collected on them is secure and controlled appropriately.

2.3.7 Application and Service Discovery

When a device moves into an ad hoc network it must have the ability to discover the resources and services in the ad hoc network that it has joined. A number of technologies have appeared which provide solutions to this problem. The three most popular are JINI [JINI], Salutation [SALU] and Universal Plug and Play [UPnP]. In [13] the list of capabilities of the that these technologies should provide are:

1. Ability to announce its presence to the network
2. Automatic discovery of devices connected to the network
3. Ability to describe its capabilities and understand the capabilities of other devices
4. Self configuration without administrative intervention
5. Seamless inter-operability with other devices

While these capabilities provide resource and service discovery a higher-level management technology needs to be deployed to facilitate the operation of the smart space as a whole.

2.3.8 Network and Application Configuration

In [14] many of the issues involved with configuring networks and applications for smart spaces which will contain hundreds of computing devices per person are examined. It is argued that such networks must be auto configuring, capable of self-organization and automatic resource and service discovery. The first two characteristics can be facilitated using the principles of policy-managed networks.

2.4 A Survey of Management in Smart Spaces

The following is a review of the technologies used to manage the smart space environment in four sample smart space applications. The four sample smart spaces that are being reviewed are:

1). MIT Artificial Intelligence Lab's Intelligent Room Project [12]
2). Ubiquitous Smart Spaces GeorgiaTech [16, FCE]
3). Distraction Free Ubiquitous Computing Carnegie Mellon [15]
4). MUSE Project at UCLA [MUSE]

These four projects will be introduced in order to provide an overview of how technology is currently being applied to the management of smart space environments.

2.4.1 MIT Artificial Intelligence Lab's Intelligent Room Project

In the MIT Artificial Intelligence Lab's Intelligent Room Project MetaGlue, the Java programming language has been extended to provide basic constructs that are suitable for the intelligent environment.

"Intelligent Environments that consistently and most importantly, predictably understand a small subset of interactions are far preferable from an HCI (Human Computer Interface) perspective to ones that always leave users guessing if some particular input will be understood. For example, in Hal, five C-language-based computer vision systems, each producing several hundred dimensional data vectors at a rate of up to 30 a second, all connect to a MetaGlue-based visual event classification system which must process all this data in real-time" [12].

An internal SQL database is used in MetaGlue for storing information about an agent's modifiable parameters. These parameters are referred to as Attributes. The database stores the agent's internal persistent state, and gives agents fast, powerful database access. The attributes represent information that might otherwise need to be represented in code in the agents and difficult to modify. An example would be what workstation the agent needs to run on or other parameters that would influence how the agent completes it tasks. MetaGlue also makes use of a directory called a Catalog where agents automatically register when they are started. The catalog can then be used to locate other agents.

Agents access the internal database by using Metaglue's freeze() and defrost() constructs. These constructs allow an agent to store and retrieve their fields from Metaglue's SQL database. The agents generally exhibit this behaviour during their shutdown and start up procedures. MetaGlue can manage and store other aspects of an agents state, e.g., their connections to other agents. Therefore these parameters do not need to be managed by the agents themselves. As of yet, there is no well-defined schema for capturing the global state of all of the agents in a running MetaGlue system. MetaGlue also provides a web-based interface that allows the values of Attributes to be updated even when the agents are running.

Agents can register with other agents in the system, including the MetaGlue system agents, to discover which events are taking place in the MetaGlue environment. "For example, an agent in Hal interested in greeting people by name when they walk inside the room, simply registers with the vision-based Entry agent to request messages about entrance events where the identity of the person can be determined. When these events occur, it receives a message and uses the agent offering speech synthesis capability to say hello to them" [12]. Event broadcasts can be used to notify groups of registered agents about context shifts in room applications.

MetaGlue makes use of a set of agents called "dealer agents" that are responsible for distributing resources to the rest of the system. Many different dealer agents have been developed and are available for use. The internal logic of these agents differs in the algorithms they use to perform their tasks. MetaGlue programmers can extend these agents to customize their operation or use them as they are currently written. Dealers can give out resources or withdraw previously allocated ones in order to redistribute them. The behavior of the agents is based on any of several priority and fairness schemes.

2.4.2 Ubiquitous Smart Spaces Georgia Tech

The Ubiquitous Smart Spaces project at Georgia Tech centers on a project to develop a Context Toolkit. "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves" [16]. A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task. A context aware system is one that uses the context information to provide additional information or services to a user, which should assist the user in completing their task.

In [16] three categories of features that a context-aware application can support are listed:

1). Presentation of information and services to a user

2). Automatic execution of a service for a user; and

3). Tagging of context to information to support later retrieval

The architecture of the Context Toolkit is built on the concept of allowing the applications to obtain the context information that is required without requiring any knowledge of how that context was retrieved or sensed from the smart space environment. A context widget that implements this concept is responsible for acquiring a certain type of context information and it makes that information available to applications in a generic manner, regardless of how it is actually sensed. Applications access context information from widgets using poll and subscribe methods.

2.4.3 The Distraction Free Ubiquitous Computing Project

In this project the architecture used represents user tasks as entities. A user proxy object called an Aura can then use models of user tasks to set up, monitor and adapt the computing environment. An implementation of this architectural framework is currently being used as a central component of Project Aura. Project Aura is a campus-wide ubiquitous computing effort at Carnegie Mellon University. The architectural framework for ubiquitous computing applications has the following key features: "first, user tasks become first class entities that are represented explicitly and autonomously from a specific environment. Second, user tasks are represented as coalitions of abstract services. Third, environments are equipped to self-monitor and renegotiate task support in the presence of run time variation of capabilities and resources" [15].

Four component types are used: the Task Manager called a Prism represents the concept of a personal Aura. Secondly the Context Observer is used to provide information on the physical context and reports relevant events in the physical context back to either the Prism or the Environment Manager. The Environment Manager embodies the gateway to the environment. Finally suppliers provide the abstract services that tasks are composed of: text editing, video playing, etc.

The explicit representation of user tasks provides a placeholder to capture user intent. This information about the users intent is used in the search for suitable configurations in each new environment. "By representing tasks as service coalitions, the infrastructure can recognize when all the essential services in a task can be supported” [15].

The solution that is being developed in Project Aura at Carnegie Mellon University is based on the concept of personal Aura. “The intuition behind a personal Aura is that it acts as a proxy for the mobile user it represents: when a user enters a new environment, their Aura marshals the appropriate resources to support the user’s task. Furthermore, an Aura captures constraints that the physical context around the user imposes on tasks” [15].

Suppliers are registered with the local Environment Manager when they are installed in the environment. The Environment Manager can then act as a registry for matching requests with services.

2.4.4 The MUSE Project at UCLA

MUSE is a middleware architecture for sensor smart spaces being developed at the UCLA Multimedia Systems Laboratory at the UCLA Department of Computer Science. The motivation behind the project is to create a middleware infrastructure, which can then be used to build applications and services for smart space environments.

The MUSE infrastructure takes the approach of representing sensor devices and contextual data providers, i.e. those who serve up contextual data derived from sensor data as services. These services belong to a community or group of services whose members may vary over time. In MUSE there are many types of services however there are two primary services which are sensor services and sensing services. Sensor services are software entities that represent a single sensor or possibly a sensor array in a service community, while sensing services are services that derive contextual understanding from the raw sensor data.

“The word MIRA is short for "Multimedia Internet Recorder and Archive." It is a subproject of MUSE being developed to add a memory component to sensor smart spaces. Sensor smart spaces provide a wealth of data that can be recorded, both sensor data from sensor devices and contextual data derived by fusion services. Such data, however, is inherently noisy, making traditional databases unsuited for managing and performing queries on such information. Traditional databases almost always assume that the data they manage is deterministic and 100% believable. Queries on sensor data and derived context data will need to deal with the non-deterministic nature of such information. Work on MIRA is being done to facilitate such a system as well as provide syntax for queries whose results or meaning are non deterministic in nature" [MUSE]. MUSE currently uses JINI as its connectivity layer; JINI will be further examined in chapter five.

2.5 Summary

This chapter has given an overview of the management technologies that will be applied to the management of smart spaces in the dissertation. The DEN initiative and the CIM information models were introduced and evaluated. Policy based networking was also discussed. Background information was then given about smart spaces and the components of a smart space system. Finally a review of management technologies in four smart space research projects was presented.

Chapter Three

3. Issues in Smart Space Management

3.1 Introduction

This chapter will examine the current management standards and concepts that are being used in existing enterprise networks. The components of a typical network management system will be discussed. The differences between smart space management and the management of an enterprise network will be evaluated. Finally the requirements for a smart space management application will be reviewed.

3.2 What is Network and System Management?

Management is defined in [17] as monitoring and controlling the resources in computers, the resources used in the connection and communication of computers, and the applications used in the computers. Network Management can be defined as OAM&P (operations, administration, maintenance and provisioning) of network services [18]. The goal of network management is to ensure that the users of a network receive the information technology service that they expect. Network management is concerned with the network resources such as hubs, switches, bridges routers, and gateways and the connectivity between them via a network. System management is the management of the systems and resources in the network. Many of the lessons learned from the development of systems management for enterprise networks can be reused when it comes to developing a smart space management system. Existing management models and how they are used will be examined in this chapter. These principles will then be applied to the management of a smart space. The information that is required to support a smart space management system will then be discussed along with the other characteristics that a smart space management system should possess.

3.2.1 Who are the Organisations defining Management Standards?

There are a number of different Organisations that are responsible for defining network management standards. The following table outlines the organizations and the management protocols or models:

Organization	Protocol or Standard
IETF	Internet Management Model Simple Network Management Protocol (SNMP)
ISO	OSI Management Model Common Management Information Protocol (CMIP) Common Management Information Service (CMIS)
ITU	Telecommunications Management Network (TMN)
DMTF	Common Information Model (CIM) Web Based Enterprise Management (WBEM)
TMF	Shared Information and Data Model (SID) Next Generation Operations Software and Support (NGOSS)

3.2.2 What are the management Models?

A number of different architectures are used to implement management systems. The most commonly used architectures are the Peer-to-Peer model and the Hierarchical model. In the peer-to-peer model the management system and the managed agent communicate directly with each other, in the hierarchical model a number of different layers of management systems are used. Only the lowest level management system communicates with the managed agents. Almost all of the models can facilitate a proxy model as a feature where a more intelligent agent can communicate with the management system for a device with very low processing capability. The proxy device translates the management commands for the target device.

3.2.3 What are the standard components of a management system?

The basic components of a management system are a management protocol, a data store, a data model, and a management station. Optionally the system may include a policy modelling schema, a policy repository and a policy distribution protocol. The management protocol defines the operations that can be carried out by the management system. The data store is used as a repository for the management information. The management information must correspond to the data model that has been defined for the information. Codd [19] defines a data model as

- A collection of data structures,
- A collection of operators, and
- A collection of integrity rules.

and McLeod [20] defines a data model as

- A data space: a collection of elements and relationships among the elements,
- A collection of type definition constraints to be imposed on the data,
- Manipulation operators support the creation, deletion, and modification of elements, and
- A predicate language used to identify and select elements from the database.

The management station is used as the human interface to the management system.

3.2.4 What are the Standards and Protocols?

The Internet based network management model defines the Simple Network Management Protocol (SNMP) [17, 18], this model is promoted by the Internet Engineering Task Force (IETF). SNMP specifies management operations, structure of management information (SMI) [17, 18] and pre-defined managed objects (MIB). SNMP is the mostly widely supported management protocol with virtually all-major vendors of network devices building SNMP support into their products. In this management model managed agents are linked to the managers by the SNMP protocol that defines simple operations that the managed agents may carry out. The structure of management information (SMI) defines the general frameworks within which Management Information Bases (MIB) can be constructed. The SMI defines the data types in the MIB and also how the resources within the MIB are named. The management information base is a structured collection of objects that represent a managed resource.

OSI network management [17, 18] is specified by the ISO and is based on the Common Management Information Protocol (CMIP) and Common Management Information Services (CMIS). The OSI management model is based on object-oriented technology with the aim of being a very flexible model. However it is a far more complex protocol to use than SNMP and this is the reason it is not as widely deployed as SNMP. The managed object is object-oriented, in contrast to the scalar representation of the managed object in SNMP. The OSI management architecture consists of seven messages representing seven services, called Common Management Information Service Elements (CMISE). The OSI network management model uses the CMIP protocol to communicate between the managed object and manager. The seven messages can cause the managed agent to carry out actions that allow the management system to control the device the agent represents.

The Desktop Management Task Force (DMTF) is attempting to bring the various management technologies under one umbrella called Web-Based Enterprise Management [18]. The aim is to integrate the different management protocols; the implementation is based on the Common Information Model (CIM). Web-Based Enterprise Management consists of five components: Web client, Common Information Model Object Manager (CIMOM), CIM Schema and specific management protocol providers. These components can be used together to create a seamless management system that can control a network in which various network elements need to use differing management protocols.

The Directory Enabled Networks (DEN) Initiative originally extended the Common Information Model (CIM) to model network elements, services and policy. The schema and information model foundation is defined for representing these devices, which allows vendors to model specific devices and mechanisms for interoperability. The goal of DEN is to allow the network to be managed as a whole and provide an overall view of the network through the directory service. The DEN specification has now been integrated into the CIM specification with the Network model being integrated into the CIM Physical Information Model, the policy model being integrated into the CIM Core Information Model and the user model being integrated into the CIM User information model. DEN-ng is the current iteration of the DEN specification, which is currently under development.

3.3 What does the term "manage a smart space" mean?

The organisations, models and standards discussed above have historically been primarily concerned with the management of telecommunications or enterprise networks. However it now becomes necessary to create a suitable framework for the management of smart spaces. Whether this will be an extension of one of the existing models or whether an entirely new management framework will be developed will become clear over the next number of years. The goal of network management used above still holds true for smart spaces as the smart space is primarily concerned with providing service to the users or inhabitants in the manner that the service is expected. The definition of smart space management could be "the monitoring and controlling of the resources in a smart space, the resources used in the connection and communication of devices in the smart space, and the applications used in the smart space in order to aid a smart space inhabitant complete his/her tasks efficiently".

3.4 Similarities between Smart Space management and network management?

Despite the differences between a smart space and an enterprise or telecommunication network there are a number of similarities that can be taken advantage of when beginning to create a model for the smart space management framework. This dissertation is not concerned with developing a full smart space management framework but rather in whether a directory service can aid in the development of a smart space management system and what types of data need to be represented in the directory service to support that management system. Lessons from other management models can be reused while researching this topic.

Each of the management models mention earlier is concerned with the gathering of data from network devices and system resources. The complexity of the protocols used to gather the information from devices and resources differs in each of the management models. Likewise any smart space management model will need to identify what information it needs to gather and what protocol it should use to collect this information.

The control of the devices and resources in the network is also approached in different ways in each of the models. A smart space management framework will need to develop a protocol or model to control the devices and resources. A policy-based model could be used with an appropriate protocol for distributing the policies and ensuring compliance with the policies. These policies will also need to be modelled and stored in an appropriate data store.

The storage and modelling of the data collected also varies from model to model, in this dissertation discusses the development of a system where the storage of the collected data will be in a directory service. The information that needs to be stored, and how this information should be modelled for storage in a directory service is discussed during the rest of this chapter.

3.5 What is needed to manage a Smart Space?

The fundamental components that are needed to manage a smart space are similar to those in any management system. Firstly a management protocol is required which will collect and distribute data to and from the managed nodes. A data model is required which will specify what and how the data to be collected and distributed is defined. This data must then be stored in any appropriate data store for processing by the management system. A management station or interface is required to allow the examining of the data collected from the managed nodes as well as facilitating the changing of system parameters.

In a smart space environment the management system must facilitate the automatic update and configuration of the devices and resources in the smart space. In order to do so it must support an active or event based system that responds to changes in the smart space. The response of the system may be controlled by policies. To allow this to happen the smart space management system must have a model for developing policies, a policy repository and a protocol for enforcing the policies on the appropriate nodes. The management station should also facilitate the creation and updating of policies.

The smart space management system must provide a mechanism for controlling access to the information stored in the data repository as unauthorized access to this information may compromise the privacy of an individual who is using the smart space. The access control mechanism must be intelligent enough to permit easy access to the data by the appropriate nodes or agents who require it.

A smart space management system must be selective about the information that it processes, the multitude of sensors and devices in a smart space could easily overflow the management system unless appropriate filtering of the data takes place. This would suggest that a hierarchal approach might be an appropriate model for any smart space management system.

3.5.1 What different management components are needed to manage a Smart Space?

The user is the centre of all activity in the smart space. Therefore in order for the management system to function correctly it must store information about the user. Existing network management systems already store certain information about users, however given that the user is such an integral part of the smart space the representation used must be far richer than that in current network management systems.

The objective of a smart space is to allow the user to complete his or her tasks efficiently. In order to do that the smart space must be able to store a representation of the users task. All the data that the smart space needs to identify and assist in the completion of the task must be available in an appropriate format.

Each smart space will consist of a number of services that the users of the smart space can avail of in order to complete their desired tasks. The smart space management system must control access to the services and ensure that the services are configured appropriately for each user. In order to complete this successfully information on each service and the appropriate configurations must be available to the management system.

Discovery of services in the smart space will be necessary for each device and user that enters the smart space. Supporting this service discovery by new and returning devices is a feature that the smart space management system must support. Many technologies such as JINI and UPnP (Universal Plug and Play) are already available to support service discovery the smart space management system must integrate easily with these pre-existing services.

The smart space management system must have methods to perform conflict resolution and resource management in the smart space. These tasks are important and can be controlled for example by a policy-based system. With out these tasks users maybe unable to access services when they expect to because of their roles or group membership.

The smart space management system must respond differently in each situation depending on the context of that situation. Representing the context of each situation and determining appropriate matching rules for context are important features for the smart space management system. Context will not always be the same set of characteristics for each situation; therefore the representation of the context must be very flexible and expandable.

3.5.2 Functional Requirements

The functional requirements of a management system are directly related to how the system carries out its core functions. In a management system the functional requirements include

1). Representation of the data, the format used for data representation must be adequate to store sufficient information to facilitate the management system

2). Storage of the data, the data must be stored in such a manner as to allow easy access by all relevant applications

3). Retrieval of the data, the management system must support methods to allow the data to be queried and retrieved efficiently

4). Update of the data, the management system must provide methods to facilitate the easy and efficient update of the stored data

5). Replication of the data, the management system must facilitate the replication and distribution of data over multiple locations

3.5.3 Non-Functional Requirements

The non-functional requirements of the management system are the requirements that the system must satisfy that are not directly related to the how the system carries out its core functionality

1). Performance, the system must perform to a consistently high-level under the expected load

2). Reliability, the system must be available for use when required

3). Scalability, the management system should be designed in a manner that allows the system to increase in size easily over time

4). Openness, the management system should facilitate the easy integration of new components over time

3.6 Information Management issues in a Management System

Any management system must have a data store; the data store could be a relational database system, a directory service, or another type of data store. Choosing the correct data store is important because it will influence how the system performs and how open and expandable the system is. However before a designer of a management system can choose the data store the data which is to be stored and the data model for that data must be identified. The data model defines how the data that is stored for the management system is represented. The data model can be data store independent however different types of data are more amenable to being stored in certain types of data stores.

Once the data store and data model have been chosen the access control rules for the data stored must be defined. This is to protect the integrity and confidentiality of the data stored. In a smart space where new devices may enter the environment every hour controlling the access of new devices to information is an extremely important challenge. The management system needs to retrieve the data that it collects quickly and efficiently, therefore appropriate query mechanisms must be defined on the data store. Once the data has been collected and stored securely it also needs to be distributed or replicated to any other locations that need access to that data, this may need to be done on a real time basis or at scheduled intervals.

3.6.1 Analysis of the Information that is needed to manage a Smart Space

The smart space management system will need to store diverse types of information. Devices and services made need simple text based configuration parameters, which will need to be stored and downloaded to the devices at various intervals. These configuration parameters may be updated only at very long intervals e.g., every six months. Text based policies may need to be stored to facilitate the management of the smart space; such policies may be updated on more frequent cycles. Other devices in the smart space may need much more complicated information stored, e.g., a projector may need to have an audio – visual presentation which is to be shown during a lecture stored and downloaded to the device before the presentation begins. Obviously one single data store may not be sufficient for all of the types of information that must be stored.

3.7 Synthesis

The components of a typical management application were introduced at the beginning of this chapter. An analysis was performed of what different components would be required to manage a smart space environment. Using this analysis as a basis the requirements for a smart space management system were developed. Finally the types of information that a smart space management system must deal with were examined.

Chapter Four

4. Prototype Design

4.1 Overview

The objective of this dissertation is to determine the suitability of a directory based management system to the task of managing a multi application smart space environment. This chapter will outline the design choices that have been made to implement a directory based management system prototype in order to correctly identify the ability of such a management system to perform efficiently in a simulated environment. The most important choices that must be made relate to what data should be stored in the directory service, how this data should be represented and what should the architecture of the management system be to leverage this data. The smart space environment that was chosen is that of a smart community space. A community centre is an area in which many different activities and events occur. The participants in these events can be members of the community centre, invited guests with a specialist role during an event or activity, a visitor to the community centre or have a multitude of other roles. The community space can be used as a public meeting facility, an education centre, and a social centre or even as a location where sporting events take place. All of these factors combine to make the environment a very diverse and dynamic one, in which any successful management system will need to be very flexible.

4.2 Requirements

The previous chapter outlined many of the characteristics of a generic management system. These included the functional and non-functional requirements of a management system. The management system for the smart community space prototype must also meet these generic requirements. The specific functional requirements of the management system for the smart community space must also include:

- Member registration, the community space must facilitate the registration of full members, guest members and visitors

- Organization and club registration, the smart community space will have a number of affiliated clubs and organizations whose members may have specialist rights in the smart space environment
- Event and Activity Creation, any smart community space will have a large number of diverse events occurring, the management system must be aware of the nature of these events
- Service Creation and Management, the smart community space will provide a number of services which the members can avail of, these services must be registered with the management system
- Resource Reservation, the members may have the ability to reserve rooms or equipment in the smart community space
- Infrastructure Management, the Management System must also be able to manage the underlying infrastructure of the smart space, this includes the power, communications, and basic services such as light and heating.

These requirements along with the generic requirements of a management system are the foundation for the design of the smart space management system prototype. The design must also take into account the fact that the management application will be operating in a highly dynamic environment in which new services, events and resources can be added at any time and into which many different people and devices will be entering and leaving.

4.3 Assumptions

In order to achieve the objectives of the dissertation in the time period available it has been necessary dedicate more time on certain areas of the research than others. In this section some of the issues that could not be fully explored in the time available have been listed.

4.3.1 User Tasks / Artificial Intelligence

The smart space management system must have the ability to represent user tasks if the system is to successfully aid the user in completion of those tasks. However for the purposes of this prototype the management application will not have the ability to learn or detect user tasks automatically rather, these tasks will be already programmed into the management system. Discovery of user tasks in the smart space environment is a very necessary function of the management system, however to achieve that function successfully a form of expert system or artificial intelligence may be required. This prototype concentrates on whether the directory service has the ability to store and represent the user's tasks effectively.

4.3.2 Integration with existing Technologies

Current research on service discovery in smart space environments is centred on four main technologies, JINI, Chai, Salutation and UPnP. This dissertation will not be examining how initial service discovery and access control is performed in the smart space. Instead the focus is on how the smart space management system can control access to resources, activities and services after the initial service discovery has been performed using any of the above technologies. For maximum flexibility the information model and architecture used by the management system should be independent of the communication technology used in the smart space environment. Once the system architecture has been outlined an overview of how the initial service discovery could be performed with each of the four aforementioned technologies will be given.

4.3.3 Security

There are a number of security challenges in developing a management application for a smart space environment which include, protecting the integrity and confidentiality of the information stored in the management system, ensuring that only correctly authenticated and authorized devices access the management system and ensuring the integrity and confidentiality of data transferred between the management system and the managed agents. Fully exploring all the topics relating to the security of the management is beyond the scope of this dissertation, however the management system needs to be designed in a security conscious way from the beginning and directories services have been shown to support such security conscious applications in many existing network applications.

4.4 Directory Design

The DEN initiative is the foremost management initiative promoting the use of directory services as the foundation of management applications. In reusing the basis of the DEN initiative as the beginnings of the smart space management application the design and implementation benefits from the years of research and development that has gone into the DEN initiative. The DEN initiative specifies the information model for representing management information and it also contains a mapping of this information model onto an LDAP schema. LDAP is an access protocol used for accessing directory services.

The design of the directory schema is the first step of the prototype design. The directory schema contains a specification of the objects that can be stored in the directory service. The directory schema must be extended to contain object classes that will represent the information that is required for the management of the smart space. From an examination of the requirements presented in the previous chapter the information that the directory service is required to store includes the following items:

- Member Information
- Organization and Club Information
- Resource and Reservation Information
- Event and Activity Information
- Service Information
- Infrastructure Information

However this list of information is not complete the management system will also need to store other types of information including:

- Policy Information
- User Tasks and Objectives
- Context Information
- Authentication Information
- Logging Information

In the following sections an outline will be given of which data is appropriate for storing in the directory service and then the approach taken to map this data onto the directory schema will be outlined.

4.4.1 Modelling Users and User Profiles in the Directory

The DEN Initiative provides an information model for representing user information in a management application. This information model was used as the basis of the information model for representing member data in the directory. The user information model specified by DEN uses the inetOrgPerson LDAP object as the basis of the LDAP mapping. The attributes contained in this mapping were sufficient to provide the basis for the creation of an object in the directory service that would represent a smart community space member. The CIM standard allows for the object classes provided in the information to be further sub classed for use in specialist applications. The User Information Model is particularly concerned with representing the user as it relates to security and user roles in a network environment. The smart space prototype that was developed was not designed to fully explore the security aspects of the information model.

However the object class, which represents the user in the CIM User Information Model, does not by default support the representation of profiles for a user for the numerous services that may be available in a smart space environment. A user profile is required for each member of the smart community space for each service or activity in the smart space, which must contain all the preferences and characteristics of that member in relation to that service. However this profile does not need to be stored as a single object in the directory service rather by combining attributes from different directory objects the overall profile can be created. In effect the directory object which represents the user does not need to store all the users preferences for each of the services and activities in the smart space, rather each service object can contain details of the preferences of each user who has used that service during they the time which they have been registered with the smart community space. This strategy allows new services and activities to be added to the directory schema without the requirement to update the directory object that is used to represent the user. The user object is then only required to store the fundamental information about a user.

4.4.2 Modelling Organization or Club Membership

The management application must have access to membership and role information for each user of the smart space therefore this information needs to be represented in the directory service. The users membership of different organisations and clubs can be stored in a basic directory object group object which can be associated with the directory object that represents the club or organisation. This facilitates the management application retrieving the grouping information efficiently. Role information for each service can also be associated with the directory object that represents the service, for example providing an attribute that lists the distinguished names of the directory objects which represents users in certain roles. The CIM User Information Model also provides generic classes for associating role information with the user object in the directory. Designing the directory schema in this way allows the schema to be easily extended to add new or extra services without modifications to the directory schema to change the user object.

The CIM Information models do not provide object classes to represent clubs or organizations. Therefore as smart space management applications move forward an information model needs to be created to represent this information. For the purposes of the prototype developed here clubs and organizations have been represented in the directory service by the use of an ordinary LDAP group and role information has been stored in the user object.

4.4.3 Modelling Resources and Reservations in the Directory

Device reservation needs to be represented in the directory service so as to avoid reservation conflicts. The method used to represent the reservation should allow the following operations to be easily achieved:

1). Cancel all of one particular user's reservations
2). Update a user's reservation
3). Check what reservations a user has
4). Add a reservation for a user or resource
5). Discover all free resources at a particular instance

One method of achieving this is to represent a reservation in the directory as multi-valued attribute of each resource. The attribute would contain the distinguished name of the individual reserving the resource as well as the time, date and duration of the reservation. The directory object that represents the individual reserving the resource would contain a multi-valued attribute containing the distinguished name of all resources, which that individual has reserved. This data could then be archived from the directory once the reservation date and time has passed.

Many existing directory service applications are calendaring and scheduling applications. In these applications the users schedule is stored in the directory service so that notifications can be issued prior to an event occurring and calendar conflicts can be detected during event scheduling. This functionality is still required in a smart space application.

4.4.4 Event and Activity Information

The smart space community space will have a number of events and activities that will be occurring during the lifetime of the smart space. The events and activities need to be represented in the directory. Examples of an event could be a history lecture which takes place once every month or annual meeting of a club committee. For each of these events or activities that could possible take place in a smart space environment a vastly different set of information may need to be stored. At a minimum a schedule of the occurrences of the event will need to be maintained. The next attribute that may need to be stored is the location of the activity. Then a list of authorized members may also be necessary. These three attributes may apply to each activity or event, however a lecture may have an audio-visual presentation associated with it and a meeting may have an agenda. Therefore it is difficult to create a directory object that can adequately represent all this information. Instead by taking the lessons of the CIM Information Model a base object class with the schedule, location and member attributes can be created. This object class can then be extended to represent information for the other events and activities as necessary. Eventually categories of these objects may evolve that will contain the attributes to represent certain classes of events appropriately.

4.4.5 Modelling of Services in the Directory

The services that exist in the smart community space need to be represented as directory objects also. The directory objects need to represent the preferences for each user who has used the service in order to configure the service for each user.

The directory object must also have attributes containing any information that is necessary for the operation of that service. The membership of each user can be managed as part of the service by using the directory group objects so that access control to the service can be achieved.

4.4.6 Modelling of Smart Space Communication Infrastructure

The data communication network in a smart space has all the characteristics of a traditional data communication network, however the network in a smart space is far more likely to have transiently connected wireless devices that a typical enterprise network. The CIM Information models can capture this network infrastructure and using the mapping in the DEN Initiative it can be successfully mapped on to the directory service.

4.4.7 Modelling of Policies in the Directory

The smart space management system may use policies to control the devices and services in the smart space. Policies can be used at a high level to start of the execution of other policies or at a lower level to control individual devices. The DEN initiative contains an information model for modelling policies that is mapped onto an LDAP implementation. The prototype application can reuse this policy model as part of its implementation. This policy model permits the easy representation of the policy actions and conditions in the directory service. The policy model can then be extended to represent other policy implementations as they are developed, which allows the management system adapt to future advances in policy based technology. The directory service stores all of the policy information for the management application. These policies can be updated easily and efficiently in the directory service. The directory service is also capable of notifying the management application when the policies are updated.

4.4.8 Modelling of User Tasks in the Directory

The focus of every smart space is facilitating users in completing their tasks more efficiently. For the management system to achieve this the tasks of each user must be represented in the directory service. Every task that a user is trying to achieve in the smart space should be related to a service, an event, an activity or an object in the smart space. In this way the smart space management system can manipulate the smart space environment to assist in completing that task successfully. If a task does not relate to the smart space environment the management system cannot aid in the completion of that task.

The management system makes use of policies to manipulate the smart space environment therefore a user's task can be represented as a set of policies which control the manipulations of the smart space to achieve that task. In this way tasks can be modelled in the directory service in a similar way to how policy information is represented in the directory service. The smart space management system can then make use of these policies to enable the users of the smart space environment to successfully achieve their tasks.

4.4.9 Modelling of Context in the Directory Service

The management system in the smart space environment must be able to determine the context of situations before it can successfully react to those situations. In order to determine the context of the situation the smart space management system must have access to all the attributes that make up the context, such as time, location, identity and resource. However these attributes can be determined from the smart space environment and attributes of the various objects in the directory service. It is therefore not necessary to create a directory object which represents the context of any situation and which is updated and stored in the directory service as the context changes.

4.4.10 Logging Information

A management system may want to store historical or statistical information about the use of the smart space. However simply because a directory service is being used to support the management application it does not mean that all data should be stored in the directory service. Information such as counters and historical logs etc. may be best stored in a relational database or flat file system. The directory service can provide a reference link to the location of this information.

4.4.11 Authentication Information

As mentioned in the assumptions section of this chapter the security aspects of the directory service would not be researched in this dissertation. However many implementations exist of applications which use a directory service to store authentication information.

4.5 Storing of Complex Data

On occasions where it is necessary to store complex data such as multimedia data, which may be related to a user's task, such data can be stored outside the directory service and a reference link provided to it as an attribute of the directory object. As an example in a situation where it is necessary to download a presentation to an audiovisual projector for display during a lecture, an attribute of the directory object representing the projector or display could contain a URI to the presentation. The projector must then implement it's own functionality to retrieve the presentation from this location and display it in an appropriate format for the content. The policy, which is assigned to control the entire activity of the presentation, should initiate the projector locating and processing the remotely stored file.

4.6 Policy Languages and Representation

One popular policy language currently being researched is Ponder. Ponder has grown from a number of projects on Policy Based Management at Imperial College in the UK [22, 23]. In Ponder policies are defined as one aspect of information, which influences the behavior of objects within the system, they are specified as objects that define a relationship between subjects (manager) and target (managed objects) [21]. A number of further research projects are currently being carried out to further enhance Ponder [PNDR]. However these projects are primarily concerned with issues such as security, resource management and quality of service as they apply to traditional enterprise networks, while beginning to appreciate the increasing mobility that is becoming a feature of those networks. Work on the development of a policy language, or extensions to an existing policy language, specifically for smart space environments is not being carried out. To support the creation of management applications for smart spaces a sufficiently expressive policy language to manage smart spaces is required. In this dissertation a minimal policy language with enough representative power to suffice for the prototype application was defined, however it will not be detailed here as it was only intended for this one off purpose.

4.7 Application Design

Once the data to facilitate the management system has been stored in the directory service it is necessary to implement a management application that will process that data. Given the complexity of the smart space environment and the many differing services, events and resources in the smart space it is unrealistic to expect one large application to manage this environment. Rather the directory service can be used as the foundation for multiple similar applications that can be used to manage the environment in a coherent and seamless way. Taking this approach allows multiple vendors to easily integrate their devices into this framework.

These applications must read in the policy data from the directory service relating to the devices that the management application is responsible for controlling. The applications must then monitor the environment, the devices and directory service for changes that occur which will cause policies that are currently active to become inactive or necessitate new policies to become active in the smart space.

These applications can become aware of changes in the environment through changes in the directory service of which the application will be notified through technologies such as persistent search, which the current generation of directory servers are supporting. Alternatively changes in the environment can be notified to the applications through communications from the devices through technologies such as JINI or UPnP. In the prototype application events in the directory service are notified to the management application to JINI and events in the smart space environment are notified to the application through the Java Message Service (JMS).

Every device in the smart space does not need to be a directory service client. If a particular device needs to access information from the directory service then the management application that is responsible for that device has the ability to read that information from the directory service and pass it to the device through a technology such as JINI or UPnP.

4.8 Summary

In this chapter the design of a smart space management application has been examined by concentrating on the data that the application needs to have access to. The ability of the directory service to store this information has then been examined. Reference has been made to the abilities of policies to play an important role in the creation of this management application. Finally the application architecture for the management system has been briefly outlined.

Chapter Five

5. Implementation

5.1 Overview

This chapter will outline the implementation of the prototype application, including the programming language choices and the background implementation technologies. The architecture of the application will be examined in detail and reasons for implementation decisions will be provided. Finally examples will be provided of how this application could be integrated with existing smart space communication technologies.

5.2 Prototype Scope

The prototype application involved the simulation of a smart space environment to examine the viability of the management architecture in a real world environment. The application models a number of services, events and devices that would be available in a smart community space. The following are examples of what has been simulated in the prototype:

- Services
  - Public Internet Access
  - Library Information Service
  - Gym Service
- Events
  - History Lecture
  - Public Meeting
  - Gym Attendance
- Devices
  - Light Fixtures
  - Heating Fixtures
  - Audio – Visual Equipment
  - Gym Equipment

5.3 Programming Language Choices

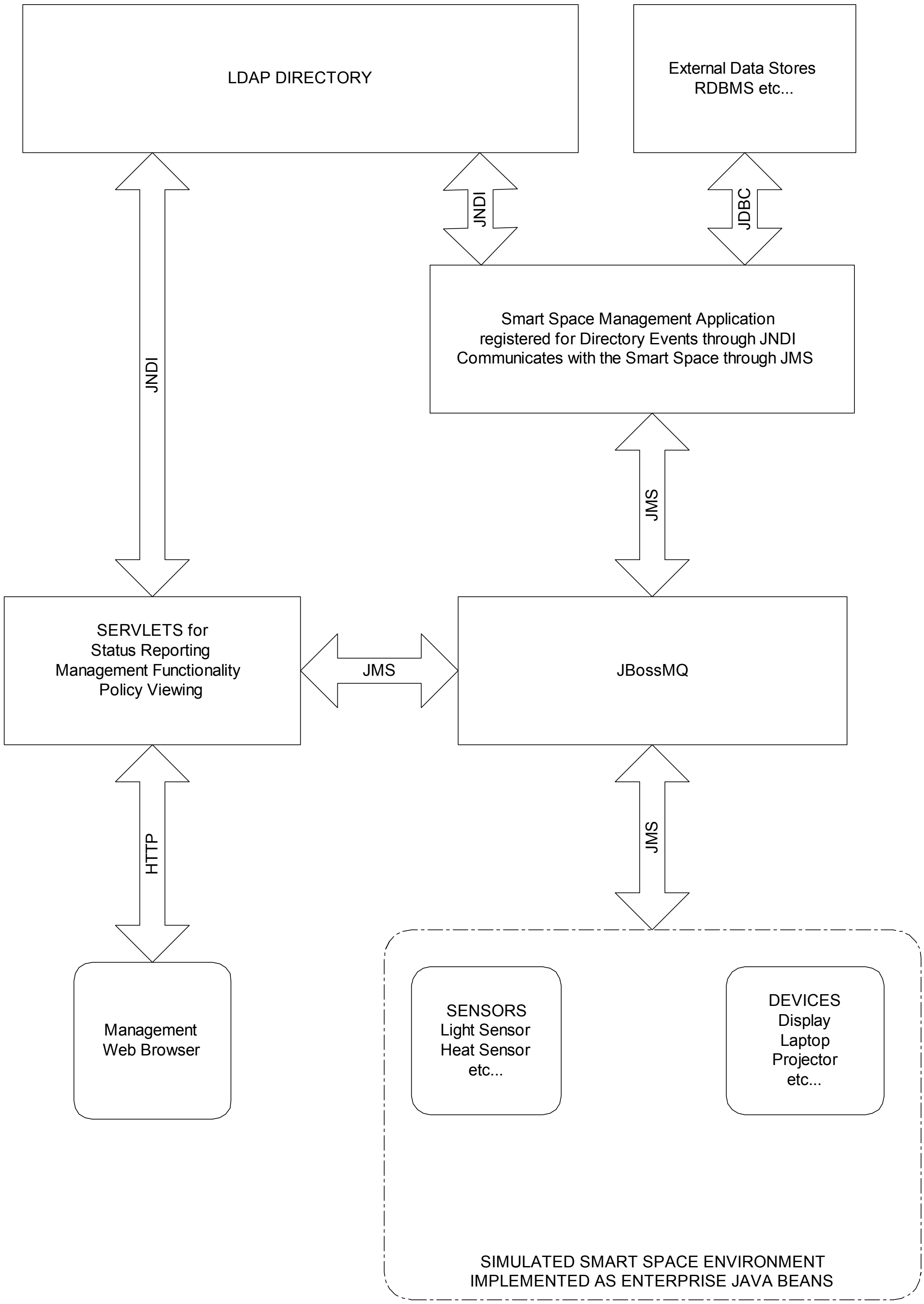
This prototype application has been developed using the Java language, however the architecture of the management system is independent of the implementation language. The prototype application could just as easily have been implemented in any other language that supports the development of LDAP clients. The Java programming language was chosen simply because of the authors experience in that development language.

5.4 Java Application Design

The management application has been designed so that initially it reads the policy information, which is stored in the directory service. Once the management application reads the policy information it proceeds to implement those policies whose conditions have been met and wait for the conditions of the other policies to be met before implementing those. The management application can determine when the conditions are met by observing changes in the directory service or by receiving notification from the simulated devices in the smart space environment. The directory service events are received through JINI and in the simulated environment the Java Message Service (JMS) is used to notify the management system of updates to the status of sensors or devices.

5.4.1 System Architecture

The following diagram provides an overview of the architecture of the prototype management application.



5.4.2 Overview of Enterprise JavaBeans

Enterprise beans are the J2EE components that implement Enterprise JavaBeans (EJB) technology. Enterprise beans run in the EJB container, a runtime environment within the J2EE server. Although transparent to the application developer, the EJB container provides system-level services such as transactions to its enterprise beans.

These services enable you to quickly build and deploy enterprise beans, which form the core of transactional J2EE (Java 2 Enterprise Edition) [J2EE] applications. The prototype makes use of Enterprise beans to implement the simulated devices in the Smart Space environment. The prototype makes use of the three different types of enterprise beans (Session, Entity and Message-Driven) to implement the various different types of devices that need to be simulated. The application server used in the development of the prototype is the JBoss Application server [JBOSS].

5.4.3 Session Beans

A session bean represents a single client inside the J2EE server. To access an application that is deployed on the server, the client invokes the session bean's methods. The session bean performs work for its client, shielding the client from complexity by executing business tasks inside the server. As its name suggests, a session bean is similar to an interactive session. A session bean is not shared - it may have just one client, in the same way that an interactive session may have just one user. Like an interactive session, a session bean is not persistent [J2EE]. The prototype application uses session beans to control the interaction with a simulated device represented as an entity bean.

5.4.4 Entity Beans

An entity bean represents a business object in a persistent storage mechanism. Examples of business objects are customers, orders, and products. In the J2EE SDK, the persistent storage mechanism is a relational database. Typically, each entity bean has an underlying table in a relational database, and each instance of the bean corresponds to a row in that table. The prototype application uses entity beans to represent the simulated devices. The relational database used as the data store in the prototype is MySQL.

5.4.5 Message Driven Beans

A message-driven bean is an enterprise bean that allows J2EE applications to process messages asynchronously. It acts as a JMS message listener, which is similar to an event listener except that it receives messages instead of events. The messages may be sent by any J2EE component--an application client, another enterprise bean, or a Web component - or by a JMS application or system that does not use J2EE technology. In the prototype application message driven beans are clients of the message service to which the management application publishes messages. Once the message driven beans receive messages they invoke sessions beans to interact with entity beans to control the simulated devices in the smart space environment. This is a very flexible model that facilitates the reuse of much code when developing simulations for new devices.

5.4.6 Enterprise JavaBeans Design Patterns

A number of standard design patterns are available for use when developing J2EE applications. This prototype application is designed to implement the message listener pattern. The pattern is designed around the use of messages to initiate and control the behaviour of the objects that entity and session beans represent in the application.

5.4.7 The Java Naming and Directory Interface (JNDI)

The Java Naming and Directory Interface is an Application Programming Interface that provides naming and directory functionality to applications written using the Java programming language. It is defined to be independent of any specific directory service implementation. Thus a variety of directories--new, emerging, and already deployed--can be accessed in a common way [JNDI]. The JNDI consists of an API and a Service Provider Interface (SPI). Applications can use the API to access the functionality of file or directory systems for which the vendors have provided a Service Provider Interface. The complexity of each of the vendor's products can be hidden behind the interface that the JNDI provides allowing application programmers to easily move their applications from one directory provider to another without re-engineering the application.

The prototype application takes advantage of the JNDI API to access the directory service. The API provides the application with the functionality to search, read and update the directory service. Functionality is also provided to the application that allows the application to register for directory events. The directory events are based on the persistent search feature of the directory service. The directory events allow the management application to be notified of changes to objects in the directory without re-querying the directory. In effect the directory service becomes active and is responsible for notifying applications of changes in the status of the directory objects. Therefore the complexity of developing a LDAP client for the directory service is simplified by reusing the components that are already available to Java based applications. The application developer is still required to understand the logic under which an LDAP based directory service operates.

5.4.8 The Java Message Service

The Java Message Service is a Java API that allows applications to create, send, receive, and read messages. Designed by Sun and several partner companies, the JMS API defines a common set of interfaces and associated semantics that allow programs written in the Java programming language to communicate with other messaging implementations [JMS]. Messaging is a method of communication between software components or applications. A messaging system is a peer-to-peer facility: A messaging client can send messages to, and receive messages from, any other client. Each client connects to a messaging agent that provides facilities for creating, sending, receiving, and reading messages. Use of the JMS API enables application developers to develop messaging applications that are independent of the underlying message service. This facilitates the easy migration of existing applications from one messaging service to another. Messaging enables distributed communication that is loosely coupled. A component sends a message to a destination, and the recipient can retrieve the message from the destination. However, the sender and the receiver do not have to be available at the same time in order to communicate. In fact, the sender does not need to know anything about the receiver; nor does the receiver need to know anything about the sender. The sender and the receiver need to know only what message format and what destination to use. In this respect, messaging differs from tightly coupled technologies, such as Remote Method Invocation (RMI), which require an application to know a remote application's methods [JMS].

The use of the JMS service in the prototype application allows for the creation of a loosely coupled architecture. The decision to use the JMS was influenced by the work of Softwired AG [SOFW] who have developed an implementation of a JMS client on the J2ME architecture. The use of the JMS facilitates the communication of the management system with managed agents and vice-versa. The management system submits messages to the messaging service that are then received by the appropriate managed agents. The agents then act on the contents of these messages depending on their internal configuration. The management system can also receive messages from the managed agents via the messaging system. These messages can influence the decisions the management system needs to take in order to control the smart space environment. The functionality that messaging service provides in the prototype application could also be provided by another technology which facilitates the production and consumption of messages by multiple producers and consumers for example a Tuple Space based technology such as JINI. The prototype application was developed using the messaging service JBossMQ, which is built into the JBoss application server.

5.5 Integration with Existing Smart Space Technologies

The following sections will examine how the prototype application could be integrated with real devices. In order for this to happen some form of communication must occur between the management application and the managed device. In current smart space research four technologies in particular are being examined as the foundation for communications between devices in a smart space environment. These three protocols / specifications are:

- JINI
- UPnP
- Chai
- Salutation

The following sections will examine how the directory-enabled prototype could be integrated with each of these four technologies. In these sections the management application is an application that uses the directory service to enable it to manage the smart space environment.

5.5.1 Initial Service Discovery and Authentication with JINI

JINI is the name for a distributed computing environment that can offer "network plug and play" [JIN2]. A device or a software service can be connected to a network and announce its presence, and clients that wish to use such a service can then locate it and call it to perform tasks. JINI is basically a specification of a set of middleware components. The specification comes from Sun Microsystems and is strongly based on the Java language. A typical JINI system consists of three main components. There is a service, a client that would like to make use of the service and a lookup service (service locator) that acts as a broker/trader/locator between services and clients.

In order to integrate the prototype with devices that wish to use JINI to communicate it would be necessary to run a lookup service. The lookup service would contain information about all the JINI services that were available in the Smart Space environment. When a client wished to available of a service it would locate that service through the lookup service. The management application for the JINI devices would be implemented as a number of JINI services which would be registered with the lookup service. Likewise the devices themselves would register the services that they provide with the lookup service. The management application would respond to changes in the directory service or the environment and would react by invoking services on the clients as appropriate. The devices can communicate with the management application through the JINI services that the management application makes available.

5.5.2 Initial Service Discovery and Authentication with Chai

Hewlett Packard developed the leading Chai Appliance Platform for embedded software in mid-range devices. Mid-range devices have a fairly powerful processor, a number of megabytes of memory available for storage and operations, a feature-rich operating system, and an option for a PC-like way of accessing functions on the display. They include printers, set-top boxes, high-end PDAs, and automotive multimedia. Chai Appliance Platform extends functionality by supporting Java programmability and Web connectivity. ChaiVM offers platform independence for application developers and allows pure Java applications to run on the devices for additional capabilities [CHAI].

For a Chai enabled to device to integrate into the prototype it would be necessary for that device to implement a ChaiServer. The ChaiServer is a http server which provides functionality such as displaying web pages which provide information on the device or providing ChaiServices that are enabled on the device. ChaiServices are services that run on a device that can be used to carry out operations on that device. The services could for example reconfigure the device, carry out diagnostics on the device or cause the device to carry out a particular function. In order for a management application to integrate with a Chai enabled device that management application must be able to support the http protocol to allow it to interact with the ChaiServer on the device.

5.5.3 Initial Service Discovery and Authentication with UPnP

The Universal Plug and Play Forum defines UPnP Device and Service Descriptions according to a common device architecture contributed by Microsoft. The Universal Plug and Play Forum is a group of 200 companies and individuals across the industry that intend to play a leading role in the authoring of specifications for UPnP devices and services. The goals of the Forum are to enable the emergence of easily connected devices and to simplify the implementation of networks in the home and corporate environments. The Forum will achieve this by defining and publishing UPnP device and service descriptions built on open, Internet-based communication standards. With UPnP, a device can dynamically join a network, obtain an IP address, convey its capabilities, and learn about the presence and capabilities of other devices—all automatically; truly enabling zero configuration networks. Devices can subsequently communicate with each other directly; thereby further enabling peer to peer networking. [UPnP]

UPnP is developed on the model of devices and control points. The protocol defines the implementation of automatic discovery and addressing therefore a lookup service is not required as in JINI. The control points can automatically discover the UPnP enabled devices in the environment from which they can learn the services that the devices provide. The control points can transfer information to the devices using the XML protocol to available of the services that the devices supply. The devices can update the control service through the event notification model that UPnP specifies.

In the prototype application the management application would be implemented as a control service that would allow it to control the UPnP enabled devices in the Smart Space environment.

5.6 What data is transported over the communication channels?

Communication in the smart space environment takes place over technologies such as UPnP or JINI because of this XML can be used in the communication messages between the management system and the devices in the smart space. XML allows the passing of simple or complex messages containing different types of data through the communication channels. The ability to use XML and communication technologies such as UPnP means that the architecture is not tied to a pure Java environment and can be easily reused with other languages.

5.7 Is it necessary to use proxies for devices?

In the instance where the device does not have enough computing power to manipulate XML and or access the communication service proxies could be used for the smart space devices. The proxy must be developed to relay information from the communication channel to smart space devices with low processing capabilities.

5.8 Summary

In this chapter an overview of the architecture of the management system has been provided. A number of the background technologies on which the simulation environment is based have also been described. Examples have been given which explain how the management system could be integrated with existing smart space technologies such as JINI and UPnP.

Chapter Six

6. Evaluation

6.1 Overview

This chapter will outline what has been learned about the suitability of applying the DEN initiative to the management of multi-application smart spaces. The smart community space environment, which has been used as the basis for the prototype application, will be used to provide example use cases to illustrate the points that are being made. The design features of directory-based applications, and what makes them appropriate for the foundation of management applications will also be discussed. A short evaluation will also be made as to the suitability of applying a policy based networking methodology to the management of multi-application smart spaces.

6.2 Objectives Revisited

At the beginning of this dissertation a set of questions was outlined which were to be examined during the dissertation. These questions were:

> 1). Is the representational power of the directory service adequate to store the required management information
>
> 2). Is the directory service dynamic enough to support the update of the management information
>
> 3). Is the use of policies beneficial in the management the smart space
>
> 4). Does the directory service support efficient retrieval of the information by the smart space management application.

The experience of designing and implementing the prototype management application has allowed conclusions to be drawn, which allow those questions to be answered. The questions will be examined separately through the next four sections in this chapter.

6.2.1 Is the representational power of the directory service adequate?

This question is attempting to determine if the directory service could represent all the information that was needed to build a smart space management application. Chapter three of the dissertation outlined the information that was needed to create a management application for a multi application smart space. Among the most important items that it was necessary to represent information for were:

- Devices
- Users
- Tasks
- Context
- Policy

A smart space consists of many diverse devices, which can be constantly moving in and out of the environment, the directory service is very capable of storing the profiles and configurations of these devices. Once the directory schema has been updated with a base set of classes, which can be used to represent a generic set of devices, this schema can be updated to represent more specific devices. The information model, which was used in the development of the prototype was CIM which is mapped onto LDAP as part of the DEN initiative. A number of classes were sub-classed successfully to represent devices in the smart space. The directory service is an appropriate medium for the storing of the information needed to successfully represent the devices in the smart space.

In a smart space the users are the most central component, if the management system is to be successfully it must have access to a full representation of each user. During the implementation of the smart community space prototype profiles of a number of sample members were stored in the directory service. Similar to modelling the devices use was made of the LDAP mappings from the DEN initiative. These mappings can be further sub-classed by additional classes, which would facilitate the directory service in storing unforeseen information about users through the lifetime of the smart space. The representation used for the members was adequate for the successful implementation of the prototype application. The decision was taken to store member profiles for each service in the smart space with the objects, which modelled the service, and not with the directory object that modelled the user. The reason for this is to provide more flexibility in the addition of new services to the smart space environment, without necessitating an update to the directory object that represented the member.

The tasks that the users of the smart space wish to accomplish must be modelled to facilitate the smart space management system in assisting with there successful completion. However prior to the tasks being represented the smart space management system needs to discover the users goals. This should take place as part of an automatic discovery process by the management system. Once the discovery has taken place the tasks can be represented in the directory in a similar fashion to a policy that the smart space management system must implement in order to assist the user in completing his or her task. In the implementation a number of member tasks have been represented and the management application can successfully assist in the completion of these tasks. An automatic task discovery system has not been implemented as this is outside the scope of the dissertation.

The model used to implement the prototype management application was based on the policy based networking model. In order to implement this model it was necessary to create a policy repository. The DEN initiative supplies a mapping of policy information onto an LDAP directory, this mapping was successfully reused as the basis of the policy mapping using in the prototype. In order to implement a policy based management application a policy language needs to be used. A simple policy language was created in the dissertation that facilitated the representation of sufficient information to create a number of policies to aid in the management of smart spaces. This language is not a fully featured policy language and its sole use will be in this dissertation however it was sufficient to examine the representative power of the directory system and DEN policy mapping for use in a smart space environment. The use of policies in smart space management will be discussed again in a later section of this chapter.

In order for the smart space management system to implement the correct policies for each situation it must be aware of the context of each situation. The number of parameters that determine the correct context for each situation can vary. In one situation time and location may be sufficient, in others the time, location, identity and activity may be necessary to determine the correct policy. Therefore the context needs to be mapped on to the conditions of the policies. Once the policy language is sufficient for the representation of the conditions of the policy the onus is then on the smart space management application to correctly determine when these conditions are met. The smart space management application must use the various sources available to it to examine if the conditions are true. The location of an individual should be available by querying a sensor in the environment and the role of that individual can be determined by querying the directory. This is the model used in the prototype application. Therefore context is not represented as one specific object in the directory but rather the management application is provided with the intelligence to use the resources available to it to determine whether the context of a situation matches the context of the policies available to it.

6.2.2 Is the directory service dynamic enough?

Much of the literature in respect of directory services outlines the fact that the original design of the directory service was for reading and searching information and not the very frequent update of this information. This is still the case and directory services perform much better for read operations than they do for write operations. For these reasons it is prudent to design directory based applications around relatively static information. Once careful design is used this characteristic of directory services does not prevent the directory service becoming the basis of smart space management applications. The designer of the application must determine what types of information are static enough for the directory service to support it without impacting on the performance of the management application. In the prototype management application, which was developed, the approach taken was to make such highly dynamic information available from the sensors in the smart space rather than the directory service. This information can then be retrieved by the management application by polling the devices or by the device notifying the management application of changes in the smart space environment.

The directory service is therefore able to support management applications that require access to highly dynamic information providing alternative data stores can be accessed by the management application. The directory service can contain links or pointers to the alternative sources of this information as attributes of directory objects.

Examples of data that maybe to dynamic to be stored in the directory service include a user's location as this may change every second depending on the accuracy of the measurement used.

6.2.3 Is the use of policies beneficial to the management of the smart space?

Policy based networking is a technology which has undergone a lot of research in the last number of years. The objective of this dissertation was not to fully examine all the issues of applying policy based networking technologies to smart space management, but rather to attempt to examine if policy based networking principles could with some modification if necessary play a part in the management of a multi-application smart space environment.

The prototype application, which was implemented, uses a policy-based system to manage the prototype environment. The prototype does so successfully, however a fully specified policy based language for the management of smart spaces needs to be developed before a longer study of the issues in using policies to control smart spaces can be carried out.

Policies can be used in two ways when managing smart spaces, the first and most obvious way is to use the policies to control the behaviour of devices in the smart space environment as the context in the smart space changes. Examples of this include downloading a member's program to a gym machine when that member enters the gym. The second way policies can be used is to change the behaviour of the management system during the lifetime of that system. An example would be modifying the directory system to change a user's role in response to an event in the smart space. In this way the management system can become far more powerful and flexible.

A policy system may also need to be used in conjunction with a form of automated learning to facilitate maximum flexibility in the management of the smart space. The management system must be able to update its behaviour as it learns from the successes and failures of the actions it takes in the smart space environment. The management system also needs the ability to learn about users tasks in a manner that does not require intervention from the everyday users of the smart space. The use of policies alone may be too rigid to allow for this adaptation over the management systems lifetime.

6.2.4 Are the query mechanisms of the directory sufficient to retrieve the information efficiently?

Directory services by their definition are designed to perform well under read operations, however this factor is not sufficient to guarantee that the retrieval of the information used to manage the smart space will happen in an efficient manner. The management data must also be stored in a method that will minimise the number of read operations required to retrieve all the data.

The data can be modelled in many different ways to minimise the number of read operations required to retrieve it. The directory service supports the creation of organizational units that can be used to categorise the directory objects into smaller groups, by doing this the number of directory objects that need to be read when performing a search on the directory can be minimized, provided it is possible to limit the search to one or two specific organizational units. Organizational units have many other uses in a directory service for example delegating security management.

The directory service provide a very capable search ability, by designing the data that is stored in the directory service with the specifics of the search functionality in mind full advantage can be taken of the directory service. In the prototype application many of the directory service objects were designed with attributes that could be taken advantage of by the directory search. In designing the system this way the performance of retrieving information from the directory service could be controlled effectively. Another alternative use of the attributes of directory service objects is to use the attributes to contain the distinguished name (DN) of other directory objects that contain additional information. Performing a simple read rather than a directory search can then retrieve these directory objects.

The directory service can support a smart space management application efficiently if during the design of that application the characteristics of the directory service are treated appropriately. By creating a mapping of the data model on to the directory schema in a manner that leverages the searching abilities of the directory and the inherent hierarchal data storage pattern of the directory service excellent performance can be achieved.

6.3 Directory Service Features

Modern directory services exhibit five characteristics, which provide a suitable platform for the development of management applications. These features are as follows:

- Hierarchal Design
- Object Oriented Modelling
- Flexible Query Support
- Replication
- Security Features

Directory services organise data in a tree like fashion in the tree there are two distinct entities Organizational Units (OU) and Objects. OUs can contain other OUs or objects. Organizing the directory in this hierarchal fashion allows the separation of data into manageable units, distinct sections. The division is often modelled on an organisations structure or geographical regions.

The directory service has a directory schema that contains rules on which object classes can be stored in the directory service. The schema also contains rules on the attributes that each object class in the directory can have. This object oriented modelling approach is very flexible and allows many different types of data to be stored in a logical manner. The schema itself can be easily extended to allow new object classes to be added to the directory.

The directory service provides query mechanisms to retrieve the data that is stored in the directory service. In particular directories provide methods to locate objects and attributes with in a scope of interest. Scopes usually consist of objects contained in a particular organizational unit or objects of a particular object class. The read performance of the directory service also assists in making the retrieval of information from the directory very efficient.

Replication in the directory service allows information to stored in more than one location. This information is then updated across all the locations once modifications have been made to the data. This ability is one of the defining characteristics of a directory service. However the price directory services pay for this ability is the loss of support for transactional updates. However there are techniques such as unique object identifiers, which can counteract many of the disadvantages of not having transactional operations.

The directory service also provides security features, which controls access to the directory objects. A user is required to provide credentials before accessing the directory service, these credentials determine which directory objects the user will be allowed to access and which operations the user will be allowed to perform on those directory objects. This fine-grained security model is necessary for directory services that will be accessed by hundreds if not thousands of users.

6.4 DEN Features

The Directory Enabled Networks initiative provides a number of lessons that can be reused in the creation of a smart space management framework. The initiative promotes the use of directory-based applications as the basis of many network management systems, likewise from the work carried out in this dissertation it is obvious that directory services are capable of supporting multi-application smart space management applications. A number of the characteristics as discussed in the previous section are the reason for this.

The initiative also promotes the use of an information model that can be further extended to support many different classes of information to represent devices, services and users. The prototype application developed made use of this information model and extended it to represent some of the devices and services that could be present in a smart community space environment. A fuller specification and extension of this information model for the management of multi-application smart spaces by an independent working group would be one option for creating a standard for directory based smart space management applications.

6.5 CIM

The Common Information Model now provides the foundation for the DEN mappings onto the LDAP schemas. Therefore the DMTF and CIM working group could play an important role in the further development of an information model for multi-application smart space applications. This work would obviously derive from the core model and require extension to some of the existing common models however a new common model may need to be developed to model information that is unique to the management of smart space environments.

6.6 Policy Based Network Management

The techniques from policy based network management can be reapplied to the management of smart spaces as has been demonstrated successfully in the prototype application. A policy language needs to be created, which has the representative power for a diverse smart space environment. The policy based management techniques need also be allied with a form of automated learning that will assist in modifying the management systems behaviour as well as assist in the discovery of user tasks and behaviours.

6.7 Summary

This chapter has reviewed the lessons learned from implementing the prototype application. The initial objectives of the dissertation have been discussed in the light of the experience gained from carrying out this research work. The benefits of reusing DEN, CIM and policy based management to aid in the management of smart space environments has been examined.

Chapter Seven

7. Conclusion

7.1 What has been achieved during this dissertation

The key challenges that exist when creating management applications for multi-application smart space environments have been examined in this dissertation. These challenges include the creation of an information model with the representative capability for the differing classes of management information. This dissertation has extended the information models, which have been provided by the DEN initiative, to represent the information that is needed to manage a smart community space. The DEN LDAP mappings have then been used to create a directory-based management application to control a simulation of the smart community space. The prototype application has then been used to test the suitability of using directory applications to manage smart space environments. The suitability of reusing the principles of the DEN initiative and policy based networking technologies to manage a smart space has also been tested. The results of the implementation of the prototype have been discussed during chapter six of the dissertation.

Completing this research work has allowed an examination of the issues in smart space management to take place and helps to identify the future challenges in developing successful management applications. The intention was not to create a completed smart space management application but rather to identify the issues involved in doing so. These issues were discussed fully during chapter three of the dissertation.

During the course of the research a number of areas that deserve further research work have been identified, these are topics are outlined in the last section of this chapter.

7.2 What interesting things have been observed

In the prototype application the policy core information model from the Common Information Model has been successfully reused to store and model policy data in a directory service. A smart space management application has then been implemented which uses this policy information to manage a simulated smart space environment. This implementation has successfully shown that the principles of policy based network management can be reused in a smart space application. It has also demonstrated the suitability of the policy core information model to represent policy information in a directory service.

The information that the management system for a smart space environment must have access to has been identified and analysed. Determining what the requirements for such a management system are has helped identify this information. The appropriate data stores for this type of information have been discussed. Where possible existing information models were used to model the data before that model was mapped onto a data store.

The User information model of the Common Information Model has also been used to model member information for the smart community space. This information model is suitable for storing the information necessary to implement a management system, which has a requirement to store user information. However the model needs to be extended to store information in relation to user profiles for services that are available in the smart space.

The DEN initiative provides mappings from the CIM information models on to an LDAP directory schema. The prototype developed has used this mapping to update the schema for the directory service that provided the basis of the implementation of the management application. The LDAP mapping is sufficient to allow the information models to be used successfully in a directory based application.

Services in the smart space environment have been modelled by extending the core information model. These extensions to the information model have then been mapped onto the directory service. This allows objects to be created in the directory to represent services. These directory objects are then populated with the management information that is required to ensure they are managed correctly.

Policies have been created in the directory service to represent user tasks in the smart space environment. The management application for the smart space can then use these policies to control the smart space and aid the user in the successful completion of their tasks. The policies have also been represented in the directory service using the policy core information model from CIM. This has also demonstrated the need for a fully-fledged policy language for managing smart space environments.

7.3 Objectives Revisited

In chapter one of the dissertation four questions were outlined to provide a focus for the research. In the next four subsections these questions will be evaluated.

7.3.1 Is the representational power of the directory service adequate to store the required management information?

An information model for representing management information is required so that smart space systems can be modelled. In the prototype application the DEN Initiative and CIM Information models were used to model the smart space. The directory service was then used to store the representations of the users, the user tasks, the activities and services that could take place in the smart space and the policies to control the smart space. The directory service was capable of this representing this management information successfully. The directory service is therefore capable of storing the management information for the smart space environment. As a result of carrying out this research the requirement for an information model, which can be used when modelling a smart space environment, became apparent.

7.3.2 Is the directory service dynamic enough to support the update of the management information?

The management system will require access to certain parameters about the smart space environment that are constantly changing such as temperature or a users location. This characteristic of the information means it is not suitable for storage in the directory service. Instead the directory service can provide a reference to the management system that allows the management system to locate the sensors that can provide the information. In this instance the directory service acts as an index to this type of information. The directory service is capable of storing other more static information such as tasks, policies or user profiles.

7.3.3 Is the use of policies beneficial in the management of the smart space?

The design and implementation of the prototype revealed that policies could play a very important role in the management of a smart space. Firstly policies can be used to control the devices in the smart space when policy conditions occur. Secondly policies can be used to control the smart space management system and change the behaviour of the management system. Thirdly user tasks can be mapped onto policies that the management system can execute to aid the user in completing his or her tasks. These three situations have been successfully modelled in the prototype of the management system. Following further research to resolve some of the outstanding issues with policy based systems it would seem that such systems would make a large contribution to the management of smart space environments.

7.3.4 Does the directory service support efficient retrieval of the management information by the smart space management application?

The directory service provides a powerful search capability that allows the management application to retrieve data efficiently from the directory provided the data is structured to take advantage of the directory service features. These features include the use of organizational units to separate directory objects and providing attributes in the object classes to aid searching the directory. These design features should be built into the information model when providing a mapping of that information model onto the directory service.

7.4 How can this dissertation be built upon?

This dissertation has demonstrated that a vast amount of research still needs to be carried out before successful multi-application smart space management applications can be developed. The areas that this research should be directed into include the following:

- Policy Language for smart spaces
- Policy Representation
- Representation and Modelling of User Tasks
- Representation and Modelling of Services in a Smart Space
- Full Extensible Standards based Data Model for Smart Spaces

7.5 Summation

Smart space environments are highly dynamic and provide a diverse range of services to the users of the environment. This fact does not prevent the successful reuse of existing management technologies to aid the management of those environments. However the representational power of those technologies needs to be extended to represent smart space services in particular. A directory service can play a very important role in the creation of a management system for these new environments. The directory service will contain a section of the management information and will act as an index to the remainder of the information. Finally policy based management will play a vital role in the management tasks once a policy language with the expressive power to represent a smart space environment is created.

8. References

[1] Friedmann Mattern, Ubiquitous Computing Infrastructures, ERCIM News Oct 2001

[2] Directory Enabled Networks, John Strassner. ISBN 1-57870-140-6

[3] Common Information Model (CIM) Tutorial, DMTF www.dmtf.org

[4] Managing Enhanced Network Services: A Pragmatic View of Policy-Based Management, Intel Corporation.

[5] IETF Policy Core Schema

[6] Building LDAP-Enabled Applications with Microsoft's Active Directory and Novell's NDS. ISBN: 0-13-062145-5

[7] JINI Architectural Overview, Sun MicroSystems

[8] Introducing Wireless JMS, Dr. Silvano Maffeis

[9] A Smart Space Management Framework, D O'Sullivan, V. Wade TCD April 2002

[10] Ambient Intelligence, Jari Ahola ERCIM News October 2001

[11] ;login: Issue December 2000, Martin Herman NIST

[12] Meeting the Computational Needs of Intelligent Environments: The Metaglue System, Michael H. Coen, Brenton Phillips, Nimrod Warshawsky, Luke Weisman, Stephen Peters, and Peter Finin.

[13] Mobile Resource Awareness, Erik Wiklander

[14] Managing Enhanced Network Services: A Pragmatic View of Policy-Based Management, Intel Corporation.

[15] Aura: an Architectural Framework for User Mobility in Ubiquitous Computing Environments, João Pedro Sousa and David Garlan

[16]Understanding and Using Context, ANIND K. DEY

[17] Network Management Systems Essentials, Udupa, Divakara K.

[18] Network Management Principles and Practice, Subramanian, Mani

[19] E.F.Codd 'A Relational Model for Large Shared Data Banks' CACM 13(6) June 1970

[20] Michael Hammer, Dennis McLeod: Database Description with SDM: A Semantic Database Model.

[21] QoS Policy Specification – A mapping from Ponder to the IETF, Oscar Díaz Alcántara, Morris Sloman Department of Computing, Imperial College,

[22] 17. Moffet, J. and M. Sloman, Policy Hierarchies for Distributed Systems Management. IEEE Journal on Selected Areas in Communications, 1993.

[23] Sloman, M., Policy Driven Management for Distributed Systems. Journal of Network and Systems Management, 1994.

9. World Wide Web Resources

BLUE www.bluetooth.com

CHAI www.hp.com/go/embedded

CSCO www.cisco.com

DMTF www.dmtf.org

FCE http://www.cc.gatech.edu/fce

IBM www.ibm.com

IETF www.ietf.org

INHAUS www.inhaus-duisburg.de/

ISO www.iso.org

J2EE java.sun.com/j2ee

JBOSS www.jboss.org

JINI www.jini.org

JIN2 http://jan.netcomp.edu.au

JMS java.sun.com/products/jms/tutorial

JNDI java.sun.com/products/jndi/tutorial

MICH www.umich.edu/~dirsvcs/ldap/

MSFT www.microsoft.com

MUSE mmsl.cs.ucla.edu/muse/

NIST w.nist.gov/smartspace/

NSCP www.netscape.com

NOVL www.novell.com

OGRP www.opengroup.org

PNDR www-dse.doc.ic.ac.uk/Research/policies/projects.shtml

SALU www.salutation.org

SMNS www.siemens.com

SLR www.ele.tut.fi/research/personalelectronics/

SOFW www.softwired-inc.com

SUNW www.sun.com

TMF www.tmforum.org

TMN http://www.iec.org/online/tutorials/tmn/

UPNP www.upnp.org

10. Abbreviations

ADSI	Active Directory Services Interface
API	Application Programming Interface
CIM	Common Information Model
CIMOM	Common Information Model Object Manager
COPS	Common Open Policy Service
CMIP	Common Management Information Protocol
CMIS	Common Management Information Service
CMISE	Common Management Information Service Element
CSCW	Computer Supported Collaborative Working
DAP	Directory Access Protocol
DEN	Directory Enabled Networks
DEN-ng	Directory Enabled Networks Next Generation
DMTF	Distributed Management Task Force
DN	Distinguished Name
EJB	Enterprise JavaBean
HCI	Human Computer Interface
IETF	Internet Engineering Task Force
ISO	International Standards Organisation
ITU	International Telecommunication Union
J2EE	Java 2 Enterprise Edition
J2ME	Java 2 Mobile Edition
JMS	Java Message Service
JNDI	Java Naming and Directory Interface
LDAP	Lightweight Directory Access Protocol
MIB	Management Information Base
MOF	Managed Object Format
NGOSS	Next Generation Operations Software and Support
OSI	Open Systems Interconnection
PBNM	Policy Based Network Management
PDA	Personnel Digital Assistant
RMI	Remote Method Invocation
SDK	Software Development Kit
SID	Shared Information Data
SMI	Structure of Management Information
SNMP	Simple Network Management Protocol
SPI	Service Provider Interface
SQL	Structured Query Language
TMF	Telemanagement Forum
TMN	Telecommunications Management Network
UPnP	Universal Plug and Play
URI	Universal Resource Identifier
URL	Universal Resource Locator
WBEM	Web Based Enterprise Management
XML	Extensible Mark-Up Lanaguage