Reducing Complexity of Consumer Electronics Interfaces Using Commonsense Reasoning

by
José Humberto Espinosa Christlieb

Ing. Computer Engineering
Universidad Nacional Autónoma de México (2000)
Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2005

Author ______________________________

José Humberto Espinosa Christlieb
Program in Media Arts and Sciences
August 12, 2005

Certified by ______________________________

Dr. Henry Lieberman
Research Scientist
Software Agents Group, MIT Media Lab
Thesis Supervisor

Accepted by ______________________________

Dr. Andrew B. Lippman
Chair, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

Reducing Complexity of Consumer Electronics Interfaces Using Commonsense Reasoning

by

José Humberto Espinosa Christlieb

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
on Agust 12, 2005, in partial fulfillment of the requirements
for the degree of Master of Science in Media Arts and Sciences

Abstract

User interfaces to consumer electronics devices – Video recorders, phones, cameras, washing machines, microwave ovens, etc. – are getting too complicated to be easily used by ordinary consumers. We believe that what is responsible for such complication is a design philosophy which simply maps functions the device can perform to controls like buttons and menu items. That leaves the users with the difficult cognitive task of mapping their goals onto the devices' capabilities – a frustrating and error-prone process. Our hypothesis is that we can provide better assistance to the user using Commonsense Reasoning leading to shorter interactions with the devices. Commonsense can infer the users' likely goals from watching their actions, and anticipate what capabilities of the device can fulfill the users' needs. As devices gain networking capabilities and interact with other devices, Commonsense can also help devices cooperate in support of the users' goals.

Thesis Supervisor: Henry Lieberman

Title: Research Scientist

Thesis Committee

Thesis Supervisor

Dr. Henry Lieberman
Research Scientist
MIT Media Lab

Thesis Reader

Dr. V. Michael Bove Jr.
Principal Research Scientist
MIT Media Lab

Thesis Reader

Dr. Kevin Brooks
Sr. Staff Researcher
Technology Storyteller Motorola Advanced Design

Acknowledgements

I want to start by acknowledging my advisor Henry Lieberman, without his advise and guideline this thesis will never been possible. I also appreciate the time and wise comments of my readers, Kevin Brooks and Michael Bove.

I also want to thank to the others members of the Software Agents Group, Alex Faaborg and Ashwani Kuman. And, the Commonsense Reasoning Group, Push Singh, Barbara Barry, Ian Eslick, Bo Morgan, Hugo Liu. Special thanks to, Rada Mihalcea for helping me to mining the knowledgebase needed in my thesis.

To the Media Lab in general for being such a vibrant and creative community, and in particular to the CELab. Also, to the sponsors for their interest and support to this project specially to Franklin Reynolds, Williams Jeff, Johnson Jed, and Daniel Kim Kyu-Jong.

This research was funded by the generous support of TELMEX

And Finally I want to thank to all my friends.

Quiero dedicar esta tesis a mi familia, a mi papa José Humberto, a mi mama Hilda y a mis hermanos Hilda, Alejandro y Rodrigo, por todo su apoyo. También quiero agradecer a mis abuelas, Tís e Hilda y a todos mis tíos, tías, primos y primas.

Y para terminar quiero agradecer a todos mis amigos.

Thank you very much to all of you. Muchas Gracias a todos.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The goal of this thesis is to make user interfaces to future consumer electronic devices such as audio equipment and kitchen appliances simpler and more effective for average users. We are convinced that conventional interface design techniques will not suffice for the coming generation of high-functionality, network-connected devices.

In conventional interface design, it is possible to design a simple, easy to use interface for a consumer electronics device if the device has few capabilities, by having a one-to-one map of controls to functions. This leads to good *affordances* [Norman 2002] – the ability for the user to infer what functions are available and how to invoke them by just seeing the characteristics of the device. Unfortunately, when the capabilities of the devices far exceed the number of controls, this direct mapping it is not possible. This leads to mapping multiple functions on a single button, or introducing such interface elements as scrolling menus, or push and hold buttons. For example, a typical consumer camera, the Canon S500, has 15 buttons, two dials, 4 x 2 mode switches, 3 menus of 5 choices in each mode, each with two or three values, 7 on-screen mode icons, etc. This design leads to interfaces that require time, expertise and considerable motivation on the part of the user to master.

On the top of that, today's devices tend to have poor button labeling and feedback. For example, in some of the current CD players there is no way to check if there is a compact-disk in it unless the door of the player is open. This behavior, as well as overloaded modes for the 'play' button, makes the device too obscure for the average person to use.

Some believe that we should return to a simpler era by reducing the number of things a device can do in order to be able to present a simpler interface [Raskin 2000] to the user. While that approach has been successful in products like the Apple iPod, unfortunately, it is not a complete solution to the problem. Maintaining a wide range of functionality while avoiding modes might then lead to a proliferation of devices themselves. For example, if we don't want to have both "daylight" and "night" modes in a digital camera, the user might have to decide in advance if he has to use a *'day camera'* or a *'night camera'* making the overall process of taking pictures even more complicated than with current digital cameras.

Others say that using direct manipulation and visualization is the right paradigm to design user interfaces [Shneiderman 1997]. Complex visualizations are preferred to present complex data to experienced users – like software to help physicians in diagnosis – but are not suitable when users are not willing to become expert before being able to accomplish their goals. Consumer electronics already provides direct manipulation of components in many cases, but direct manipulation of large numbers of controls proves confusing. Designing a visualization tool that meaningfully shows all the states of all the devices available that novices want to use in their homes is not an easy task. This makes the direct manipulation paradigm not scalable as the complexity of consumer electronics grows.

The usability of consumer electronics can be aggravated as microprocessor-based devices accumulate features. Adding a new feature is accomplished by software with a minimal increase in cost of the device, since the software is installed in multiple devices. On the other hand, the cost of adding a new physical control to the device is not minimized by mass production [Brouwer-Janse 1992]. Furthermore, consumers cannot fully evaluate ease-of-use at purchase time (especially in a retail store), but functionality and perceived cost are important criteria. These considerations encourage design of devices with growing functionality, but minimally usable interfaces [Brouwer-Janse 1992].

The consumer electronics industry often identifies two kinds of users from the usability point of view. The *wizards* are technically savvy users with an interest in exploring the details of the devices. The so-called *couch potatoes* are the users that are more goal-oriented; they are interested in accomplishing specific tasks with the devices rather than learning how to use the devices [Brower-Janse 1992] for their own sake. Current devices' user interfaces are often designed for the *wizards*. The naïve users are then forced to learn the internal machinery of their consumer electronics in order to us their devices as tools for their goals.

Device designers should take advantage of the microprocessors inside the devices to improve the usability of their products. In theory, at least, microprocessors let the designers control the interfaces independently of the hardware characteristics, allowing software controls to replace hardware ones. Furthermore, devices can be controlled remotely using network connections. This gives us the opportunity to start exploring new methods of interaction with everyday devices.

In this thesis, we present ROADIE, a prototype consumer electronics interface oriented towards the needs of non-expert users, including so-called *couch potatoes*. The project name comes from the person who is in charge of setting up the audio and video devices during music concert tours. The principal ROADIE objectives are to help the user in the following scenarios:

- *What can I do "out of the box"?* When the user first acquires the device, how do they know what it can do? How do they know what its capabilities and limitations are? Devices should be self-aware, self-explaining, and self-revealing. Onboard memory, processing and networking can access and display information like introductory tutorials, user group messages, examples of use, etc. just when they are needed.
- *Oops, it doesn't work!* Devices should also be self-debugging. Devices should know what the possibilities for error are, and give users sensible options for investigating the problem, mapping the behavior of the device to their expectations, and plausible routes to a solution or to seeking more assistance.
- *Don't do that to me again!* Devices should accept feedback on their behavior and modify their behavior accordingly. They should have the capability of customizing device operation for particular situations and automating common patterns of usage.

- *I should be able to…* Devices should enable unanticipated, but plausible, patterns of use. Especially when several devices are networked together, users should be able to compose the results of using one device with the input of another without learning arcane procedures; converting file formats, patching cables, etc.
- *Why didn't you tell me that before?* Devices should inform users about the technical trade-offs they need to make, so they can make informed decisions and avoid getting caught by unexpected consequences.

To sum up, our goal is to create an interface that is goal-oriented, self-describing, self-revealing and self-debugging.

1.1 Goal-oriented Interfaces

The information presented to the user, and the information exchanged between the user and the system, should always be in the context of the user's goals. For this kind of dialogue, the system needs to have a good idea of the relations between the user's actions and the goals he or she is trying to accomplish.

For example, a reasonable goal for the user after opening the freezer is to want to defrost something. Thus if a system can sense the door opening, the microwave should suggest the defrosting function.

1.2 Self-describing

The system should describe its capabilities and limitations in terms that the user can understand and comprehend. If the devices cannot perform a desired action, it should give the user an explanation of why it failed and how to fix it. If the state of the device interferes with another action, it should inform the user of the actions necessary to set the device in the right mode.

1.3 Self-revealing

In the normal process of manipulating the devices, the user needs to make choices in situations where he might not have a good understanding of their consequences. In this situation, the system should inform the user about the trade-off of each choice. Informing the user of the consequences of trade-offs has the following advantages: (a) it prevents the user from experiencing an undesirable and potentially irreversible consequence of a system action (b) it helps the user back trace to the right point if he or she wants to change the behavior of the system.

1.4 Self-debugging

Fixing devices when things go wrong is one of the most frustrating things about dealing with consumer electronics. A common reason for problems is when the user's use of the device is outside the designer's anticipated scenario of use, or there is a piece that it is not working as expected. Fixing this problem forces the user to introspect about the system's internal state – which might be hidden by the device designer – and to figure out what is wrong and devise a strategy to fix it.

The device interface should free the user of this task, and generate hypotheses concerning what might have gone wrong. It should test those hypotheses automatically, when possible. If the system cannot test a hypothesis it should give to the user an explanation of what might be wrong, how he or she can test it, and the steps he or she should follow to correct the problem.

1.5 Commonsense Knowledge

In order to enable this goal-oriented approach, ROADIE uses EventNet, a plan recognizer incorporating the OpenMind common sense knowledge base [Singh 2002]. It uses temporal knowledge from this corpus and spreading activation to infer a possible set of antecedent or subsequent actions. The details of the implementation of EventNet are explain in section 2.2.

ROADIE uses EventNet to infer the user's goals form his or her actions, and proposes specific device functions that might accomplish the user's goal.

1.6 The Architecture of ROADIE

To accomplish the design guidelines described above, ROADIE is composed of two components: a user interaction module, and a device controller. A detailed description of the architecture is explained in Chapter 3.

The user interaction module maps the actions, goals and desires of the user to a format that the planner can understand. This component works as a complement to the normal device's interface, sensing the user's interactions with the devices. It uses EventNet to find the implications of the user's actions. For example, if the user plugs in his or her guitar, the system infers that it is likely that the user wants to play music and communicates this to the device controller. It is also responsible for providing an explanation about the behavior and functionality of ROADIE and the devices it controls.

The device controller has the capability to sense and change the state of the devices. It has knowledge about how the devices work, and can deal with all the intricate steps inherent in managing the complex functions of the devices. This layer is responsible for inferring that, in order to watch a recorded show, the devices have to be on, they have to be connected, and the proper input of the television must be selected. This layer has a built-in planner to provide a flexible inference engine for performing the device's functions, providing the knowledge that a *wizard* user has and the *couch potato* lacks. In addition, the device controller has a model of the capabilities of the available devices. This knowledge helps to constrain the broad options provided by EventNet. If the user says that she or he wants to hear some music, EventNet might retrieve that dancing is related to music, but since no capabilities of the device relate to dancing, those irrelevant nodes will be filtered away.

In addition, the architecture can find and set the states of the devices that are relevant for a particular configuration. For example, if to listen to music from the computer on the stereo, it is necessary to change the setting of the home router, ROADIE can detect that piece of information and change the configuration, saving the user the trouble of having to debug the problem.

Chapter 2

Commonsense Reasoning

Giving computers commonsense has been one of the biggest goals of Artificial Intelligence since the beginning of the field. A program should know that in order to use your car it should be in the same place you are (or at least at reasonable distance) [McCarthy 1959]. To accomplish this goal it is necessary to solve two main problems: a) collecting and storing all the knowledge needed b) building reasoning algorithms capable of using this knowledge. The biggest and longest-standing effort in this direction is Lenat's CYC [Lenat 1995]. CYC is produced by a team of knowledge engineers who have worked for two decades to carefully encode Common Sense in a formal language, CYCL.

2.1 OpenMind Commonsense Project

In contrast, the Media Lab's OpenMind Commonsense Project collects common sense knowledge from volunteers over the web. It is a website where the users are asked to fill templates like *"Something that might happened when you go to ____ is that you might ____"* using plain English [Singh 2002]. By the spring of 2005, the site has collected around 750,000 sentences from 16,000 contributors. An example of the knowledge found in this knowledge base is *"Something that might happen when you go to the zoo is that you might see exotic animals"* and *"The effect of walking in the rain is getting wet.*

By mining the templates in OpenMind and applying Natural Language Processing (NLP) techniques, a semantic network, called ConceptNet, was created [Liu 2004]. This network has 300,000 nodes and 1.6 millions links, like *[Subevent "go to the zoo" "see exotic animals"]* or *[EffectOf "walk in the rain" "get wet"]*. ConceptNet has a variety of operations like getting the context of a given topic, analogy making, topic spotting, and classification [Liu 2004]. ConceptNet has been used to embed Commonsense reasoning into interactive applications [Lieberman, Liu, Singh, Barry 2005].

Using the temporal links from ConceptNet a dynamic Bayesian network called LifeNet was created. LifeNet is formed by "egocentric" nodes of the form *"I go to the zoo"* and a set of weights linking the nodes. This network uses belief propagation [Pearl 1998] to perform a variety of temporal operations like predicting what else might be true now, in the near future or in the near past, explaining why some event happened, or filtering out nodes that are not likely to be true. Probabilistic reasoning was introduced into LifeNet due to its semantic imprecision [Push 2003].

2.2 EventNet

EventNet uses the temporal nodes in LifeNet to create an association network. It can make predictions of the events most likely to occur before or after a certain set of events, in contrast to LifeNet's single-event predictions. Also, it provides paths between nodes providing a plausible sequence of partially-ordered events occurring between two given events. It is able to infer that in order to watch a movie it is necessary to buy a ticket and that a person is likely to buy popcorn. EventNet is a suitable inference engine for applications that have to watch users' actions and give them advice or suggestions. (See Chapter 4 for examples).

The EventNet links are expressed as triplets of the form *(0.504 "I go to a zoo" "I see exotic animals")*. The first element expresses the weight of the link, the second is the parent node and the third element is the son node. The parent node expresses an event that happened before the son node.

In this network all the nodes are expressed in an egocentric way with no distinctions between the cases where the subject is executing or receiving the action. Examples of EventNet nodes are "I run," "I eat breakfast," or "I am sick."

2.2.1 EventNet inference Algorithm

EventNet uses spreading activation algorithm to do its inferences. At each step, every energized node spread a fraction of its energy to its adjacent nodes. The value of the spread energy is directly proportional to the weight between the nodes.

The energy of any node after a spreading step is calculated using the formula,

$$n_i = \sum_{j = links(n_i)} energy(n_j) * weight(n_i, n_j) \qquad \textbf{(1)}$$

This causes the nodes with high connectivity to increase their likelihood, and filters out the unlikely and irrelevant links. Spreading energy from the node "I rain" [1] will reach

[1] Note that the node is "I rain" and not "It rain" since LifeNet and EventNet are formed of egocentric nodes.

the concepts “I paint someone’s house,” “I get wet,” “I go to baseball game,” “I walk in rain,” “I go to zoo” in the first iteration. In the second iteration the top-ten concepts are “I walk in rain,” “I catch a cold,” “I get wet,” “I am cold,” “I wash someone’s car,” “I wash away dirt,” “I have clothes to clean,” “I repair my umbrella,” “I play in water,” “I have a bath.” Note that the unlikely nodes (“I paint someone’s house” and “I go to a baseball game”) are filtered out in the second iteration (Figure 2-1).

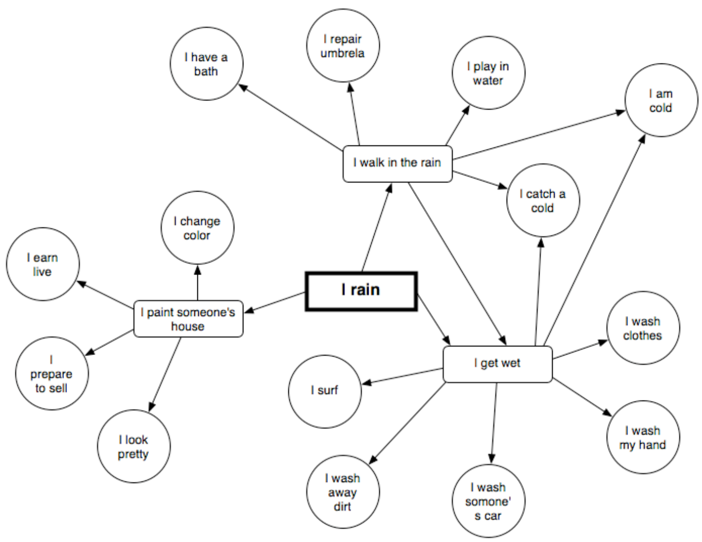


Figure 2-1: Spreading activation over one node. The bold rectangle node is the query node, the rounded-rectangle nodes are the nodes reached on the first iteration, the circular nodes are reached in the second iteration. The unlikely node “I paint someone’s house” is pulled down since it is not heavily connected as opposed to the nodes “I get wet” and “I walk in the rain”

Despite EventNet’s origin as an extraction from a Bayesian network, it uses spreading activation instead of a Bayes-rule inference algorithm. The latter method does not allow changing the topology of the network during the inference; this is needed to add the semantic links. Also, spreading activation *emphasizes* highly connected nodes by adding multiple paths while Bayes networks *subtracts* the extra overlap.

2.2.2 EventNet Temporal Toolkit

EventNet is conceived as a temporal reasoning toolkit. It provides two basic operations: plan recognition and paths between events. EventNet is implemented in Common Lisp; in addition the main API has been packaged as an XML-RPC server to be easily accessed. It provide two basic operations:

a) **Plan recognizer.** This operation is able to infer what are the user's likely next actions or goals based on a set of observed events. This operation works by applying a certain amount of energy $\square_{node}$ to one or more observed events, then applying the spreading activation algorithm descried in the previous section. This operation infers a set of possible next events, previous events or temporally related events. The future events are calculated by spreading the energy forward from the parents to its sons. The past events are calculated by spreading the energy backward from the sons to its parents. The temporally related events treat EventNet as a undirected graph, giving you the composition of the next and past events.

   The API for the plan recognizer allows specifying the number of times the spreading step will be applied before yielding the answer and the desired size of the answer. For example: calling the function (find-next-state "wake up" :size 100 :ntimes 2) gives you the top-100 ranked nodes associated with future events of "wake up" after spreading the node two times.

b) **Paths between events:** This operation finds a plausible explanation between two temporally separated events. This algorithm is inspired on a planning algorithm originally proposed by Maes [Maes 1989].

The algorithm works as follows:

1 Each of the source and goal nodes is excited with energy $\square_{source}$ and $\square_{goal}$ respectively. In addition, the source nodes are marked as activated.

2 The source nodes spread energy to their children and the goal nodes spread energy to their parents. The amount of energy injected is directly proportional to the current energy of the node and the weight of the link, as expressed in function 1.

3 All the nodes that have received energy from at least one of their parents keep spreading the energy to their sons. In the same way, all the nodes that have received energy from at least one of their sons keep spreading the energy to their parents.

4 After each spreading step, the energy of the nodes is averaged to a certain value $\square$. This step keeps the total energy within the network constant.

5 All the nodes that have at least one of their parents marked as activated and their energy above a threshold $\square$ are marked as active. If no nodes are marked as activated the threshold $\square$ is decreased by 10%.

6 If in a single iteration no new nodes are excited (the number of excited parents and children remain constant) there are no paths between the source and goal nodes. In this case the network uses the semantic information within the nodes to generate a new link and connect the two otherwise unconnected subgraphs. See the explanation about semantic link calculus below.

7 Repeat steps 2, 3, 4, 5, and 6 until there is at least one path of activated nodes between the source node and the goal node.

8 This operation provides an explanation of plausible events, without committing itself to whether the events are necessary or just stereotypical. A path between the event "I wake up" and "I go to work" can be "I wake up," "I take a bath," "I put my clothes on," "I eat breakfast," "I drive my car," and "I go to work." In this path, just the node "I put my clothes on" is *necessary* to accomplish the goal, but it is safe to assume that the other nodes are also true.

c) **Create semantic links.** EventNet is formed by 10,000 nodes and 30,000 links. It is not a fully connected graph; therefore there is not always a path between two nodes. To bridge this knowledge gap, the system can dynamically create new links between two nodes that have semantically similar meaning. This similarity is calculated using synonyms from WordNet [Fellbaum 1998] and analogies from ConceptNet [Liu 2004]. This operation is an extension of Cohen's WHIRL [Cohen 2000] using WordNet's and ConceptNet's semantic information.

For finding a path between the nodes "I get money" and "I eat lunch", we find that the nodes are not connected. The algorithm generates new links based on the semantic similarity of the nodes (Figure 2-2). In this case the nodes "I buy pizza" is semantically linked with the nodes "I have hamburger" and "I have hot dog".

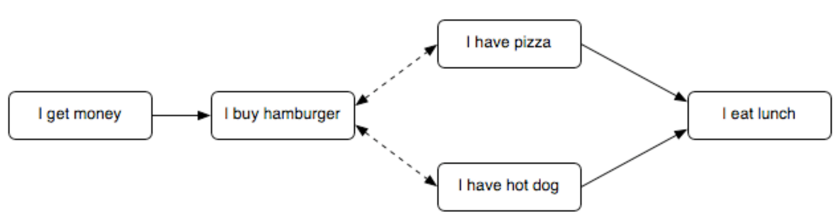


Figure 2-2: Example of using dynamic links to find a path between the nodes "I get money" and "I eat lunch". The solid lines show the temporal links existing in EventNet. The dotted lines show the semantic links

This semantic expansion also is used to match existing nodes to process text queries, getting a close match not by plain keyword matching, but by semantic similarity.

Chapter 3

Description of the System

Figure 3-1 diagrams the main components of the system: the device interface, a planner, EventNet – Commonsense reasoner – and the user interface. In addition, the system is divided in two modules from the interaction point of view, the user module and the device module.

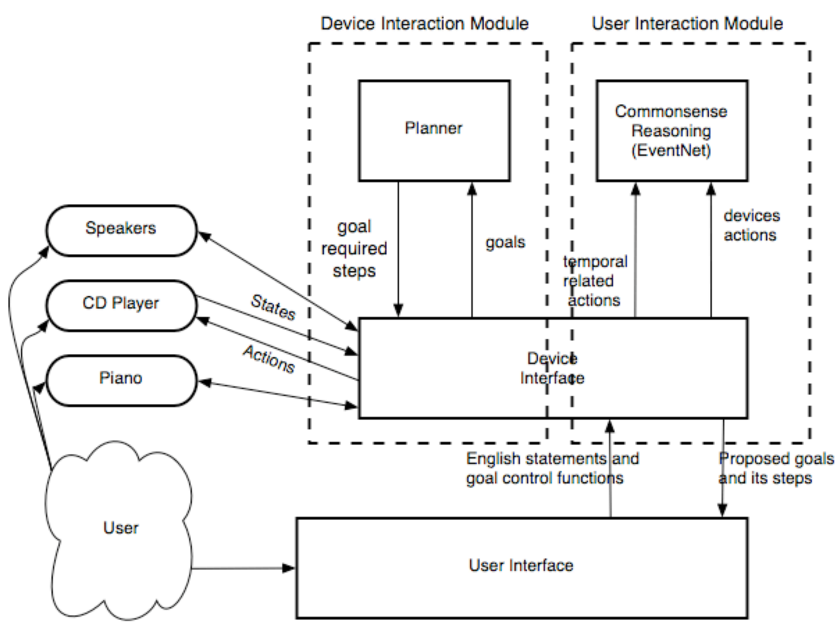


Figure 3-1: ROADIE Diagram

3.1 ROADIE Device Requirements

ROADIE is designed to operate with devices that 1) provide means to control their functions, and 2) that can query their state by external software. The first requirement allows ROADIE to control the devices on the user's behalf; the second allows ROADIE not only to watch the state changes of the devices and interpret them as the user's actions, but it also monitors the devices by looking for direct user interaction.

Unfortunately, the devices available to us at this time do not meet these two requirements. The first requirement can be easily mimicked by using an infrared controller to override the device's remote controls. But there is no easy approach to extending API's of current devices to get their states. Controlling the devices without getting their states leads to clumsy design, since the controller's internal representation of the devices becomes unsynchronized every time the user interacts with the devices using the traditional front panel interface controls.

The devices' manufacturers are aware of this problem and created UPnP [UPnP 2000] as a standard for remote interaction with home appliances. Unfortunately, the manufacturers have not started to build devices that fully comply with this standard. Furthermore, even when devices in theory are compliant with the UPnP standard, they sometimes do not fully expose to the applications programmer all the necessary controls and states to accomplish a given task. Also, some manufacturers are interested in implementing sets of branded devices that coordinate using a proprietary protocol or proprietary functionality that prevents systems like ROADIE from fully implementing general interaction with the device.

To overcome this problem, we created a set of simulated devices to test ROADIE. We hope that in the near future UPnP or a similar standard will become widely accepted allowing deploying ROADIE in real devices.

3.1.1 Simulated Device Design

The simulated representation of each device is divided in two classes, the first contains the device's states, and the second contains the visual interaction elements. This design was chosen since it will allow easy modifications to deploy ROADIE in new devices. ROADIE's device simulation includes:

- *Front panel controls:* e.g. power on/off button, radio tuning knob.
- *Front panel display indicators:* e.g. power on/off light, radio frequency display.
- *Back panel input/output jacks:* When connected, these display what they are connected to.
- *Image of the device hardware.*
- Where possible, *functionality of the device:* e.g. playing music when a PLAY button is pressed.

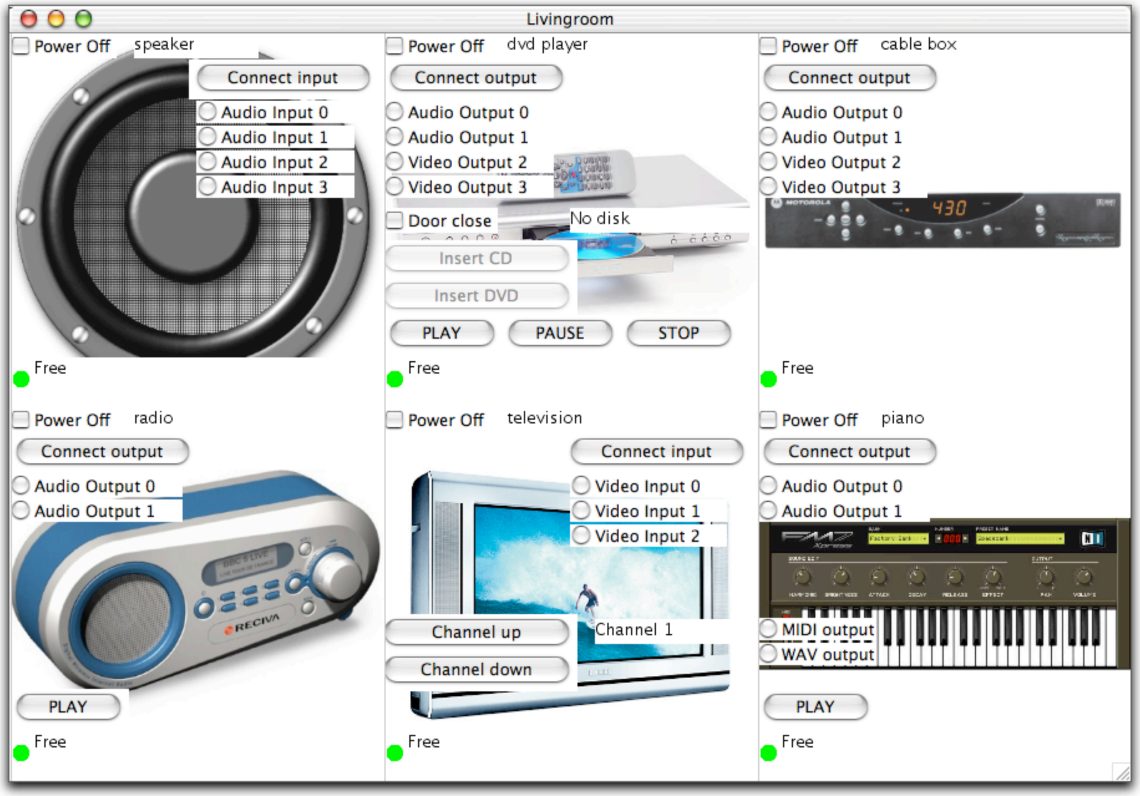


Figure 3-2: Simulated devices used by ROADIE

In addition, the capabilities of the devices were inherited from simpler device models. For example, each device inherits common capabilities from a base class like turning on and off or connecting the device. This class hierarchy helps to create new devices as a composition of old ones; creating the logical infrastructure for a television with a DVD player is as simple as creating a new class inherited from the television and the DVD player. This architecture can be used in addition to the UPnP standard by modeling each device around its UPnP services.

3.2 Planner

The planner is used to infer the set of actions that need to be performed to satisfy the user's goal. It is responsible for knowing and planning all the steps, and checking prerequisites and conflicts inherent in operating the devices. The planner decomposes the desired states to single actions that the devices can execute and creates alternative actions when something unexpected occurs.

In addition, the planner keeps track of the recently performed actions and whether they succeed or not. If it is impossible to accomplish the goal, the system uses this information to provide the user advice to debug and potentially correct the problem.

ROADIE uses the planner to find the steps to configure the devices. It finds the prerequisites necessary to perform each action, and the consequences of each action. Then, it returns an ordered set of actions to accomplish the stated goal.

ROADIE uses the standard Graphplan [Blum 1997] implementation. This planner uses a paradigm called Planning Graph Analysis to reduce the search space of the planning problem, and find a partial order plan. The partial order plan groups the actions that might be possible to perform together finding the shortest path from the initial state to the final one. For example, in order to listen a music CD, the planner will state that it you should connect both devices and turn them on in step one, open the CD door and select their inputs and outputs of the devices at step two, and insert the disk at step three. Grouping the actions means that there is no conflicts in the order these actions can be performed, but at least one of the actions in the current step is the prerequisite of one of the actions of the next one.

3.2.1 Solving Goal Conflicts

There are times when a goal cannot be reached since some of the devices needed to accomplish it are already in use. ROADIE can infer what the conflicting goals are, and when possible, propose solutions to them. To solve these conflicts the planner has built some heuristics to try to minimize the user's interruption.

The first rule is to try to find a free device with similar capabilities that might accomplish the goal. If a device is found, that device is bound into the planner slot.

If no free device is found, the system infers the conflicting goals and tries to assign new devices to these goals. Then it uses the freed devices to accomplish the proposed goal. If this step succeeds, two options are presented: move the conflicting goal to the new configuration, warn the user of a possible disruption and then set the new goal; or stop the conflicting goal and then set the new one.

In the case the second heuristic fails and there is no way to accomplish both tasks at the same time, then the system proposes that one of the conflicting goals be abandoned.

3.3 Commonsense Reasoning – EventNet

ROADIE uses EventNet to infer the user's goals. This inference is done by using the temporal-related operation from EventNet. The input nodes are calculated using templates with English descriptions of the device's changes of state. Then the output nodes are matched against a text description of the available goals.

Since the Open Mind Common Sense collection does not include enough knowledge about the possible interaction between people and home appliances, we decided instead to use the Web to automatically collect pairs of device actions linked by temporal relations. Using a set of templates, we derived a set of queries that were posed against a Web search engine, resulting into a collection of sequences of temporally related actions. The action pairs collected in this way were further post-processed using a syntactic parser that removed the potentially noisy, ungrammatical entries [Mihalcea 2005]. Each pair of actions was also assigned with a weight reflecting its occurrence probability, determined using the same method to assign the LifeNet probabilities.

3.4 Device Interface

The device interface is the module responsible for making the devices communicate with the rest of the system. It is responsible for controlling and monitoring the devices, querying EventNet and sending the goals to the planner. It is the glue between the rigid structure of the Device Interaction Module formed by the planner and the devices and the flexible infrastructure of EventNet, and the ROADIE user interface.

This module has a text string for each change in state of the device, like "turn on the device," "I insert a music CD." These natural language phases are passed to EventNet.

In addition, it has all the possible goals that might be reached with the current devices: both natural language, and as a planner goal with the slots and its acceptable types. For example, it has *<"play the music CD", (play-music-cd [cd-player-device] [speaker-device])>* for playing music CD. So, to set up the action *play-music-cd* it looks for CD players and speakers and sets those particular devices into the planner. Also, it has English templates for each possible planner step and uses them to create explanations in natural language of each step of the planner. The matching between two phrases are made using EventNet's semantic link algorithm.

Even if this knowledge is currently embedded into the device interface, it is mainly expressed in natural language. We can envision then a set of goals and device actions that can be dynamically added every time a device is found. Since these English phrases are linked to EventNet, ROADIE can be expanded to manage a new device's capabilities and goals.

Another advantage to using natural language in contrast to specifying handlers symbolically, is that the approach can be extended to allow the user to control the devices by voice or text. From the device interface point of view, there is very little difference between a device or a user commanding "Turn on the television."

3.4.1 Debugging Information

Though one of the functions of the ROADIE's planner is to give robustness to the system, we do not assume that action sequences will never fail. Problems inherent to devices - malfunctions or misunderstandings between the user and ROADIE - might emerge.

To solve this problem ROADIE provides some basic debugging interface. Debugging consists of looking for the causes of unexpected results. To give the user the knowledge to perform this task, ROADIE can display relevant information for each of the steps to perform the current goal.

The information shown for each step is why the step is important, how the user can perform the step, what the consequences are of do not doing this step, what the results are of performing it, and the things that might go wrong while trying to perform the step. In case the user does not find this information sufficient to solve the problem, the system can automatically send queries to online search engines, user manuals, user group forums, etc. to give the user more detailed information.

Also, for some steps, it is possible that some additional user input will be needed, like choosing the output format. In this case the similarities and differences of each option are presented to the user.

3.5 User Interface

The ROADIE User Interface is the part of the system in charge of the communication between ROADIE and the user. It is a lightweight interface capable to displaying GUI elements following the device manager directions. The interface is currently deployed as a window on a computer screen, but it can be ported onto a PDA, a cell phone or a Universal Remote Control.

At the top of the interface are the suggested goals. When the user picks one of the options, the planner calculates a plan to reach the goal. The answer is mapped to English by the device interface, and rendered by the user interface, highlighting the action that is going to be executed next. Also, the interface has a "Show EventNet nodes" where the user sees the predicted goals that made the action be suggested.

The user can control the execution of the steps by using the "Perform this action" and "Do next step," and get extra information with the "Tell me more" and "Oops, it does not work!" buttons. In addition, the interface has a "What do you want to do?" text box where the user can use natural language to communicate with the system.

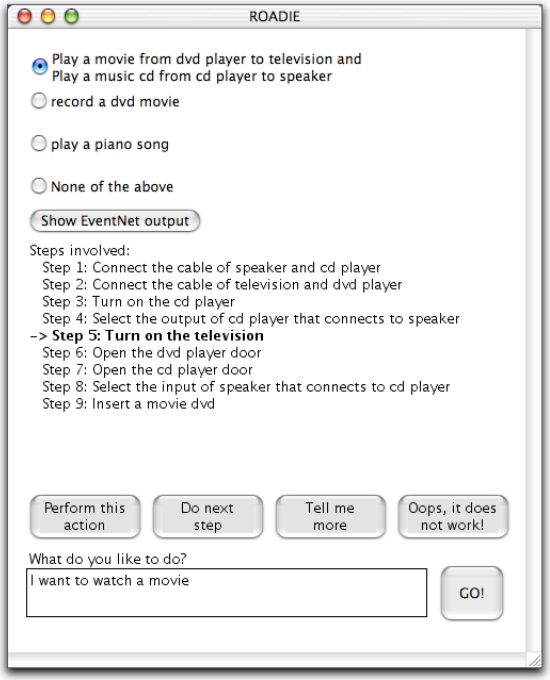


Figure 3-3: Screen shoot of ROADIE's User Interface

In Figure 3-3 you can see a screen shot of ROADIE's user interface. You can see the suggested options on the top, the listing of the necessary steps on the middle and the user goal at the button.

A description of the interaction between the user and the system is given in Chapter 4.

Chapter 4

Users Scenarios

The following sections present several scenarios of interaction with ROADIE. We are assuming that we have a digitally connected network of consumer audio and video devices, such as an amplifier/radio receiver, CD/DVD players, television, speakers, musical instruments, etc. We assume that all the functions of each device can be invoked by software, and all the pertinent states of the device (on/off, volume control, input selection, etc.) can be sensed by the software. Currently, ROADIE functions only in simulation, with software simulating the behavior, states and controls of the devices. Industry is working on initiatives such as Universal Plug & Play [UPnP 2000] and others, but the current devices and protocols available to us for this project fell far short of the necessary capabilities for ROADIE.

4.1 Listening to a CD

The user turns on the DVD player, using its front panel switch. ROADIE queries EventNet for the set of *temporally related events* for the action *"turn the DVD player on."* The system has built-in text descriptions of the actions that the devices can do;

these descriptions serve to communicate the devices' actions to EventNet. EventNet uses the spreading activation algorithm as described in section 2.2 to predict the most likely temporally related events relevant to the recent user actions.

Keep in mind that the predictions generated by EventNet are produced from mining Commonsense facts that appear in the OpenMind Common Sense knowledge base and from the web. Such facts were not entered by the OpenMind contributors for the express purpose of predicting events. Therefore, EventNet predictions are *not* intended to be a complete set of the logical consequences of a particular action, nor do they represent an estimation of the conditional probability of the event. They may arise from very specific situations.

However, since we have not programmed in advance all the possible goals that the user might have, and all the implications of these goals, EventNet is useful in generating at least *some* plausible possibilities for subsequent events, no matter what the user's goal and situation is, as long it could reasonably be considered part of Common Sense knowledge.

For the action "turn the DVD player on" the EventNet answers are

"watch hours of worlds best nature programs,"

"hit play,"

"insert your recorded cd,"

"listen to music,"

"insert disk,"

"insert dvd,"

"leave the room,"

"push television," and

"turn on home theater projector."

Some of the actions, like *"leave the room,"* are ambiguous. Others are just true in a very narrow context, such as "*watch hours of the world's best nature programs*". Again, the idea to generate a broad range of possibilities, and let further constraints from the context, other actions, and interaction with the user narrow down the search space.

Then, ROADIE tries to match the EventNet answers with a description of the device's goals. Each device has a built-in set of goals represented by text (to communicate with EventNet) and symbolic descriptions (as input to the planner). Using text matching as a method to find the likely goals allows flexibility to add new goals to the set of devices, while filtering the nodes that are out of context since they do not match any goal.

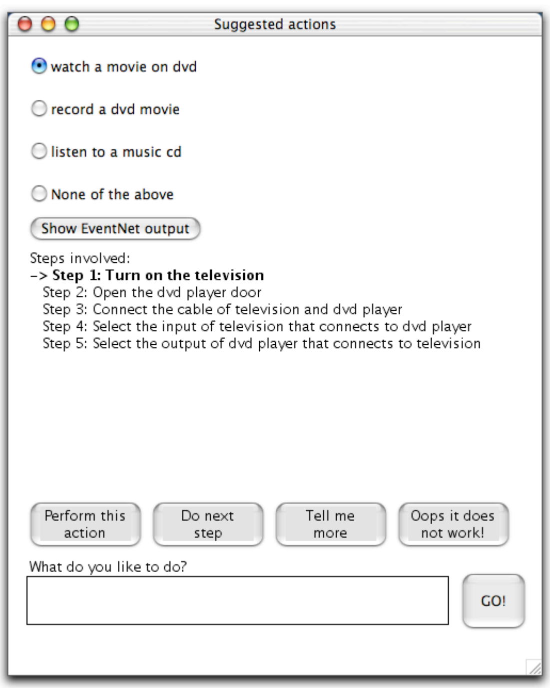


Figure 4-1: ROADIE's suggestion window after the user turns on the DVD player

The set of suggested actions are: *"watch a movie on dvd," "record a dvd movie," and "listen to a music cd."* The ROADIE interface has a button that is called "Show EventNet output". If the user clicks this button, the system displays the EventNet actions and the subset of those actions that triggered the selected actions. This is not normally necessary for an end user, but helps us debug and explain the system.

The user wants to play a CD, so he picks the last choice. This goal needs two parameters: a speaker and a CD player. The system keeps track of the recency of usage of the devices, and knows that the DVD player can also be used as a CD player. Thus ROADIE picks the DVD player instead of the CD player to play the compact disk and asks the planner for a set of actions to accomplish the goal *(play-music-cd dvd-player speaker)*

The planner calculates a plan. Like most AI planners, the job of ROADIE's planner is to compute which sequences of actions will accomplish a given goal, taking into account the requirements of each action step, which steps depend on which other steps, which goals can be accomplished at the same time, and which need to be sequential, which require certain resources and have certain effects, etc. The output of the planner is a partially ordered set of actions. One of the advantages of using a planner is that the system is able to find the configuration to accomplish the goal even if it is necessary to change some settings deeply buried on a device interface, or have to set the state of a remote device into a particular mode.

ROADIE uses the planner output and a set of English templates – one for each possible planner step – to communicate to the user the steps involved in performing this task.

The planner's explanation is shown in the ROADIE interface:

1. Turn on the speakers
2. Connect the cable of the speakers and the DVD player
3. Open the DVD player door
4. Select the output of the DVD player that connects to the speaker
5. Select the input of the speaker that connects to the DVD player
6. Insert the music CD
7. Close the DVD player door

Note that some of these actions can be performed directly by the system, while others (like inserting the CD) cannot.

In addition to the necessary steps ROADIE shows four control buttons: "Perform this action," "Do next step," "Tell me more," "Oops, it does not work!"

- "Perform this action" This button will perform all the steps listed to accomplish the goal at once. If one of the actions needs the user's manual intervention, the system will instruct the user about what he or she needs to do. ROADIE queries the device's states and knows if a step fails. In this case the planner is called again to find an alternative plan. If there is no alternative plan, the system will tell the user which step of the process went wrong along with suggestions concerning how to solve the problem.
- "Do next step" This button behaves like the button "Perform this action" but instead of executing all the steps at once, it executes them one step at the time. This permits the user to observe physical effects of each action. It is useful in learning about the sequence of actions or debugging problems that may occur.

- "Tell me more" This button displays detailed explanation of each step of the process. It tells the user why each step is important, how he can perform the step, what can happen if the step is not finished, and how he can determine if the step has been performed correctly. The aim of this button is to provide the user with technical information for improving his or her knowledge of how the devices work and give him or her the knowledge to debug the steps in case something goes wrong. See Appendix B for an example of the help provided by the system.
- "Oops, it does not work!" This button is used when the system does something unexpected. This button queries an online search engine for information about the step it just performed. This button can be specialized to use the device's user forums or vendor-provided information to give more accurate answers. We are assuming future consumer electronic devices will have wireless network communication chips and will easily be able to connect to the broader Internet for supplementary information. Knowledge about user goals, device states, and other context items can be fed to the search engines directly by the device, rather than asking the user to end their interactions with the device and log into a conventional computer.

Showing the steps to accomplish the goal, along with the information provided by the buttons "Tell me more" and "Oops, it does not work!" helps the user to debug the devices if they do not behave as expected and learn the low level functionality if the users are interested.

The user picks the button "Perform this action," and ROADIE starts to execute the steps until it reaches the action *"Connect the cable of the speakers and the DVD player"*.

The system cannot perform the action by itself, so it asks the user to perform this action. In order to instruct the user how to do it, ROADIE shows the user a picture of the correct input and connector. The interface also displays a "Tell me more" button that explains to the user what a connection is, the different jack types, and the differences between input and output devices and other relevant information about this step. A similar dialog is displayed when the system needs the user to insert the music disk.

When all the steps are completed, the music CD starts to play.

After listening a couple of songs, the user types, *"I want to watch a movie"* in the "What would you like to do?" dialog box. ROADIE recognizes the pattern *"I want to"* as a user goal, then passes it to EventNet to figure out the desired goal. Since ROADIE uses natural language to glue its different parts together, the user's natural language goals are internally handled using the same mechanism that handles the normal interaction.

ROADIE queries EventNet and matches the user's goal to the functions "watch a DVD movie," and "watch television". The user selects the option "watch a DVD movie"

ROADIE realizes that it is not possible to use the DVD player since it is being currently used to play the music CD, but there is also a hardware CD player unit that is capable of playing the CD player. At this point the user has three possible options,

- Perform both actions,
- Play the only the movie, or
- Play the CD.

To explain to the user their options, ROADIE displays the dialog shown in Figure 4-2.

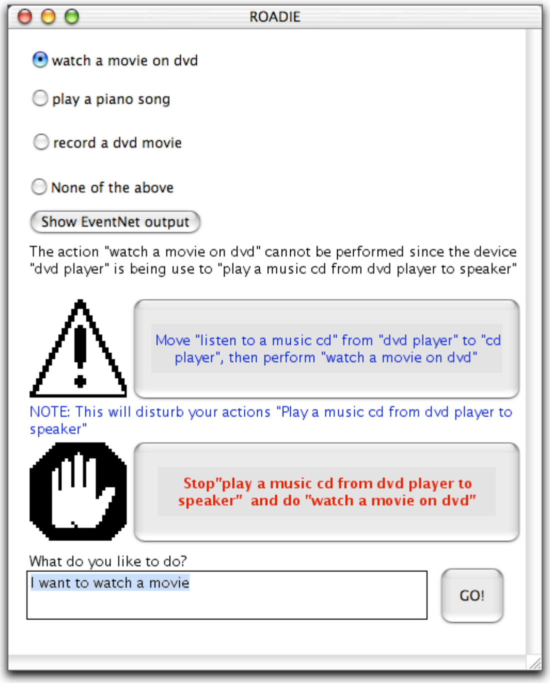


Figure 4-2: ROADIE showing two possible ways of resolving conflicting goals

Note that many of today's CD and DVD players do not give any visible indication whether there is a disk loaded in the tray – you have to open the tray to find out whether there is a disk in it.

In this dialog the system explains options for changing the devices' configuration. If the user ignores the suggestion, the current configuration is kept. ROADIE first displays a message that says *"The action "watch a movie on dvd" cannot be performed since the device "dvd player" is being used to "play a music cd from dvd player to speaker."* And then it displays two buttons explaining the available options. The first button says *"Move "listen to a music cd" from "dvd player" to "cd player," then perform "watch a movie on dvd"* and a note warning the user that the current action "Play a music cd from dvd player to speaker" will be disturbed. The second button says *"Stop "play a music cd from dvd player to speaker" and do "watch a movie on dvd"*

The new desired goal is sent to the planner and the control buttons are displayed.

While this scenario is simple, it illustrates ROADIE's capability of dealing with the problem of conflicting goals. Conflicting goals are a common source of difficulty and problems in operating devices. People experienced in operating audio and video equipment often have sophisticated and successful techniques for resolving goal conflicts.

4.2 Watch the News

The user types into the "What do you want to do?" dialog box the phrase "I want to get the news." This goal is sent to EventNet and then matched with the available goals; the proposed actions are "watch television," and "listening to the radio." The user selects "watch television" and sets the devices by clicking the "Perform this action" button.

A second user turns on the DVD player making the options *"watch movie on dvd," "record dvd movie,"* and *"listen to music cd"* appear, and he selects the first option. ROADIE realizes that the television is busy watching the news, and remembers that also "listening the radio" might satisfy the goal. This will free the television to watch the DVD while satisfying the goal of listening to the news. To warn the user about this conflict and a possible solution ROADIE displays a similar dialog to the one in the previous scenario. This scenario also shows how ROADIE can track device states and user actions, and find concrete actions compatible with multiple high-level goals.

4.3 Playing the Piano

The user has a new piano that has output in MIDI, WAV, and MP3 formats. He plugs it into the amplifier. ROADIE suggests the options *"play piano music," "record piano music,"* and *"listen to a music cd."* The user selects the second option and clicks the "Perform this action" button; in this case there is a choice of output format. On one hand ROADIE does not have enough information to prefer one particular device. On the other hand, the user might not be familiar with the different options. To solve this problem, ROADIE displays a choice dialog. This window explains the basic differences between MIDI, WAV and MP3 formats. In addition, the dialog box has the "Tell me more" button with the necessary information, in addition to links explaining the options in detail. In this window, the user also has the choice of setting a selected option as the default.

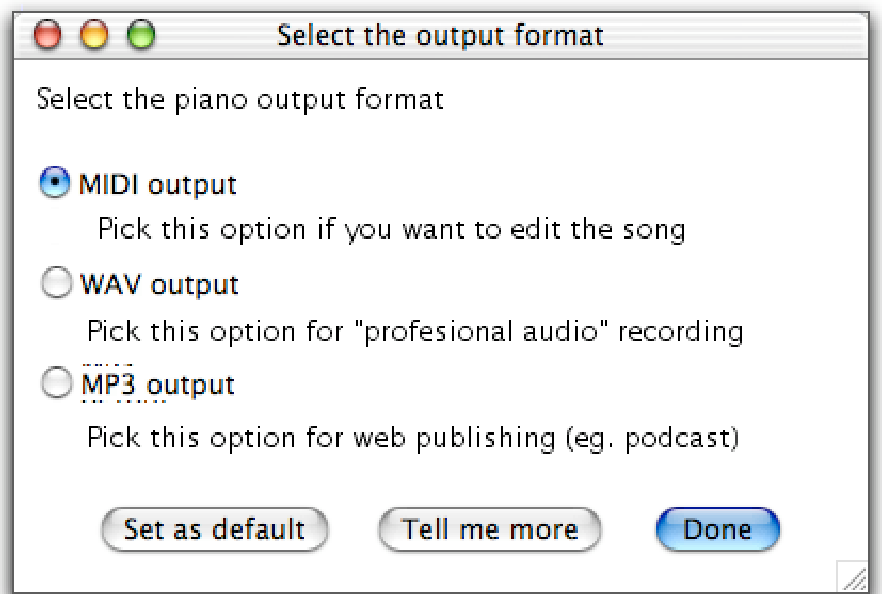


Figure 4-3: ROADIE asking for the output format

This scenario illustrates how ROADIE can elicit goals from the user by presenting choices among the device's configuration options. Had the user explained their goal in advance, using the natural language input box, the system would not have had to ask this question and could have picked between the formats automatically.

4.4 KitchenSense

In this section we will describe KitchenSense, an Augmented Reality Kitchen that uses the same techniques that ROADIE uses to provide the people cooking with context aware information. KitchenSense is a collaboration with Jackie Lee of the MIT Media Lab's Context-Aware Computing Group [Lee 2005]. KitchenSense is equipped with a variety of sensors and digital projectors. One of the main goals of KitchenSense is to motivate the user to be aware of concurrent tasks, keep good cleaning habits, and simplify the interaction with kitchen appliances.

The environment uses its built-in sensors to tag the user's activities, for example a drawer sensor is attached to the sentence "I open the kitchen drawer." Then, this sentence is send to EventNet, and the output is processed by KitchenSense to trigger relevant functions.

4.4.1 Increasing User Awareness

One of the motivations behind KitchenSense is to make the user aware of the potential dangers inherent to the kitchen environment. Imagine that the cook is at the counter and water is boiling on the stove when a child comes into the kitchen. KitchenSense sends the nodes "I boil water" and "I see a child" among other nodes like "I am on the counter" to EventNet. EventNet's inference engine identifies the conjunction of the boiling water and a child as a potentially dangerous situation. Then KitchenSense uses a set of built-in rules to activate its ambient displays based on the cook's position to warn him about this potential danger.

Note the usefulness of a commonsense inference engine with broad knowledge base to enable this application. It uses application-dependant methods to discriminate between the useful implications. In this scenario, KitchenSense might warn the user about the potential danger, while ROADIE might ignore the action.

4.4.2 Increasing usability

In this scenario KitchenSense uses the EventNet inference engine to proactively infer possible actions that might help people cooking. In this scenario, KitchenSense behaves similar to ROADIE. Normal kitchen appliances do not have complex functions, they are usually a very large set of simple operations – e.g. cook meat, boil water, defrost food, etc, … – and these functions are placed on hierarchical menus.

KitchenSense uses the information from its sensors and the EventNet plan recognizer to show device functions that might be relevant to the user activity. For example, when the user opens the refrigerator and gets close to the microwave, KitchenSense sends the sentences *"I open the freezer,"* and *"I walk to the microwave"* to EventNet, the top answers are:

"I cook food,"

"I eat lunch,"

"I reheat food,"

"I take ice cream out,"

"I read newspaper,"

"I set cup on table,"

"I breathe fresh air," and

"I took food out of the fridge."

Then KitchenSense matches these sentences to the functions in the electronic appliances, suggesting the functions "Cook" and "Reheat" of the microwave.

Figure 4-4: This figures shows the suggested functions "Cook" and "Reheat" on the microwave after the refrigerator door is opened.

A variety of similar scenarios have been implemented in KitchenSense. See [Lee 2005] for a detailed description of the kitchen.

Chapter 5

ROADIE Evaluation

We performed experiments to evaluate the contribution of ROADIE to making consumer electronics interfaces more user-friendly and effective. The scenarios we chose to test are ones in which consumers are likely to face problems, such as (a) familiarizing themselves with new devices, (b) performing complex multi-step processes involving multiple devices and requiring data transfer among devices, and (c) debugging problems when things goes wrong.

We would have liked to test ROADIE with physical devices controlled by software, to present a more realistic scenario to the user. As explained above, we were unable to implement ROADIE with physical devices, and so were forced to perform tests on our simulation. However, there were some advantages to using a simulation. Because we pushed participants out of their "comfort zone" and familiar devices, they had to pay more attention. When it happened that they did make mistakes that they might not have made with a physical device, this provided an opportunity to test our debugging capabilities.

5.1 Experiment Hypotheses

The most important aspect to test is whether users will spend less time configuring the devices with the agent turned on or off. We hypothesized that *the users will perform the task faster and in fewer steps with the proposed agent turned on than using only conventional interfaces.* To test this hypothesis we created a set of scenarios where the user has to accomplish certain goal by configuring a set of devices.

In addition, we wanted to test other hypotheses associated with the agent: the potential acceptability of the agent in the participant's home, how intuitive the operation of the devices is using the agent, and how helpful is the interface in solving problems.

5.2 Experimental Design

The experiment consists of four scenarios. The first trains the users on how to use the devices and the agent. During this scenario the users were able ask the experimenter questions. The next three experiments were the test experiments.

5.2.1 Experiment Scenarios

The first scenario consists of playing the piano and watching television. The users have to interact with the devices presented in Figure 5-1. The first task was performed by the experimenter and the second by the participant. During this task, the users can ask questions.

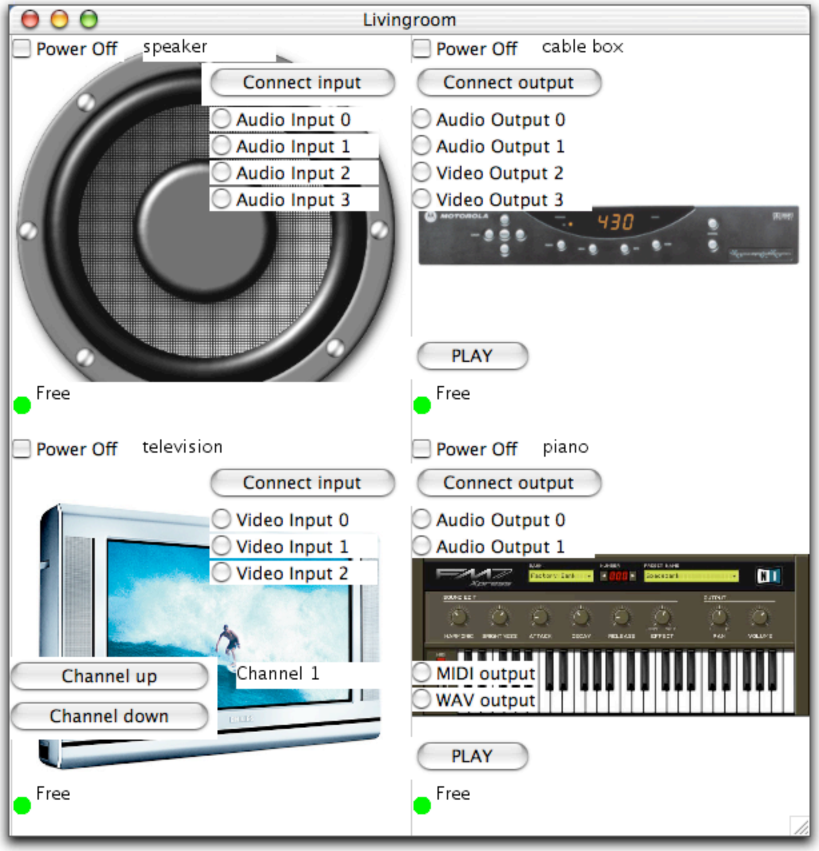


Figure 5-1: Scenario one, play the piano and watch television

For the second scenario, the user was presented with a television, a set of speakers, a CD player and a DVD player (Figure 5-2). This scenario consists of two tasks. The first one involves playing a music CD using the DVD player. The second involves playing a CD and a DVD at the same time. In the second part, since the DVD player is busy playing a CD, the user has to change the configuration from the first part. This causes the user to perform a chain of steps. If the user makes a mistake performing any of the steps the participant has to review the configuration until the goal is accomplished.

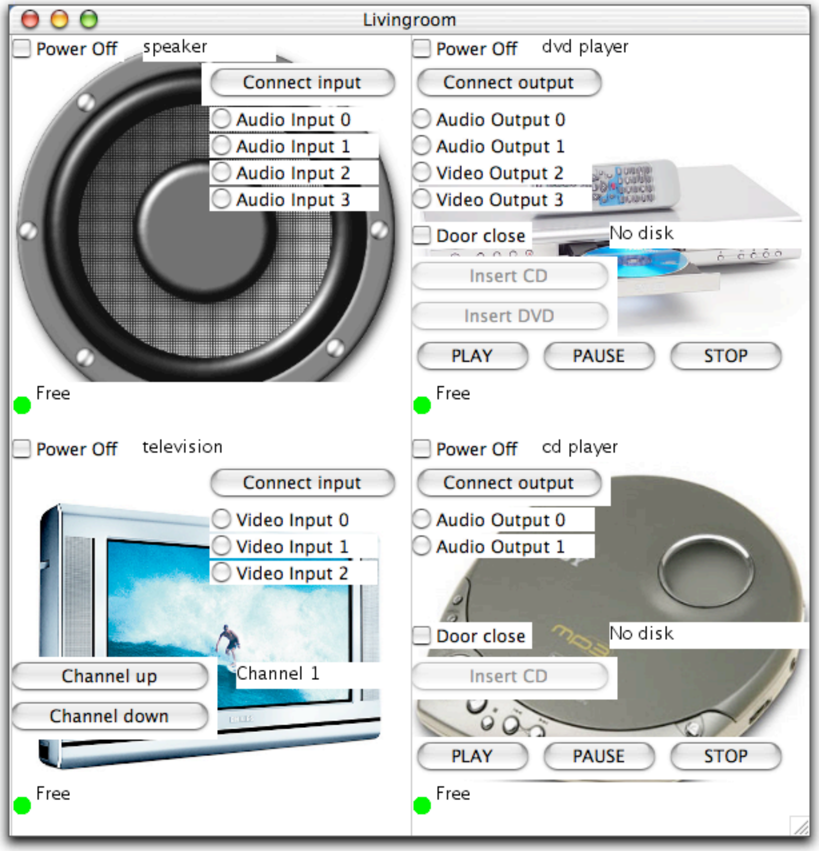


Figure 5-2: Scenario two, play a music CD and a movie DVD

The third scenario involves recording the piano. In this scenario the piano can be connected to just one device at the time, and in order to record and hear the music an amplifier is needed (See Figure 5-3). The users were not given any clear signal of how the devices work. In order to successfully perform this task the participants need to debug the devices until he or she figures out how they work.

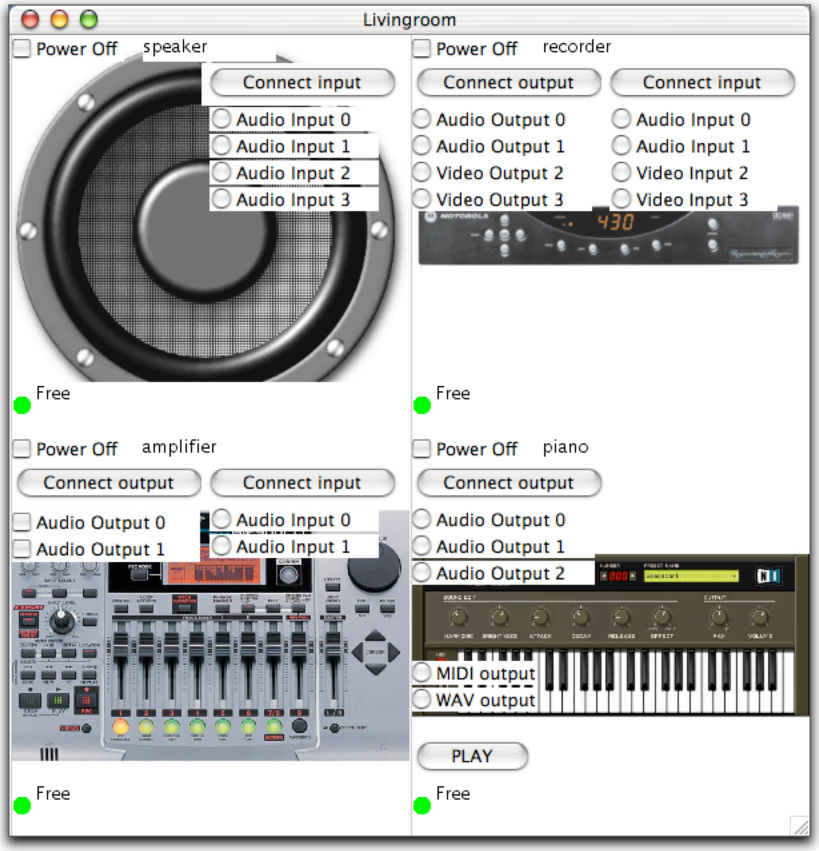


Figure 5-3: Scenario three, record the piano

The forth and last scenario consists of playing a movie DVD, the interface can be seen in Figure 5-4. This scenario is similar to the playing a CD on the DVD; its goal is to get within-subjects data.

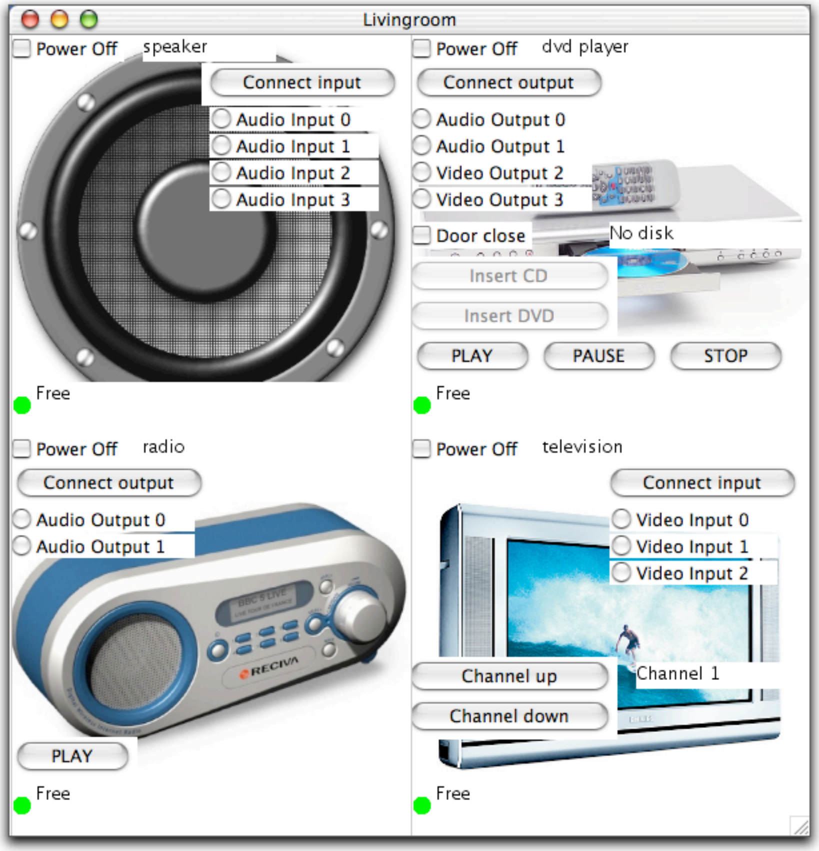


Figure 5-4: Scenario four, watch a DVD

5.2.2 Experimental Design

Before the study begins, users answer a questionnaire with information corresponding to their experience with consumer electronics. They then were handed written instructions about how to perform the low level operations of the devices; these instructions are presented in Appendix A. The subjects were randomly divided in two experimental groups.

5.2.3 Experiment Protocol

5.2.3.1 Condition one

1. Fill out the consent form.

2. Fill out the Pre-Test questionnaire.
3. Read the devices' instructions.
4. Perform the training scenario with ROADIE turned off.
5. Perform the second scenario with ROADIE turned off. A Pre-Task and Post-Task questionnaire are filled out at the beginning and end of the first part ("play the CD on the DVD player") and the second part ("play the CD and the DVD at the same time")
6. Perform the third scenario with ROADIE turned off. A Pre-Task and Post-Task questionnaire are filled out at the beginning and end of the task.
7. Fill the Post-Test questionnaire.
8. Perform the training scenario with ROADIE turned on.
9. Perform the forth scenario with ROADIE turned on. A Pre-Task and Post-Task questionnaire are filled out at the beginning and end of the task.

5.2.3.2 Condition two

1. Fill out the consent form.
2. Fill out the Pre-Test questionnaire.
3. Read the devices' instructions.
4. Perform the training scenario with ROADIE turned off.
5. Perform the training scenario with ROADIE turned on.
6. Perform the second scenario with ROADIE turned on. A Pre-Task and Post-Task questionnaire are filled out at the beginning and end of the first part (play the CD on the DVD player) and the second part (play the CD and the DVD at the same time)

7. Perform the third scenario with ROADIE turned on. A Pre-Task and Post-Task questionnaire are filled out at the beginning and end of the task.
8. Fill the Post-Test questionnaire.
9. Perform the forth scenario with ROADIE turned off. A Pre-Task and Post-Task questionnaire were filled at the beginning and end of the task.

5.2.4 Number of Participants Involved

The test included 12 participants, six assigned to each experimental condition

5.2.5 Method of Recruitment

Participants were recruited using fliers on MIT campus.

5.2.6 Length of Participant involvement

Participants participated between 30 and 60 minutes.

5.2.7 Location of the Research

The experiment was conduced in an office in the Media Lab, using a computer with the ROADIE agent.

5.2.8 Procedures for Obtaining Informed Consent

Prior to start the experiment, participants signed a paper consent form stating that they were aware of the circumstances of the study, including the recording of their interactions with the application.

5.3 Study Results

This study find that the participants finished the task faster and with fewer clicks with ROADIE turned on that with ROADIE turned off in both the between-subjects and within-subjects experiments. Also, when the users performed a task with the agent turned on, they judged the task to be easier, than they did with the agent turned off. Furthermore, when the devices were on, the participants found the interface more intuitive and helpful. Due the small sample size it is impossible to find a statistical significance of any of these results.

Despite these encouraging results, the participants did not state that they understood the interface better, or that they wanted to have it at home. Using a simulation rather than physical devices was distracting in many respects, and users had trouble separating artifacts of the simulation from conceptual aspects of a ROADIE-like interface. We interviewed the participants who gave ROADIE a low rating to find the problem and they answered that the interface made them to read too much text to accomplish simple tasks. We suggest a solution in section 6.2.2.3.

5.3.1 Pre-Test Answers

	Condition one	Condition two
Male	1	1
Female	5	5

Table 5-1: Gender

Table 5-1 shows the gender of the participants. Note that the number of females to males is 5 to 1. This is consistent with the finding that women feel more frustrated with consumer electronics, since the industry does not pay specific attention to them and their preferences, despite figures that show that they are spending more money in high tech devices than men do [MSNBC 2004, Whitesel 2004].

The participants had an average age of 26.5 years for condition one and 26.6 years for condition two.

	Condition one	Condition two
Television	5	5
VCR	4	4
DVD Player	4	4
DVR	0	1
Electronic Musical Instrument	3	1
CD Player	6	6
MP3 Player	3	4
Radio	6	6
Other	4	2
Mean number of devices per subject	5.5	5.6

Table 5-2: Which consumer electronics do you have at home?

	Code	Condition one	Condition two
Less than 15 minutes	1	3	5
Between 15 minutes and half an hour	2	3	1
Between half an hour and one hour	3	0	0
More than one hour	4	0	0
	Media	1.5	1
	Range	1	1

Table 5-3: How much time per week do you spend configuring your consumer electronic devices?

	Code	Condition one	Condition two
Less than an hour	1	1	0
Between one hour and three hours	2	0	1
Between three hours and six hours	3	1	2
More than six hours	4	4	3
	Media	4	3.5
	Range	3	2

Table 5-4: How much time per week do you spend using your consumer electronic devices?

	Condition one	Condition two
Read the user manual	2	5
Consult online forums	2	3
Tinker until I find the problem	5	4
Call a friend or an expert for assistance	4	2

Table 5-5: When you have a problem configuring your home appliances, what strategies do you use to solve the problem? (Mark all that apply)

	Code	Condition one	Condition two
0	1	3	0
1	2	1	1
2 or 3	3	2	4
More than 3	4	0	1
	Median	1.5	3
	Range	2	2

Table 5-6: Have you taken any classes involving programming? If so, how many?

Additionally, the participants were asked to recall a time when they had a problem configuring their appliances and something went wrong. We asked them, how did you find out what the problem was? And how did you fix it? Some answers are:

- *Fiddled around until I figured out what the problem was.*

- *During the installation, the device did not work after I hooked it up, so I followed the manual.*
- *My DVD player couldn't turn off the French subroutine. I solved it by tinkering with it.*
- *Taking parts out one by one to see what wasn't properly in place.*
- *Had someone else help me with the problem.*
- *Pushed a lot of buttons until I figured out that the receiver wasn't on.*
- *If didn't do what I wanted it to do, it just didn't work.*

	Code	Condition one	Condition two
Less than 15 minutes	1	2	3
Between 15 minutes and half an hour	2	1	2
Between half an hour and one hour	3	2	1
More than one hour	4	1	0
	Media	2.5	1.5
	Range	3	2

Table 5-7: How long did you spend solving this problem?

5.3.2 Test Results

5.3.2.1 Between-subjects Analysis

		ROADIE ON	ROADIE OFF
Play a CD on the DVD player	Success	6	6
	Failure	0	0
Play a DVD and a CD at the same time	Success	6	5
	Failure	0	1
Record the piano	Success	6	4
	Failure	0	2
Play a DVD on the DVD player	Success	6	6
	Failure	0	0

Table 5-8: Success rates in configuring the devices

		ROADIE ON	ROADIE OFF
Play a CD on the DVD player	Mean	88.33	111.50
	Std. Dev	37.49	90.29
Play a DVD and a CD at the same time	Mean	171.33	179.83
	Std. Dev	99.11	114.47
Record the piano	Mean	202.17	444.00
	Std. Dev	194.03	239.96
Play a DVD on the DVD player	Mean	54.67	77.17
	Std. Dev	26.40	57.92

Table 5-9: Time in seconds to configuring the devices

		ROADIE ON	ROADIE OFF
Play a CD on the DVD player	Mean	10.33	13.67
	Std. Dev	4.72	7.23
Play a DVD and a CD at the same time	Mean	19.00	32.83
	Std. Dev	8.39	9.52
Record the piano	Mean	23.00	59.33
	Std. Dev	21.81	15.37
Play a DVD on the DVD player	Mean	7.33	15.83
	Std. Dev	3.83	8.86

Table 5-10: Number of steps involved in configuring the devices

		ROADIE ON
Play a CD on the DVD player	Mean	15.83
	Std. Dev	8.75
Play a DVD and a CD at the same time	Mean	37.80
	Std. Dev	62.89
Record the piano	Mean	16.33
	Std. Dev	11.18
Play a DVD on the DVD player	Mean	15.17
	Std. Dev	8.42

Table 5-11: Time before ROADIE suggests the desired function

		ROADIE ON
Play a CD on the DVD player	Mean	1.17
	Std. Dev	0.41
Play a DVD and a CD at the same time	Mean	1.40
	Std. Dev	0.89
Record the piano	Mean	1.17
	Std. Dev	0.41
Play a DVD on the DVD player	Mean	1.33
	Std. Dev	0.82

Table 5-12: Number of events before ROADIE suggests the desired function

Some participants took ROADIE's planner's output as a set of instructions of the steps to follow, and not as an explanation of the system's intentions. This led to them overlooking the opportunity to have ROADIE perform tasks automatically. Perhaps we should have set their expectations in advance to head off this phenomenon.

For this reason, we analyzed when the system suggested the desired function. This does not represent the time it will take the users to complete the task, but rather the time when the system starts to provide useful information to the users.

	ROADIE ON	ROADIE OFF
Play a CD on the DVD player	1.33	-0.17
Playing a DVD and a CD at the same time	-0.50	-0.33
Record the piano	0.50	-1.17
Play a DVD on the DVD player	0.00	-0.67

Table 5-13: It is easy to configure the devices to perform this task. (Negative numbers represent and increase of the perceived difficult)

At the beginning and at the end of each task the participants were asked to quantify the easiness of the task in a scale from –3 to 3. The answers of the pre-task and the post-task were subtracted to measure the how their perception of the difficulty of the task changed before and after it was performed. The results are presented in Table 5-13.

	ROADIE ON	ROADIE OFF
Play a CD on the DVD player	1.67	0.50
Playing a DVD and a CD at the same time	1.17	0.33
Record the piano	0.17	-1.50
Play a DVD on the DVD player	2.83	0.50

Table 5-14: The interaction with the devices was intuitive. (-3 Strongly disagree, 3 completely agree)

	ROADIE ON	ROADIE OFF
Play a CD on the DVD player	0.67	1.00
Playing a DVD and a CD at the same time	0.33	0.17
Record the piano	1.17	-1.33
Play a DVD on the DVD player	2.67	-0.67

Table 5-15: The interface helps to foreseen any problem that I might have. (-3 Strongly disagree, 3 completely agree)

5.3.2.2 Within-subject Analysis

For this analysis just the tasks “Play a CD” and “Watch a DVD”

	ROADIE ON	ROADIE OFF
Mean	71.50	94.33
Standard Deviation	35.56	74.51

Table 5-16: Time to “Play the CD” or “Watch the DVD” for the within subjects analysis

	ROADIE ON	ROADIE OFF
Mean	8.83	14.75
Standard Deviation	4.39	7.79

Table 5-17: Number of events to "Play the CD" or "Watch the DVD" for the within subjects analysis

	ROADIE ON	ROADIE OFF
Easy of use	0.67	-0.42
Intuitiveness	2.25	0.50
Helpfulness	1.67	0.16

Table 5-18: Easy of use, Intuitiveness and Usefulness for the within subjects analysis

5.3.3 Post-Test Answers

The Post-Test questionnaire was applied after configuring the devices to record the piano. At this point the users have just been exposed to one condition, having ROADIE on or off. The data in this questionnaire reflects their impression of the system.

	ROADIE ON	ROADIE OFF
Play a CD on the DVD player	0	0
Playing a DVD and a CD at the same time	2	1
Record the piano	4	5

Table 5-19: What task do you think was the most difficult?

	ROADIE ON	ROADIE OFF
Average	0.67	0.50
Standard Deviation	1.21	1.76

Table 5-20: I felt like I understand how to use these devices effectively. Positive three represents Completely agree and negative three strongly disagree.

	ROADIE ON	ROADIE OFF
Average	0.67	0.33
Standard Deviation	2.42	0.82

Table 5-21: I like to have an interface like this for interaction with my home appliances. Positive three represents Completely agree and negative three strongly disagree

The users were asked to rate the understandability of the interface by rating it between –3 (low understandability) and +3 (high understandability), the compiled answers are shown in Table 5-20. In addition, they were asked if they would like to have the interface at home by rating it from –3 (I do not like the interface) and +3 (I like the interface), the compiled answers are shown in Table 5-21.

Chapter 6

Discussion

6.1 Related Work

Our discussion of related work will fall into four categories. First, we look at the few projects that have directly tried to tackle the problem of simplifying consumer electronics interfaces and making them effective for the problems we are considering, such as planning complex actions, making device behavior context-sensitive, and debugging. Next, we consider related work regarding some of the particular AI interface techniques used by ROADIE, namely mixed-initiative interfaces, goal-oriented and Commonsense interfaces, and self-explanatory interfaces.

6.1.1 Interfaces to Consumer Electronics Devices

To overcome the current usability crisis, researchers have proposed constructing Ambient Intelligent environments "that are sensitive and responsive to the presence of people" [de Ruyter 2004]. This approach classifies user scenarios in terms of *rituals* and *routines*. A ritual is something that the user values as a meaningful experience while routines are mundane tasks in everyday life. This distinction varies among individuals [de Ruyter

2005]. One of the goals of the Ambient Intelligence approach is to magnify the pleasure in the rituals and minimize the interaction in the routines.

De Ruyter created a context-aware remote control where the state can be changed in response to the input of home sensors. In order to make the end user to feel in control of the context-aware remote control, the user can modify its look-and-feel and contextual rules using special tools. His work acknowledges that it is hard for the end user to significantly decrease the amount of programming in the devices, and recognizes that a new programming metaphor needs to be developed [de Ruyter 2005].

However, de Ruyter does not propose any fundamentally new approaches either to controlling individual devices, or to programming behaviors for sets of devices. There is no provision for expressing high-level goals that are not directly reflected in concrete device operations, nor any provision for planning or debugging, both strengths of ROADIE. ROADIE also has rules, whose sources is EventNet, but whose representation is natural language. It is easy to imagine expanding the current ROADIE functionality to allow correction of the rules while the system is in use.

The subfield of Model-Based Interface Design seeks to develop declarative descriptions of interfaces that can be automatically adapted or reconfigured for different platforms. MOBI-D is a general framework to map abstract to concrete actions. This system is able to map the user's specifications for data needs to user interfaces [Puerta 1998].

Kühme exposes a technique called *adaptive action prompter* interface where the tools that are shown change based on the user's previous actions. This system uses modes and recency of use to show the user the most relevant tasks in a contextual menu [Kühme

1993]. While much work in this area concerns desktop computers, some work on adaptive interfaces has taken phones and consumer electronics devices as a target domain.

One popular approach is to build universal remote controls, able to control every single device [Zimmermann 2002]. This is done by downloading the specification from the appliance to the Personal Universal Control (PUC), and it will render a suitable interface. The main goal of this research is to generate a general framework where the current interfaces are mapped to portable computers or cell phones [Nichols 2002]. Having these dynamic interfaces allows hiding or graying-out functions that are not available in a given context.

An approach to simplifying the internal operation of device networks is to model each device as an autonomous agent and use multi-agent communications and negotiation techniques to build an autonomous home [Lesser 1999]. These models try to predict the user's actions and intentions [Cook 2003]. The main problem with this approach is that, because of their statistical approach, they cannot help the users when they are trying to perform a novel action, or when some agent of the system is not functioning as designed. They are also better suited to device-to-device communication rather than direct interaction with the user.

The MIT House_n project built a home equipped with multiple sensors. Volunteers live in this house for periods of time while the house collects real usage data [Intille 2005]. This research aims to build houses that help their habitants to live long and healthy lives, reduce energy consumption, and integrate learning into their everyday activities in the home [Intille 2002]. Similar experimental house-of-the-future projects also exist at Georgia Tech, Microsoft, Phillips, and elsewhere. These houses contain appliances such as washing machines and microwave ovens that could be targets for our approach. These projects emphasize sensor technology rather than explicit user input. We have already explored kitchen applications in the section describing our work with Jackie Lee on KitchenSense.

Another approach to simplifying the user interface interaction is to use voice commands instead of controls. PRECISE is able to reliably translate *semantically tractable* sentences to SQL queries [Popescu 2003]. This approach has been applied to generate PDDL statements to control consumer electronic devices [Yates 2003]. This allows users to correctly control devices with direct verbal orders like "Increase the temperature 5 degrees." This requires the user and the system to share the same vocabulary for talking about the concrete device operations. ROADIE will be able to expand this approach to have more open-ended sentences like "It is too cold here" and infer that the user wants to increase the temperature. Both approaches can be merged by reformulating EventNet nodes to a format that complies with PRECISE's semantic tractability requirements.

6.1.2 Mixed-Initiative Interfaces

Collaborative or *mixed-initiative* interfaces are software agents that cooperate with the user to satisfy certain goals. The outstanding system within this paradigm is Collagen. This system uses SharedPlan [Grosz 1990] as a conversational model between the user and the interface, which enables it to effectively keep track of its interactions with the user. In addition, the agent interacts with the user interface in the same way a real user does, giving the user feedback concerning the operations performed by the agent [Rich 1998].

The Collagen architecture has been applied to consumer electronics [Rich 2005], in a system called DiamondHelp. This approach reduces the configuration overhead when the user knows what function of the device to use. DiamondHelp is a help system that explains procedures that accomplish high-level tasks to the user in terms of their concrete device operations.

Collagen works by having two avatars, one representing an agent and the other the user. Both agents can communicate by directly manipulating a shared user interface. This leads to an interactive paradigm where both are able to observe the actions and comment on them. In addition to direct manipulation, they can use their avatars to communicate to each other, using scripted dialogs using the SharedPlan theory. The user can communicate with the agent by clicking one of the proposed alternatives. In order to work, this approach needs both the agent and the user to have mutual beliefs; if there is a substantial belief discrepancy the SharedPlan scripted dialogs might not make sense. To solve this problem DeKoven [DeKoven 2004] states that for the successful implementation of a collaborative interface, it needs to express what it believes the users' goals are and what the steps are to accomplishing the goal.

ROADIE follows DeKoven's suggestions by presenting the user with an explanation of what it wants to do and the set of relevant EventNet nodes. These pieces of information shown the current system goals and beliefs.

One of the current problems with Collagen is its inability to deal with natural language, although exploratory work has been done [Sidner 2004]. ROADIE is able to enable the user to implement a collaborative approach where the user can state his goals in plain English, and then the system proposes a set of goals.

ROADIE guesses the user's goal and then configures the devices. And then, if a problem occurs the system provides information to find and solve the problem. Collagen takes a different approach by modeling the interaction between the user and the devices as a dialog. During the conversation the user and the agent agree on the function to use and try to reach the goal and in a cooperative way even when problems appear.

6.1.3 Goal-oriented and Commonsense Interfaces

An early application that used commonsense computing to infer the user's goals is Hugo Liu's Goose, a goal-oriented search engine. This search engine goes one step beyond current keyword matching of current search engines by reformulating user's queries to satisfy their goal. It is able to reformulate the query "*my cat is sick*" to *"veterinarians"* [Liu 2002].

Furthermore, commonsense computing has been used to create a proactive calendar that suggests To-Do tasks based on the user's events. When the user adds a new appointment of event into the calendar, the application asks EventNet for the temporal related nodes. The answered nodes are filtered by the application, picking the ones that have patterns used in a To-Do list, the selected phrases are shown to the user. This application was designed to help the user remember common steps and prerequisites needed to perform an action. For example, if the user adds the event "Go to the beach" the system will suggest bringing a beach ball and sunscreen [Espinosa 2005].

Other examples of commonsense interfaces can be found in [Lieberman, Liu, Singh, Barry 2005]. Many of these interfaces share ROADIE's approach of using

Commonsense to infer goals from concrete actions. ROADIE is a goal-oriented interface in the same way than Goose is; it maps the user's goals to actions in a format and operations that the device can understand.

A variety of applications have used AI planners interactively. A planner has been used for controlling the configuration and operation of a disk management on the HPUX system [Barruffi 1997], scheduling in a manufacturing line [Harvey 2001], in air traffic control [Hatzack 2001], and in scheduling for a call center [Fukunaga 2002]. Writer Aid is a collaborative writing assistant that helps to identify and insert relevant papers from online resources. This application uses a planner to identify both a relevant bibliography and the action to perform [Babaian 2002]. Puerta's MOBI-D, discussed earlier, also uses a planner to plan interface configurations. All these systems, along with ROADIE, use planners to gain flexibility in the ways they can control a certain machine. The planners allow them to successfully accomplish unanticipated scenarios if they are possible.

6.1.4 Self-explanatory Interfaces

The ability of systems to introspect their state and change it is called *reflection*. This ability means that a system is able to change its own behavior in order to satisfy its goals [Dourish 1995]. This ability is important in computer systems since it allows the system to reason about its state when something does not go as expected. Furthermore, in case the system is not able to solve the problem, it can provide a good insight of what went wrong, possible causes, and what can be done to solve the problem.

Although reflection in programming languages have been just been implemented in a handful of *no mainstream* languages [see references in Maes 1987] some popular languages allow introspection. They can know the capabilities of the objects and variables at running time [O'Brien 2002]. Having programming languages with built-in support of introspection and reflection will help to build programs and applications using this programming paradigm.

The most significant system using this approach is EXPECT, a knowledge acquisition and reasoning tool [Blythe 2001]. This system has the ability to infer which pieces of knowledge are required, which are necessary to perform certain reasoning, and provide an explanation of why. For example: if the system is trying to ship merchandise to Los Angeles it will ask for the airports and seaports in this city and explain that this information is needed as an input of the desired inference operation [Gil 1994].

Woodstein is a debugging interface for web processes like purchases. It provides reflection by allowing the user to go back to the webpage where an action occurred and introspect if the data shows it is correct or not. To introspect the pieces of information, the user can ask further information about why and how something happened, and it can also tag the data as successful or unsuccessful [Wagner 2003].

Reflection has also been applied to programming environments. ZStep 95 [Lieberman 1997] is a programming environment that keeps a memory of all the states of the program's execution. This history allows, while debugging, not only a full history of the execution of the code that can be stepped forward and backward, but it also answers questions like "When did the variable have that value?" or "Where in the code was that expression evaluated?"

ROADIE provides introspection since it is able to change the configuration of its devices to satisfy the user's goals. In addition, it adds introspection to its internal beliefs by providing an explanation of why a certain action is suggested.

6.2 Future work

6.2.1 Integration with SMPL

The SMPL (Simple Media Protocol Layer) [Rocha 2005] is a core technology for decentralization of devices to their simpler parts. A telephone is nothing more than a microphone, a speaker, and a data link. SMPL will allow scenarios where a phone call can be emulated using an MP3 player speaker, a Bluetooth microphone, and a WiFi connection, even if each of these parts were not designed to be part of a telephone.

This approach promises to push the Internet to the next level, where people will interact not with machines, but with services. Unfortunately this will not come for free: the SMPL architecture will be so complex that the average user will be unable to unlock all its power. Fortunately, the SMPL modular architecture can be easily enriched if it is used in conjunction with ROADIE.

6.2.2 Expanding ROADIE Capabilities

6.2.2.1 Learning form User's Habits

Learning from user's habits can be done in two ways. First, we can raise the weight of the links when goals are chosen and lower them when they are not chosen. Also, since ROADIE can show the output of EventNet temporal traces, the user should be able to mark the output links that are incorrect.

6.2.2.2 Allowing the User set Custom Goals

ROADIE's current use is to map the whole possibility of user's activities to a small set of prefixed goals. This approach has a fundamental flaw, as the system designer should program in advance the available goals or the goals that have to be added from *somewhere else*.

This limited scenario can be expanded by adding Programming by Example [Lieberman 2001] techniques. Here, the user can train the system to perform certain tasks, then these tasks can be automatically suggested using the techniques presented in this thesis.

6.2.2.3 Improving ROADIE user interface

One of the main problems with ROADIE is that it heavily depends on natural language to communicate with the user. This characteristic makes the system very intrusive when the users wants to perform simple tasks that they already know how to do and the system behaves as expected. On the other hand, when problems arise it is good to have a rich interactive communication channel.

This can be improved by changing the ROADIE interaction schema. First, the system should treat the tasks the user should perform and the ones that the system can perform differently. Then, if the suggested goal does not need user interaction, it should be presented in a different way than if it needs the user's attention. Furthermore, the system can assign icons to the more frequent operations, heavily reducing the amount of text displayed by the system. This brings the problem of how to easily and intuitively assign icons to similar tasks like playing a CD on the CD player or on the DVD player.

Then, when problems arise – an operation fails, the user complains about certain behavior of the system, etc – ROADIE will show its current interface.

6.3 Conclusions

6.3.1 Implications of Goal-oriented Interfaces

Raskin [Raskin 2000] argues we should rethink the computer interfaces as a small set of always accessible core operations and then build more complex operations around this small set. His approach, while appealing, has two fundamental limitations. First, there is no one-size-fits-all method for selecting those small core operations. So the user ends having a huge set of core capabilities, and then wondering which capability he wants and how to access it. Second, it is easy to map a simple goal to simple actions – Control-B transforms the selected text to bold – but the task of mapping from the goal "emphasize these ideas" to "make the text bold" is still left to the user.

With a goal-oriented interface, like ROADIE, this problem gets blurred. Since the system can map high-level goals to low level actions, there are any constraints to keep the core functions small.

6.3.2 Towards Simplicity

It is known that having few tools is easier to manage them than having hundreds of them. But if our current set of tools cannot perform a certain desired task, we create a new specialized tool. This means that there is a natural tendency to add more and more tools increasing the features, but decreasing the usability.

There is another solution to the problem. If we can design just a few powerful and easy to use tools – like a Swiss Army knife – successfully creating these tools will lead to full-featured, easy to use devices. The MIT Media Laboratory Simplicity [Maeda 2004]

Initiative looks for these powerful tools and organizes them to increase functionality without decreasing usability.

ROADIE looks for simplicity by organizing the functions of the devices not around their basic operations, but around the goals they help the user to reach.

6.3.3 A Plan Recognizer with Commonsense vs. other Plan Recognizing Approaches

Statistical or logical plan recognizers can be easily used to mimic the Basic EventNet operations Statistical plan recognizers can be trained on a corpus of associations between sequences of actions and statements of goals, and build up correlations incrementally. Logical plan recognizers deduce correspondences between actions and goals from first-principles axiomatizations of specific knowledge domains.

The first advantage of using EventNet is that the systems built on top of it are able to provide the user with the reason for the suggestion, unlike statistical approaches, and allow the user to correct the system's knowledge base. Second, EventNet can work with more open-ended scenarios than is typically the case with logical recognizers.

6.3.4 Should Technology be Invisible or Transparent?

Norman [Norman 1999] states that in order to make technology easier to use, it is necessary to "hide" the computer. The analogy that he uses is the electric motors, they are ubiquitous in modern appliances; almost all modern appliances have at least one, even computers.

We agree that people do not want to know unnecessary details about computer algorithms or device architectures, simply in order to perform tasks like sending an email or watching television, just as people do not want to know about electric motors to grind their coffee in the morning or to listen an audio type… until the devices break.

With current devices, when they break, people call technical support or other experts. If your coffee grinder does not work, you ask an expert to fix it. It is not feasible for the end user to perform diagnostic or repair procedures. This model is due to the simplicity of states available on these devices; if there is electricity and the grinder is connected, I can assure you that you need to call the technician.

Unfortunately, this clean approach is not realistic with complex devices. If you cannot watch a movie on the DVD player, there are a number of things that can go wrong making impossible to know if something is broken or it is a configuration problem.

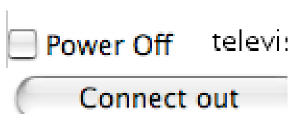
The challenge we face in make consumer electronics easy to use is *not* to completely hide details of their operation, but to present only those details which the user needs to know, when they need to know them, and in such a way that the average user can easily understand them and get done what they want to get done.

Appendix A

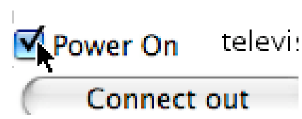
How to Operate the Devices

How to turn on or off the devices

To turn on or off the devices click on the check box that says, “Power Off"/Power On”

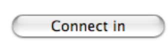


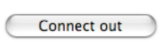
Device turn off



Device turn on

How to connect two devices

To connect two devices you need to click the “Connect in” 

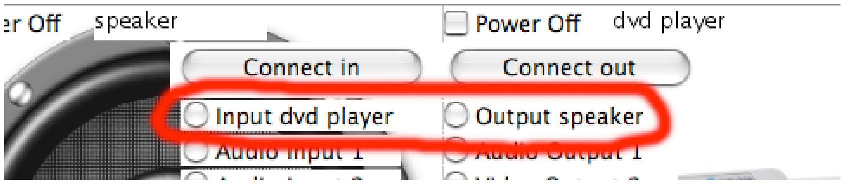
and the “Connect out” 

button.

When you click the “Connect in” or “Connect out” button the text will turn blue showing that the device is selected for connection.

This action simulates physically connecting the devices using cables.

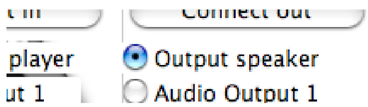
To know what devices are connected look the names of the devices in the radio buttons of the device.



The DVD player and the speaker are connected

How to select the input and output of the devices

To select the input or output of the device select the radio button with the name of the device you want to connect.



The output of this device to the speaker is selected

How to open or close the CD or DVD door

To open or close the CD or DVD door click on the check box that says, "Door close"/"Door open."

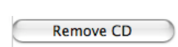
DVD player with the door close

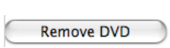
DVD player with the door open

How to insert or remove CD or DVD disks

To insert or remove the CD or DVD disk click on the buttons that says "Insert CD," 

"Insert DVD," 

"Remove CD," 

or "Remove DVD" 

buttons.

How to know if the device is in use or not

If the device is not being use it shows a green point and the word free at the button of it.

To use a device, first configure the device and then click the "play" 

button. If the device is configured correctly you will here a congratulations message.

The radio in not in use

If the device is being use it shows a red point and a legend saying what is it doing at the button of it.

The radio in use

Appendix B

Information Provided by the System for Playing a CD

In this appendix we shown the help information provided by the system when the button "Tell me more" is pressed when the user is trying to play a CD. This information has the goal to help the user to solve unexpected problems.

- ## Connect the devices

Why is this important?

- Connecting the devices allows you to move the audio signal from one device to another

How can I connect the devices?

- Be sure that you have the right cable type for your connector
- Be sure that you plugged the input into the input jack and the output into the output jack

What happen if I do not connect the devices?

- If you cannot see the device name on the selector of the other device, the devices are not connected
- If the devices appear to be configured correctly but you cannot hear or see anything make sure that the connections are correct

How will I know if the devices have been connected?

- The device name will be on the selector of the other device

What can go wrong?

- Make sure that you connect the input into the input jack and the output into the output jack

- **Turn on the device**

Why is this important?

- Power gives the device the electricity necessary to work

How can I turn on the device?

- Pushing the power button turns the device on

What happen if I do not turn on the device?

- You cannot use the device
- Other devices cannot access the device

How will I know if the device has been turn on?

- You can see a green light on near the "power" label

What can go wrong?

- Check that the device is plugged in

- **Select the input**

Why is this important?

- Selecting the input to the device the audio signal come from

How can I select the input?

- Click the selector with the name of the desired device

What happen if I do not select the input?

- The indication light of the desired device will be selected

How will I know if this has been done?

- If the indication light of the desired device will be selected

What can go wrong?

- Make sure that the connection of the devices is correct

- **Select the output**

Why is this important?

- Select the output to the device the audio or video signal has to go

How can I select the output?

- Click the selector with the name of the desired device

What happen if I do not select the output?

- If you hear or a signal from the wrong place, it is very likely that you have the wrong selection

How will I know if this has been done?

- The indication light of the desired device will be selected

What can go wrong?

- Make sure that the connection of the devices is correct

- **Open the disk player door**

Why is this important?

- Opening the disk door is necessary to introduce the disk

How can I do open the disk player door?

- Push the open/close device button

What can go wrong?

- Some devices need to be turned on in order to open the door

- Insert the music CD

How can I insert the music CD?

- Insert the music CD in the open dock

What happen if I do not insert the CD?

- If the play button is pressed but you cannot hear anything, there is probably no disk inserted

What can go wrong?

- Make sure that the disk is inserted into the selected input device

- **Close the disk player door**

Why is this important?

- The disk case should be closed in order to use it

How can I close the disk player door?

- Push the open/close device button

What can go wrong?

- Some devices need to be turned on in order to open the door.

References

[Babaian 2002] Tamara Babaian, Barbara J. Grosz, Stuart M. Shieber. A writer's collaborative assistant. *Proceedings of the 7th International conference on Intelligent User Interfaces*, pages 7-14, ACM Press, 2002

[Barruffi 1997] Rosy Barruffi, E. Lemma, Michela Milano, P. Mello. Application of Planning Techniques for System Configuration Tasks. *In Proceedings if the 4th Workshop of the OpenView University Association, OVUA'97*, Madrid, Spain, April 2-4, 1997.

[Blum 1997] Avrim Blum, Merrick Furst. Fast Planning Through Planning Graph Analysis. Proc. Of the 14th International Joint Conference on Artificial Intelligence (IJCAI 1997), Montreal, Canada, pages 1636-1642, 1997.

[Blythe 2001] Jim Blythe, Jihie Kim, Surya Ramachandra and Yolanda Gil. An Integrated Environment for Knowledge Acquisition. *Proceedings of the 6th International Conference on Intelligent User Interfaces*, pages 13-20, ACM Press, 2001

[Brouwer-Janse 1992] Maddy D. Brouwer-Janse, Raymond W. Bennett, Takaya Endo, Floris L. van Nes, Hugo J. Strubbe, Donald Gentner. Interfaces for consumer products: "How to camouflage the computer?" *In Proceedings of CHI conference on Human factors in computing systems*, pages 187-190, ACM Press, 1992.

[Cohen 2000] William Cohen. WHIRL: A word-based information representation language. *Journal of Artificial Intelligence*, pages 163-169, 2000.

[Cook 2003] Diane J. Cook, Michael Youngblood, Edwin O. Heierman, Karthik Gopalratnam, Sira Rao, Aandrey Litvin, Farhan Khawaja. “MavHome: An Agent-Based Smart Home”, *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, pages 521-524, IEEE press, 2003.

[de Ruyter 2004] Boris de Ruyter, Emile Aarts. Ambient Intelligence: visualizing the future. *Proceedings of the working conference on Advance visual interfaces*, pages 203-208, ACM Press, 2004.

[de Ruyter 2005] Boris de Ruyter, Richard van de Sluis. Challenges for End-User Development in Intelligent Environments. In Henry Lieberman, Fabio Paterno, Volker Wulf, eds, *End-User Development*, Kluwer Academic Publisher, (to appear).

[DeKoven 2004] Elyon A.M. DeKoven. Help Me Help You: Designing Support for Person-Product Collaboration. PhD. Thesis 2004. Delft University of Technology

[Dourish 1995] Paul Dourish. Developing a reflective model of collaborative systems. *ACM Transactions on Computer-Human Interaction*, pages 40-46, ACM Press, 1996.

[Espinosa 2005] Jose Espinosa, Henry Lieberman. EventNet: Inferring Temporal Relations Between Commonsense Events. *Proc. Fourth Mexican International Conference on Artificial Intelligence*, Springer Publisher. November 14-18, 2005. Monterrey, Nuevo Leon, MEXICO (to appear)

[Fellbaum 1998] Christiane Fellbaum (Ed.): WordNet: An electronic lexical database. MIT Press, 1998.

[Fukunaga 2002] Alex Fukunaga, Ed Hamilton, Jason Fama, David Andre, Ofer Matan, Illan Nourbakhsh. Staff Scheduling for Inbound Call Center and Customer Contact Centers. *AI Magazine,* 23(4), Winter 2002

[Gil 1994] Yolanda Gil. Knowledge Refinement in a Reflective Architecture. *Proceedings of the Twelfth National Conference of Artificial Intelligence (AAAI-94)*, volume 1, pages 520-526, AAAI, 1994.

[Grosz 1990] B.J. Grosz, S. Kraus. Plans for Discourse. In P.R. Cohen, J.L. Morgan, M.E. Pollack, eds, *Intentions and Communications*, pages 417-444, MIT Press, Cambridge, MA, 1990.

[Harvey 2001] Terrence Harvey, Keith Decker. Planning Alternatives for an Intelligent Scheduler. *IJCAI 2001 AI in Manufacturing Workshop*, 2001.

[Hatzack 2001] Wolfgang Hatzack, Bernhard Nebel. The Operational Traffic Control Problem: Compuational Complexity and Solutions, *in: A. Cesta (ed.), Proceedings of the 6th European Conference on Planning (ECP'01),* 2001.

[Intille 2002] Stephen S. Intille. Designing a Home of the Future. *IEEE Pervasive Computing*, vol. April-June, pages 80-86, 2002.

[Intille 2005] Stephen S. Intille, K. Larson, J.S. Beaudin, J. Nawyn, E. Munguia Tapia, P. Kaushik. A living laboratory for the design and evaluation of ubiquitous computing technologies. *Extended Abstracts of the 2005 Conference on Human Factors in Computing Systems,* New York, NY: ACM Press, 2004.

[Kühme 1993] Thomas Kühme, Uwe Malinowski, James D. Foley. Facilitating interactive tool selection by adaptive prompting. *INTERACT ´93 and CHI 93 conference companion on Human factors in computing systems*, pages 149-150, ACM Press, 1993.

[Lee 2005] Jackie Chia-Hsun Lee. Spatial User Interfaces: Augmenting Human Sensibilities in a Domestic Kitchen. Master Thesis. School of Architecture and Planning, MIT Media Laboratory.

[Lenat 1995] Douglas Lenat. CYC: A Large-Scale Investment in Knowledge Infrastructure. *Communications of the AMC*, pages 33-38, Vol. 38, No. 11, November 1995.

[Lesser 1999] Victor Lesser, Michael Atighetchi, Brett Benyo, Brett Horling, Anita Raja, Regis Vincent, Thomas Wagner, Xuan Ping, Shelley XQ Zhang. The Intelligent Home Testbed. *Proceedings of the Autonomy control Software Workshop (Autonomous Agent Workshop)* January 1999.

[Lieberman 1997] Henry Lieberman, Christopher Fry. Step 95: A reversible, animated source code stepper. *In Software Visualization: Programming as a Multimedia Experience, J. Stasko, J. Domingue, M. Brown, and B. Price, Eds*. MIT Press, 1997.

[Lieberman 2001] Henry Lieberman (Ed.). Your Wish is My Command: Programming By Example. The Morgan Kaufmann Series in Interactive Technologies, 2001

[Lieberman, Liu, Singh, Barry 2005] Henry Lieberman, Hugo Liu, Push Singh, and Barbara Barry. Beating Common Sense into Interactive Applications. *AI Magazine* 25(4): Winter 2005, 63-76.

[Liu 2002] Hugo Liu, Henry Lieberman, Ted Selker. Goose: A Goal-Oriented Search Engine with Commonsense. *Proceedings of the second international conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 253-263, ACM Press, 2002.

[Liu 2004] Hugo Liu, Push Singh. ConceptNet – a practical commonsense reasoning toolkit. *BT Technology Journal*, 22(4): 211-226 2004.

[Maeda 2004] John Maeda. Simplicity. *BT Technology Journal*, 22(4): 285-286 2004.

[Maes 1987] Pattie Maes. Concepts and experiments in computational reflection. *Conference proceedings on Object-oriented programming systems, languages and applications*, pages 147-155, ACM Press, 1987.

[Maes 1989] Pattie Maes. How to Do the Right Thing. *Connection Science Journal, Vol. 1, No. 3., also MIT AI-Memo #1180*, 1989.

[McCarthy 1959] John McCarthy. Programs with Common Sense. *In Mechanization of Thought Process. Proceedings of the Symposium of National Physics Laboratory*, pages 77-84, London, HMOS, 1959.

[Mihalcea 2005] Rada Mihalcea. Personal Communication. Summer 2005

[MSNBC 2004] MSNBC News. Consumer electronics companies woo women. Women spent more on technology then men in 2003. Press note. Available at http://msnbc.msn.com/id/3966261

[Nicholas 2002] Jeffrey Nichols, Brad Myers, Thomas Harris, Roni Rosenfeld, Stefanie Shriver, Michael Higgins, Joseph Hughes. Requirements for Automatically Generating Multi-Modal Interfaces for Complex Appliances. *Fourth IEEE International Conference on Multimodal Interfaces*, pages 377, IEEE Press, 2002.

[Norman 1999] Donald A. Norman. The Invisible Computer: Why Good Products Can Fail, the Personal Computer Is So Complex, and Information Appliances Are the Solution. The MIT Press, reprint edition, 1999.

[Norman 2002] Donald Norman. The Psychology of Everyday Things. Perseus, New York, 2002.

[O'Brien 2002] Patrick O'Brien. Guide to Python Introspection. http://www-128.ibm.com/developerworks/library/l-pyint.html

[Pearl 1998] Pearl, Judea. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. San Mateo, CA: Morgan Kaufman. 1988.

[Popescu 2003] Ana-Maria Popescu, Oren Etzioni, Henry Kautz. Towards a theory of natural language interfaces to databases. *Proceedings of the 8th international Conference on Intelligent user interfaces*, pages 149-157, ACM Press, 2003.

[Puerta 1998] Angle Puerta, Jacod Eisenstein. Towards a general computational framework for model-based interface development system. *Proceedings of the 4th International Conference on Intelligent User Interfaces*, pages 171-178, ACM Press 1998.

[Push 2003] Push Singh, William Williams. LifeNet: A Propositional Model of Ordinary Human Activity. Proc. *Workshop on Distributed and Collaborative Knowledge Capture.* Sanibel Island, FL 2003.

[Raskin 2000] Jef Raskin. The Humane Interface: New Directions for Designing Interactive Systems. *Addison-Wesley Professional*, First Edition, 2000

[Rich 1998] Charles Rich, Candy Sidner. Collagen: A Collaboration Manager Software *Interface Agents. Journal of User Modeling and User-Adapted Interaction*, pages 315-350, 8(3/4), 1998

[Rich 2005] Charles Rich, Candy Sidner, Neal Lesh, Andrew Garland, Shane Booth, Markus Chimani. DiamonHelp: A Graphical User Interface Framework for Human-Computer Collaboration. *IEEE International Conference on Distributed Computing Systems Workshops*, pages 514-519, June 2005.

[Rocha 2005] Carlos Rocha. SMPL: A network architecture for collaborative distributed services. Master Thesis. School of Architecture and Planning, MIT Media Laboratory.

[Shneiderman 1997] Ben Shneiderman. Designing the User Interface. *Addison-Wesley*, Third edition, 1997

[Sinder 2002] Candace L. Sinder, Clifton Forlines. Subset Languages for Conversing with Collaborative Interface Agents. *International Conference on Spoken Language Processing (ICSLP)*, September 2002.

[Singh 2002] Push Singh. The public acquisition of Commonsense knowledge. *Proc. AAAI Spring Symposium on Acquiring (and Using) Linguistic (and World) Knowledge for Information Access*, 2002.

[UPnP 2000] Universal Plug and Play Device Architecture. 2000. Available: http://www.upnp.org/download/UPnPDA10_20000613.htm

[Wagner 2003] Earl Wagner. Woodstein: A Web Interface Agent for Debugging E-Commerce. Master Thesis. School of Architecture and Planning, MIT-Media Lab.

[Whitesel 2004] Terri A. Whitesel. The Electronics Market – Only A Man's World? Interpret-Her 2004. Available at http://www.interpret-her.com/pdf/ACF7D8E.pdf

[Yates 2003] Alexander Yates, Oren Etzioni, Daniel Weld. A reliable natural language interface to household appliances. *Proceedings of the 8th international Conference on Intelligent user interfaces*, pages 189-196, ACM Press, 2003.

[Zimmermann 2002] Gottfried Zimmermann, Gregg Vanderheiden, Al Gilman. Prototype Implementations for a Universal Remote Console Specification. *ACM Special Interest Group on Computer-Human Interaction*, pages 510-511, ACM Press, 2002.