Reasoning in a Smart Home

Lauri Kainulainen

AIOME 2006

1 Introduction

Smart home environments have been thought of for decades, but are only now slowly becoming reality. The biggest problem so far is the lack of user acceptance: smart home technologies are not perceived as necessary additions to the common home. Other related problems include technical problems and high cost.

Homes are very personal areas. The technology that is required to create a smarter home must be unobtrusive and preferably hidden. This is like Weiser's notion of Ubiquitous computing as technology that is completely hidden from the user, but still adds value to the daily routines. This unobtrusiveness also affect the AI of a smart home. While helping the occupant get on with his daily routines, the AI shouldn't take a too strong role in the running of the house.

In this short paper I'll go through two different approaches to create a smart home artificial intelligence and take a look at a third approach that should solve the caveats of the two. I'll go through the technical parts of the AI algorithms, but I'll also discuss the overall principles and effects the specific algorithm exposes to the system.

2 Requirements from a Smart Home AI

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 two parents, two children family, 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.

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.

The second category 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 favourite sources when she sits down to eat breakfast or the automatic notification of marked appointments and upcoming followed TV-series.

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 home environment organization, like the closing of curtains or switching 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 behaviour 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.

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. It's quite interesting that this requirement does not appear on Röcker's study, but that does not mean we can ignore it when designing a smart home.

3 Two Different Approaches to Smart Home Reasoning

In the following chapters two fundamentally different approaches to smart home artificial intelligence are presented. Both approaches have the same aim of making the inhabitant's life easier, but they try to reach that goal through different means.

3.1 MavHome and Active-LeZi

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 automatingand 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.

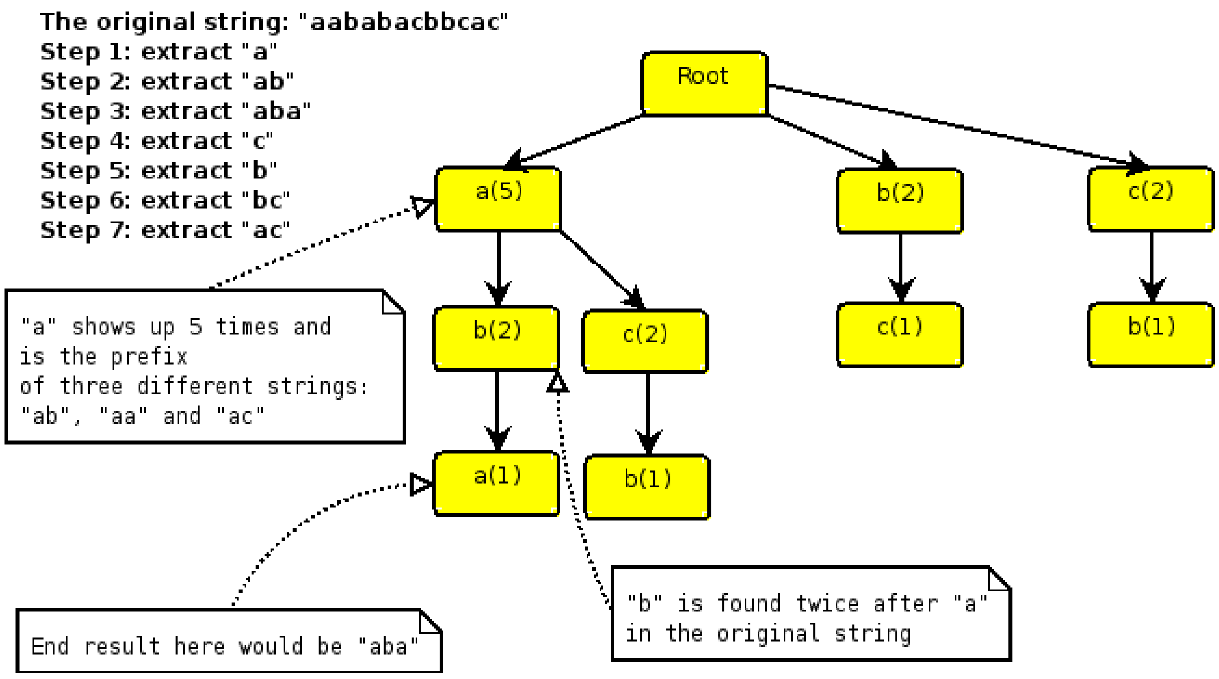
Each agent is composed out of four layers. The decision layer does the thinking and chooses the actions to take based on information that comes from the Information layer. The third layer is the communication layer that handles communication with other agents. The last layer is the physical layer that contains all the hardware and individual devices.

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.

3.1.1 Understanding Active-LeZi

The Active-LeZi algorithm 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 basically operates in two distinct phases. Firstly it observes the user and remembers all the actions and their sequences. Secondly it tries to predict what the user would do next. For this prediction it uses an algorithm based on the LZ78 (see [Cook et al., 2003] or http://en.wikipedia.org/wiki/LZ78 for a more detailed explanation). This algorithm is an online prediction algorithm mostly used in compression. It takes an input sequence and forms a trie of it. This trie is sorted by the prefixes of substrings extracted from the original input sequence. For example the short input string of *"aababacbbcac"* would form a trie like the one depicted in illustration 1.


The original string: "aababacbbcac"
Step 1: extract "a"
Step 2: extract "ab"
Step 3: extract "aba"
Step 4: extract "c"
Step 5: extract "b"
Step 6: extract "bc"
Step 7: extract "ac"
Root
a(5)
b(2)
c(2)
b(2)
c(2)
c(1)
b(1)
a(1)
b(1)
"a" shows up 5 times and is the prefix of three different strings: "ab", "aa" and "ac"
"b" is found twice after "a" in the original string
End result here would be "aba"
Illustration 1: The resulting trie of the LZ78 parsing of "aababacbbcac"

The second phase of Active-LeZi does the probability calculations. Looking at the trie we can see that it's possible to predict the possibility of the next character as soon as we see the first one. For example when the current character would be the letter "a" and we would like to guess the following one, we would have a two out of five possibility of it being a "b" or a "c" and one out of five possibility of it being a null. If we found out that it would be "ab" then we could say that there's a 50% chance the next character is an "a". When dealing with a more bigger and at least some what deterministic input string we could easily count the probabilities of the following characters. The overall possibility of a substring is also easy to calculate by multiplying the possibilities along the path that leads to that substring.

Of course the calculations in MavHome are not done with plain characters. Instead every time the user does something that can be seen as an *event*, the smart home makes a notion of it by storing a triple consisting of the device the user interacted with, the change that occurred in that device and the time of the event. When storing the usage history of a device in a group of $\{x_1, x_2, x_3, \ldots, x_i\}$ it is the task of the algorithm to predict the event x_{i+1}.

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.

3.2 EasyLiving and the Geometric Model

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.

3.3 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 fulfils the desires of the users. The easier one to tackle is MavHome: with it's no-questions-asked kind of operation 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 next actions and doing them instead of the user.

On the other hand 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.

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 behaviour 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.

4 Goal-based Approach

As a solution Heider and Kirste [2005] propose in their paper a goal-based approach that lets the user define explicit goals, which are then analysed and hopefully reached 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 a goal-based approach as defined by Heider and Kirste 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.

Using this kind of technique might also be described as *weak proactivity,* a term introduced by Frans Mäyrä and his colleagues in their Morphome study on smart homes [Mäyrä et al., 2005]. *Strong proactivity* would mean a system like MavHome, which takes control of the situation and does a lot of things automatically. A system that follows the principle of *weak proactivity* will always ask the user first if he would like to automate something. The Morphome study found that this possibility for the user to state if something should be done or not is one of the main factors in a smart home environment that create the sense of control for the user. Seeing that the need for control was found to be the most important thing in Röcker's study, we can quite safely assume that a weak proactive system is the correct approach to take when designing an artificial intelligence for the smart home environment.

4.1 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. Since a smart home contains multiple devices and the user interface must provide at least both visual and aural feedback and it has to be accessible in all the different areas of the home, it is not sensible to restrict the control of the house to just one device or interface. Instead a smart home should have many different user interfaces that all serve their own purpose. This was also discovered in a research conducted at the Technical University of Tampere, where three different smart home user interfaces – a smart phone. a laptop and a TV with a media terminal – were used in actual environments for a period of six months [Koskela ja Väänänen-Vainio-Mattila, 2004]. All three user interfaces had their own purpose: while the TV was great for doing chores and browsing media content *together* with other people, the laptop was more suitable for more complicated tasks. The smart phone on the other hand was perceived to be a very nice solution for remote monitoring and control of the user interface.

Thus the first step of the goal-based approach 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.

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 the perfect system could be a blackboard type of solution that combines multiple different AI techniques such as a declarative planner and a neural network.

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 how powerful it is, what it needs for operation and what are the effects when it has been turned on.

One relatively easy 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 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 could 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 to define.

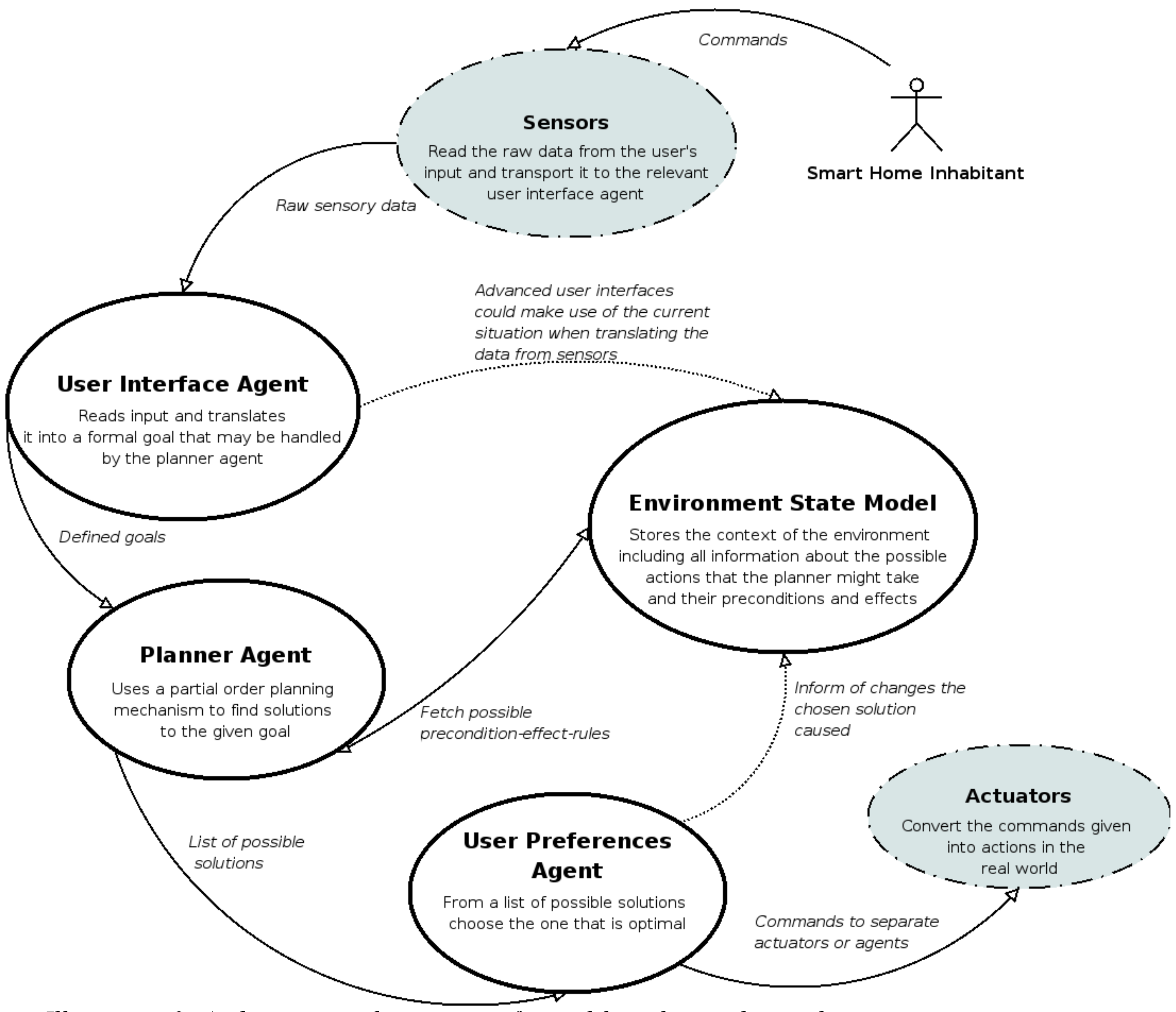
Instead of calculating the quality beforehand, the system might let the user decide which approach would be best. When faced with multiple solutions to reach a specific goal the system could suggest one and then let the user switch to the next one if the previous solution was not satisfactory. After 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 category one (need for control) and four (user preferences should be saved) better.

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 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"?).

A different approach to meet the requirement of a learning house would be to introduce *"modes"* 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 the closest 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.

To keep things simple the planner agent should be separated from the agent that chooses the best approach for the problem based on user preferences and environment conditions. This chooser agent would probably be the same one that stores all user preferences. Illustration 2 depicts a simple overview of the architecture and communication between the different smart home agents.



Commands
Sensors
Read the raw data from the user's input and transport it to the relevant user interface agent
Smart Home Inhabitant
Raw sensory data
Advanced user interfaces could make use of the current situation when translating the data from sensors
User Interface Agent
Reads input and translates it into a formal goal that may be handled by the planner agent
Environment State Model
Stores the context of the environment including all information about the possible actions that the planner might take and their preconditions and effects
Defined goals
Planner Agent
Uses a partial order planning mechanism to find solutions to the given goal
Fetch possible precondition-effect-rules
Inform of changes the chosen solution caused
Actuators
Convert the commands given into actions in the real world
List of possible solutions
User Preferences Agent
From a list of possible solutions choose the one that is optimal
Commands to separate actuators or agents

Illustration 2: Architecture and operation of a goal-based smart home planner

The communication arrow from the user preferences agent to the environment state model is not solid to emphasize that this might not be the best approach. While it would be very simple to just let the chosen solution affect the environment state through its deletes- and achieves-facts it would cause problems when the chosen operation failed for some reason after the commands have been issued. Perhaps the optimal solution in keeping the environment state model up to date would be to create an observer-pattern between it and the sensors where the model would observe changes in the environment and through those update its own internal state.

5 All Three Approaches Side-by-Side

To summarize a part of the results found in this paper, I'll construct a table that matches the 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 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 taking the approach proposed in chapter 4.1, it's possible to see that 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.

6 Conclusion

I have presented one goal-based approach in defining a smart home artificial intelligence. This approach would in its simplicity emphasize the use of *weak proactivity*[Mäyrä et al., 2005] coupled with modular design that survives in a dynamic environment such as the smart home. The system would allow the developers to meet the requirements found by Röcker and his colleagues [Röcker et al., 2004] and it would be easily extendable. On paper the approach seems quite functional, but the real value of the design can only be tested with a real prototype that can then be reliably compared with other smart home solutions. Hopefully this will be possible someday.

7 References

[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

[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

[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

[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

[Koskela ja 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

[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 ja Ilpo Koskinen (ed.), Tampereen yliopisto, 2005, 7-27

[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