Making Existing Homes Smart

Lauri Kainulainen

University of Tampere
Department of Computer Sciences
Computer Science
M.Sc. thesis
Supervisor: Roope Raisamo
October 2007

University of Tampere
Department of Computer Sciences
Computer Science
Lauri Kainulainen: Making Existing Homes Smart
M.Sc. thesis, 93 pages, 7 index and appendix pages
October 2007

Abstract

Smart homes have been a central theme in ubiquitous computing and intelligent environments. Various research projects and adventurous companies have tried to tackle the challenge, but still smart homes are nowhere to be seen on the consumer market. So far the research has focused on new housing and laboratory prototypes instead of our current homes, and without much consideration for rented households. In my thesis I will focus on the new challenges brought by these two factors, take a look into the current state of smart home research, summarize the wishes made by potential users and write about the various themes in smart home design. I will conclude by proposing a simple design that has the potential of lifting our current homes and the homes of the future to the realm of intelligent environments.

Key words and terms: smart homes, intelligent environments, ubiquitous computing, software architectures

Table of Contents

1 Introduction

Smart homes are one those topics that are often mentioned in scientific publications and magazine articles, but rarely defined properly. The domain has been researched furiously for decades and various large-scale prototype implementations have been made. Some actual housing projects have also proclaimed to be producing smart homes. Regarding computer science, smart homes provide an interesting field as they contain many popular domains of modern research. These domains include such topics as "*tangible interfaces", "ubiquitous computing", "ambient intelligence"* and *"speech interfaces"*.

Despite all the efforts, wide spread success of intelligent homes is yet to come. One of the major problems is that home automation solutions are mostly based on a technological "push" rather than a consumer "pull" [Mäyrä et al., 2005]. Although the technology is almost at our grasp, successful deployment of home automation requires research on other areas, such as sociology, industrial design and usability, in addition to the directly computer science related fields like proactive computing and multimodal interfaces.

The problem with the technological push can also be seen in many parts of modern research. Multiple papers provide innovative ways to bring proactive computers into our homes, but very few try to find out if they are actually welcome. Although user requirements for smart homes have been researched (for example see research by Röcker [2004] or by Mäyrä et al. [2005]), the tests seem to be separated from actual implementations or designs. In addition to this home environments are especially hard to analyze and comprehend. Studies have shown that homes are not thought of as simply places of dwelling, but instead more as a 'state-of-mind' [Mäyrä et al., 2005]. Good examples of research and development exist as well. For instance the LinuxMCE project delivers a convincing smart home solution that is free of charge as long as you possess the hardware. In addition all smart home projects contribute to the overall understanding of intelligent environments in one way or the other.

Many of the modern large-scale projects deliver a lot of wonderful features that they see useful in the everyday life of smart home occupants. Unfortunately many research groups start from a personal agenda and not from actual user requirements. For instance the MavHome project from the Arlington University in Texas created a smart home environment that learns from the user's actions and tries to predict her next move [Cook et al., 2003]. This kind a of *strong proactivity* is however one of the features that users fear the most [Röcker et al., 2004]. Good studies on smart home user requirements exist and they should be put to actual use [Röcker et al., 2004; Koskela and Väänänen-Vainio-Mattila, 2004].

Another crucial problem with smart homes is the difficulty of bringing the technology to existing households. Many of the smart home projects are implemented

as *"living laboratories"* and built to accommodate the smart home system [Crabtree and Rodden, 2004]. In a real situation the smart home system should be the one that is adapted to the home environment. Not the other way around.

In my thesis I will start by introducing a few central concepts and research on smart homes. After that I'll introduce relevant smart home projects and compare them. The next chapters are dedicated to smart home networking, software architectures, artificial intelligence and user interfaces. During these chapters I'll sum up the different solutions in each domain and finally present a system concept that would have the potential to meet the user requirements and start the process of turning our existing homes into intelligent environments.

2 History and Motivation

It is easy say that smart homes are not a new discovery. Like many other technological concepts, variations of smart homes and intelligent environments have already been present in fantasy literature and science fiction works such as George Orwell's famed novel 1984. Human imagination has long conceived living environments. Only the explanations of what make them work have changed during the years.

The classical intelligent environment is the wizard's tower in fantasy literature. A popular example is Saruman's Isengard in The Lord of the Rings where the autonomous nature of the environment is explained through the use of magic. During the 20th century the use of magic was accompanied by the science fiction environments that used new imaginative technology. Perhaps one of the most known examples is the *"Heuristically programmed Algorithmic computer"* also known as HAL 9000 that originally came from the Arthur C. Clarke's Space Odyssey saga. Although HAL and many other artificially intelligent environments often play the role of the bad guys, some environments, such as Rickety Rick from the children's cartoon Doctor Snuggles, also play a supporting role on the hero's side.

Moving on from fiction, some real life housing projects have already claimed to produce real smart homes. One example comes from South Korea where the leading electronics manufacturer LG built over 100 homes that, according to the manufacturer, offer smart technology. For example the houses offer a wall panel which can be used to control appliances and lights in the house as well as keep track of electricity consumption and other household data [Simmons, 2006]. The question is how we can determine if these homes really are intelligent and whether they contain features that users find useful.

2.1 Defining the Smart Home

Officially the word "smart" in conjunction with technology was first used during the seventies. It originated, like many other scientific terms, from the military and was used to describe bombs and other weapons that had some capabilities to guide themselves. During the technological boom of the 1980's the adjective caught another meaning: it became to represent objects that contained microchips, such as computers and advanced house appliances. This meaning has subsided during the years and hardly nobody today would define a modern computer smart even though they are magnitudes faster than the first microchips of the eighties.

The actual term "smart home" was coined by the American Association of House Builders in the year 1984. Today the definitions vary and are generally more technologically oriented. One rather simple way of defining the concept smart home comes from the DTI Smart Homes Project:

> "A dwelling incorporating a communications network that connects the key electrical appliances and services, and allows them to be remotely controlled, monitored or accessed."

Where remotely controlled means that the appliances and services may be controlled within or outside the dwelling [King, 2003]. This definition works with most of the smart home scenarios and projects since they usually contain interacting, networked appliances. Some projects such as the UMASS Intelligent Home Project even focus solely on researching an optimal way for appliances to cooperate [Lesser et al., 1999].

Some people would disagree that the network is the thing that makes the home smart. Many would resort in defining the smartness of a home in terms of artificial intelligence, like the proactive nature of the system or the way it can learn from the actions of the inhabitants. Trying to come up with a definition that uses artificial intelligence as a meter is a dead-end. The first problems already arise when trying to find out how to determine if a system really is smart.

The focus on the networked nature of smart homes puts the focus on the *competence* of the system. The networked house has the potential of becoming smart and helping the inhabitant during her daily life. However the network itself is not smart. Rather the network is as smart as the cooperation it enables. Using this as the starting point we find a new definition for the smartness of an intelligent environment. Instead of traditional artificial intelligence, the smart nature of the system comes from the interactions between the inhabitant and the system. Instead of being a characteristic of the system or the user, smartness refers to the whole environment of humans and technology, and the possibilities this environment enables in finding new ways of getting things done [Norros et al., 2007].

2.2 Smart Homes and Ubiquitous Computing

In the beginning of the 90's Weiser introduced the concept of ubiquitous computing to explain the future scenario of people–computer-interaction where computers and intelligent devices of all sizes would occupy various parts of our environment, and our interaction with them would be a normal and continuous part of our everyday lives [Abowd and Mynatt, 2000]. The research on ubiquitous computing has since grown to be one of the most active areas of computer science encompassing multiple domains such as intelligent environments and pervasive computing.

Like many other scientists, the researchers at Xerox PARC started working on ubiquitous computing (ubicomp). Two of them, Abowd and Mynatt [2000], wrote their paper almost a decade after Weiser and concluded three remaining challenges for ubicomp: the need for *natural interfaces*, *context-awareness* and *the automation of capture and access of live experiences*. Natural interfaces would allow us to break free from the traditional desktop metaphor of computer interaction and let us use more

expressive input techniques like speech or hand writing. A context aware system would present only relevant information for us and help against the information burden, and by automatically capturing our everyday live experiences it would be much easier to return to moments of importance, like an important decision in a meeting or the shopping list discussion from the morning.

Abowd and Mynatt also introduce the concept of *everyday computing* to explain the emerging area of interaction research where computing is no longer done in front of desks using a keyboard and a mouse, but instead the interaction becomes a constant action that can be done with mobile devices from anywhere. Everyday computing rarely has any explicit ending or starting point and tasks are usually interrupted at some point and continued later on. In addition to this multiple activities are done concurrently and it should be made easy for the users to monitor the ones on the background.

One of the key issues Abowd and Mynatt emphasize is the use of time as a discriminator. Although humans use time often to categorize events, computer applications often overlook it. Another thing seen as important by Abowd and Mynatt is the use of associative models of information. Meaning that humans often have multiple different views – in addition to the chronologic view – into remembering information. Sorting documents in a folder hierarchy might work as long as the hierarchy does not need to be changed later on, but when it does, deciding on a new hierarchy might be difficult. Instead of explicit folders, users could for example have multiple views that use other discriminators like time, the titles of documents and their contents as sorting criteria.

Abowd and Mynatt also layout two major challenges for future ubicomp research. The first one is the challenge of evaluation. Ubicomp systems need to serve a real need instead of being just proof-of-concept systems. Scenario driven development should be carried out to ensure that the system has a specific purpose. Secondly the evaluation should be carried out in a real environment and not in laboratory surroundings. The remaining challenge with evaluation is that normal usability testing is mostly task-centric and this does not fit well into the area of ubiquitous computing.

The second challenge comes from social issues. One problem is the intrusion of privacy that people face when surrounded by embedded devices. Multiple cameras and other sensors will make people uneasy and therefore it should be clearly visible when you are being watched in an ubiquitous computing environment. In addition to privacy and visibility, ubicomp design has to pay close attention to control and security so that users stay in control of their systems and that their information stays intact.

Smart home research is one of the major sub-domains of ubiquitous computing. The concept of everyday computing with its needs for context awareness and natural interfaces apply straight into smart home design. Many of the issues such as user fears of losing control or finding the actual human need are key issues within smart home research.

The ideas outlined by Abowd and Mynatt have been used by various smart home projects. For example multiple projects have gathered user expectations and requirements through scenario-driven interviews [Battarbee and Kuusela, 2005; Röcker et al., 2004; Sainz de Salces et al., 2005] and many other projects have built prototype buildings to evaluate the functionality of technologies in an authentic environment [Cook et al., 2003; Spinellis, 2003;Koskela and Väänänen-Vainio-Mattila, 2004]. All these studies do contribute much to smart home research and ubicomp research in general.

2.3 Smart Homes in the Light of the Actor-Network Theory

The Actor-Network theory provides one simple viewpoint into understanding the relation between the smart home occupant and the software system. The debated theory was originally mainly formed by French scholars Michel Callon and Bruno Latour accompanied by the British sociologist John Law [Wikipedia], and it gives an interesting insight into the evaluation of information systems. Actor-network theory, or shortly ANT, is a social theory that tries to explain the interactions between material and semiotic agents. The theory follows the idea of *generalized symmetry* which means that all elements in the network are seen as equal. This is because, according to the theory, the differences of the elements are created through the interactions between them and therefore should not be presupposed. Unlike in many other theories which define agents as autonomous entities, in ANT the static elements are also counted as agents and treated equal with their autonomous counterparts.

In their research paper Cypher and Richardson [2006] use the example of a person wielding a gun. Both agents – the gun and the person – affect each other. How third parties view the person has been changed now that he possesses the gun and can use it to threaten or kill them. Also the gun has changed because it is now in a relationship with the person. A gun on a table is not seen as dangerous as a gun wielded by someone. When these two agents join, the possibilities that this combination enables is greater that the possibilities when they are separated. The non-human agent influences the human agent and vice versa.

Cypher and Richardson discuss the use of Actor-network theory to better understand and design games and virtual environments. They see that the use of the theory might help in getting away from the traditional user-centered view and bring the focus more on the interaction between the different participants. The same insight can be brought to the world of smart home design. It's important to realize that when the smart home designer takes a stand on how a certain routine is handled in the house, she affects both sides: the system and the user. The design decisions might collide with the routines the user already has and have a negative impact on user satisfaction.

Another way to apply the principle is to look at the deployment of intelligent environments. When the system is introduced into a new surrounding, it needs to be

installed into a places where it can function, like near electrical outputs or close to ethernet wiring. In other words the system needs to adapt to the new environment. At the same time however the people in the environment will need adapt to the new system whether they use it or not. This is even more crucial to note in very dynamic environments such as homes.

The notions of actor-network theory are quite close to the basic understandings in ubiquitous computing. The computational parts of the system are no longer seen as devices that support one or more tasks, but instead as *agents* that together with their human counterparts create new possibilities for interaction. This puts new pressure on design of these systems as old desktop computing oriented techniques are no longer enough.

2.4 The Structure of Buildings

One crucial part in finding out how smart homes could be brought to our everyday lives is understanding the characteristics of current houses. Rodden and Benford extend the notion originally put forth by Steward Brand to generate a better understanding on how buildings change. In the core of the model lies the concept of the “six S's”: *site*, *structure*, *skin*, *services*, *space plan* and *stuff*. These six terms reflect the layers that are present in a building. Site is the outermost layer and stuff is the one in the middle [Rodden and Benford, 2003].

Site refers to the geographical location of the building. This layer is mostly regulated by governmental authority in terms of building contracts and land deeds. Changes to this layer are usually very slow and usually site remains unchanged across generations of buildings. In addition the changes done to this layer are usually executed by professionals, such as civil engineers and builders, and doing the changes usually lasts from months to years.

Structure is the foundations and central elements of the building and is also very hard to change. Usually once the structure of the building needs changing it is demolished completely. Changes take from weeks to months and generally require people with professional expertise or inhabitants with advanced do-it-yourself (DIY) skills. Ages of structures range from thirty years to three hundred years.

Skin encompasses the external look of the building and usually changes to keep up with the latest trends or technology. An estimation for the usual age of the skin layer ranges from twenty to thirty years. Work is usually carried out by professionals, but DIY enthusiasts might be able to complete the work themselves.

Services are the internal organs of the building and contain parts like plumbing, electrical wiring and telephone lines. Like the skin layer, services usually last from twenty to thirty years. Work can also be carried out by DIY enthusiasts, but on most cases it's done by professionals, such as plumbers, who are coordinated by the occupants of the house. Work on services is shorter than with skin and usually only

takes days.

Space plan stands for the interior layout of the building. It dictates where things like walls, ceilings and doors go. Space plan changes depend a lot on the owners of the house. In some households the space plan might change every three years, while in more settled environments it could last for thirty years. While changes are usually made by the occupants of the house, the help or consultation of outside experts is often used. Doing the actual changes might take from hours to days.

Stuff is the last layer and the one that is always in total control of the occupants. It compromises of the objects inside the home, such as lamps, chairs, tables, beds and plants. The stuff layer changes daily and the changes can be done in minutes or hours. This means the that stuff layer is the most flexible of all the layers.

The layers are interrelated and changes on one layer impact others as well. This is more evident when changes are done to an outer layer. For instance changing the structure level is surely visible in the space plan since the foundations of the whole building could have changed. One important notion is subtle chain reactions that occur between the layers. Rodden and Benford [2003] present the example of the television set, which is often placed near the antennae and electrical outputs to avoid the need of pulling wires across the room. The placing of the television in turn affects the placing of sofas and chairs in the room as they are often placed so that the television acts as the focal point. This example illustrates the way that the decisions done on the services layer affect the layout of stuff.

Rodden and Benford also remind that some of the layers are dependent on the cultural environment. For example changes to the skin layer are hard to do in some parts of urban Great Britain where the outlook of buildings is much more regulated than in the United States. However, I'd be ready to argue that the changes to the stuff layer are pretty much in control of the inhabitants all over the world. Thus the stuff layer can be seen as the only totally “safe” layer, where changes are initiated by the inhabitants alone.

2.5 Rented Apartments

One of the overlooked areas in smart home research is the design of intelligent environments for rented housing. Although many of the smart home research principles also apply with rented housing, some extra challenges are introduced. In this Subchapter I'll outline why rented apartments differ and what challenges they introduce to design.

According to Tilastokeskus 72% of households, where the oldest person was under 30 years, were living on rent in Finland during the end of 2005. When looking at the situation in all age groups the total percentage of people living on rent is still almost 32 [Tilastokeskus, 2005]. When taking into consideration that possibly the most promising clientele of future smart homes are young people, it is rather surprising that so little

effort has gone into the research of bringing smartness into rented apartments.

One obvious difficulty with rented apartments is that they are not owned by the people that live in them. Rodden and Benford [2003] take this into account when they compare the different entities that initiate the work to do changes on an office, a rented house and an owned house. On their six layers the office is the one where the actual inhabitant, or in this case the worker, has the least control over the environment. In an office the inhabitant may usually only change the stuff layer, while other layers are in control of the landlord, the employer and service providers. In a rented home the inhabitant is also quite powerless: the layers on which she can initiate changes are the stuff layer, the space plan and the services layer. Usually changes to the space plan and services are also controlled by the landlord. In owned homes the inhabitant has much more control over all the layers, although changes to the site or structure usually need governmental approval.

When the only layer that can be safely kept in the control of inhabitants is the stuff layer, it becomes evident that the only way to create a smart home system that the inhabitants of rented apartments can safely incorporate into their homes, is to keep the system on the outermost layer. Surely changes to the other layers are also possible, but they require the permission of the landlord who might be less interested in bringing intelligence to his buildings. The key issue is keeping the minimum parts required for the operation of the smart home system on the stuff layer. Optional components can and should be offered to the other layers as well.

3 Designing Smart Homes

In this Chapter I'll go through the special characteristics that homes possess as environments and outline some of the most important requirements found. I'll continue by defining important roles that smart homes might fulfill in a home setting and introduce a new way of looking into the design process of smart home environments. I'll conclude with some thoughts on the licensing and releasing of a smart home system.

3.1 AMIGO Research on Smart Home Requirements

In a cross cultural user study on smart homes Röcker et al.[2004] tried to find what people actually would like to see from the concept of smart homes. They conducted their study in multiple European countries. All the studies were constructed from three different stages: a quantitative evaluation of different scenarios, structured discussion addressing different topics and an open-ended discussion on people's expectations.

Using four different scenarios featuring a futuristic home and a fictional family with two parents and two children, the researchers were able to find out what people expected from the homes of the future. Based on these results the researchers found six categories, each containing from one to four requirements, that should be met in an ideal smart home. The categories are prioritized – the most important one being the first one. I'll continue by shortly describing the categories and the key requirements. These requirements can then be used to evaluate the usefulness of the smart home solutions that will be presented later.

3.1.1 AMIGO Results

The first category consisted out of non-functional issues. The foremost requirement was the need for control, which meant that the smart home system should always obey the occupant. Also needs for privacy, security, home comfort and safety were presented. One of the requirements stated that the system should offer real added value over the existing infrastructure, but it should never replace the direct communication between people. Looking at this category it is perceivable that the users want the new technology to help them, but at the same time they want to keep their home pretty much as it was before.

Category number two contained only one requirement that expressed the need for help on the information burden: the system should offer correct, context dependent information to the right users at the right time. This directly presents the heavy need for context awareness on smart homes. Possible scenarios for this category could be that the system offers the user news headlines from her favorite sources when she sits down to eat breakfast or the automatic notification of marked appointments and upcoming TV-series. This category is one of the most easiest to convert to actual features. It's also

the one direct way of providing the *"added value"* requested in the first category. Worth noticing is the high priority given to this requirement in respect to the others.

The third category contained a more pragmatic set of requirements, which stated that the system should reduce the time needed for common household chores, do as much of the cleaning as possible, integrate and combine the functionality of appliances and be energy and cost saving. For a smart home system this presents a set of functional requirements that should be met.

The fourth category expresses needs that surfaced during scenarios that described follow-me content and the playing of games. These requirements contained the need for supporting the planning and organizing of activities for multiple persons at home and between home and work. The need for security was again brought up with a requirement relating to protection against data loss and system abuse or intrusion by malicious hackers. Occupants also wished that their user preferences would be saved and that the access to the system should be controllable and based on authorities. As actual feature these could mean the need for a way to login to the system and gain privileges to change more sensitive settings relating to the system.

The fifth category contained the need for assistance in home environment organization, like the closing of curtains or switching on the lights. During these activities the requirement that the system should always take the environment and the local conditions into account surfaced. This requirement repeats the need for context awareness that was already implied in the second category.

The last category of requirements are related to the need for people to stay in contact with others. For this the users saw that it's important for the system to take implicit social rules of behavior into account and that it should protect the privacy of the inhabitants at all times. As an actual feature this could be realized by asking confirmations before accepting incoming video calls and before providing information automatically about the user's location or timetable.

The requirement for communication can be seen as a minor conflict with the requirement expressed in category one about not replacing the direct communication between other people. Thinking it further it becomes evident that the key point is keeping people responsible for doing all the communication. The smart home system should provide assistance and make the communication as easy as possible, but it should never replace humans. This effectively means that, for instance agents that automatically try to schedule common meetings for their masters by checking their calendars, or voice operating applications that automatically contact other people, are not sorely needed.

3.2 Other Requirements Related Research

Another similar research conducted by the Samsung Corporation in cooperation with the American Institutes for Research tried to find requirements by interviewing and

monitoring candidates in the United States and South Korea [Chung et al., 2003]. The requirements found correlate quite nicely with Röcker's research. The need for harmonious cooperation between appliances and the need for context awareness and ease of organization were also spotted. A few more concrete requirements were also found, such as the need to reduce the wiring inside a home and the need for centralized entertainment resources.

One important requirement found by the Samsung & AIR study was the ability to customize one's home. The same need surfaced during smart home research done by the Tampere University Hypermedia laboratory [Mäyrä et al., 2005]. Homes are very personal spaces and therefore the smart home system needs to adapt to the environment the way the user wants it to, not the other way around. Indeed the first challenge in building a smart home system is to ensure that users would want the thing in their houses in the first place.

Not all of the research is in complete agreement. For example a study on 20 Jewish families and their usage of smart home systems revealed that they experienced the loss of control to the smart home system as a positive feeling. The feeling had some very strong connections to religious beliefs and the loss of autonomy that comes from believing in a higher being [Woodruff et al., 2007]. While this provides a very interesting counter-example, I'd argue that in a normal situation the strong need for user control still applies. Giving up the control to the system should be decided by the user, not by the designer.

Most of the requirements gathered in all these researches are not new. In fact the paper by Abowd and Mynatt [2000], which was introduced in the previous Chapter, already reflects many of these. They concluded that the social issues with ubiquitous computing boil down to four major themes: security, visibility, control and privacy. These are all evident in the list of requirements from smart homes. Also Abowd's and Mynatt's evaluation recommendation of using stories and scenarios was used in all the projects introduced here.

3.3 The Roles of a Smart Home System

Home environments are notoriously difficult to analyze and predict. Designers can never see all the different usage scenarios that arise from the interaction between the inhabitants and the smart home system.

In the following three chapters I'll introduce three characteristics of smart home systems that greatly affect the design. First is the big role media management plays in many intelligent environment scenarios and how that affects design. Then comes the need for the system to support communications inside and outside the house and thirdly one of the driving forces behind ubiquitous environments: the assistance it promises for elderly and disabled people.

3.3.1 The Media-centric and Private Nature of Smart Home Systems

One of the most important but often overlooked properties of a smart home system is that it presumably also acts as a media center that contains personal data such as movies, music and digital photos. When a family moves to another house they will want their photos and other digital content with them. Another aspect related to the media content is that often families have lots of shared content such as holiday photographs and movies bought for viewing pleasure of all members. Next I'll focus on these two issues imposed by the media-centric nature of a smart home system: the need for a way of sharing media and the possibility to take the media with them when they move.

The first apparent solution for the sharing question is to provide a common file storage that all inhabitants may use. However, as Grinter et al. [2005] found in their ethnographic study, the use of a simple network storage to make it possible to share media content between all family members won't always work as expected. For example in one family all the members stored their own digital photos on their personal machines. Merging them had not worked since they organized their photos in various different ways. The distributed way of storing the photos raised concern in the occupants that their collaborative experiences might be destroyed or rendered inaccessible when – for example – one of the computers was disconnected.

The organization of media needs to be personal, but the storage should shared among all the occupants. This is possible by providing multiple views to the underlying storage. So even though the family photos would reside on one physical network storage device, all the end points that are used to view them – such as computers, media terminals or mobile phones – may define their personal ways to organize the content. One approach would be to use a relational file system that allows SQL-like queries to the data. Then using the results of the queries to provide the mechanism to categorize the data. Unfortunately this would require building a scalable, efficient and trustworthy file system that would allow those queries. Modern operating systems already allow this kind of functionality through more sophisticated search engines that work on top of the legacy file systems. For example the search tool from Beagle-project (http://beagle-project.org) builds an internal database from the contents of your hard drive and allows searching by various different criteria. Supported media files are searched for metadata and documents are scanned so that the user may for example search for a concept he wrote or saw in one paper. The results of searches may be stored as *"smart folders"* that update their contents as new data matching the original search expression shows up.

The other aspect created by the media-centric nature is the moving of the data when moving from one household to the other. The transition can basically be done in two different ways: physically or digitally. The former meaning simply the transportation of the smart home system or at least the component that contains the data to the new address. Moving the digitally means that the data is copied through a network or over a transport medium, such as an external hard drive. Using the latter approach might make

some inhabitants suspicious: “will my data be safe?”, “can the new inhabitants access my deleted data?”.

In addition to basic media data, the smart home system could contain user preferences and personal tweaks. For instance the LinuxMCE and Pluto smart home systems support the usage of *moods* that basically are definable environment settings that affect things like lights and air conditioning. One mood that dims the lights and closes the curtains could for instance be defined for DVD-viewing. All these user definable settings should also be transportable. While exporting the settings and transporting them to another system could be quite straightforward, the importing of the settings into the new system could be a bit more difficult. This would require seamless interoperability between different smart home systems. A goal that is most likely unachievable within the current situation.

The storing of private data on the smart home system makes the system personal and therefore smart home systems can be better compared to personal computers than refrigerators or showers that can be found pre-installed in most homes. A personal system is probably something that the inhabitants would like to take with them if they move. This creates a new aspect to smart home design: mobility. This aspect has often been overlooked in smart home research; perhaps intentionally to avoid complicating the already difficult area of smart home requirements capture.

It's quite possible that the inhabitants can't take all the “smart” bits of the house with them when they move. Things like mounted motion detectors, freezers and kitchen sinks are probably there to stay. Depending on the wiring solutions in the house, such things as light switches might be mobile or not. The most important pieces of the smart home would be the storage and logic units that define how the home works and store the personal data of the inhabitants. These units need to be mobile and easily disconnected from the network.

When talking about flats where space might be very limited, it's important to take into consideration the size of the smart home system. The components that store personal data should be quite compact. Also it would be useful to design components that fit into the décor of most houses, so that the components need not be hidden in closets.

3.3.2 Domestic Communications and Organization

A great deal of our home routines evolve around communications. In their study on these routines Crabtree and Rodden [2004] found that: *“great many of the information resources in the home are implicated in the collaborative production of outgoing communications and consumption of incoming communications”*. These communications also happen inside the house as a method of internal organization.

A good example of a communications oriented routine is the handling mail in a family. Crabtree and Rodden found that mail could be collected by anybody in the

family. The collection happens from a central point (e.g. a mailbox). The collector goes through the mail, sorting out which ones belong to him. The rest of the mail is ordered spatially around the household depending on the relevancy of the mail and to who it's written to. For example letters with higher priority might be put on top of the kitchen table, while personal letters to the husband of the family might go on top of the table where he usually works.

The same need can be implicitly seen in a research conducted by Taylor and Swan [2005] where through an ethnographic research the creative way in which family members use various tools for coordination was discovered. By studying the every-day life of eight different households they discovered the way in which the person responsible for coordinating everyone's actions – in all cases the mother – would come up with very artful ways of communicating and coordinating her own daily events.

For instance one family used a big paper and pen based calendar on the fridge door to keep track of where everyone was. One mother used the location of different papers to reflect their importance and role. Another used a small paper book to keep track of the nappy changing of her youngest one. Once she grew out of the nappies the journal became a recipe book. In addition to the personal use of these objects, many of the objects were used in conjunction with others. For instance a simple paper board was not enough for coordinating more complex actions that involved other objects, such as letters, and therefore a more suitable system was chosen for coordination. In this case it was the spatial dimensions offered by the kitchen table.

These examples reflect the complex way in which every day objects are used in homes. Through these observations and interviews with the families, Taylor and Swan set forth three major challenges that need to be reached in smart homes. The first requirement is that heterogeneous devices should be artfully combined. This implies the same need as the second category in the AMIGO research that required help with the information burden, but also emphasizes the fact that designers won't know all the various usage contexts beforehand.

The second need was for *pliable systems of organization*, which means that coordination systems should be re-designable by replacing some of the devices involved. The function of the coordination system could be changed this way to reflect the new needs of the user and the devices themselves could act in different roles in different coordination systems.

The last challenge was for integration with existing organization systems. This is the most technically challenging part as it requires that the prevailing paper and pen based systems should be compatible with the new systems. The authors suggest embedded RFID-tags that could tie the traditional devices with their electronic counterparts.

Both studies suggest that special emphasis should be given to the design of communication and organization enabling components in the smart home system.

Taylor and Swan also like to point out the versatile ways in which coordination is done. The system can't force the users to organize their lives in a certain way. Instead it should be adaptable so that it may become part of the current organization routine in the household.

Organizing systems such as the Gate Reminder have been built before [Kim et al., 2004]. One of the key issues with these prototypes is that they are stand-alone applications that don't connect to other organizing systems. For example the Gate Reminder should connect to a shared family calendar instead of providing its own. When designing these systems, the designers should try to steer away from the feared information burden that – according to the AMIGO research – is one of the key requirements of inhabitants.

3.3.3 Assistance for the Elderly and Disabled

Some studies approximate that by 2010 half of the population of Europe will be over 65 years of age. This creates a huge market for home assistance technology [Green et al., 2004]. What this means that it becomes crucial for the smart home system to support people with disabilities or weaker senses. Many studies already advocate the usage of multimodal interfaces in intelligent environments [Sainz de Salces et al., 2005; Laarni et al., 2007]. This is definitely a very good stand since multimodal interfaces improve the usability and accessibility of the system. Multimodal systems work better for people with weaker sensory abilities, such as elderly or disabled persons, since the transmission of data happens on many sensory channels.

Another important feature would be the possibility to use the technology for distant monitoring. For example the system could detect when the users blood pressure is too high and warn him. In an emergency it could alert hospital staff who could see the situation through their own monitoring devices. Improved communication possibilities would promise more contact with family and loved ones and allow better usage of delivery services, such as food deliveries, for persons who have movement disabilities.

All these features would allow disabled and elderly people to achieve a higher level of autonomy in their lives. However Sainz de Salces et al. [2005] point out that special smart home systems should not be made for people with disabilities. Instead the ideal smart home system will allow all to grow old and still be able to use the system without problems.

3.4 Design Methodology

In traditional software design the *Human Centred Design Processes for Interactive Systems*, also known as ISO standard 13407, has been widely accepted. It was originally defined in 1999 to form common understanding on usability driven design. The whole process has been designed to be adaptable to different development environments from straight waterfall models to more complicated agile models [EMMUS, 1999].

The standard forms a process with five consecutive steps. The first step is to plan the human centered process. During this step all parties involved in the design process should be committed to the user-centered design philosophy and a plan that ensures sufficient time for user requirements elicitation and interface validation. The validation plan is the major document produced during this step. It outlines the time plans and iteration needs of the project as well as the criteria on which the application is deemed successful.

Secondly the context of use should be specified. For a project that is creating a totally new system, methods such as interviews and meetings with the users and other stakeholders should be used to gain better understanding. Usually when building a system to replace an existing one, data about the usage and context is widely available. In these cases items such as help desk reports and user feedback about the old system may provide valuable insight into the context.

In addition to learning about the environment and users of the system, it's also crucial to understand the different user categories and their tasks as well as have a good understanding of the big picture – the global task that the system should help achieve. The results of this step should be documented in a Context of Use Description that outlines the relevant characteristics of the users, tasks and environment, and finds out the ones that are important during system design.

During the third step the user and organizational requirements should be specified. This is step of requirements elicitation is generally accepted as the most important step in software development. User-centered design needs to include the search for additional requirements related to the quality of the user interface and workstations used for operation, the quality of tasks that are done by the identified users, task performance requirements, communication requirements and the required performance of the system in the light of operational and financial objectives.

During the third step the model in ISO standard 9241 part 11 can be used to define objectives for each category of users. The first of the three objectives is efficiency, which dictates the criteria whereby the minimum level of effective performance may be noted. Second objective is for effectiveness, which points out the criteria that can be used to say if a task is failed or succeeded. Lastly one criteria should be used to measure the satisfaction of the users [EMMUS, 1999].

The fourth step is about producing design solutions. Usually the process inside this step advances incrementally in iterations. First the existing knowledge of the context and users is used in defining a design proposition. Then more concrete models, such as paper prototypes or simulations are produced from this proposition. These concrete models are then shown to users and their usage is observed. At last the feedback from the models is used to design and improve the existing models. It's better to focus on doing longer studies with a few people instead of doing small tests on many participants.

Finally the fifth step is about evaluating the made designs against user requirements. At least two different types of evaluation can be used: summative and formative. The former is about assessing if the product meets the requirements set during the process. The latter is about gathering user input to help improve the design further.

Depending on the outcome of the fifth step, the process may enter an iterative pattern by going back to step number two and repeating the steps until step five again. If, for example, a serious usability issue was found during the evaluation period, the designers should change the user interface in order to get around the problem by going through steps two to four and then end up re-evaluating the interface in step number five.

The process that is outlined in the standard is quite general and usually in use one way or another. In the end it's about first doing requirements elicitation, then designing, showing those designs to actual, identified users and then using the feedback to possibly improve the design. The methodology is easy to use when faced with simple, contained applications such as desktop tools. When dealing with larger systems, like industrial robots or airplanes, simulation and user feedback gathering becomes much more harder.

3.4.1 A New Way to Design Intelligent Environments

With intelligent environments, such as smart homes, that contain ubiquitous devices and have a practically endless supply of different usage contexts it becomes hard if not impossible to do extensive user research. It is practically impossible for the designers to see all possible usage scenarios beforehand. This creates a difficulty in the usage of methods like ISO 13407. When you don't exactly know what the system will do in the end or how the user is planning on using it, you can't really start to define explicit requirements and base your design on that. The problem with ISO 13407 is that it focuses on usage instances in certain scenarios.

The problem with the current software methodologies and their incompetence in the design and development of intelligent environments is not a new issue. According to a report by the EU's Information Society Technologies Advisory Group (ISTAG), one of the head research centers on information technology in the EU, the current situation with financial models, methodologies and service providers does not allow us to advance greatly with all the new technology, and we should search for a more comprehensive solution [Kuutti et al., 2007].

Instead of comparing the design of intelligent environments to traditional software design it would be more fruitful to see that design process is more like the design of cities [Kuutti et al., 2007]. As with city design – or perhaps planning is a more appropriate term – the starting point is never a clean plate: you always have something from which you start building. With cities it might be an early settlement that has started to grow and needs official planning to evolve right and with smart homes you

are bringing the technology into existing homes. Even if the actual physical home could be designed from scratch, you still have to think about the environment's special characteristics and the meanings people give to their homes through emotional and cultural ties.

Another insightful view into the city planning analogy is that the planning never ends, but instead moves forward in small steps. Once the plans have been realized to some extend, the designers or planners need to take a look on the current situation and plan their next move. This is also true about smart homes. These are environments that evolve and change all the time. The system's actual value to the inhabitants becomes more clearer as time goes on and the interaction between the inhabitants and the system becomes part of the daily routines.

One way of solving the issue is taking the focus away from explicit usage instances and designing the system to *enable* certain ways of use. A proposed new view into design contains two new concepts: *remote* and *immediate* design [Kuutti et al., 2007]. The former refers to design done by smart home designers and the latter to design done by the actual users of the system.

The point of remote design is to move the focus further from the basic product design and work on general solutions that enable products designed through immediate design to solve real issues close to the users. Remote design aims to produce infrastructure that connects to the existing surroundings. In addition to the technological design, governmental decisions, new laws and standards can also be seen as products of the remote design process.

The main point is that since it's impossible to predict the exact future uses of the system, the system designers should emphasize on designing a system that enables users to reach their goals. This reaching is done in immediate design where users define their issues and solve them by using the infrastructures and technologies provided through remote design.

The concepts of remote and immediate design are well suited for the smart home domain. The design should start from creating an environment where users can improve their daily lives by using features provided by the smart home system. The highly praised need to do customizations to the smart home system is evident here.

The question is how to provide the users with ways to do immediate design. Since it can be assumed that most of the smart home occupants won't be interested or capable of directly programming the smart home system it is clear that a certain mechanism for personalization should be offered. Obviously the system should be very modular and offer the user optional features that can be enabled easily.

3.5 Releasing a Smart Home System

The difficulty in handling the design on intelligent environments is not the only problem smart home system designers face. The other big question is how to release the

system once it's ready to be used in real environments and especially how to offer support for the users. Depending on how big of a role the system takes in the home, the supporting might be a 24/7 job.

As outlined in previously, enabling the users to customize their system is a definitive requirement. This is also supported by the requirements in one of Samsung's studies [Chung et al., 2003], the Morphome study [Mäyrä et al., 2005] and by Rodden's and Benford's work on current domestic environments [Rodden and Benford, 2003]. The model used to deliver system that takes a somewhat bigger role in a personal home environment can't follow the traditional proprietary software-as-a-product paradigm, where a company would sell the system to the user for a ready defined price and possibly provide a limited period of support.

The first reason is the difficulty in providing support. The high price of support has already been widely noticed in traditional software development where the cost of supporting users and fixing bugs can be much higher than the original cost of developing the product. The risks grow exponentially when instead of targeting a few different computer environments, the system should fit into an unique home.

Another issue against the closed model of software-as-a-product approach is mentioned already in the analogy of city planning: the difficulty in creating a proper smart home system is that it's possible that the system will never be ready. According to Kaasinen et al. [2007] the key feature of an intelligent environment is its constructiveness, which means that hardly any environment can be implemented in one go. Instead the environment evolves gradually through the work of all participating members. This piecemeal integration idea is also supported by other researchers such as Rodden and Benford [2003].

One viable option would be to deliver an open source system under a free software license. This would allow technically oriented users to see how the underlying system works and build improvements that suit their household. Building a working community of volunteers around the system would suit a smart home system very well since it would generate more possible viewpoints and core developers would have a better chance of knowing what the users are doing with the system.

Using this kind of a model with household appliances is not a new thing. One modern example is the Neuros' OSD media player (http://www.neurosaudio.com/osd/osd.asp) that was released as an open product in beta phase and has gradually gained new features developed by paid developers and volunteers. Neuros also used small cash bounties that were paid to volunteers if they developed a certain feature to the player.

An open source model also fits nicely with the immediate-remote-design ideology. While the paid core developers and possibly enthusiastic volunteers focus on the development of critical core functionalities in the system, the community members can participate and do immediate design by producing smaller improvements that use the

infrastructure provided by the core. This of course requires that the core system uses a plugin architecture of sorts that allows the building of features without knowing how the core itself works.

Pluto and LinuxMCE are two real-life examples that use the open source approach. Both are based on the same code base and can benefit from the improvements made to the other. Pluto is the more commercial one that sells ready made units and service for installation, while LinuxMCE is a non-profit fork of Pluto code run entirely by a community of volunteers. Both will be introduced in the next Chapter.

Replacing the software-as-a-product ideology with a software-as-a-service approach would suit the development of smart home systems much better. However, this approach is not perfect either. As Kaasinen et al. [2007] point out, problems arise when trying to find out who is liable for the software. First guesses would probably target the company officially behind the product, but what about if the piece of software in question was provided by a volunteer somewhere or by another company. Another challenge comes from the upkeep of the system when people all over the world are developing it.

However the better suitability of a service-based smart home system is also supported by the notion that smart home support requires professional support and the way that this support is going to realize in the future is still uncertain. Rodden and Benford [2003] ask three questions: what are the representations of the digital services, who are involved in supporting them and how will the inhabitants engage with these services and the professionals that maintain them? All these questions and notions show that it's more fruitful to see the smart home system as a supportive service in the house, just like plumbing or electric wiring, than another computational device.

The Linux operating system provides one way to think about the support scheme for an open smart home system. Instead of one central corporation that provides support, Linux support is organized by various different entities ranging from huge corporations like Novell to smaller specialized firms with only a few employees. The openness creates a huge market for support services that in turn help in improving the original system.

4 Smart Home Projects

In the following subchapters I'll go through some smart home implementations. The first two are smart home research projects done in laboratory environments and the third is a system that has been installed in an authentic home environment. The last one is a explanation about the history and features of a real world system that you can build yourself. In addition to the projects themselves, the contrast between their focus is quite interesting. Especially the contrast between MavHome and EasyLiving gives insight to the freedom of design that many of today's smart home projects have.

I won't go into elaborate details about each project. The purpose is to explain the basics so that the names can be easily referenced later in text.

4.1 The MavHome Project

Managing An Intelligent Versatile Home is a project from the Arlington University in Texas. It focuses on the creation of an environment that acts like an intelligent agent. The AI studies the way the inhabitants live and tries to maximize their comfort and productivity by automating and predicting tasks in the house [Cook et al., 2003].

The MavHome artificial intelligence takes a strong role in the running of the house. It makes its own decisions and changes the state of the house the way it sees fit. The following short scenario was presented by the MavHome designers:

> At 6:45am, MavHome turns up the heat because it has learned that the home needs 15 minutes to warm to optimal waking temperature. The alarm sounds at 7:00, after which the bedroom light and kitchen coffee maker turn on. Bob steps into the bathroom and turns on the light. MavHome records this interaction, displays the morning news on the bathroom video screen, and turns on the shower. When Bob finishes grooming, the bathroom light turns off while the kitchen light and display turn on, and the news program moves to the kitchen screen. During breakfast, Bob requests the janitor robot to clean the house. When Bob leaves for work, MavHome secures the home, and starts the lawn sprinklers despite knowing the 30% predicted chance of rain. Because the refrigerator is low on milk and cheese, MavHome places a grocery order. When Bob arrives home, his grocery order has arrived and the hot tub is waiting for him.

The MavHome environment is composed of autonomous agents that are laid out in a specific hierarchy. One agent might be in charge of the refrigerator while another one activates the sprinklers. The hierarchy dictates which agents have more power over the decisions. The one on the top of the hierarchy is the most powerful one, while the agents on the bottom usually only control actuators or monitor sensors.

Each agent is composed out of four layers. The decision layer does the thinking and

chooses the actions to take based on information that comes from the layer underneath it, called the information layer. The third layer is the communication layer that handles communication with other agents. The last layer is the physical layer that contains all the hardware and individual devices, which may be present in especially in the agents on the bottom of the hierarchy.

Perception happens in a bottom-up manner with the physical layer notifying about changes in sensors to the communication layer which may then alert other agents that are interested in the information. The information layer saves the state and the decision layer chooses an appropriate action based on the information.

MavHome provides is a very interesting project due to its totally different approach to the problem. It's clear that the very proactive approach is something that people would not want since it goes directly against the first and foremost requirement for user control in the AMIGO research. However the agent architecture used provides an interesting view into the software design of a smart home. Generally speaking from a user requirements perspective MavHome can be seen as a relatively bad example of how things should be done, but from a software design perspective it provides nice insight.

Clearly much of the motivation for MavHome has come from artificial intelligence and agent architecture research. The Active-Lezi algorithm that they use to predict and understand the routines of the user has proved to be quite effective and longer testing in an authentic environment would be very interesting. However, the implementation as such is not suitable for general discussion on future smart homes.

4.2 EasyLiving from Microsoft

One approach to tackle the multiple difficulties related to intelligent environments is the EasyLiving project from Microsoft. The clear focus of this project is to provide an environment in which multiple I/O-devices can cooperate. Unlike MavHome, which has a direct relation to AI research and learning algorithms, Easyliving is more about finding a working architecture and a convenient communications method for all the cooperating devices and applications [Brumitt et al., 2000].

To provide the means for cooperation the platform offers the EasyLiving Geometric Model. This model makes it possible for applications to query about others in their vicinity. In the model the basic object is an entity that represents a physical object in the real world. Measurements are used to define relationships between different entities. After a set of measurements and entities have been given to the model, queries may be performed. One query could be *"which display to use to notify Jack of guests on the front door"*. Since sensory data may be very lacking and information about the context incorrect, an uncertainty factor has been taken into account when doing the measurements.

After deciding which devices to use based on the information provided by the

Geometric Model, the applications may converse using InConcert – a cross-process communication framework, designed for asynchronous messaging and machine independent addressing with a XML-based language syntax.

The actual system intelligence that tells the environment what to do and when is implemented as a hard-coded set of rules. This quality has a definitive negative impact on the learning capabilities of the system. The intelligent environment does not adapt to its user. Instead the occupants have to learn how their home works. The contrast to the learning and acting MavHome project is obvious. The richness of the EasyLiving research lays in the development and design of a working network and communications protocols.

Regarding the requirements discussed in Chapter 3, the flaws of EasyLiving are easy to spot. Most apparent is the lack of customization that is needed for smart home systems. Then again EasyLiving is a laboratory experiment that tries to shed light to the problem of connecting devices in a dynamic, heterogeneous environment and provide a solution on the sharing of context information between the devices to reach the best solutions.

4.3 Aladdin Home Networking Project

Another project influenced strongly by Microsoft is the Aladdin Home Networking Project. As the name implies, the focus is again on the networking concept of the smart home, but this time the system is deployed in an authentic environment, namely the three story house of one of the developers [Wang et al., 2000].

The network's backbone works through wired ethernet inside the house. This is something not available pre-installed on many of today's homes. The system presents an architecture which connects devices communicating through different mediums, such as infra-red or radio-frequency, through specialized bridge devices.

User Access Points or UAPs are devices which the inhabitants of the house may use to access the internet or home information such as calendars. According to the research team the ideal UAPs would be standalone or wall-mounted flat-panels dispersed throughout the house, but in the current implementation the crew had to resort to Windows 98 PCs.

The implementation allows the user to use three different interfaces to communicate with the smart home system. The first one is the browser interface which runs through a normal web-browser. Another interface enables distant communication via the use of email and thirdly it's possible to use a voice-interface to command the system. These interfaces can be used, for example, through a mobile phone so that distant monitoring of the house is possible.

Evidently the Aladdin project is much more aimed to work in real life circumstances than the EasyLiving experiment. The best insight from the project comes from the architecture and the soft-state store it uses to keep track of the context. Also

the ideas behind the device discovery and connection handling will be referenced later in this paper.

4.4 Einstein, Pluto and LinuxMCE

During the 2006 CEDIA Expo a firm called Monster unveiled the product they had been working on. Codenamed *"Einstein"*, Monster Cable's flagship product promised a cost-effective and open solution to home automation. When asked which product was the most interesting one at the expo, Richard Green, the director of the CEDIA's new technology council nominated Einstein [O'Heir, 2006]. He described the system itself as a high-end networked approach to audio and video streaming. As of July 2007 there is no mention about Einstein on the Monster Cable's front page at www.monstercable.com.

Monster cable licensed their software from a company called Pluto (www.plutohome.com). Probably one of the most far reaching decisions on Pluto's part was to also release the source code under a free, open source license. Pluto's technology was highly praised by professionals and some claimed it to be the most promising thing for Linux to grab the living room PC market. Despite the promising review, the system did not grow a significant user base [Webber, 2007].

Paul Webber was one of the people that tried the system. He noticed how the it was very tightly integrated to Pluto's own single-purpose built custom Linux distribution. According to Webber the people at Pluto would have preferred a more generic approach, but their paying customers demanded a black-box solution that would work with things like Digital Rights Management (DRM).

Webber took the platform with its 3 million lines of code and created a fork called LinuxMCE. LinuxMCE has pretty much the same feature base as Pluto's home automation system. However, the development focus is to provide all the functionalities of a smart home system as a non-intrusive, installable package that would be possible to install as a component to turn a normal Linux powered computer into a home control system. The official system supported by the LinuxMCE project is Ubuntu (www.ubuntu.com), a free and – according to www.distrowatch.com – the most popular Linux-distribution during July 2007.

Since they share the same code base, Pluto and LinuxMCE use pretty much the same concepts. The system is compromised of four main components: a core, a media director, an orbiter and a mobile orbiter. The core is the central piece of the system. Every home needs one and only one core. If bought straight as a ready device from Pluto the core is shipped as a commercial-grade Linux server. The media director component is needed whenever the user wishes to attach TV-screens or A/V components to the main system. At least with Pluto, the need for a media director may be avoided by purchasing a more inexpensive interface module.

Orbiters are the controllers for the household. The normal Pluto orbiter has a 8.4

inch touch screen and some buttons that enable you to control the Pluto system. It has built-in WiFi and a battery, which allow you to carry the orbiter around the house. In addition to the official orbiter you can use mobile orbiters which are pretty much smart home control software installed into your own mobile phone. According to the Pluto description at http://plutohome.com/index.php?section=pluto_system_overview mobile orbiters can use Bluetooth inside the house and switch to cellular service when the inhabitant leaves the coverage of the Bluetooth network. This enables controlling your smart home from anywhere where you can reach a cell phone network.

In addition to these main components the Pluto system contains optional devices such as WiFi routers, ethernet switches and security cameras that can be integrated to the Pluto system.

Pluto and LinuxMCE provide a truly interesting approach into bringing smart homes to the public. Not only is the system freely available for those that wish to build their own, it's also possible to install it to an existing household. Both projects provide basic smart home features in addition to the media handling capabilities. The features include such things as follow-me content, automatic surveillance and controlling the house via a cellphone. Obviously the system can't live up to the AI standards of MavHome or the inter-device communications of EasyLiving, but the openness of the projects make them a good starting point for any smart home endeavor.

5 Smart Home Networking

As outlined in the smart home definition of Chapter 2, a crucial characteristic of a smart home is the network. Also by looking at the requirements stated in Chapter 3 it can be seen that a robust network solution is needed. The versatile network environment of a smart home with various different networking technologies and devices, and the need to be able to react to new devices that enter the network makes smart home networking one of the most complex areas of the implementation, but at the same time it is without a doubt one of the most important ones.

In their research paper on current computer networks deployed in the ordinary households of today Grinter and her colleagues agree with claim a originally made by Richard Harper that some modern homes contain such sophisticated computational networks that they are already – consciously or not – moving us toward the vision of smart homes [Grinter et al., 2005]. During their research they discovered that some homes had very complicated networks to enable Internet access and the sharing of resources, such as printers or network storage. These networks enabled collaborative production and consumption of media and services, but on the other hand required quite a lot of work to set up and keep working.

Grinter's group admitted that the families under surveillance in their research were “more-affluent-than-average middle-class” and thus generally more open and knowing about new technology. Still the network technology caused even the most technically competent members problems, not to mention the other non-techie members of the family who rarely understood much anything about the inner workings of the home network. These huge usability issues are one of the big problems that need to be solved before home networks arrive in the majority of households.

In this Chapter I'll go through some of the different approaches that could be taken to create a smart home network. I'll start by introducing the key communication technologies and their capabilities, then do a quick introduction on different network types and finish by outlining recommendations for the networking of intelligent environments so that the user requirements mentioned earlier could be reached.

5.1 Technology

Despite the variety of different smart home projects, some clear themes can be found from the usage of technology. One apparent one is the favoring of wireless technology, such as Bluetooth, RFID and Wifi for connectivity. Even though many of these technologies need external power supplies for transmission and are unreliable at times, the advantage of not needing any cabling seems to outweigh the cons. Some papers such as the Morphome study go even a bit further and claim that wireless technologies are a strict requirement for a smart home solution [Mäyrä et al., 2005].

When bringing networking sensors to existing homes, wireless options look even

better. One clear issue is that the cost of installing wiring for wired sensors would become very high. According to Meyer and Rakotonirainy [2003] the price of installing wiring for a sensor might be 50 – 90% the cost of the sensor itself. Other obvious advantages also exits. For example wireless networks enable moving the devices around without worrying about the wires and make it possible to easily bring new devices to the network. Incorporating wireless technologies to legacy devices, such as light switches would make it possible to easily reconfigure their places and replace them when necessary.

Even with all the advantages of wireless technology traditional cabling is not a totally discarded option. Although cabling is more costly to construct and much more harder to change in the future, it provides distinct advantages such as reliability, security from eavesdropping and high throughput – features that are often seen as the weaknesses of wireless technologies.

In the following subchapters I'll go through some of the key communication technologies that are used in smart home and intelligent environment projects these days. I'll go to show that the technologies fill a certain niche in an intelligent environment and that these have to be taken into account during the design process.

5.1.1 X10

As a communications method invented in 1975, X10 is the oldest of the communication techniques presented here. It's also the only one that relies on wiring. Despite its age, the technology is in wide use even today. Especially in smart home projects such as the Aladdin project [Wang et al., 2000].

Undoubtedly the best part of X10 is that it uses the existing electrical network in a home for the data transfer and therefore doesn't need special cabling. The transmission of data is done by sending the information on a carrier signal of 120kHz through the electrical outputs in the house. The data is compromised of a four bit *house code* and a four bit *unit code* followed by a command that can be four bits in length. The combinations of unit code and house code allow for 256 different entries, which means that one X10 network may have up to 256 individual devices. It is however possible to use the same address for multiple devices.

Although the X10 technology allows for any devices connected to the electrical network of a home to communicate, the technology has a few critical shortcomings. First is the limited bandwidth of the protocol and secondly the well known lack of reliability with the protocol [Wang et al., 2000].

The low throughput prevents the usage of X10 for applications like web browsing, instant messaging or transmitting A/V signals. Instead X10 is mostly suitable for simple “on/off” controlling of legacy devices such as lamps or video recorders or polling the states of X10 capable devices. Although still very popular, the X10 protocol should not be the first choice as a communication technology for smart home systems or devices.

But it serves a good role as a supportive technology that allows the controlling of older devices through the smart home system.

5.1.2 Bluetooth

Bluetooth is a wireless technology originally designed to replace cables when connecting devices like mobile phones and laptops. According to Wikipedia (http://en.wikipedia.org/wiki/Bluetooth) the technology was named after Harald "Bluetooth" Gormson, an ancient king of Norway. Since its inception Bluetooth has evolved into a standard for short range wireless connectivity.

According to Dideles [2003] the design of Bluetooth was driven by several usage scenarios. For example the developers envisioned a conference where all the participants could share data easily with their mobile phones or other Bluetooth enabled devices. One other usage scenario had the idea of a mobile phone that would automatically switch from using a normal GSM-connection to a Bluetooth connection that would utilize a land line when it would detect the presence of such a service.

On the technical side, Bluetooth works on the unlicensed Industrial Scientific Medical (ISM) band at 2.4 GHz. The ISM band is open to everyone and therefore all systems using it have to be prepared to encounter other devices utilizing the same area. Bluetooth does this by using Frequency Spectrum Hopping Technology (FHSS), which means that two connected Bluetooth devices hop from one frequency channel to the other after each sent packet. In effect this means that if one channel happens to be occupied by another device, the Bluetooth enabled devices will change to another one and the communication will only be mildly affected by the crowded spectrum. FHSS also ensures that multiple overlapping Bluetooth networks may coexist without hampering each others connections.

Bluetooth enabled devices may form *piconets* – networks that contain two to eight devices. One of these devices acts as a master that relays traffic to all the other participants that are called slaves. Only one master may exist inside a piconet, but the role can be easily switched if one of the slaves would like to become the master. While acting as the central hub for communication, the master also schedules when the slaves may send data and therefore controls the overall bandwidth of the network.

When one or more of the devices in the piconet are connected to another piconet, a *scatternet* is formed. Bluetooth devices may act as a slave in several piconets, but may only act as the master of one. Devices connected to more than one piconet act as relays that transmit data from the other piconet to the other. Bluetooth devices may only send data in one piconet at a time and when connected to multiple nets they use the Time Division Multiplex (TDM) basis, which means that the devices switch between the networks on a specific time interval. Naturally this has a somewhat negative effect on the bandwidth.

The average range of Bluetooth devices is around 10 meters, but more powerful

devices might even reach 100m [Dideles, 2003]. Generally speaking Bluetooth can be seen as a short range wireless technology when compared to technologies like the IEEE 802.11-family of wireless technologies. However, the short range of Bluetooth networks may also be used as an advantage. For instance a mobile device that uses the Bluetooth network is most certainly very close to the receiver. This context information could then be used in the smart home system. For instance Pluto and LinuxMCE smart home systems use the relative signal strengths of multiple, scattered Bluetooth receivers to pinpoint the location of the user, or rather the user's Bluetooth phone. This in turn is used to provide follow-me content.

5.1.3 IEEE 802.11

The IEEE 802.11 family of wireless technologies has been widely accepted and has experienced much growth in the last few years. Often called WiFi or simply WLAN, the main driver for its rapid success has been the need to replace traditional, cable-based networking in offices, homes and public places, such as cafeterias [McFarland, 2003]. The wide spread success coupled with the rather cheap price tag on 802.11 capable hardware, such as access points, has brought easy-to-use and often free Internet access in many public places to the masses.

The network topology in the currently used technologies is centralized. This means that all clients in a network connect to a central entity, often referred to as an access point. This entity keeps track of the clients and routes data accordingly. Work to provide real mesh networks is being done. With a mesh network the IEEE 802.11 devices would not need a central entity. Instead the devices would act as peers and route data from one another. One of the main drivers for this research is the possibility to relay wireless access to desolate places [Hiertz et al., 2006].

The strong point of the newest 802.11 based technologies is the speed of the network. With the newest IEEE 802.11 technologies such as 802.11n, data rates over 500 Mb/s are feasible [Hiertz et al., 2006]. The technologies work on the 2.4 GHz band and according to McFarland [2003] the power need for various WLAN cards by different manufacturers falls between 0.14W and 2.06W when the device is idle, so the price for high throughput is not great, but somewhat large for the smallest mobile devices.

5.1.4 ZigBee

As the newest of the communications technologies, version 1.0 of the ZigBee specification was ratified on December 14th, 2004 [Wikipedia 2]. The technology is based on the IEEE 802.15.4 standard and like the two other wireless technologies, it too works on the unlicensed ISM band at 2.4 GHz and has a transmission range of up to 100 meters with a maximum speed of 250 kilobits per second. ZigBee is also capable of acting on frequencies of 915 MHz and 868 MHz [Schindler, 2004].

ZigBee's target is to provide communications for sensor and control devices that don't require bandwidth but do require long operation times with batteries, and flexible network topologies. To make it possible to build devices that require the absolute minimum in power, ZigBee devices come in two distinct flavors: full function devices (FFD) and reduced function devices (RFD). The former ones are devices that act pretty actively and consume much more power than the latter ones that mostly sleep and send transmissions [Kinney, 2003].

The RFD:s can only act as the endpoints of a network and need at least one FFD to communicate. This means that a network of one FFD and multiple RFD:s can only form a star topology, where all the endpoint devices (RFD:s) connect to one central device, which is the FFD. However, by using multiple FFD:s the network may form a peer-to-peer topology or a mesh. In these networks the FFD:s act as routers that transmit data between clusters of the network with one FFD acting as the network coordinator [Kinney, 2003].

5.1.5 RFID

Radio Frequency Identification was first described by Harry Stockman in a ground breaking paper titled “Communication by Means of Reflected Power” in 1948. Resembling technology was already used during the Second World War, where the British Royal Air Force tagged their planes with radio transmitters that enabled them to see which planes on the radar are their own . More research was carried on the subject during the fifties and the sixties, but the real breakthroughs in technical development were seen once microchips and integrated circuits hit the markets during the seventies. The first real application of RFID technology was seen during the sixties in the electronic article surveillance business when a very simple RFID tag was attached to high value items in retail stores to prevent theft. The usage and research of RFID technology continued to grow during the next decades and during the 1990's the first single chip RFID tags were brought to the market [Roberts, 2006].

Today RFID is generally used to describe technology that is used to identify objects using radio waves. A typical system has three parts: a RFID device, a RFID reader with an antennae and a host system or a connection to one. These days the RFID devices are commonly known as *tags*. These tags generally fall into two categories: active and passive tags. Active tags have their own power supply and passive tags work by inducing the power from the transmission made by the RFID reader and reply using that. Active tags are generally read/write, while passive ones are read only. Semi-passive tags that have their own power supply for the internal circuit, but still use the induced power from the reader also exist [Roberts, 2006].

Because of their own power supply, active tags are bulkier than passive ones. Their life is also limited by the lifetime of the battery even thought this can be as high as ten years. Passive ones have a theoretically unlimited life span and they are lighter, smaller

and cheaper. The trade-off is a smaller data storage, shorter transmission range and the requirement for a more high powered RFID reader. Tags come in various different sizes, but the smallest ones can be as small as 0.4mm x 0.4mm and thinner than paper. RFID has not been fully standardized yet and thus RFID devices tend to work on various different frequencies. Low frequency devices use the band from 100 kHz to 500 kHz, intermediate frequency devices transmit on the 10 to 15 MHz band and high frequency devices generally work between 850 and 950 MHz or from 2.4 GHz to 5.8 GHz. Which frequency area the device uses depends on the application and range requirements. Generally the higher the frequency, the better the transmission range. Low frequency devices tend to have an effective range of about 30cm, while passive tags working on the high frequencies can transmit up to 3 to 5 meters. The use of active tags can boost the range up to 100 meters [Roberts, 2006].

Roberts [2006] also writes about the negative sides of RFID. The biggest problem is the lack of privacy as simple RFID readers can be bought easily and used to read all kinds of RFID tags. In addition to the possibility of doing corporate espionage and information gathering by reading the tags of others, it's possible to do tag spoofing where an RFID tag is duplicated. RFID technology is also vulnerable to denial of service attacks and other forms of technical attacks. It has also been proven that it's possible to reprogram some the tags. These issues and the fact that the tags themselves are physically quite fragile must be taken into account.

In relation to smart home research RFID tags have been mostly used to bring tangible and ubiquitous interfaces to various devices. For instance the research carried out at Samsung Electronics by Kim et al. [2004] created a working proof-of-concept of a smart gate reminder. The reminder was placed next to the front door and it would remind the inhabitants of the house before they left the building about things they had written up in their calendars. The smartness of the gate reminder came from two points: it knew who was on the front door and it knew the time of day and the events in the calendar. The real value was on knowing who the user was and this was achieved with RFID tags that were carried by the users.

In related research by Elliot et al. [2007], a set of smart household objects were devised. These devices had embedded RFID-tags that could be used to identify which device was currently close to the RFID reader. Also when a device was found close, a certain action might be triggered. This approach is used in the Elope-project [Pering et al., 2005] where certain device actions are triggered when a device is moved close to a reader device. For instance one could bring his RFID tagged Bluetooth phone close to a RFID reader to trigger the forming of a Bluetooth network between the mobile phone and a nearby computer.

In summary it may be said that RFID tags provide cheap and efficient means to construct devices that can be utilized in ubiquitous, tangible interfaces or to provide device dependent information. Also the active tags may be used to save information.

The passive nature of RFID makes it a viable technology for the wireless smart home as it creates only a minimal amount of traffic thus not affecting the other wireless technologies that much. Also the possibility to use different frequencies makes it less intrusive. The privacy concerns must be taken into account and limit the use of RFID to less crucial tasks. For instance the usage of a sole RFID tag to work as a house key is not wise as the key can be copied easily by other parties. When such identification measures are wanted RFID tags should be incorporated with other identification techniques like passwords or facial recognition.

5.1.6 Summary of the Technologies

So how should the different technologies be used? None of the technologies replace one another completely. All have their own distinct roles in a intelligent environment setting. The overlapping has been a common question especially between Bluetooth and ZigBee which would seem to be similar technologies at a first glance. However the roles of the two are quite different. Patrick Kinney [2003] sums up the differences by saying that Bluetooth seems to be much better suited for things like PDA to printer connections or hands-free audio, while ZigBee is better for controlling an environment with lots of devices that need long battery lives.

On the other side the long transmission range and high bandwidth makes WLAN networks the best choice for high throughput requiring transmissions like video conferencing or Internet access. The network topologies provided by the current WLAN implementations are not as diverse as with the two other technologies, but normally one central point for Internet access should be enough for a normal household and can also boosts security as it limits the possible attack routes for outside hackers and malicious programs. The usage of WLAN relays might also be suitable for building networks with better coverage.

As a practical example it might be conceivable that low power consumption ZigBee devices are used in the places that they were designed for, like in light switches, thermostats and smoke alarms. Bluetooth on the other hand is excellent for quick local connections, like syncing one's mobile phone with the desktop computer, remote controlling or transmitting sound to external speakers. Longer range communications such as A/V transmission between rooms is best handled with WLAN. With a working central entity that enables cross-communication between the different networks the benefits of these technologies would be greatly improved. For example the inhabitant could use her mobile phone to connect to the home network and turn on the ZigBee enabled lights or browse the files on her PC.

RFID is a trickier technology to fit into a smart home environment. Since the real power of RFID lies in the possibility to make things easily recognizable by the computer, heavy usage depends on the support from external industries, such as food or clothes manufacturers. Some direct possibilities do exist for practical home

applications. One would be to have RFID stickers that could be attached to objects that tend to get lost once in a while, like keys or remotes. These devices could then be found easily with a powerful RFID-reader.

It is obvious that the technologies should coexist peacefully. This might be more difficult on the wireless side as all of the three technologies act on the same ISM bandwidth. Procedures like changing WLAN channels and using the two lower frequencies for ZigBee devices should make it possible to get around this problem. On the design level it becomes the responsibility of the smart home system to make the technologies talk to each other, so that people can control their ZigBee devices via Bluetooth or WLAN, or turn on the X10 connected sauna through their mobile phone.

5.2 Network Topologies

In the following subchapters I'll give a short introduction to the different network topologies. Some of them have already been mentioned in the technology section. I'll cover the ones central to the smart home environment. These are star, bus, mesh and hybrid topologies.

5.2.1 Star Topology

Often also called a centralized topology in networking, this network relies on one central entity that relays the traffic between the endpoints of the network. In other words all traffic on the network goes through one entity. This is the topology used in the current WLAN technologies as well as in solitary Bluetooth piconets and ZigBee networks with only one FFD.

An extension to the star topology can be reached by using relays that can connect multiple star topologies together. This way the network becomes a sum of many star networks, hence some call these Star-of-Stars-networks. This approach was used by Hiertz et al. [2006] in their research on WLAN networks and it's the way scatternets form in Bluetooth. Relay-technology brings big advantages like extensibility and robustness to the network.

Once the number of relays grows big enough, the topology of the network starts to look less and less like a star. Usually relay technology that is applied to devices designed originally to work in a star topology has some cons. For instance connecting multiple Bluetooth piconets together decreases the overall throughput of the network since the devices acting as relays must focus on two different networks.

Another weak point of a centralized network is that the whole network relies on the central entity to function. Endpoints are completely unable to find new routes if their central entity falls and thus the network will collapse. When using relays, the fall of the central entity will crash the whole subnet and end all transmissions that have to pass though the crashed network to reach other networks.

5.2.2 Bus Topology

A bus topology forms around a *backbone*. The backbone carries all the transmissions from one node to the other and all nodes are directly connected only to the backbone. For instance the Aladdin project [Wang et al., 2000] used a ethernet-based bus topology to network all the devices in their smart homes.

A linear bus topology has exactly two end points, often called terminators. All the devices are connected to the bus between these points. All the transmissions these devices do are also available to all the other devices on the network. A bus topology is more reliable than a star, because it does not have a central entity that controls the traffic. On the other hand the negative side of a bus topology is that all the traffic can be monitored by all the parties and therefore all the components connected to the bus should be trustworthy.

5.2.3 Mesh Topology

A pure mesh network has no central device. Each device is capable of forming connections with any other device on the network and relaying data from others. This means that in a three device network where A and B see each other and B and C see each other, A should be capable of speaking with C even though no straight route can be formed. In this case the traffic between A and C would be routed through the device B.

Well functioning mesh networks have the advantage of being very robust. Unlike in the star topology, the network won't fail if one central device fails. Another advantage is that the speed of the network is increased when devices may form direct connections instead of relaying transmissions via a third party. Thirdly the network range increases dramatically and is not dependent on a central device, but instead on the range of the outermost devices. And as one last advantage the load on the network is distributed more nicely when the devices use the best possible routes.

With the many advantages comes a few disadvantages as well. Firstly the devices require much more computational power to calculate the best routes and transmit all the data. This mainly is the reason why Zigbee Reduced Function Devices can't connect to multiple hosts. Secondly mesh networks present a serious security problem, when the access point of one individual device can't be strictly controlled. While in a star topology you can be sure that the connection was made with the central entity, in a mesh network the connection might be initiated through any of the participants.

5.2.4 Hybrid Topology

A topology where two or more of the simpler topologies are combined is called a hybrid. One example is the aforementioned Star-of-Stars-topology which combines a multiple star topologies together. Another can be formed by combining a star topology with a bus topology so that the devices connected to the bus backbone can act as central hubs on a star topology.

Given the array of different networking technologies a smart home could use and the limitations of those techniques, a hybrid network is probably the eventual result if all the devices are to be connected. In a hybrid network case it is wise to construct the network so that the most robust technology is in the middle. For example if networks A, B and C are to be connected to each other and they all have different protocols, the most robust one should be the central network that acts as the relay between the two. Of course the ideal solution for robustness would be to have all the networks some how connected to each other so that the failing of one would not harm the others. On a more practical note something like the bus topology with the ethernet backbone used in the Aladdin project would provide a robust center for a smart home network.

5.3 A/V Networks and Computer Networks

Grinter et al. [2005] discovered that homes generally have two different networks: audio/visual device networks and computer networks. A/V networks are constructed out of multimedia devices, such as stereos, DVD-players and televisions. The cabling in A/V networks is usually just composite wiring or equal, and the communication between the devices stays quite simple.

Computer networks join all the family's computers and peripherals, like printers, into a common network. Usually the advantage from this network is shared internet access or printing resources. Computer networks may be formed with traditional ethernet wiring or by using wireless networking.

Grinter's group also noticed that these two networks are used in different contexts. Usually A/V networks are packed quite tightly into groups and are quite easy to see already on the outside. Computer networks on the other hand are usually much more hidden. Especially when using wireless technologies. Grinter's study also noted that the computer networks proved out to be surprisingly complex. For instance some families wanted to separate their “work” network from the “home” network. Another distinction between the two networks was that the computers on a computer network usually belonged to someone, while the devices on the A/V networks were usually considered to belong to everyone even though some of the A/V networks were solely used by one inhabitant.

One of the goals for a smart home network is to join these two networks into one. Today's large LCD screens and audio devices work well as output devices for a smart home user interface. Also networking between these devices brings advantages like having all recorded videos in one place where all the in-house video players may reach them. On the networking side it's quite easy to say that the computer networks will form the core of a smart home system while A/V networks are present on the borders.

5.4 Towards the Ideal Solution

As outlined in the beginning of this Chapter the difficulty of setting up and maintaining

a home network is one of the big obstacles in bringing these networks to ordinary homes. As the network forms the core of the intelligent environment it's crucial that the network is robust and easily understandable. Wireless networks provide one way to bring easily configurable networks to home environments, but they lack the robustness to suit for every situation. Wired networks on the other hand provide reliability and throughput, but are static and expensive to install properly.

When designing a network for a mobile smart home system it becomes evident that we can't rely on one technology. No presumptions can be made about the future environment. For instance relying on a single WLAN network to cover the whole house is a mistake if the house happens to be an old structure with thick stone walls that cut the transmission range of WiFi drastically. The home network that fits into existing homes should be dynamic in addition to the robustness. Network bridges should make it possible to swap from one networking technology to the other if the previous one fails. For instance the inhabitants in the old stone-walled house could use an ethernet backbone that connects multiple wireless routers together to form a more comprehensive network.

When thinking about the concepts of immediate design and remote design that were introduced in Chapter 3, it becomes clear that in order to enable immediate design, the system should make connection forming as easy as possible. Ideally the smart home system should abstract the networking technology and provide a unified way to access any device. This kind of network abstraction means that there has to be a central entity that knows all the devices in the house or at least knows where to forward the transmissions. These issues will be dealt more clearly in Chapter 7.

6 Smart Home User Interfaces

It's impossible to find one definitive direction in user interfaces when looking at the vast array of smart home projects. The fact is that most of the projects have different goals and areas that they are interested in smart homes. For instance the MavHome project [Cook et al., 2003] was mostly interested in researching a learning algorithm that could detect patterns in the occupants daily chores, the UMASS project [Lesser et al., 1999] wanted to research appliance cooperation with a new resource sharing algorithm and the research done at Samsung innovated multiple smaller home appliances that were smart on their own [Park et al., 2003].

When looking at the various different projects from a distance one can however see that they share a few common trends that affect the general functionality of the system and the user interface more or less. These trends are in a way the minimum requirements that the system must meet in order to be qualified as a competent smart home system. The fact that most of the projects have ended up implementing these trends shows that they are important.

Defining these trends explicitly is needed so that they may be taken into account during design. In this Chapter I plan to do this and continue by categorizing different parts of the user interface. I'll conclude by writing about the different user roles in a smart home system and about the physical placement of these user interfaces.

6.1 Trend One: Divide and Conquer

One strong trend in smart home systems is the need for a centralized control. Projects such as UMASS [Lesser et al., 1999] and MavHome [Cook et al., 2003] do research that is very dependent on this one characteristic. The strongest points of having centralized control are obvious: resource and action coordination become much more simpler when there is one component that controls the others. The clear minus side is that the total functionality of the system is greatly dependent on this one, ruling component.

Looking at the requirements defined by the AMIGO-project about energy saving, appliance integration and cost-effectiveness in the third category, it's quite obvious that a requirement for some kind of centralized control is justified. The resource coordination algorithm produced by the UMASS project [Lesser et al., 1999] is one very promising way of meeting this requirement.

It's crucial to point out that having a central component does not necessarily mean that one component has to directly control all the others. One god-component could even turn out to be a crucial mistake. The safest way is to take the Aristotelian middle road and go for a architecture where some components control some aspects of the system. This was the approach in the UMASS-project where the resource coordination

agents would control their own resources and this way they would be mostly unaware of the other resource agents in the system. This kind of division of responsibilities makes development and debugging errors much easier.

6.2 Trend Two: Multimodality

As outlined in Chapter 3, one of the major research drives for smart homes are living quarters for the elderly. Solutions are researched to make life easier for people with disabilities and make them less dependent on external help. This area is the one where a lot of the research for accessible user interfaces is.

Multimodality is at the core of this research. The term is meant to describe a system or interface that is capable of using one or more modalities in order to interpret input and produce output. In a user interface this could for instance mean the capability to take commands through voice and keyboard. Generally a well implemented multimodal user interface is more accessible since it provides the user with more ways to interact. In the case of smart homes a multimodal user interface is the only way to go if you wish to cater for all the different types of users: hearing impaired users would have difficulties with an audio-only system, while visually impaired couldn't use a system based solely on graphical output.

A research project conducted at the John Moore University in Liverpool set to design an optimal user interface for a smart home [Sainz de Salces et al., 2005]. They made a comparison between three distinct user interfaces: one entirely visual, one based on audio and one multimodal that used both audio and visual information. Based on user tests they found out that the visual user interface was the most effective one, with the multimodal coming as a close second. When we count in the accessibility gains that come with multimodal interfaces and the strong need to have good accessibility in a smart home, we can safely say that one requirement from a smart home system is to provide more that one modality to conduct at least a big part of the actions.

6.3 Trend Three: Variety of Interfaces

In a research conducted Koskela and Väänänen-Vainio-Mattila at the Technical University of Tampere three different user interfaces were compared in focus-groups, interviews and in a real smart home setting for a period of six months. The user interfaces where built into normal everyday appliances that are found in most homes: a media terminal controlled via a remote control, a personal computer and a mobile phone [Koskela and Väänänen-Vainio-Mattila, 2004].

In their research they discovered that the daily activities of the inhabitants focus around different *action centers* in the home. For instance the sofa would be one action center where the inhabitants would gather to play games, watch TV or read. These action centers were placed all around the house and therefore showed that using one central position for smart home control would be ineffective.

They also found out that users have different contexts in which they wish to use the interfaces. The most crucial divider was the surrounding company: sometimes users preferred to do actions all alone, while some other times the presence of other people was welcomed. The media terminal user interface was found to be the best one when doing tasks together and the mobile phone proved to be quite ideal for personal or private tasks. A very large plus with the mobile phone was also the capability for distant use outside the house.

In the end the users found the media terminal quite clumsy and complained that it got in the way when watching television. The personal computer with its graphical user interface was seen as the best solution for more complex tasks or ones that require more accuracy, such as setting timers for lighting. The phone was the best user interface for day-to-day operation.

Taylor and Swan [2005] noticed something very similar in their survey done in real contexts of eight different homes. They chose the term *activity centers* to reflect the various different locations around the house that act as hubs for distinct actions. For instance they found out that the refrigerator door was a significant communication channel for all members of the family and often acted as the place where people would coordinate their actions, such as schedule a common meal time for the evening.

According to this research a smart home system should offer at least one way of controlling the house through a mobile device. This requirement is also present in the AMIGO-research, where in category four the need for helping with tasks from the home or workplace was brought up. The notion of different action centers brings up another requirement. Since the users have various different use contexts and there is no single user interface that could meet all the requirements, a selection of different user interfaces should be offered.

6.4 Trend Four: Weak Proactivity

One of the most controversial topics in smart home research is the question of how smart the home actually should be. The Morphome study conducted in the Hypermedia Laboratory at the University of Tampere ended up recommending a scheme that follows the principle of *weak proactivity*. Instead of strong proactivity such as the learning and automatically adapting environment in the MavHome project [Cook et al., 2003], weak proactivity essentially means that before the system does anything on its own it first asks the user for permission [Mäyrä et al., 2005].

For instance if the system notices that the user always closes the curtains just before going to sleep it would ask the user next evening if the user would like the curtains to close automatically before going to bed and the user may approve or disapprove the action. This scheme effectively puts the control into the users hands and as the first category of the AMIGO research pointed out, this is very important for the users. In the Morphome study this possibility to approve or disapprove schemes was also raised as

one of the key factors in creating a sense of control for the inhabitant.

6.5 Trend Five: Customizability

The Morphome research also concluded that every home is unique at least to its inhabitants [Mäyrä et al., 2005]. A notion shared by various other smart home papers and ethnographic studies on homes. In most cases this uniqueness is expressed by customizing the home – mostly by decorating. This need for customizability also applies to the smart home system and causes a need to balance between the normal user interface guidelines for consistency and the need for personal user interfaces.

One way of offering customizability on the outlook is to implement support for themes. However it is questionable if theme support permits enough options for more demanding users. A more complex approach is to allow restructuring the user interface. The term used to describe these is *adaptable user interfaces*. For example the user could reorder menu items, hide functions and define new button texts for the user interface. Adaptable user interfaces usually create a positive sense of control for the user, but it must be remembered that not all users want to reconfigure their user interfaces [Arhippainen, 2007].

An even more technically advanced user interface could change to reflect the most used practices of the user. These kinds of user interfaces are called *adaptive user interfaces*. Even though an adaptive user interface might increase the effectiveness on common tasks without any extra effort from the user, it introduces negative effects such as the fear of losing control when it might not be evident to the user why the interface has changed since the last time she used it [Arhippainen, 2007].

6.6 Categorization of Different User Interfaces

Categorizing the user interfaces helps the developers and users to talk about the technology and functionality. Something like this is very important when creating different scenarios to present in a co-design process with the possible future users of the system. It also helps to recognize the various hardware type dependent user requirements and common usage contexts, such as the main purposes for a mobile phone user interface or the role of a kitchen touch-screen user interface.

The simplest way of dividing the user interfaces into categories is to use the device type as the category. Mobile phones and PDA:s are devices that can be used inside or outside the house, while personal computers, media terminals and touch-screens are in-house devices. It's worth noticing that if the smart home system provides a web-based user interface, static devices, such as PCs, might also be used to access the smart home system from outside the house.

The division into mobile devices and static, in-house devices makes discussion easier. One of the most critical requirements in the AMIGO-research was the need for security and privacy. Creating a user interface that gives access to all the functionalities

of the smart home core from outside the house, would present a horror scenario for the inhabitants if one malicious hacker could take control of their home.

The only solid way of making sure that nobody could take control of the house from the outside is to make sure that all powerful user interfaces remain inside. This way the security of the system equals the security of the house. Once a burglar gets inside the house, the security of the smart home system becomes a non-relevant issue. The security scheme of the smart home will then reflect the current state of most houses: all devices, like lights, washing machines, refrigerators and media players, can be used freely if the user gets next to the device or in range of the remote control.

Limiting too much the functionality of the user interfaces that might be accessed outside the house is not a perfect solution. In the usage study of Koskela and Väänänen-Vainio-Mattila [2004] the possibility of accessing the smart home functionality from outside was found to be one of the favorite features of the users. A flexible access policy scheme could serve the purpose. Marking one type of user interfaces as the administrator user interface that could change the access policy of others, would give the user freedom to affect the functionality of her different user interfaces. The only restriction should be that the administration of these policies could only be handled from inside the house and preferably by giving a passphrase.

One way of granting the mobile user interfaces more power would be to detect the network they are using. For example a laptop that is logged into an unidentified wireless network is more of a risk than a laptop that is logged into the wired network inside the house. Once the system could detect that the mobile interfaces are actually inside the house, it could grant them the permits that user interfaces inside the house get. Using hardware constraints to enforce the detection is a relatively secure way of doing this. A mobile phone could also get in-house permits by using the Bluetooth network that has a much smaller coverage than the mobile phone network. This would permit the mobile user interfaces to stay mobile, while being more powerful.

6.6.1 The Ubiquitous User Interface

A special case of an in-house user interface is the ubiquitous user interface. As explained before in Chapter 2, a ubiquitous view of a user interface fits well in the smart home environment. While conventional user interfaces such as graphical ones work well and probably more efficiently for tasks that require precision , the real added value to smart home usage is generated by the ubiquitous interface that – as the definition of ubiquitous says it – is available all over the house.

Users could use the system to do simple tasks like turn on the lights or open the curtains of the room in which they are currently in, and because the ubiquitous user interface is always present and can often reach a user inside the house, it works perfectly as the main communication channel when the user is not in front of another interface. Audio or video notifications of arrived mail or an incoming phone call can be

given through this interface and directed to the interface nearest to the user. This way the ubiquitous user interface takes the role of a notification agent that tries to get the attention of the user and possibly asks her to use the nearest conventional user interface that is capable of handling the task that needs attention.

The ubiquitous user interface is relatively tightly integrated into the actual infrastructure of the house. It needs speakers and microphones in each room just to cover the whole house with the possibility to use audio input and output. Using video would require installation of screens and possibly cameras in all the rooms of the house. While this might seem interesting from a technical point of view, it goes straight against the highly cherished requirement for privacy by introducing the scenario of remote surveillance of the inhabitants.

Video gathering and surveillance in general is one of the key questions in research related to ubiquitous computing. As Abowd and Mynatt [2000] wrote in their research paper, one of the problems is the fear that users encounter when they don't know what a certain computational system is doing. The same fear is probably related to the fear of losing control that was discovered in the AMIGO research. Abowd and Mynatt go further to conclude that this fear is also related to surveillance systems and other sensing parts of a computational system. Just as on the physical side people know when they are in a public place or whether others can easily see them, it should be obvious also on the virtual side. People need cues to signal them that they are being watched. Video surveillance is the most obvious technology where these issues need to be considered, but many of the principles also apply to other surveillance technologies such as audio and motion.

Video surveillance can be seen as a double-edged sword in smart homes. On the other hand it provides video footage that can be used to track down burglars and permits using the video input to create novel interfaces, such as gesture-based systems, but on the other hand it can be seen as a huge privacy issue. This is especially true with cameras that are placed indoors. One common reason to use cameras in smart home projects is the possibility to follow where the user is and this in turn provides the system with the important context information that it needs.

When deciding on the usage of sensors the high prioritized and frequently mentioned need for privacy should be taken into account. Meyer and Rakotonirainy propose the use of smart sensors that contain embedded chips and are capable of processing their own input data. This way the amount of critical private data can be cut already at this phase as the data from the sensor does not leave without processing [Meyer and Rakotonirainy, 2003]. While this may sound technically quite secure, I would argue that the resident of the house still feels like being watched. Probably even more so, because the camera provider advertised that the sensors were “smart”. Instead of using intrusive ways to monitor the house, the system should survive using less intrusive sensors.

While video cameras have many advantages, the idea of having a video camera in your bedroom might not sound like a good idea to most users. Because the home is a very private domain for it's occupants, relying too heavily on video input might create an Orwellian nightmare. Placing one video camera outside the front door and perhaps one in the living room can however be seen as justified and provide nice features such as seeing who is at the door from anywhere in the house. Also providing physical ways such as privacy shutters to turn off the recording might help in conveying a sense of control.

Video cameras should be seen as optional components that can be installed mostly for security reasons. Less intrusive ways to collect context data exist, such as the usage of mobile phones in the Pluto system or the use of RFID tags in projects like Elope [Pering et al., 2005]. While for the technically minded these approaches could seem less interesting because they require the presence of an external device, the approach could seem even more better to an inhabitant as she could purposefully fool the system by leaving her cellphone in another room. An this in turn could increase her sense of control.

One of the main challenges with ubiquitous user interfaces is the correct interpretation of input. The most straightforward approach into improving interpretation accuracy is to limit the possibility of different inputs. The best way to do this is to analyze the context of usage. For example if the user is in a room that has no light sources, the system can safely ignore all commands that have to do with lighting when searching for a matching command. Especially with audio-based systems, the possibility to reduce the space of possible inputs greatly increases the reliability of the system.

Another way to improve interpretation was used in the Open Agent Architecture -project that implemented a Modality Coordination agent that was responsible for gathering data from all the different inputs, creating a common interpretation that was based on all of the different modalities and sending the command forward [Moran et al., 1997]. Making use of all the different modalities would of course improve detection, but it would also require the support for their usage, meaning that for example both audio and video should be recorded at the same time whenever the user issues commands. Therefore an interpretation scheme based on good context information might be more suitable.

6.6.2 Tangible User Interfaces

In addition to the user interfaces exposed through devices such as mobile phones and desktop computers, the home environment contains multiple user interfaces that are tangible by nature. The concept tangible means that their usage is done via physical interaction like pushing or grabbing. For example our everyday homes are already filled with tangible interfaces like door handles, light switches and water taps. The tangible

devices of today are tangible because the require the use of physical power. The challenge lies in bringing the advantages of tangible interfaces to help us in the complex world of information appliances like computers and A/V hardware.

Luckily new technology like RFID-tags provide new interesting approaches to the creation of tangible interfaces. This is exactly what the Elope-project has been aiming for in their work [Pering et al., 2005]. The basic idea is to trigger actions by scanning RFID-tags on objects. This allows random devices to know about each other and their capabilities and takes away the user's pain of setting up a communications venue between them.

A scenario proposed by Pering et al., [2005] involves a woman walking into a conference room to hold a presentation. She has the slides for the presentation stored on a smart phone that she is carrying and she would like to transfer them to the big screen in the room. A participant in the room hands her a remote that is used for presentations on the big screen. The woman scans the remote with her smart phone that is capable of reading RFID-tags. This makes the smart phone form a network with the big screen and transmit the presentations there, where she can then pick the one she wants to show. The advantages of this technology may be even further seen when imagining different remotes that would allow different actions, like playing music from the phone or forming an Internet connection.

The RFID-approach brings tangible features to current technical devices, but another challenge lies in integrating the tangible interfaces of today into the smart home network. For instance how could the old, inherited table lamp be attached to the system so that it could be controlled via the smart home user interfaces as well as with the traditional switch. On some cases the integration of a legacy device might be too difficult without taking away one of the use methods. In the lamp case a simple ZigBee powered controller won't work if the light also needs to be operated through the traditional switch and both rely on controlling the same electrical output.

Another interesting theme is the research of new, innovative tangible interfaces that connect to the smart home network. Some examples of these “smart” devices already exist on the consumer market. For example the Nabaztag-series of lamps, that are available in places such as www.thinkgeek.com may be connected wirelessly to the Internet and made to react to events like incoming mail, RSS-feeds or weather forecasts. Bringing input to these devices would allow the user to interact more freely with the objects in her household. With a sense for touch, common household items like lamps, flowerpots or paintings might enable new powerful ways of interaction.

6.7 User Roles and Device Permissions

In the fourth category of requirements found during the AMIGO-research, the need for storing personal preferences was noted. This also goes hand to hand with the need for customization. On the user interface side this implies two things: the need for support of

changing the interface to better fit the needs of the user, and the storing of user preferences. The smart home system also needs to support various different user groups. House inhabitants might be granted personal accounts, but the need to log in every time to do the most basic tasks is a bit questionable. As with the user interface types one or two of the users should act as the administrators that solely have access to the access policies of others. In a family this might be the parents.

A basic user division would be to give each member of the house, that is capable of using the smart home functionality, a personalized user account. When fiddling with the smart home setting the user would login and could access the features that are allowed to him. One extra user account would be available for guests and anyone who wants to do basic operations that require no privileges, like turning off the lights.

Some research, such as the paper by Taylor and Swan [2005], suggest that explicitly defined privileges are not needed in a home environment. Thus there would not be any need for passwords or user names. The concept might sound strange but the argument is quite valid: Taylor and Swan say that location already defines the access rules for different objects. A personal diary provides a nice analogy, as it's most certainly stored in a safe place such as a locked drawer or beside the bed. Therefore its location conveys the privacy rules. The principle has been working so far in the physical world so why wouldn't it work in the digital world?

My argument is that location is not enough for digital data. When comparing the differences between a conventional paper diary and the contents of a personal digital document, the first obvious difference is the ease of creating millions of copies from a digital document compared to the difficulty of creating even one near perfect copy of the conventional one. Thus the consequences of a digital document falling into the wrong hands might be much more severe. This and the highly prioritized requirement for personal privacy makes it sensible to protect user data with passwords. Of course in the name of customizability and user control, passwords should be made optional to let the users choose their preferred method.

Location should however be considered when deciding what different user interfaces are allowed to do. As mentioned earlier the in-house user interfaces can be allowed more power since they are already protected by the physical space. In comparison all external venues to control or monitor the house should always be behind authentication and the possibility to shutdown outside access completely should be available.

When designing a mobile and dynamic smart home system it becomes evident that the types of user interface devices can't be known beforehand. In a centralized system one device could be named as the main user interface and additional user interfaces could be integrated by coupling them with the main user interface and approving them as a part of the network. For instance one could join a mobile phone to the network by connecting it to the main user interface and then accepting it through the main interface.

This approach of accepting separate entities is already familiar to many PC users from firewall software that asks is local applications are allowed to connect to the network.

The abilities of the other user interfaces should also be configured from the main user interface. For instance the touch-screen in the kitchen could be configured to get Internet access and allow the setting of lights and music in the kitchen. A centralized approach would make the system more secure by limiting the damage that external user-interfaces can cause if they end in the hands of wrong people. The main user interface itself should be static.

6.8 Dynamic Interfaces

One big challenge in the development of the various user interfaces is the ever changing environment of the smart home. What happens when the user adds a new lamp to the living room? What if an unknown, new kind of device becomes part of the smart home network?

Solving these issues requires that user interfaces don't assume much about the environment. Instead of only the user interfaces saying what they can do, the environment tells its capabilities to the interfaces. This means that the responsibility of presenting features falls on the devices themselves. User interfaces in turn have the responsibility to present these features in a logical way. This way once a new device is added into the network, the system notifies user interfaces and other components that are interested in devices of that type. The device then shows up on the affected interfaces as a seamless part of the environment.

One way of reaching this is to use a definition database of devices that dictates what these devices can do and how they do it. In other words the database contains machine-readable use instructions and descriptions of each smart home device. This kind of approach was used for example in the Aladdin project where the *Attribute-Based Lookup Service* was used to create a shared understanding of the capabilities of each device. Once the right device for the job is found the *Name-Based Lookup Service* can be used to find the actual location and address of the device.

However, in order to automatically show the new device to the user interfaces it becomes necessary to know who is permitted to see the device. Of course the user could one-by-one define the permissions for new devices, but this could prove to be very cumbersome. Some kind of categorization of devices is therefore recommendable. Obvious categories for devices are device types, such as screens, window blinds, kitchen appliances, washing machines and A/V devices. Another important way to categorize them is by location. This makes the already mentioned feature of setting permissions by location possible.

Since the user interfaces need to be able to construct a sensible view into the smart home functionality, they need to be able to understand the information coming from a lookup service. Categories ease the task considerably. Already with the location and

type category we can construct a rather sensible user interface that is able to organize its functionalities into sensible bits. Hardcoding some basic type categories is needed so that the user interfaces may provide more powerful means of interaction. For example hardcoding the type category of “lamps” makes it possible to build a more powerful interface for lighting control since the user interface can assume that we are talking about lamps and not ovens.

These two ways of categorization are by far the minimum that should be expected from a device description. However, more data about the device would be better. The users should also be allowed to categorize them as well. For example through subcategories or by more flexible means like tagging.

6.9 The Placement of User Interfaces

One big design issue that has to be thought of is the placement of each user interface. As Koskela and Väänänen-Vainio-Mattila [2004] pointed out, the actions that the inhabitants carry out during the day are mostly centered around specific points. In addition these action centers serve different purposes. For instance the couch in the living room might serve as an action center where the family gathers to watch TV or play games. In a much wider study Crabtree and Rodden [2004] discovered the same result. They did an ethnographic study that involved the following of 22 families across England. The survey lasted for two years, during which time monitoring equipment and interviews were used to gain insight into the lives of the family members.

Drawing from their results Crabtree and Rodden formed three different concepts to describe the communications patterns and routines that the inhabitants follow. *Ecological Habitats* are places where communications media can be found. They are plain visible sites where the residents of the house go and find the media they need. For instance a corner in which the family's computer is placed might be seen as an ecological habitat. In the same way a postal box may be seen as an ecological habitat that supplies postal, paper-based media.

The concept of *Activity Centers* is used to describe the actual places where the gathered media is consumed and used. These can often be different places from the Ecological Habitats. For instance the mail might be collected from the postal box on the front porch, but the actual reading of the mail could usually occur inside by the kitchen table. This concept of activity centers by Rodden and Crabtree is quite close to the similarly named concept from by Koskela and Väänänen-Vainio-Mattila [2004], but the latter focuses more on the actual acts of doing something, like watching television, while the former focuses on the usage and consumption of media items. Both researches do however support one another and the concept of activity centers should be taken well into account when designing.

The third concept introduced by Crabtree and Rodden [2004] was the idea of *Coordinate Displays* to describe the places which are used to coordinate the actions

related to the usage of media items. For instance a kitchen table might be a coordinate display for mail items where the most important letters are placed in a distinct corner and the bills that should be paid immediately are placed in the centre. This importance of spatial information that is used in coordination was also present in the aforementioned study of Taylor and Swan [2005] with an emphasis on the personalized and artful way in which the different locations are used for inaudible coordination.

The three location types often overlap. For instance a notice board might contain important notices and therefore act as an ecological habitat. The notices might be read right there making the site an activity centre as well. Lastly the board might be ordered so that certain areas contain more important notices and the inhabitants might move the notices around to make sure that a certain family member will find them and act on them. This would effectively make the location also a coordinate display.

Places where all the three location types overlap should be considered when thinking about where to place the digital user interfaces in the house [Taylor and Swan, 2005]. Also the way that these locations are used in the everyday coordination of activities should be taken into account. The most important thing is however to remember that no two families have the same routines and the placement of the devices has to be adjustable by the residents.

6.10 3rd Party Development

In this Chapter I've explained the basic characteristics of smart home user interfaces. Without a doubt user interfaces are one of the key components in a smart home network. However, it's worth noticing that the user interfaces are not a part of the core of the smart home.

Following the aforementioned principles of immediate and remote design, it should be noted that user interfaces are mainly a part of the former, while core development belongs to the latter. This means that one of the first things many techie people will probably want to do is build a new user interface to control their houses. User interfaces are also the visible part of the smart home system. They are the part which can be decorated to match with the rest of the household.

The issue with decoration is key in understanding that the default user interfaces deployed with the core should be as easily configurable as possible to serve the needs of the basic users. Another point is that user interface development should be as simple as possible to allow the development of 3rd party user interfaces. In a good system the security issues and connections to other devices are abstracted and the user interface developers can focus on the interface itself.

However, the developers of the first user interface can't assume that the inhabitants are capable or willing to configure their system and therefore should make sure that the default user interfaces shipped with the system are as usable and accessible as possible.

7 Smart Home Components and Architecture

Many of the aforementioned smart home challenges and user requirements suggest various characteristics that are required from the smart home environment. Firstly the uniqueness of homes and the need for customizability suggests that the architecture has to be flexible and modular so that new parts can be added freely.

Secondly – as mentioned earlier – smart home systems can't be thought of as normal computers or machines that can be turned off and on when needed. Instead a smart home system has to effectively be on all the time. This puts great demands on the uptime of the system. However no system is foolproof and therefore the smart home system should be prepared to recover from occasional crashes independently. Hopefully without the user noticing anything. This means that the system needs to be stable and, at least partly, autonomous.

A certain kind of intelligence is also expected of a smart home system. Even though various research projects, like Morphome [Mäyrä et al., 2005] suggest that full proactivity should be something to steer clear from, the need to collect context information and respond to the various different usage scenarios around the home already call for some amount of intelligence on part of the system.

Looking at these three broad requirements from the system it becomes quite evident that software agents could the optimal solution from which to move on. According to Franklin and Graesser [1996] a software agent is:

> "*a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future.*"

When we add the normal characteristic of cooperation – that is often associated with agents – to the definition, we see that the nature of software agents work well in smart home environments: autonomy, intelligence, awareness and cooperation are good qualities for an intelligent entity in a smart home. In addition to this the modularity that agent-based planning enforces makes it easier to design and describe such systems.

The applicability of agent technology for smart homes is also visible from the more wider concept of ubiquitous computing. As explained by various ubicomp researchers, such as Abowd and Mynatt [2000], the computing parts of a smart home system are entities that work towards a common goal. The features of the system are realized in the cooperation of the various components, not by any one component. These components are called agents to emphasize their independent nature.

Using agents in smart homes is by no means a new idea. Various projects such as MavHome (Managing an Intelligent Versatile Home) [Cook et al., 2003] from the University of Texas in Arlington or the UMASS Intelligent Home Project [Lesser et al.,

1999] have used agents. Both systems utilize a hierarchy of agents. The focus of the MavHome project was on the artificial intelligence side, while the UMASS project choose agents as the base on which to develop a resource coordination protocol.

The AMIGO project has already provided a software design based on their research on user requirements [Georgantas, 2005]. This design is done with a service-oriented methodology, is fairly complex and goes into close detail. The project aims to bring four different domains together: consumer electronics, mobile computing, personal computers and home automation. This brings a huge overhead to design as the team has to please the players on all the four teams. The AMIGO project is mainly geared towards creating a networked home in which different entities from all the four domains can work together.

The problem with the AMIGO design is the assumption that a common ground can be found for the different parties and that this will fuel the development of cooperating appliances that people would like to purchase for their homes. Even though the design is based on the requirement research done earlier, a faint notion of technological push is evident. As a firm believer in the piecemeal approval and user participatory co-design of smart homes, I will start the design from a clean plate, focusing on a working, flexible and robust core that enables user driven immediate design.

I'll start this Chapter by outlining a few critical components that are needed from a core system. A big part is dedicated to the concept of context and how it should be handled in a smart home. After context I'll move on to logging, agents and then some final discussion on the subject.

7.1 Central Components

In their paper on context aware homes Meyer and Rakotonirainy [2003] propose three critical components for a smart home system. The first component is a hardware abstraction layer that abstracts the communication between the underlying devices and the software components. The second minimum component is a context manager which gets data from the hardware abstraction layer and organizes that data into context information that can be easily used by other components that require context awareness. The last component Meyer and Rakotonirainy see needed is a privacy manager that makes sure that only the minimum amount of needed information leaves the privacy domain.

The first two components seem like a good foundation for a multi-agent system: hardware abstraction eases the addition and control of different devices and is a basic part of any modern operating system, and the context manager eases the programming of the other software components and makes them more suitable for a general purpose use. The need for a separate privacy manager however seems a bit redundant: clear permissions on what a component or a user is allowed to access can be realized on a per-component-basis, with a coordinating agent or simply with user-group-permissions,

like the ones in UNIX-like operating systems. However the requirement for the privacy agent may also be considered as a need for the core to make sure that it does not give any more information to the agents of the system than what they are allowed to get.

The flow of data between the layers can be described like this: the hardware abstraction layer provides sensor data for the context manager which in turn notifies the software agents of the changes that they are interested in. For example an agent responsible for the front door might be interested if the context manager gets information about the motion sensor spotting movement there. After getting the information the front door agent could directly tap into the data stream that the front door camera is giving and process it in the preferred way. This direct feed could also be used by a user interface agent to display the video footage to the user. The communication can also go the other way. For example if the front door agent identifies the user standing in front of the door, it can tell the context manager to update it's information.

The context manager should also be aware of the situation of the user and her capabilities. For example if the occupant is hearing impaired, the manager should realize that no audio should be used for notification, or if the user is discussing with his friends in the living room and would not like to be disturbed, the system could resort to non-intrusive ways to notify him, such as dimming the lights a bit.

The requirements about indoor follow-me content can also be satisfied with the context manager. This way agents in charge of different follow-me contents can simply specify a target user. The context manager will then make sure that the output is directed to a appropriate device near the user. The agents should naturally also be able to target certain spaces inside the house. For example during a party the occupant probably won't want the music to follow him in to the bathroom while the guests in the living room are left with silence.

The downside in this design is the complexity of the context manager. However the difficulty of the whole context management issue suggests that it should be better to have one component handle context and provide easy-to-use information to the other components. The abstraction of the context makes the development of other components a lot simpler. For example the possibility for an audio agent to target a user instead of first finding out where that user is and then following her movements, makes the implementation of the audio agent a lot more easier. Figure 1 below shows a possible example of an architecture.

In comparison to the three main components, the Aladdin project has built a smart home system that focuses on the issues of smart home networking [Wang et al., 2000]. Their system revolves around a *soft-state store* that keeps track of all the devices and sensors in the house. The soft-state store waits for periodical updates from all the attached devices. If no update has been given in a specific interval the soft-state store presumes that the device is no longer connected to the network.

In the dynamic smart home environment the usage of the soft-state store has a few advantages over a “hard-state” store where the devices would be accountable to send their data when joining the network and logging out when they are removed. Firstly the software is more robust when the devices of the house are unreliable and don't often get to send their logout messages and secondly the system survives accidental removing of devices when, for example, the inhabitant trips on a power cord and the media terminal shuts down.

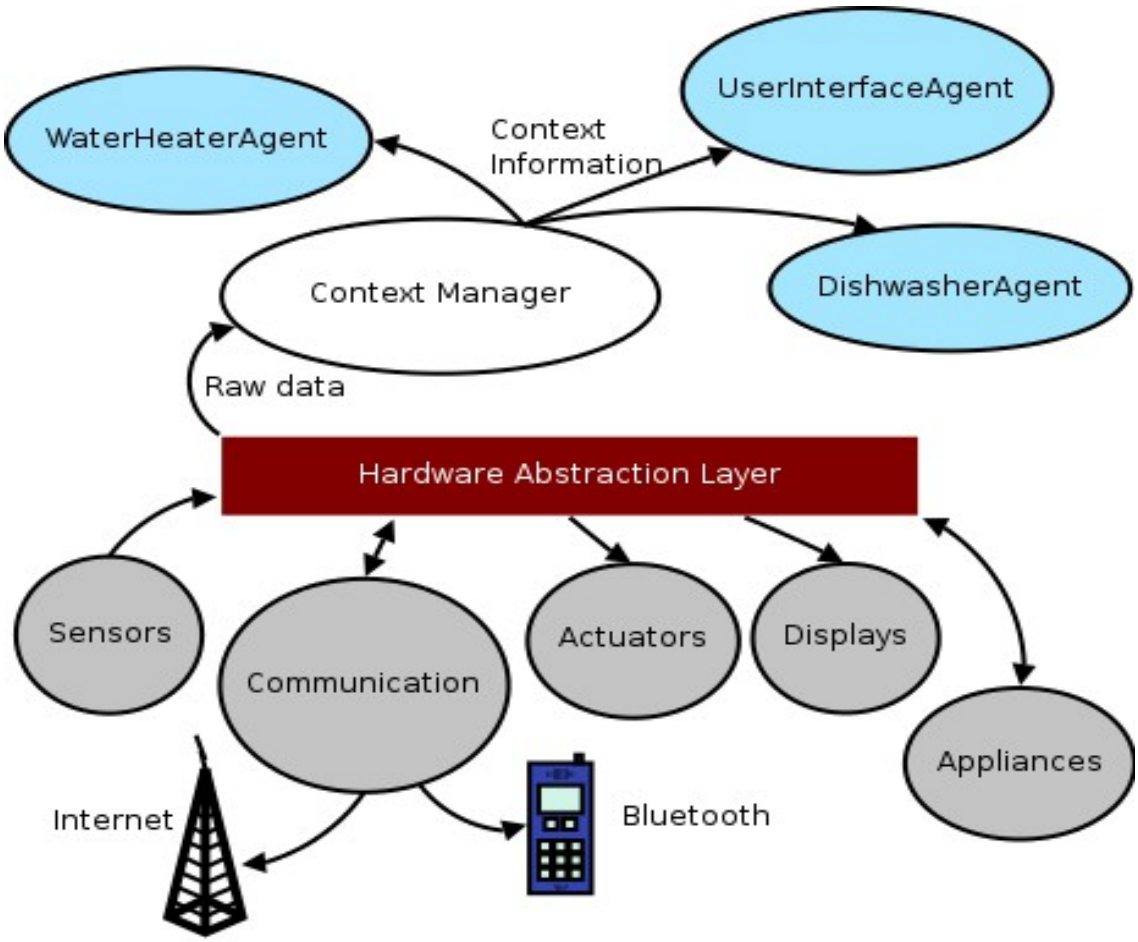

Figure 1: One possible architecture solution for a smart home system

7.2 What is Context

A soft-state store provides a nice approach into implementing the context manager, since it is relatively simple in its function and quite reliable in an unreliable environment like the home. The big question is however, what is context? What should the context manager keep track of without growing into a huge god-component of the system?

In the Aladdin project [Wang et al., 2000] one of the soft-state store's key responsibilities was to keep record of all the devices in the network. It had two different components for this. The Attribute-Based Lookup Service provided information about

the devices and agents in the system and what they could do. The Name-Based Lookup Service in turn stored the addresses of those components so that they could be contacted properly.

The five Ws introduced by Abowd and Mynatt [2000] produce a good reference when considering the definition of context gathering in a smart home environment. It must be emphasized that Abowd and Mynatt consider these the minimal set of necessary context. The five Ws are: *who, what, where, when* and *why*.

Who points to the subject as well as other people in the environment. Abowd and Mynatt add that current systems focus too much on one central identity in the environment. This usually being the person committing an act of some interest. As humans we plan our activities and recall them from the past based on the presence of other people. This should be noted also by computing systems.

What explains the purpose of the user. Abowd and Mynatt admit that perceiving and interpreting human action is a complicated matter. This is however a requirement from context-driven devices that need to know what the human is trying to achieve so that they may offer the user related, useful information.

Where simply answers to the question of the location. According to Abowd and Mynatt this area of context has enjoyed the most attention of the five and has been explored more. Another interest is to couple where with other context information such as when.

When is about the temporal context. Abowd and Mynatt argue that with the exception of indexing captured records with time or counting how long a person has stayed at one point, time is one of the most overlooked areas in context-driven systems. One area of interest is understanding relative changes in time so that human activity could be better understood. With the usage of time certain patterns or routines can be discovered. Especially in a home environment understanding the routines is very important. One application area mentioned by Abowd and Mynatt is to detect if an elderly person deviates from his morning routines.

Why is the hardest of the context areas. It's far more difficult to see why something is being done than what is being done. Abowd and Mynatt suggest the usage of other sensors such as a person's body temperature, heart rate and galvanic skin response to try to find out the person's affective state.

Extracting all the five Ws from the context of a smart home will definitely be quite difficult and it's possible that at least the first smart homes won't necessarily support full context understanding. However, considering the importance of context in a smart home system, it's quite crucial that a great deal of time and resources is devoted into the development of context understanding.

7.2.1 Quality of Context

One of the central issues with current sensors and sensing software is that the results are

never 100% accurate. For instance a face recognition software won't produce accurate results all the time. Especially if the environment, such as lighting conditions, change from time to time. Quality of context, or QoC for short, is one way of handling these issues in a more reliable way.

QoC means using a certain metric to determine how dependable the information is. In a research on context-aware systems Huebscher and McCann [2004] propose a middleware design for a context providing service. In their system one of the main themes is the quality of context that is provided by the components that generate context information.

Huebscher's and McCann's QoC has five attributes: precision, probability of correctness, resolution, up-to-dateness and refresh rate. They also add that other attributes may be available for different areas of context, but these are variables that are present in all context related information. Also the QoC in their design is dynamic, meaning that for instance a video camera may degrade its QoC if the lights are turned off in the room.

Precision measures how well the provided context information relates to the real world. With location context the precision means how accurate the placement of the information is. For example the context providers might know that there is a 3cm error possibility in measuring that the user is in front of the window.

A high probability of correctness means that it can be assumed that the context information is correct. Huebscher and McCann give the example of a video camera that tries to detect the posture of the user. The probability of correctness with the camera is different than the same attribute with pressure sensors integrated to furniture, that can be quite certain to know when the inhabitant is sitting on top of them.

Resolution tells about the granularity of the information. If the resolution is low the context information may be more of an approximation of the current state. With high resolution more exact information is available from the sensor input. A low resolution audio input might only know that there is sound while a more high resolution device could also determine the direction and position of the sound source.

The two last ones are quite straightforward: up-to-dateness is simply the age of the information and how likely it is that this information is still valid, and refresh rate is a related attribute that tells how quickly we can or should ask for a new measurement from the context provider.

In their middleware Huebscher and McCann divide the context creation and distribution to three parts. *Sensors* are on the lowest level providing their data to the upper layer of *context providers*. These in turn feed the context to *context services* that handle the interaction with actual applications that are interested in the context. Their work is largely based on the Context Toolkit by Dey and Abowd, but provides a more flexible approach since the entities on the levels are not tied to just one component on the next. This means that multiple sensors may send their data to multiple different

context providers, but the providers can only feed to one context service. These services may, and usually do, get context from various providers.

Another central part of the design is the *adaptation engine*, which is responsible for controlling the context services to pick the best context provider based on the requirements it gets from applications. The best suitable context provider is chosen with a function that determines how "satisfied" the application is with a certain context providers QoC.

Huebscher and McCann's design is intriguing, but has a few shortcomings. The first problem is that the QoC values are predefined by the context providers. This creates the problem of trust and is also noted by the researchers themselves, who propose a new QoC attribute of *trustworthiness*. This attribute would be measured by *complaints* and *praises,* that are given by the applications that use the context information. For instance if an application can determine via another context that the information it got from the first provider is false, then it can complain about that provider whose trustworthiness will degrade as a result. Another way to determine the correctness would be explicit user feedback.

Although a possible solution to the trust issue, the new QoC attribute complicates largely the development of new applications as they need to be able to conduct if their context is correct or not. The whole middleware also makes it harder to include new context providers as they need to be able to provide QoC. Finally the trustworthiness attribute does not totally take away the problem that other QoC attributes are still solely defined by the providers themselves and not by a trusted third parties.

Another question that raises is why bother with the difficulty of picking the best QoC match for each application? Why can't we just pick the best QoC and stick to that context provider? The difficulty is that when we are picking *between* context providers we end up with the problem of determining the best one for the job. There might be no clear winner as the some might be stronger with some attributes and weaker in others. There is however a way to get rid of the adaptation engine and this will be covered next.

7.2.2 Context Fusion

Instead of always picking the best context provider for the current need, we could join all the context sources and determine the context attributes through that. For instance if the video camera knows with a 95% probability of correctness that someone is in the living room and the Bluetooth monitoring agent knows that George's mobile phone is in the living room and nobody else's, then we can presume that the person in the living room is George. Collecting all this context information and joining it to determine the final context attributes is called context fusion and has been already proposed before to help with context issues in ubicomp systems [Abowd and Mynatt, 2000].

Of course fusing the bits of information means that there has to be a central entity that gets all the related data. Of course these calculations may be shared among

different devices with concepts like grid computing, but still effectively one agent should be in charge of the overall context management. An arguing intelligent environment with multiple context managing agents is not sensible to build.

Context fusion makes the development of external sensors and context providers much easier since they don't necessarily need to know their QoC and development of context-using applications becomes easier when they don't need to blame or praise the received data. The complexity is moved to the central context service. It needs to be able to learn the QoCs of different context providers and how to combine them in various different scenarios.

In an ideal situation the context service wouldn't have any assumptions on the QoC values of different devices. It would – for example – handle the Bluetooth-based context provider and the motion-sensing context provider on the same level of trustworthiness. The difficulty comes from building an algorithm that is capable of saying that one of them is wrong or right so that their QoC may be adjusted accordingly. One way of determining that one provider is wrong is to spot the situations where it totally disagrees with all the other providers. However, the only almost fool-proof way is to explicitly ask the user or provide a way for the user to complain to the system that the context was misinterpreted. For example if George notices that the follow-me content is not doing its job correctly he can complain to the system and this will make the context service change its QoC values for the responsible sensors.

Also worth noticing is that usually all sensors and context providers are good in only interpreting a certain type of context. For example the pressure sensors on the couch know about the "where" and something about the "what" and "who" of context, but perhaps can't give information on "when" and "why". An entity doing context fusion can however determine the "when" from the system time and try to understand the "why" by observing the other context information available. For example if the user sits on the couch and the television opposite of her is on, the system may determine that the user is watching television and thus – at least partially – answering the question "why".

Context usage and interpretation is one of the most difficult and most important parts of a smart home software system. Because of the difficulty of handling context, it should be left to one specialized, central entity. In addition to the difficulty in interpretation the system should also be able to work together with the rest of the network. Huebscher and McCann's[2004] abstraction of context providers to handle different sensors would be helpful here since the context service would only need to work with them. Defining a flexible method of communication is still a task for the smart home developer.

7.2.3 Environment Modeling

One big areas under context management is the understanding the system has about the

environment. In the case of a smart home this means the layout of rooms and devices in the house. Modeling the environment affects the cooperation of the various agents in the house as many of the devices are dependable on the devices near to them. For example when the user wants to view a DVD in the living room and the room should be made darker, the system needs to know where the corresponding display device is, and which curtains and lights to dim.

Many conventional smart home projects – such as EasyLiving and MavHome – have relied on having a predefined environment with static device locations. This would not be the case with a real life smart home system and thus a method to model the environment should be conceived. The system should also be reprogrammable since the environment changes every time the people move to another building or just do changes to their space plan or interior layout. For instance if the inhabitants wish to move their television, which is not capable of reporting its location, they have to inform the system that the location of the device has changed. Thus unless the operation is completely automated, it has to be as user friendly as possible.

Usually the first level of modeling the environment is the space layout as defined by the six Ss of building structure [Rodden and Benford, 2003]. In a big single family house the modeling could start from the site-level as the inhabitants would describe the locations of their external buildings such as garages and sheds. This means that in order for the environment model in a smart home system to fulfill all these roles, it should be freely scalable.

Collecting the information for the environment model can be done through various means. In a smart home project with a bigger budget one could envision the usage of wireless sensors spread around the corners of the house, or the usage of one wireless device that the user would carry around the boundaries of the house to create a model for the system. Also the usage of robotic devices to build a model of the site and building is not impossible. For instance the integration of a vacuum cleaning robot like Roomba (http://www.irobot.com) would allow for the smart home system to construct a model of the interior of the building.

In the end the system must be able to survive without the usage of these external devices. The user should be able to model the environment manually through an intuitive user interface. When designing the interface for such an application it's wise to look at the current defacto way of defining environments, such as the usage of CAD programs. Through this user interface the occupant should be able to draw simple models of rooms and windows in the house and add locations of devices, such as lamps, display screens and audio equipment. Essential for the good interaction between the core system and the devices is that the devices are first registered with the core and then dropped to their actual locations through the user interface. This ensures the proper coupling of devices and their respective locations.

7.3 Logging

Logging is the system's way of remembering what has occurred earlier. Already quite important with more complex software systems of today, logging is such a key issue in an intelligent environment, where actions are conducted between multiple independent agents, that it deserves a section of its own.

A central logging agent that would be able to gather all the information from all the devices would allow for the best log viewing experience since all the information would be in one place. This approach has however multiple negative sides. Firstly collecting log information from all the pervasive devices in the network would require a lot of transmissions that are functional-wise unnecessary. This would also lead to a more complicated software architecture since a common way of sharing log information would have to be found.

Another reason is the problem for security. A central logging entity would have access to all the logs. If this one component is compromised then all logs can end up in the hands of the wrong people. For example a hacker that gets access to a server's log files may use those files to determine the software and network setup, and find a weak spot he can exploit to get into the system.

Finally having one central entity to do the tasks of others goes against the agent philosophy, especially the independence of the agents. A better approach is to make all devices responsible for their own logging. The ones that are able to see what they have been doing can present a readable attribute that returns the logs from that device. This attribute can then be represented through user interfaces that have the permission to read it.

7.4 The Agents

As explained earlier, agents provide a nice way to abstract the components in the system. The core services such as the context manager and the hardware abstraction layer provide a good foundation for the agents live in. An autonomous agent is hardly useful by itself. In order to cooperate with others the agent must have a way to find them. For this some kind of agent directory is needed. This kind of directory service could then contain a attribute-based lookup service and a name-based lookup service for finding right agents and their locations as suggested by the Aladdin project [Wang et al., 2000]. As information about the availability of agents is context information, I propose that the lookup services are integrated as a part of the more general context service.

Now that the agents can find each other, a common way of cooperation has to be found. The resource management framework introduced in the UMASS Intelligent Home Project could provide a good start for cooperating agents. The UMASS home is controlled by a horizontally organized group of agents that are capable of communicating and agreeing on resource usage. The framework still has some rough corners, but the principle has been shown to work [Lesser et al., 1999]. For example if

the dishwasher agent would like to do it's chores, it has to ask the agent in charge of hot water to provide it some. If the occupant is taking a shower at the same time, the hot water agent may decide to deny the request and then the dishwasher will try again later. Another excellent idea in the UMASS project was the use of noise as a resource. It provides a nice way to regulate the disturbance caused by all the household appliances.

Using similar resource management tactics as proposed by the UMASS project, the agents are capable of saving energy, cutting cost and acting more efficiently. Thus meeting the requirements of the third level in the AMIGO project. Most resources like hot water and noise are present in all common households today. The development of these agents is quite central to the functionality of the rest of the household and thus should be planned by a central entity.

Implementing this kind of resource coordination is quite difficult as it requires that the distribution of the resources can be done through software and therefore this kind of resource coordination should not be forced. Managing the usage of hot water for example requires the installation of computer controlled devices to the water pipelines. When designing a mobile smart home system that can fit into any environment without much work, resource coordination should not be a strict requirement, but the system should be able to support one if needed.

7.4.1 Coordination and Control

User control was on the first level of requirements. The requirement decreases the need for strong proactivity. Instead of having a very proactive system – like MavHome [Cook et al., 2003] – that does most of the chores for the users, the weaker proactive system only suggests different tactics and acts without user intervention only when the user has approved such actions. Following the aforementioned principles of weak proactivity are more appropriate for the smart home environment [Mäyrä et al., 2005].

Exceptions to the rule of weak proactivity are situations that can be interpreted as hazards. For example a fire detector could inform the fire department automatically when it discovers a fire, or the shower could be shut down if it is running and nobody is near the bathroom. Even these situations must be possible to be overdriven by the user. For example a malfunctioning fire alert will cause the inhabitants a lot of grief if they can't turn off the automatic alerting feature.

To ensure that the system stays in line and agents don't go around doing their chores without user intervention, one agent is needed to oversee the others. This agent is the coordination agent. All other agents require a permission from the coordination agent to do their chores. Small tasks mostly including communication between agents, such as resource coordination, can be handled without intervention from the coordination agent. In addition to controlling the other agents, the coordination agent could also be responsible for the proactivity of the system by detecting usage patterns. If the agent notices that the occupant likes to close the curtains when she starts to watch the

television, it follows the principles of weak proactivity and asks the user next time she turns on the TV if she wants the curtains to be closed automatically.

Figure 2 below depicts agent communication in a situation where the washing machine has just completed its task. This sends an event to the context service which sees that the coordination agent was interested in this and thus the event continues to the coordination agent. The coordination agent checks which occupant was interested in this and seeks the best interface to communicate with the her. After finding the best route, the appropriate user interface agent is instructed to notify about the event. Depending on the location of the occupant, the user interface chosen could be graphical, audio or even a mobile telephone

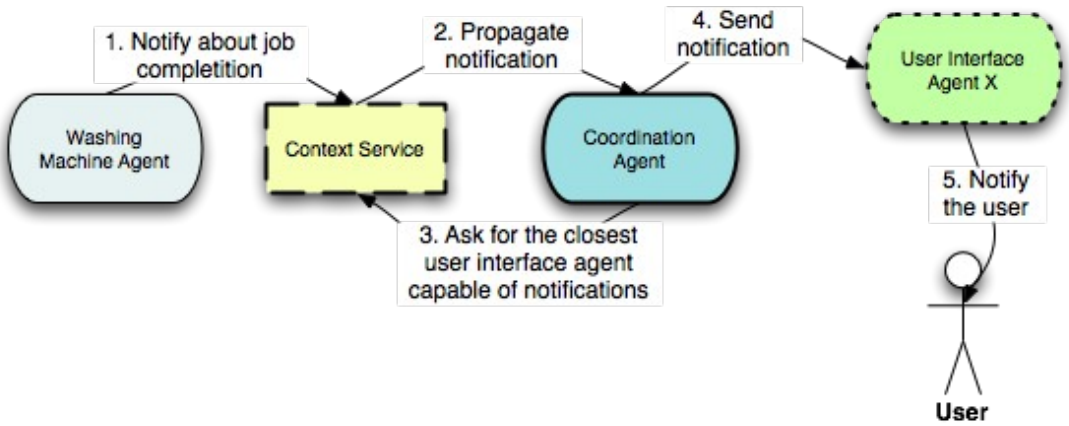

Figure 2: Example of agent communication

7.4.2 The User Interface Agent

One smart home contains multiple user interface agents. These agents can usually reside on the devices they are using for interaction and can be reached via a network connection. They monitor the context manager for changes that interest them and read their own sensors for input. The agent inside the touch screen in the kitchen could for example receive a notification that someone has entered the room and start listening for input on the screen or through an embedded microphone. Another UI agent could be operated with a remote through the television in the living room.

Multiple user interfaces scattered around the house and designed to be used by everyone, present a challenge to privacy: using information from the context manager and its sensors, the user interface agent may try to guess who the user communicating with it is, but the information might not be totally correct. Trying to automatically display a personal user interface to the person in front of the display might be too risky. One false guess might result in a total lack of trust in the system.

Because privacy was such a heavy requirement, the system should provide a solid and secure way to recognize someone through the user interface agents. The default way could be normal, traditional passwords, but the system could also use more

advanced methods such as fingerprint recognition. A basic user interface should be available to at least anyone inside the household, but more critical commands require secure identification and permissions to carry them out. Different user interface agents may have different ways to provide authentication. For example a user interface agent in a mobile phone may approve a simple PIN-based authentication, while the agent inside the occupants laptop might rely on a stronger pass phrase.

Since we are talking about an agent-based architecture where one of the key requirements is that new agents can be added dynamically to the the network, it becomes obvious that there needs to a method that can ensure the trustworthiness of the agents. In effect this means that there are two levels of authentication: user authentication and agent authentication. The first one occurs between the user and the user interface and the second is between the agent and the smart home core that supplies the central services such as agent directories and context.

Having the coordination agent as a kind of a local gate keeper inside the house helps security, because the coordination agent can keep track of the approved user interfaces and other agents, partly preventing malicious agents from getting access to the system. All wireless sensitive information should be encrypted and a procedure has to be invented to prevent malicious third party agents from acting as a coordination agent and commanding the others. One possible way could be that agents only rely on the coordination agent that has been specified on the context manager. This way the context manager could act as a kind of a trusted third party between the agents.

An example of user interaction with the system can be seen in Figure 3 where the occupant contacts the smart home system with a mobile phone through GPRS. Because the connection comes from outside the house, the user interface agent in the phone requires authentication for all actions. In this scenario the occupant wishes to start the coffee machine. After the authentication, the user interface agent contacts the coordination agent that asks the coffee maker agent to start the machine. The coffee machine agent needs water and asks the water agent for the resource. The water agent informs that it can provide water and the coffee maker agent starts the process. After the coffee maker agent informs that it has started the action, feedback is given to the user.

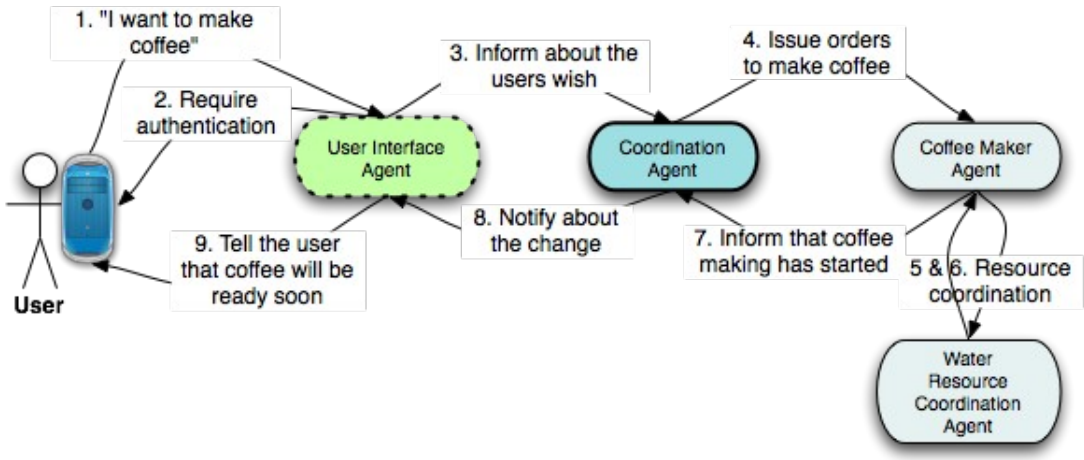

Figure 3: Communications involved in coffee making

7.4.3 Examples of Useful Agents

Here I'll present a couple of ideas about different agents, whose existence would be justified by looking at the requirements. These agents are mostly helpful features that were explicitly mentioned in the requirements studies. It can be argued that most of the user involved development and immediate design will happen on the borders of the system as understanding the core functionalities such as context management and agent lookup is much more difficult than building agents that make use of these provided abstractions. In that case the development of these agents may be left to third party developers, but at least some rough examples could be included in the default system as well.

The first example is meant to help with the notorious information burden. The smart home system could contain an information retrieval agent that collects important information relevant to the occupants. The information retrieval agent may also provide general information such as weather forecasts and traffic information to everyone and then seek more personal information such as emails, calendar events and selected RSS-feeds when the user has logged in. This of course means that the information retrieval agent has to be able to save the preferred information sources of all the users. By providing external information such as weather forecasts the information retrieval agent would also extend the context information offered through the core. For example the reminder agent next to the main entrance could tap into the context information provided by the retrieval agent and warn the user that because rain has been promised for today she should take her umbrella with her.

Third level requirements about easing household chores can be achieved with agents that coordinate the operation of robotic devices such as cleaning robots. Accident prevention is possible by having agents monitor the potentially dangerous devices. For example the agent in charge of the kitchen oven could receive notification

that the user has left the kitchen while the oven is empty and on. If she won't be back in five minutes the agent could notify the coordination agent, which would ask the user if the oven should be shut down. Having the oven on for fifteen minutes without anything inside, could be declared as a hazard and the oven agent would then be authorized to shut down the oven automatically.

Video conferencing is already a quite common technology. This could easily used in the smart home system. The output and input could be directed to the right room with the help from the context manager. Video could be used if the user so wishes and without it the system could default to audio.

Additional context providing agents should be easy to install as less intrusive ways of retrieving context-information. Normal passive motion detectors are an easy and cheap way to detect presence inside and outside the house. Many normal household items can also be modified to provide context-information: intelligent pot plants and all kinds of other items, such as the ones designed at Samsung labs [Park et al., 2003], have already been proposed.

7.5 The Last Issue of Communication

In this Chapter I've argued that an agent-based approach is the most appropriate beginning point for smart home design. Although an agent approach helps in dealing with increasing complexity, enabling dynamic interaction, allowing the development of third party components and understanding the agent nature of the home environment better, it also brings some problems. The central problem is the decision of the communication language between agents.

In order to simplify the development of extra components and the core itself the system should default to a very simple communications scheme. But because of the wide array of other possible agent communication methods, the system should allow communicating through other means as well. For example the UMASS project uses a communication protocol called SHARP for the resource coordination [Lesser et al., 1999]. This doesn't however meet all the demands for an ideal solution and it's highly unlikely that it'll meet the needs of the architecture presented in this paper. Using a standardized agent communication language such as FIPA-ACL or KQML could provide a robust solution, but the all-purpose nature of these languages could also bring some unnecessary overhead to the system. In addition to abstracting the actual communication venue such as Bluetooth, TCP and GPRS, the smart home system should also allow connections between agents that use different communication languages and ontologies without changes to the agents themselves. This is only possible via means of interpretation by a central entity. This is yet another argument for a centralized device in the household.

In order to facilitate cooperation between devices that don't know about each others existence beforehand, a common ontology needs to be agreed upon. Common device

categories like lights and audio outputs need to be defined so that the devices can find and recognize other agents and devices that they can cooperate with.

8 Artificial Intelligence

This Chapter is dedicated to the delicate and somewhat controversial matter of smart home intelligence. The problem of the matter arises from the question of whether we really need artificial intelligence as such in a smart home. Of course some might argue that an intelligent environment is not intelligent if we don't have an artificial intelligence running matters for the users. To counter this concept in this Chapter I'll use a different approach into understanding what constitutes smartness in a smart home system.

To further investigate smart home intelligence I'll introduce two fundamentally different approaches to smart home AI. Both approaches have the same aim of making the inhabitant's life easier, but they try to reach that goal through different means. I'll present these solutions as examples of how different the approaches are and lastly introduce an idea of a rather simple, but very powerful solution that might suit smart homes better.

8.1 MavHome and Active-LeZi

Managing An Intelligent Versatile Home is a project from the Arlington University in Texas, which was already introduced in Chapter 4. It focuses on the creation of an environment that acts like an intelligent agent. The AI studies the way the inhabitants live and tries to maximize their comfort and productivity by automating and predicting tasks in the house [Cook et al., 2003]. By creating a scheme for categorizing the actions of the users, the system finds patterns from their daily routines and then aims to automate them.

One of the main focuses of the project is the Active-LeZi algorithm that was designed to predict the occupant's actions and automate tasks accordingly. Two major requirements from the algorithm were speed so that prediction and actions would happen in real time without delay, and accuracy that would ensure the correctness of the actions and avoid the need to undo the decisions made by the household.

Active-LeZi is based on the LZ78 online prediction algorithm, which is mostly used in compression. For more information on the Active-LeZi algorithm or the LZ-family of compression algorithms I suggest the MavHome paper [Cook et al., 2003] or the wikipedia article at http://en.wikipedia.org/wiki/LZ78. The MavHome team state that the prediction accuracy of this approach is quite sufficient with an accuracy plateau of approximately 86% on a dataset of 2000 events gathered from a smart home scenario. The algorithm was also tested for 30 days in a real MavHome testing environment and it reached 100% accuracy on the real data. We can therefore safely assume that a compression algorithm works accurately in predicting the outcomes of household routines and can be used to assist the user. The bigger question is whether we should

employ one in a smart home system or not.

8.2 EasyLiving and the Geometric Model

Another approach to tackle the multiple difficulties related to intelligent environments is the aforementioned EasyLiving project from Microsoft. The clear focus of this project is to provide an environment in which multiple I/O-devices can cooperate. Unlike MavHome, which has a direct relation to AI-research and learning algorithms, Easyliving is more about finding a working architecture and a convenient communications method for all the cooperating devices and applications [Brumitt et al., 2000].

The actual system intelligence that tells the environment what to do and when is implemented as a hard-coded set of rules. This characteristic has a definitive negative impact on the learning capabilities of the system. The intelligent environment does not adapt to its user. Instead the occupants have to learn how their home works. While quite a common quality in desktop software, it would hinder the acceptance of home control software greatly.

EasyLiving is – consciously or not – emphasizing the enabling feature of the network itself instead of doing traditional AI. This was already mentioned to be a healthy direction in the new design ideas of Chapter 2.

8.3 Shortfalls of Both Approaches

Neither of the two approaches is perfect. They can't even be compared on a good scale, because they differ so greatly from each other: while MavHome can be seen as a venture in making a house that tries to learn from the user, EasyLiving is a system that is designed and programmed in advance and the user has to learn how it works. When we observe both in the light of the requirements, we see that neither fulfills the desires of the users.

The easier one to tackle is MavHome: with its no-questions-asked kind of methodology and direct involvement in the operation of the household it breaks the most important requirement: the need for the occupant to stay in control. One quick fix for this would be to introduce a user interface that asked the user if she would like to automate something. However introducing a simple yes-no user interface would not suffice, because the user should be able to modify the actions as she sees fit. The whole environment and the Active-LeZi algorithm has been geared towards predicting the user's next actions and doing them automatically.

In comparison, the Achilles' heel of the EasyLiving system is its inability to learn and adapt to the needs of the users, which makes it a lot harder to configure the system the way the inhabitants wants it. This goes against the requirement for the system to be customizable and puts the responsibility of operating the house back to the user. No help will be offered unless the user knows how to operate her house. It's worth noticing

that both systems cause the same problem: lack of customizability by the user. MavHome tries to do all the customizing itself and EasyLiving leaves customization to the original software designer.

In their paper Heider and Kirste [2005] also compare the two approaches. First they take on the EasyLiving approach of making the system designer responsible for the different strategies the environment uses. This approach is fundamentally flawed when we see the smart home as a dynamic assembly of different cooperating appliances. For instance when a user brings a new laptop to the household and wants to show his holiday pictures from it via the TV screen, the end result might be satisfactory if the designer had planned a feature that supports this kind of activity beforehand. However when things get more complicated than this (as they surely will), it becomes impossible for the developer to handle every scenario.

According to Heider and Kirste the MavHome approach of learning behavior patterns from the user becomes invalid as soon as we're faced with the notion of ubiquitous technology, meaning technology that can't really be seen, technology that is integrated to the environment. In an ubiquitous environment a substantial amount of the devices are invisible to the user and therefore it's basically impossible to monitor the usage patterns of those devices.

8.4 Goal-based Approach

As a solution to the issue of smart home artificial intelligence, Heider and Kirste [2005] propose a goal-based approach that lets the user define explicit goals, which are then analyzed and pursued by the system. This way the user does not need to know the internal structure of the system and the system does not need to wait for the user to do something before it can do it by itself.

This approach makes the system more usable since the user doesn't have to be aware of the way the system works. It also improves the occupant's sense of control which was seen as a very important requirement from a smart home environment. Heider and Kirste justify their approach based on findings in cognitive psychology which says that humans are accustomed to think of *tasks* as goals that need to be reached rather than a set of functions that need to be done in order to reach that goal.

Using their goal-based approach involves two basic steps between the input from the user and the actualization of the command in the environment. The first step is *intention analysis* and it deals with the understanding of the message. After that the second step of *strategy planning* takes this machine-interpreted target and tries to find the best way to reach that goal.

The first step is the one where most of the errors happen. The amount of errors depends on the manner of input. For example if the user uses a graphical user interface and a pointer device to tell the system about her goals, then the system can be pretty sure about the intentions of the user. On the other hand if the user uses gestures or voice

to explain her intentions, the chance of a misinterpretation by the system is quite large.

As an example of the process, the first step might involve the user saying *"open curtains"*. With this audio input the system first has to do an analysis on the data it receives and construct a meaningful representation of the sentence (phonology, morphology and syntax understanding needed). After this the system needs to find the semantic meaning of the utterance. This step needs reliable information about the context: the system needs to know where the user is looking at or which curtains are the closest ones so that the command will reach the intended target. Some reasoning has to happen on this level as well. For example if the room where the user is has two sets of curtains and the set closer to the user is already open, then we can deduct that the user meant the other set.

After a definitive goal has been found the control is moved to the second step of the process. During this step the system uses a planning algorithm to reach the explicit goal. In the example the opening of the curtains might just require a simple command to the actuator in charge of the mechanical operation, but more complex goals, like *"make room darker"*, might involve many steps and different approaches.

When combined with a learning or predicting algorithm like Active-LeZi from the MavHome project this kind of technique might reach the requirements for weak proactivity by being able to understand the actions of the user and predicting what she'll do next. As weak proactivity has a huge positive impact on the feeling of control that a user has [Mäyrä et al., 2005] and since user control is the key requirement in many studies, such as the AMIGO research, it becomes evident that following a weak proactive route is necessary.

8.5 Possible Technical Solutions

The first interpretation focused step of the goal-based approach naturally requires a user interface through which the occupant of the house may express her will. As discussed in Chapter 6 a smart home system needs various different user interfaces that are used in various different contexts and through various different modalities.

Thus the first step of gathering input and forming the understanding of it becomes the responsibility of the user interface in question and the exact techniques used depend heavily on the type of the interface. However, the second step of Heider and Kirste's approach is the more crucial one in the light of artificial intelligence research. The fundamental problem during this phase is the transformation of the occupant's goal to a set of primitive operations that realize it.

Heider and Kirste propose the use of a *partial-order planner* which may then take this desired goal and process it by using a set of possible operations that are all described as *precondition-effect* -rules. The preconditions state which conditions in the current environment must be true in order for the specific operation to work. The effect declares the end result if the operation is completed. The precondition-effect -rules must

come from the devices that are present in the current environment. Heider and Kirste call this database of possible operations the *environment state model*. This model can be understood as the entity that keeps track of the state of the house, in other words it stores context information. As the planner entity needs to integrate quite tightly to the context service it's quite safe to assume that it should be a part of the core system along with the other context-related parts discussed in the previous Chapter, such as the lookup services.

The actual planner takes the goal provided by the intention analysis step, checks the current state of the world and all the possible operations from the environment state model and then proceeds to find a set of operations. Heider and Kirste list the various planner systems they have tried for their application domain and conclude that the best alternative so far has been the Metric-FF system that is capable of passing the problem to other systems if it does not find a viable solution. This could imply that one appropriate solution could be a blackboard type of system that combines multiple different AI techniques such as a declarative planner and a neural network, and always uses the best one to reach the target.

Since all the operations are provided by devices added dynamically to the smart home ensemble, it becomes obvious that a standard must be found to define the different preconditions and effects they provide. In addition to this ontology, the way the qualities of the environment are stored must be standardized. For example a new bed room light must tell the system what it is and how powerful it is, what it needs for operation and what are the effects when it has been turned on.

One relatively easy way of achieving a planner that can handle precondition-effect-rules would to use a Prolog implementation of a partial-order-planner(POP). One example of a a Prolog POP implemented using the STRIPS (**St**anford **R**esearch **I**nstitute **P**roblem **S**olver) notation is available from http://www.cs.ubc.ca/spider/poole/ci/ci_code.html. For each precondition-effect-rule three different Prolog facts must be defined. A simple STRIPS example of the rule of dimming a light is here:

```
preconditions(dim_light( X ),[light_is_on( X ), light_is_not_dimmed( X )]).

achieves(dim_light( X ), [light_is_dimmed( X )]).

deletes( dim_light( X ), [light_is_not_dimmed( X )] ).
```

The first part of all the three facts defines the operation which the fact stands for. In this case “dim_light”. The second part contains a list of prerequisites or effects the command has. The precondition rule above can be read as follows: *“operation dim_light for lamp X requires that the lamp X is on and that lamp X is not dimmed”*. The achieves rule shows what additions happen to the environment if the operation is done. It could be read like this: *“if the operation dim_light for the lamp X is completed then the state that 'light X is dimmed' is valid”*. The deletes fact lists the states that are

no longer valid if the operation is carried out. In this case it would read: *"if the operation dim_light for the lamp X is deleted then the state that 'light X is not dimmed' is no longer valid"*.

In addition to the operations, the environment itself should be described using a suitable ontology. For instance the connections between rooms, the states of the doors between them, the windows of the house and the states which always hold should be defined.

A simple POP does a good job when faced with a set of precondition-effect -rules, but this might not be enough. Many situations have multiple different solutions that reach the desired goal. In these situations it becomes necessary to select the most optimal solution. Heider and Kirste propose a maximum quality function that calculates the advantage of a solution. While this approach would work in a predefined environment it is obvious that it would not work that well when transferred into a totally new environment or situation. Thus the maximum quality function approach creates the same problem that Heider and Kirste found in the EasyLiving solution. Furthermore quality is something that is in many cases perceived more as a personal preference than a calculable variable. For instance whether George likes to answer his video calls in the living room or the bedroom and whether he likes to use the PDA or the big TV-screen depends on his situation, who is calling and how he is feeling that day. A quality function for these kinds of complicated, personal situations is hard if not impossible to define.

Instead of calculating the quality beforehand, the system might let the user decide which approach would be best. When faced with multiple solutions to reach a specific goal the system could suggest one and then let the user switch to the next one if the previous solution was not satisfactory. After the user has selected the best approach the system would remember it the next the same user wants to reach the same goal in a similar situation. The solution that has been chosen the most would become the standard. This approach would reach the requirements in AMIGO-categories one (need for control) and four (user preferences should be saved) better. Of course it should be possible to choose another way to do things if the user changes his mind.

The most simple approach in making the system take into account the preferences would be to just count the solutions the user has chosen and propose the most popular solution each time the user does the same action. This is however not the best approach because the context of the request is not taken into account. For example if George is holding a party for a few of his friends and he wishes to show them photos he would probably like to use the big screen in the living room. But when George is alone he would perhaps want to view them quickly on his laptop or perhaps on his PDA. In cases like this the system must also take the state of the environment into account when doing decisions.

One way of taking the context into account when doing the decisions would be to

store the context information of every request and compare the different solutions to reach the goal in the light of the current context. This would however be quite complicated and require explicit knowledge on what variables are important to note in the context (is the state of the kitchen light important when the goal is to “play music”?).

Another approach to meet the requirement of a learning environment would be to introduce *“modes”* – which are already in use in LinuxMCE and Pluto systems – that define different states of the nearby environment. The occupants of the house could then define this modes to suit their tastes and the smart home system would store them for later use. George could for example define a “movie mode” by sitting on the living room couch, ordering the lights to dim down, the music to stop and the DVD-player and TV to turn on. Then he would declare this state to be stored as a mode and the next time he would like to watch movies he would go to the couch and issue the change to the “movie mode” through a user interface. This would make it possible for users to quickly switch from one state to another, to customize the behavior of their house and to stay in control during the whole procedure.

One problem with the POP-approach is evident when the effect-rules don't apply to the current situation. For example let's say the user wants to have breakfast and the effect rules for that goal define that the coffee maker will be on. This works as long as the coffee maker is operational. If this is not the case then the system state might become invalid as it thinks that the coffee maker is on when it's actually broken. This can be solved by integrating the precondition-effect-rules with the context and making at least part of the effects realize only through noticed changes in the environment instead of direct changes through POP. This way the context service would notice that the coffee maker is broken and the effect rule would not apply.

As mentioned previously, the planner agent is quite tightly integrated into the process of surveying the context and doing decisions. The latter quality brings the planner agent quite close to the coordination agent and therefore one rational point would be to combine the coordination and planning activities to one agent. A simple example is given below in Figure 4.

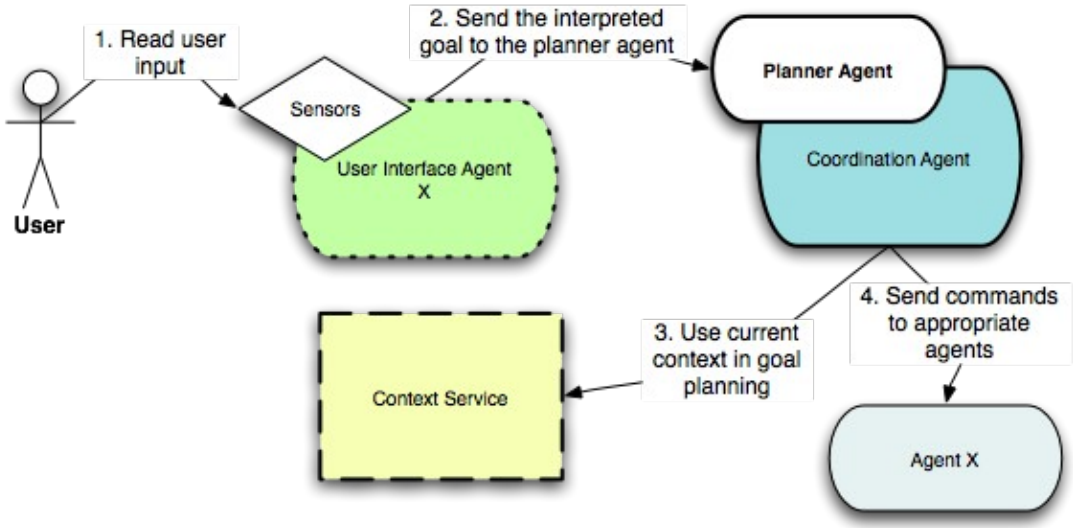

Figure 4: POP and planning integrated with the coordination agent

8.6 All Three Approaches Side-by-Side

To summarize a part of the results on the issue of artificial intelligence, I'll construct a simple table that matches the AMIGO smart home requirements to the different approaches. To keep it simple I'll use a evaluation scale of "+ +" to "- - " with "0" being the middle. A value of "+ +" means that the specific requirement was very well met in the smart home and a value of "- -" means that this requirement is a real problem to this approach.

Requirement Category	MavHome	EasyLiving	Goal-based approach
1. Control, security, safety and privacy	- -	+	+
2. Help with the information burden	+	+	+*
3. Help with household chores, integrate appliances	+	+	++**
4. Support planning & organizing, allow different authorities, save user preferences	+	+	+
5. Context awareness, help with common tasks	+	+	+
6. Help people stay in contact with one another	0	+	+*
Extra 1. customization	-	-	+

Table 1: Different AI approaches side-by-side

* depends on the application developed for the system

** depends on whether the devices provide correct precondition-effect-rules

From the table we can see that it's very difficult to say how well the goal-based approach would meet the requirements, because the it has not been implemented or designed in detail. However in its simplicity the goal-based way of doing things would not hinder the development of external applications that meet these requirements. The weakness of the goal-based approach would be that it requires all the appliances to

define their set of precondition-effect-rules in order to make use of them. This requires approved standards to work and probably would cause problems in the beginning. This is however not a huge problem on itself, because no matter what the exact approach is, a dynamic, customizable environment like the smart home requires standards so that the appliances may cooperate. Keeping the system open and possible for the user to define her own rules is an easy way to enable the addition of previously incompatible devices.

8.7 Discussion on the Artificial Intelligence

Many projects treat artificial intelligence as the most important feature of a smart home. The all-seeing, all-knowing smart home system makes the life of the inhabitants easier and takes care of the dull chores automatically. The first question and biggest with these kinds of visions is *do we need them?* Artificial intelligence is one of the most vivid areas of smart home research, but should the system be smart? And if it should, then to what extent? A system that can call the authorities when a burglar enters, or can make coffee ready for the user before she wakes up can already be achieved with a simpler AI that is capable of running timed tasks. The big question is whether the pros of a more advanced artificial intelligence overcome the cons.

As discussed previously an entirely proactive AI is something that users don't want. Therefore the sensible choice falls between a weak proactive system and a simple reactive system. From the user's perspective the weak proactive system might be very close to a reactive system if the inhabitant has very alternating or complex routines that the system can't see and thus suggest automating them.

A goal based approach could prove a simple way to give the user power to define new commands and their effects. This would make the system more customizable and thus more desirable to the users. To follow the important requirement of user control the system should in start only support very straightforward and unambiguous commands that can't be misunderstood. For instance if Peter would like to show his holiday photos in a party and he issued the command *'show photos from last week'* and let the system decide the appropriate location and device, the system might do an error and display the photos on his PDA. These kinds of situations can be avoided with very explicit commands. Peter could have said *'show photos from last week on the television and dim lights'*. In addition to the much better probability of success with this command, Peter also gets to use more explicit commands. This might have a very large impact on the feeling of control that Peter has. If Peter decided to omit the target location then the system could try to do an educated guess. If no valid choices could be thought of then the system could ask the user where he wants the photos to be displayed.

As already mentioned in Chapter 2, the focus needs to be taken away from defining smart homes based on the artificial intelligence they possess and instead working with the definition used by Norros and her colleagues where the emphasis is on the enabling nature of a smart home system [Norros et al., 2007]. The intelligence comes from the

interactions between the user and the system, and the *new practices* that this enables. Smartness lies in is the possibility to do things more efficiently, pleasantly and intelligently than before.

9 The Design

In the following subchapters I'll try to describe the characteristics of a smart home system that would have the potential of meeting the user requirements. The solution is divided into two distinct parts: the hardware layout and the software architecture. In the former I'll shortly outline what kind of hardware could work well for a smart home system. The rationale for the choices are introduced in previous chapters. In the latter I'll use concepts and notions provided by previous chapters to describe a flexible software architecture on a very conceptual level. I won't go into details with UML-diagrams or accurate algorithm descriptions.

In general the solution is meant to be taken more as a suggestion for a good direction than an accurate description of a system. Because of the ever-changing area of smart home technology, the design is kept on a more abstract level.

9.1 Hardware Layout

Due to the requirements for mobility and a centralized approach defined in Chapter 2 and 3, the hardware side should be kept light so that the minimum amount of devices would be as low as possible. In a small apartment the ideal system would fill other roles as well. For example many students use their laptops to view DVD-movies, listen to music, play games, write documents and browse the Internet. An ideal smart home system could fill these roles as well.

One immediate physical requirement to the hardware selection is that the computing unit won't overheat when kept on for long periods of time. Especially in a smart home environment the system would be practically on all the time. A closely related matter to overheating is the cooling system used. In a smart home core the best choice would be passive cooling since that would allow noise-free operation. Unlike in the case of Spinellis' smart home core which resided in the basement, the average small apartment inhabitant won't have a proper secluded place to store the system [Spinellis, 2003].

The other benefits mentioned by Spinellis on the usage of a central device do apply to this design. Especially we gain two advantages: all the outside connections are terminated to the same destination and thus wiring becomes easier, and the system can be physically secured from burglars and other evildoers as we only have one device to really worry about. In addition to these advantages, having a main frame that acts like a communications hub between the devices in the house, and between the house and the outside world enables the use of features that are common in everyday office environments, such as in-house printer sharing, shared Internet firewall protection and wireless access, and backup storage for personal files. An UPS-unit may also be used to ensure that the system survives power shortages. Putting these services under one controllable entity could make the network environment more understandable and

easier to maintain as many people are finding these home networks quite cumbersome [Grinter et al., 2005].

Critical appliances such as burglar alarms and smoke detectors should however be capable of working on their own without help from the main frame. Optional communication with the smart home system could however provide additional value. For example the smoke detector could inform the user through the mobile phone that it has called the fire department while the user was away. It's however critical that the devices don't rely on the smart home network being available.

Various different choices for hardware exist on today's market, but one promising alternative for a core system could be a small-factor personal computer with passive cooling and low power consumption, such as the Zonbox (http://www.zonbu.com/home/). Additional devices, such as lighting controls, sensors, external displays and user interfaces could be added to the network as desired, but only one device should be compulsory for a smart home system to begin functioning. Of course one device on its own does not constitute a great network, but it provides the seed on which to build and customize the desired functionality with additional devices. Many devices, such as mobile phones, televisions and other computers, that are capable of interacting with the smart home system are already available in many homes. Forcing people to buy more of these devices to create an intelligent environment is not sensible.

9.2 Software Architecture

The main rationale for the layout presented here was given in Chapter 7. One convenient way to describe the system architecture is to divide it into four layers: hardware abstraction, communications layer, context service and agents. Hardware abstraction is provided on the operating system level of the core device through device drivers. Since most modern operating systems offer hardware abstraction by default and building one is very time consuming, it would be wise to stick to the existing solutions instead of reinventing the wheel. All major operating systems are viable alternatives, but Linux provides the most flexible choice as all levels of the operating system may be modified as necessary. Not to mention the other advantages of having a developer friendly environment, stability and long uptime.

The communications layer is built upon the hardware abstraction layer. The purpose of the communications layer is to further abstract the transmissions so that different technologies such as Bluetooth and WiFi can be used transparently without any need to further program the applications on top of the communications layer. For example if the context service wishes to notify a touch-screen in the bedroom that the oven has heated to the desired temperature, it shouldn't worry about the method to use. However, the upper layers should be able to ask which communications method is being used so that devices capable of using several methods may be treated differently according to the current way of transmission. For instance mobile phones that are used via Bluetooth are

safer than ones that use GPRS since the Bluetooth phones are surely close to the Bluetooth receivers in the house.

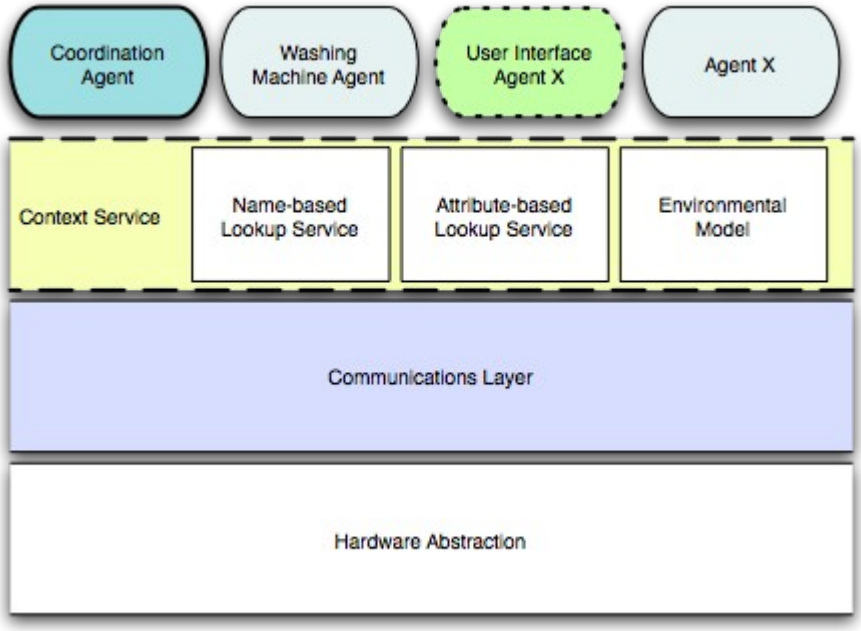

Figure 5: Software layers

The context service is the most complicated component of the architecture. Internally the context service can be divided into three separate parts: attribute-based lookup service, name-based lookup service and the environmental model. The first two are already familiar from Chapter 7 and the Aladdin project introduced in Chapter 4. Attribute-based lookup is the primary way which devices use to find available interesting data sources and the name-based lookup service provides the methods to connect to those devices. For instance it enables connecting the primary user interface to an information retrieval agent that can provide weather forecasts to the inhabitant.

The environmental model ties the devices to their respective locations inside the environment. This operation is crucial to enable location-based cooperation. For example providing follow-me content is dependent on knowing the locations of the right devices. The default way of providing the environmental model is to use a layout editor through the primary user interface. While this method might seem a bit cumbersome, it effectively removes any dependencies to external helper devices. Of course these devices could be supported later on.

The topmost layer is the agent layer. Using the functionalities provided by the layers below, the agents on this level can be implemented quite simply without any prior knowledge about the structure of the household or its inhabitants. This layer is the one where most user-driven immediate design takes place. The coordination agent introduced in Chapter 7 is a key element provided by the smart home system. It enables the controlled cooperation of agents and implementation of weak proactivity. As mentioned in Chapter 8, the goal-based AI approach could be implemented as a part of

the coordination agent. The different software layers are summarized in Figure 5.

9.2.1 User Interfaces

The only required user interface is the primary interface provided by the core system. This user interface is mainly used for specifying the environment model, adding devices to the system and defining more advanced settings. As suggested by previous research the preferred way to do precision demanding settings is through a conventional desktop user interface with a keyboard and mouse [Koskela and Väänänen-Vainio-Mattila, 2004]. As the system is running a conventional operating system, the easiest way to provide a user interface is to build it using the graphical user interface toolkits available to that device.

The system core can also host other user interface agents such as a graphical user interface more convenient for usage through a television screen with a remote. Another supported user interface agent would function through a web-browser. This makes it possible to control the smart home system to some extent via any device that supports a web-browser, such as a mobile phone, an internet-tablet or a PC. Of course these other agents don't have the same privileges that the primary user interface possesses.

9.2.2 Trusted Agents via Public Key Authentication

As mentioned before, the need for security and privacy was one of the key requirements. Securing the system from outside is mostly a matter related to conventional techniques like firewalls, user access restrictions and passwords. The more interesting challenge is making sure that we can trust all the components communicating inside the smart home system. In the ever-changing and heterogeneous environment of the smart home we can't trust that all devices that reach the home network are trustworthy. Especially since we're using wireless technologies.

The first step of including a new agent or device to the system is to approve the stranger. Approval can only be done through the primary interface since the possibility to give this privilege to other interfaces would make it more easier to smuggle unwanted agents to the system. Of course it's possible to control agents that are a part of the smart home system through external components not known to the home network. For example the PC in the bedroom might be approved as part of the smart home network, but the user could access and use the smart home by logging in to the PC via SSH. These kinds of situations can't be avoided. Once an agent is approved as a part of the smart home network, it becomes the responsibility of the agent to make sure malicious agents can't get access through it.

After approval we need to be sure that we are always talking to the same device. Unless we somehow make sure we are dealing with the same device we authenticated with in the beginning, an attacker could use a *spoofing attack* and masquerade as an already authenticated device to get access to the system. Another issue we need to be

prepared for is *eavesdropping*, where the attacker listens to the transmissions between two devices inside the home network. While this won't give him direct access to the system, it's certainly a huge privacy issue.

To tackle both of these threats I propose an authentication and communication method based on public key authentication. This method is widely used today in areas like digital signing. The idea is that devices have two keys: a public key and a private key. The public key can be used to encrypt messages that can be decrypted with the private key and vice versa. The public keys are usually shared with other devices so that they may make private messages to the entity that owns the private key. Effectively this means that unless the private key is leaked to other entities, the message can't be deciphered (at least inside a sensible timeframe) by anyone else but the owner of the private key.

When joining the network and once approved through the primary user interface, the devices would get a copy of the public key of the core and respond by giving their public key to them. Onward from this moment all traffic would then be ciphered. If the inhabitant has a reason to suspect that someone has gained access to some of his private keys, he can always create new ones and reform the network.

While constantly using encryption in message transmission has a rather big negative impact on the performance, I'd argue that the positive sides of increased security and privacy outweigh the cons. Also with the ever increasing computational power that our everyday microchips possess it's possible that the users wouldn't even spot the difference.

Unencrypted messages would also be possible, but should be considered harmful and the user should be warned if sensitive information was sent without encryption. The devices that could authenticate themselves through the public key method would be considered trusted hosts. The way that those hosts consider that devices in their control, such as external sensors and input/output devices, is up to them.

9.3 User Involvement

As discussed in Chapter 2 and 3, people need to be able to design their own homes. This design should not be limited in any way. Thinking that changing the colors of the user interface or using adaptable user interfaces should be enough for all users is a misconception. In a truly open system that encourages immediate design the possibilities for the users to affect the functionality of their system should be available on all levels from hardware to software.

Biggest issue on the hardware level is the avoidance of vendor lock-in. This is accomplished by avoiding it on the software level as well. The system should enable addition of new devices and communication technologies. 3rd party agents and components might still facilitate vendor lock-in. For example a surveillance subsystem might only support certain types of cameras and be distributed as a closed system. This

can't be avoided. As open and/or free source code is still a intimidating factor for many companies, it's better to remember that a closed system is usually better than no system at all.

On the software level user involvement should be embraced in two ways. First it should be possible to replace any layer or subsystem with another one. This can be achieved with good component-based design and clear interfaces. The second way of encouraging immediate design is to implement plug-in architectures so that new functionality can be inserted with little effort. This needs to be possible on all levels from device drivers on the hardware abstraction level to new agents, or even agent subsystems on the top level.

10 Conclusion

The original purpose of my thesis was to provide a solid solution into building smart home systems. However, during the research and design process I became aware of the huge amount of complexities involved. Smart homes are one of the major science themes of our time and drive areas like ubiquitous computing, artificial intelligence and social research on home environments forward. Covering all this in one thesis is impossible.

Instead I've focused on the current status of smart homes and presented the new problems that arise when we bring existing households, rental and small apartments into the equation. Drawing examples from real-life smart home projects and the most important areas inside smart home research, I've suggested various design ideas to take into consideration. The design proposed in the last Chapter tries to combine the ideas so that the requirements from the users and the problems defined earlier may be solvable.

A big part of the message is the realization that successful smart home planning takes the design process from the laboratories to the actual homes and does not assume to ever reach a completed stage. The inhabitants of the households are the real designers of the system and taking this power away from them is not possible when we're trying to build a real, desirable smart home system.

In the end there is no direct answer, nor a direct roadmap on how to make our existing homes smart. The initiative has to be taken by the consumer. The problem of the designer is to provide the incentive for smart home building. Following the guidelines provided by this thesis coupled with good features, lots of development effort and solid marketing I believe that the vision of bringing smart, networked homes to mainstream is possible with current technology.

References

[Abowd and Mynatt, 2000] – Gregory D. Abowd and Elizabeth D. Mynatt, *Charting Past, Present and Future Research in Ubiquitous Computing.* ACM Transactions on Computer-Human Interaction, **7** (1), ACM Press, 2000, 29-58

[Arhippainen, 2007] – Leena Arhippainen, Mukautuvan vuorovaikutuksen kokeminen. In the book *Älykkäiden ympäristöjen suunnittelu*, Eija Kaasinen and Leena Norros (ed.), Teknologiateollisuus, Helsinki, 2007, 196-200

[Battarbee and Kuusela, 2005] – Katja Battarbee and Kristo Kuusela, Interacting with Proactive Technology, In: Frans Mäyrä and Ilpo Koskinen (eds.), *The Metamorphosis of Home.* University of Tampere, 2005, 71-85

[Brumitt et al., 2000] – Barry Brumitt, Brian Meyers, John Krumm, Amanda Kern and Steven A. Shafer, EasyLiving: Technologies for Intelligent Environments, *Proceedings of the 2nd international symposium on Handheld and Ubiquitous Computing*, 2000, 12 – 29

[Chung et al., 2003] – Kook Hyun Chung, Kyoung Soon Oh, Cheong Hyun Lee, Jae Hyun Park, Sunae Kim, Soon Hee Kim, Beth Loring, Chris Hass, A User-Centric Approach to Designing Home Network Devices. *CHI '03 extended abstracts on Human factors in computing systems*, 2003, 648 - 649

[Cook et al., 2003] – D.J. Cook, M. Huber, K. Gopalratnam and M. Youngblood, *Learning to Control a Smart Home Environment.* Innovative Applications of Artificial Intelligence, 2003, available as: http://ranger.uta.edu/~holder/courses/cse6362/pubs/Cook03.pdf

[Crabtree and Rodden, 2004] – Andy Crabtree and Tom Rodden, *Domestic Routines and Design for the Home.* Computer Supported Cooperative Work, **13** (2), Kluwer Academic Publishers, 2004, 191-220

[Cypher and Richardson, 2006] – Mark Cypher and Ingrid Richardson, An actor-network approach to games and virtual environments, *Proceedings of the 2006 international conference on Game research and development*. Murdoch University, 2006, 254-259

[Dideles, 2003] – Myra Dideles, Bluetooth: a technical overview. *Crossroads,* **8** (4), ACM Press, 2003, 11-18

[Elliot et al., 2007] – Kathryn Elliot, Mark Watson, Carman Neustaedter and Saul Greenberg, Location-dependent information appliances for the home. *Proceedings of Graphics Interface 2007*, ACM Press, 2007, 151-158

[EMMUS, 1999] – Introduction to ISO 13 407, University of College Cork web publication. 1999, Available online at http://www.ucc.ie/hfrg/emmus/methods/iso.html

[Franklin and Graesser, 1996] - Stan Franklin and Art Graesser, Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*, Springer-Verlag, 1996, Also available as: http://www.msci.memphis.edu/~franklin/AgentProg.html

[Georgantas, 2005] - Nikolaos Georgantas (ed.), Specification of the Amigo Abstract Middleware Architecture, *Available as http://www.hitech-projects.com/euprojects/amigo/deliverables/Amigo_WP2_D2.1_v10%20final.pdf*, 2005

[Green et al., 2004] – William Green, Diane Gyi, Roy Kalawsky and David Atkins, Capturing user requirements for an integrated home environment. *Proceedings of the third Nordic conference on Human-computer interaction*, ACM Press, 2004, 255-258

[Grinter et al., 2005] – Rebecca E. Grinter, W. Keith Edwards, Mark W. Newman, Nicolas Ducheneaut, The work to make a home network work. *Proceedings of the ninth conference on European Conference on Computer Supported Cooperative Work ECSCW'05*, Springer-Verlag, 2005, 469-488

[Heider and Kirste, 2005] – Thomas Heider and Thomas Kirste, Multimodal appliance cooperation based on explicit goals: concepts & potentials. *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*, 2005, 271-276

[Hiertz et al., 2006] - Guido R. Hiertz, Sebastian Max, Erik Weiß, Lars Berlemann, Dee Denteneer and Stefan Mangold, Mesh technology enabling ubiquitous wireless networks: invited paper. *Proceedings of the 2nd annual international workshop on Wireless internet*, ACM Press, 2006

[Huebscher and McCann, 2004] – Markus C. Huebscher, Julie A. McCann, Adaptive middleware for context-aware applications in smart-homes. *Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing MPAC '04*, ACM Press, 2004, 111 – 116

[Kaasinen et al., 2007] – Eija Kaasinen, Turkka Keinonen, Salu Ylirisku and Johan Plomp, Älykkäiden ympäristöjen piirteitä. In the book *Älykkäiden ympäristöjen suunnittelu*, Eija Kaasinen and Leena Norros (ed.), Teknologiateollisuus, Helsinki, 2007, 95-113

[Kim et al., 2004] – Sung Woo Kim, Min Chul Kim, Sang Hyun Park, Young Kyu Jin, Woo Sik Choi, Gate reminder: a design case of a smart reminder. *Proceedings of the 2004 conference on Designing interactive systems: processes, practices, methods, and techniques*, ACM Press, 2004, 81-90

[King, 2003] – Nicola King, Smart Home – A Definition. Intertek Research. Available on-line at http://www.cat.csip.org.uk/_library/docs/Housing/smarthome.pdf

[Kinney, 2003] – Patrick Kinney, *ZigBee Technology: Wireless Control that Simply*

Works. ZigBee Alliance Whitepapers, 2003, available online from http://www.zigbee.org/en/resources/whitepapers.asp

[Koskela and Väänänen-Vainio-Mattila, 2004] – Tiiu Koskela and Kaisa Väänänen-Vainio-Mattila, Evolution towards smart home environments: empirical evaluation of three user interfaces, *Personal and Ubiquitous Computing*, **8** (3-4), Springer-Verlag, 2004, 234 – 240

[Kuutti et al., 2007] – Kari Kuutti, Turkka Keinonen, Leena Norros and Eija Kaasinen, Älykäs ympäristö suunnittelun haasteena. In the book *Älykkäiden ympäristöjen suunnittelu*, Eija Kaasinen and Leena Norros (ed.), Teknologiateollisuus, Helsinki, 2007, 32 – 51

[Laarni et al., 2007] – Jari Laarni, Risto Näsänen, Tomas Lindberg, Hannu Soronen, Anneli Pulkkis, Pekka Appelqvist, Katja Battarbee, Salu Ylirisku, Ismo Alakärppä, Eija Kaasinen and Leena Norros, Ihmisen toiminta älykkäissä ympäristöissä. In the book *Älykkäiden ympäristöjen suunnittelu*, Eija Kaasinen and Leena Norros (ed.), Teknologiateollisuus, Helsinki, 2007, 114 – 166

[Lesser et al., 1999] - Victor Lesser, Michael Atighetchi, Brett Benyo, Bryan Horling, Anita Raja, Régis Vincent, Thomas Wagner, Ping Xuan and Shelley XQ. Zhang, The UMASS Intelligent Home Project, *Proceedings of the third annual conference on Autonomous Agents*, ACM Press, 1999, 291 – 298

[McFarland, 2003] – Bill McFarland, The Family Dynamics of 802.11, *Queue*. **1** (3), ACM Press, 2003, 28-38

[Meyer and Rakotonirainy, 2003] - Sven Meyer and Andry Rakotonirainy, A survey of research on context-aware homes, *Proceedings of the Australasian information security workshop conference on ACSW frontiers.* **Volume 21**, 2003, 159 – 168

[Moran et al., 1997] - Douglas B. Moran, Adam J. Cheyer, Luc E. Julia, David L. Martin and Sangkyu Park, Multimodal user interfaces in the Open Agent Architecture. *Proceedings of the 2nd international conference on Intelligent user interfaces,* ACM Press, 1997, 61-68

[Mäyrä et al., 2005] – Frans Mäyrä, Tere Vadén and Ilpo Koskinen, Introduction: Living in Metamorphosis – the Whys and Hows of Proactive Home Design Research. In the book *The Metamorphosis of Home,* Frans Mäyrä and Ilpo Koskinen (ed.)*,* Tampereen yliopisto, 2005, 7-27

[Norros et al., 2007] – Leena Norros, Kari Kuutti, Pirkko Rämä and Ismo Alakärppä, Ekologisen suunnittelukonseptin kehittäminen. In the book *Älykkäiden ympäristöjen suunnittelu*, Eija Kaasinen and Leena Norros (ed.), Teknologiateollisuus, Helsinki, 2007, 52-91

[O'Heir, 2006] – Jeff O'Heir, *Monster's Linux Device Brings Intelligence To Home Control.* ChannelWeb online article, 2006, Available from http://crn.com/digital-home/193104836

[Park et al., 2003] - Sang Hyun Park, So Hee Won, Jong Bong Lee and Sung Woo Kim,

Smart home – digitally engineered domestic life. *Personal and Ubiquitous Computing*, **7** (3-4), Springer, 2003, 189 – 196

[Pering et al., 2005] - Trevor Pering, Rafael Ballagas, Roy Want, RFID: tagging the world: Spontaneous marriages of mobile devices and interactive spaces. *Communications of the ACM*, **48** (9), ACM Press, 2005, 53-59

[Roberts, 2006] - Roberts, C.M., *Radio Frequency Identification (RFID).* Online publication, 2006, available online from http://eprints.otago.ac.nz/280/01/RFID_Pre_Publication.pdf

[Rodden and Benford, 2003] – Tom Rodden and Steve Benford, The evolution of buildings and the implications for the design of ubiquitous domestic environments. *Proceedings of the SIGCHI conference on Human factors in computing systems CHI '03*, 2003, 9 - 16

[Röcker et al., 2004] - Carsten Röcker, Maddy D. Janse, Nathalie Portolan and Norbert Streitz, User Requirements for Intelligent Home Environments: A Scenario-Driven Approach and Empirical Cross-Cultural Study. *Joint sOc-EUSAI conference*, 2004, 111-116, Also available as http://www.hitech-projects.com/euprojects/amigo/publications/roecker_et_al.pdf

[Sainz de Salces et al., 2005] – Fausto J. Sainz de Salces, David England, David Llewellyn-Jones, Designing for all in the house. *Proceedings of the 2005 Latin American conference on Human-computer interaction*, ACM Press, 2005, 283 – 288

[Simmons, 2006] – Dan Simmons, Smart homes a reality in S Korea. *BBC Click online* article, 2006, Available online at http://news.bbc.co.uk/2/hi/programmes/click_online/6179868.stm

[Schindler, 2004] – Esther Schindler, The Buzz in the Background, *NetWorker.* **8** (3), ACM Press, 2004, 24-29

[Spinellis, 2003] - Diomidis D. Spinellis, The information furnace: consolidated home control. *Personal and Ubiquitous Computing*, **Volume 7 Issue 1**, 2003, 53 – 69

[Taylor and Swan, 2005] – Alex S. Taylor and Laurel Swan, Artful Systems in the Home, *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, 2005, 641-650

[Tilastokeskus, 2005] – Tilastokeskus, *Kolme neljästä alle 30-vuotiaiden taloudesta asui vuokralla*. 2005, Available from http://www.stat.fi/til/asuolo/2005/asuolo_2005_2006-10-25_tie_002.html

[Webber, 2007] – Paul Webber, *History of LinuxMCE, why I started this project, and the goals*. Online wikipage, 2007, Available from http://wiki.linuxmce.com/index.php/History

[Wang et al., 2000] – Yi-Min Wang, Wilf Russell, Anish Arora, A toolkit for building dependable and extensible home networking applications. *Proceedings of the 4th conference on USENIX Windows Systems Symposium - Volume 4 WSS'00*,

USENIX Association, 2000

[Wikipedia] – Actor-network theory, Wikipedia online article. Available online at http://en.wikipedia.org/wiki/Actor-network_theory

[Wikipedia 2] – ZigBee, Wikipedia online article. Available online at http://en.wikipedia.org/wiki/ZigBee

[Woodruff et al., 2007] – Allison Woodruff, Sally Augustin and Brooke Foucault, Sabbath day home automation: "it's like mixing technology and religion". *Proceedings of the SIGCHI conference on Human factors in computing* systems, ACM Press, 2007, 527-536