A Goal-Oriented Interface to Consumer Electronics using Planning and Commonsense Reasoning

Henry Lieberman
MIT Media Lab
20 Ames St. E15-384A
Cambridge, MA 02139, USA
+1-617-253-0315
lieber@media.mit.edu

José Espinosa
MIT Media Lab
20 Ames St. E15-383
Cambridge, MA 02139 USA
+617-253-0315
jhe@media.mit.edu

ABSTRACT
We are reaching a crisis with design of user interfaces for consumer electronics. Flashing 12:00 time indicators, push-and-hold buttons, and interminable modes and menus are all symptoms of trying to maintain a one-to-one correspondence between functions and physical controls, which becomes hopeless as the number of capabilities of devices grows. We propose instead to orient interfaces around the *goals* that users have for the use of devices.

We present Roadie, a user interface agent that provides intelligent context-sensitive help and assistance for a network of consumer devices. Roadie uses a Commonsense knowledge base to map between user goals and functions of the devices, and an AI partial-order planner to provide mixed-initiative assistance with executing multi-step procedures and debugging help when things go wrong.

Categories and Subject Descriptors
H.5.2 [**User Interfaces**]: Interaction styles. D.2.2 [**Design Tools and Techniques**]: User Interfaces. I.2 [**Artificial Intelligence**]. J.7 [**Computers in other systems**]: Consumer products.

General Terms
Design, Human Factors.

Keywords
Commonsense Reasoning, planning, consumer electronics, goal-oriented interfaces.

1. THE CRISIS IN CONSUMER ELECTRONICS INTERFACES
Current consumer electronics are getting more and more complicated, threatening to outstrip the competence that can be reasonably expected from their intended users. 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.

IUI'06, January 29–February 1, 2006, Sydney, Australia.

We attribute the growing complexity of consumer electronics interface design to the desire to maintain the one-to-one correspondence between functions and controls that worked well for simpler devices. But as the number of functions of a device grows, controls get overloaded, leading to heavily-moded interfaces, push-and-hold buttons, long and deep menus, and other confusing and error-prone interface elements. The next generation of consumer electronics devices will incorporate processing and networking, making things potentially more complex if we stick to manual operation, but also opening up new possibilities for automating co-operation between multiple devices.

We propose to re-orient the interface around the *goals* of the user, rather than the functions of the device. Something, then, has to map between the user's goals and the concrete functions of the device. We propose to fill this gap with Roadie, an interface that makes use of Commonsense knowledge and a partial-order planner to give the user proactive advice, automate complex tasks, and provide debugging help when things go wrong.

2. USERS NEED HELP WITH MANY SCENARIOS OF USE
It is not only the "normal operation" of the device that users need help with. There are other scenarios associated with consumer devices that users need help with. The advent of powerful computing and communication in devices gives us the potential of providing help with these scenarios, as well as merely invoking functions of the device.

- *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. The system should describe its capabilities and limitations in terms that the user can understand and comprehend.

• *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. Fixing problems sometimes forces the user to introspect about the system's internal state – which might be hidden by the device designer. The interface should help 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.

- *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.

- *I want to do …* 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.

3. INTRODUCING ROADIE

In this paper, we present *Roadie*, a prototype consumer electronics interface oriented towards the needs of non-expert users. The project name comes from the person who is in charge of setting up the audio and video devices during music concert tours.

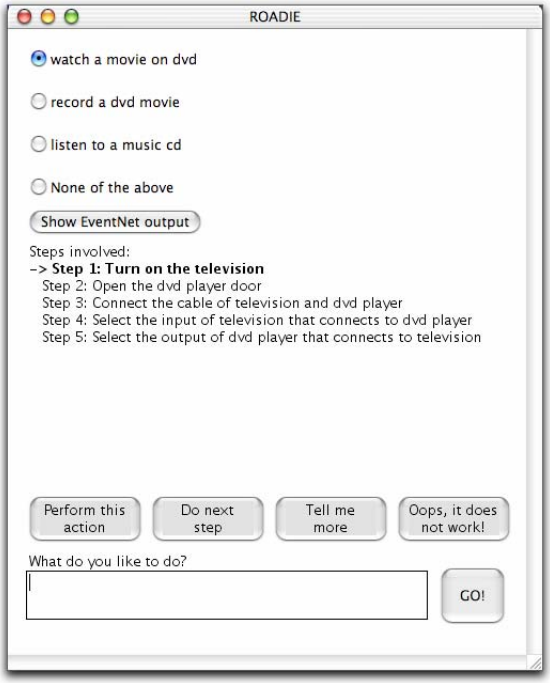

Figure 1: Screen shoot of Roadie's User Interface

Roadie's 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. In the long-term future, it might indeed be advantageous to totally redesign the hardware control interface to each device to be more goal-oriented. But for the moment, it is not our intention to completely replace the conventional button-and-knob interface to each device.

The Roadie interface [see Figure 1] shows dynamic dialog boxes that

- Show steps of a procedure
- Provide controls for executing the procedure
- Provide explanation
- Display alternatives for what to do next
- Provide facilities for help and giving feedback
- Provide a box for unrestricted natural language input for the user to state goals or ask questions.

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.

The user can control the execution of the steps by using the "Perform this action" (do all the steps at once), and the single-step "Do next step" button. The "Tell me more" button provides more detailed explanation of why the steps help accomplish the goal, and the "Oops, it does not work!" button launches a debugging dialog.

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. While we are using some natural language understanding, as explained below, we do not rely on being able to completely understand arbitrary English. We also are anticipating the possibility that we could use speech input for the natural language component.

Roadie's interface is intended to be *fail-soft* – provide intelligent assistance when it is helpful, but not replace conventional push-the-button interaction if that turns out to be more convenient in a given case, or if Roadie does not have the knowledge or language understanding capability to correctly understand and implement the user's intention.

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 devices' manufacturers are aware of this problem and created the UPnP [15] standard. Unfortunately, the manufacturers have not started to build devices that fully comply with this standard. Furthermore, they sometimes do not fully expose to the applications programmer all the necessary controls and states to accomplish a given task. Some manufacturers are interested in implementing sets of branded devices that coordinate using a proprietary protocol 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. Each device is represented in the simulation by a window containing an image of the device, and conventional dialog box "widgets" representing the conventional hardware controls. Where feasible, we simulate the operation of the device

(e.g. an MP3 file plays when you hit the "Play" button on the CD player to show that a simulated CD would be playing; a Quicktime video represents the operation of the television). We represent actions that the user would normally have to perform manually by a dialog box instructing the user to perform that operation. Figure 2 shows an example of the simulated devices used by Roadie.

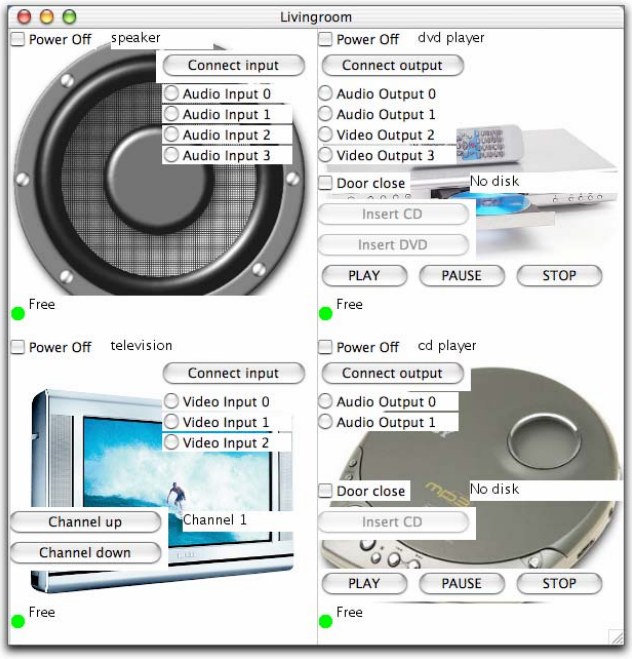

Figure 2: Roadie's Device simulation

4. ROADIE'S INTERNAL ARCHITECTURE

Figure 3 shows a diagram of Roadie's system architecture.

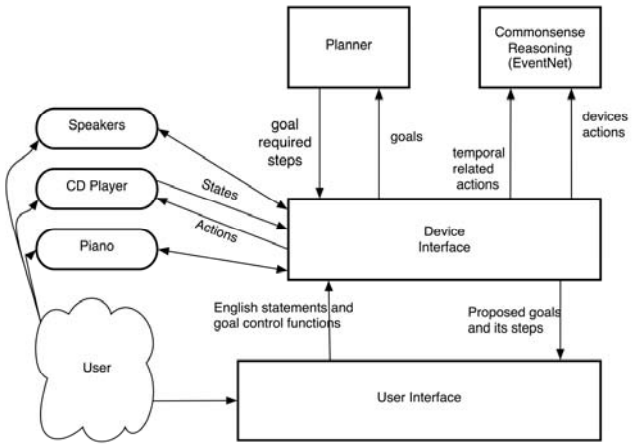

Figure 3: Roadie system architecture

4.1 Commonsense Knowledge and EventNet

To implement Roadie, the devices need to have knowledge about what the motivations, desires and goals of the users are. For getting this knowledge, we created a plan recognizer called EventNet [3]. This plan recognizer uses knowledge mined from the OpenMind Commonsense knowledge base [14], a knowledge base of 770,000 English sentences describing everyday life, contributed by volunteers on the Web. We also use ConceptNet, a semantic network derived from parsing the sentences in OpenMind and applying a spreading activation algorithm. EventNet uses temporal knowledge from this corpus and spreading activation to infer a possible set of antecedent or subsequent actions.

ROADIE uses EventNet to infer the user's goals from his or her actions, and proposes specific device functions that might accomplish the user's goal. This is illustrated below in the scenario.

4.2 User Interaction module, Commonsense plan recognizer

The user interaction module maps the actions, goals and desires of the user to a format that the device controller 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. We also used the Web to automatically collect pairs of device actions linked by temporal relations [Mihalcea, personal communication].

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. It is also responsible for providing an explanation about the behavior and functionality. 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.

The planner is used to infer the set of actions needed to configure the devices to satisfy the user's goal. The planner decomposes the desired states to single actions that the devices can execute and creates alternative actions when something unexpected occurs. Also, 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 standard Graphplan [1] implementation.

In addition, the planner's goals serve as 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. The planner is also responsible for finding the states of the devices that are conflicting with the desired goal.

4.3 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. This module has a text string for each change in state of the device, like "turn on the device," "I insert a music CD." 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 a 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, English templates for each possible planner step are used to create explanations in natural language. The matching between two phrases are made using EventNet's semantic link algorithm.

4.3.1 Debugging Information

We do not assume that action sequences will never fail. Problems inherent to devices - malfunctions or misunderstandings between the user and Roadie - might emerge. Debugging consists of looking for the causes of unexpected results. For each step, we can show why the step is important, how the user can perform the step, what the consequences are of *not* doing this step, what the results are of performing it, and the things that might go wrong while trying to perform the step. If 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.

5. USER SCENARIO

5.1 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."* For this action, EventNet answers: *"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.

Keep in mind that 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.

Then, Roadie tries to match the EventNet answers with a device description. 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.

The set of suggested actions are: *"watch a movie on dvd," "record a dvd movie," and "listen to a music cd."* The user wants to transfer a CD to his home system, so he picks the second choice. This goal needs two parameters: a recorder and a DVD player. The system keeps track of the recency of usage of the devices and asks the planner for a set of actions to accomplish the goal *(record-movie-dvd dvd-player recorder)*

The planner calculates a plan. 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 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 recorder
2. Connect the cable of the recorder and the DVD player
3. Open the DVD player door
4. Select the DVD player output that connects to the speaker
5. Select the speaker input that connects to the DVD player
6. Insert the movie DVD
7. Close the DVD player door

Note that some of these actions can be performed directly by the system, while others (like inserting the DVD) cannot. Roadie shows four control buttons:

- **"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 knows if a step fails, in which case the planner is called again to find an alternative. If there is no alternative plan, the system will tell the user which step of the process went wrong along with suggestions for 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 a time. This permits the user to observe physical effects of each action.

- **"Tell me more"** This button 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.

- **"Oops, it does not work!"** This button queries an online search engine for information about the step. This button can be specialized to use the device's user forums or vendor-provided 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.

These facilities not only provide the user with concrete information and things to do, but also facilitate the user's learning more about the devices' principles of operation.

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 recorder and the DVD player"*. The system cannot perform the action by itself, so it asks the user to perform this action. Roadie shows the user a picture of the correct input and connector. The "Tell me more" button 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 disk.

The user inserted a music CD and not a movie DVD as the system is expecting. The system knows that the CD can be recorded but not it consist on just music and not on video. The system notes this difference to the user and starts the recording process.

After recording 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. 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 record the music CD, but there is also a CD player unit that is capable of recording the CD player. At this point the user has three possible options,

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

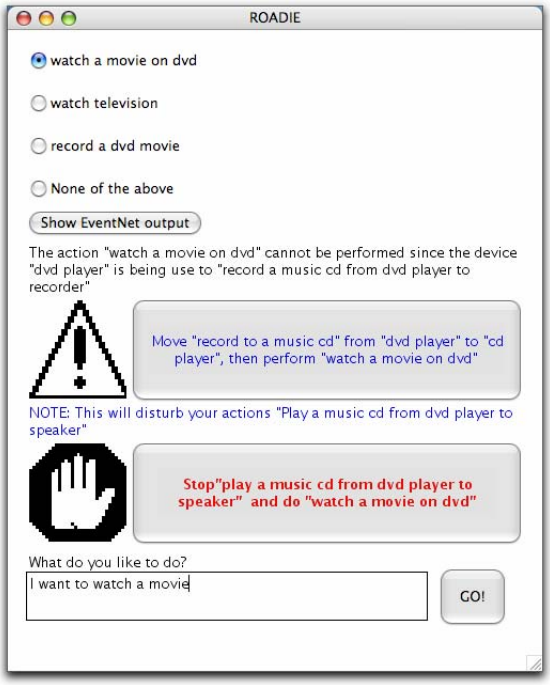

Figure 4: Roadie showing two possible ways of resolving conflicting goals

Roadie displays the dialog shown in Figure 4. 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 "record a music cd from dvd player to recorder."* The first button says *"Move "record 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 "Recording a music cd from dvd player to speaker" will be disturbed. The second button says *"Stop "recording 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.

5.2 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.

To perform this action, the user needs to connect the cable box to the television. But the user does not make the connection correctly. The user realizes that something is wrong, then type in the "What do you want to do?" box, "Why can't I see any image?" Roadie identifies the pattern "Why …" as if something is wrong, then it tries to find the problem and correct it. Roadie uses the devices sensors to look for the cause of the problem. Also it knows that it cannot sense the states of the cables, and that it is a frequent source of mistakes. It then shows a picture of the correct way to connect the cables, and recommend the user to check the connections. Furthermore, Roadie provides "Tell me more" option where the system gives more explanation about how to know if the connection is correct, information about the different types of inputs, outputs, jacks, etc.

Figure 5: Picture of how to connect a television

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.

5.3 KitchenSense

KitchenSense is an Augmented Reality Kitchen that uses the same techniques that Roadie uses to provide the people cooking with context aware information [6]. Whereas the first scenario dealt with audio and video equipment, similar user interface problems exist for kitchen appliances such as microwaves, dishwashers and food processors.

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 refrigerator,"* 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.

6. 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.

The testing scenarios was configuring the DVD player to play a music CD, then move the music CD to the CD player and play a movie DVD as explained in section 5.1. And recording the piano, this scenario involves using an amplifier whose functionality is not shown explicitly forcing the user to play with the devices in order to succeed with the task.

We designed the experiment using a between-subjects design with six participants in each group. As a result we find that the users finish the task in less time and using less steps with Roadie turn on than off, the results are shown in Table 1. (Due the small sample size, we do not give confidence levels).

One surprising finding during the experiment was that the users used the explanation of the steps not as an presentation of the system plan, but as a list of steps for the user to follow. Perhaps that could be cured by better explanation to the user as to the function of the list.

As the table below shows, users were able to complete the tasks significantly faster and with fewer clicks with Roadie than without.

Table 1. Average time and number of clicks before the user ended the given task

		Roadie ON	Roadie OFF
Play a CD on the DVD player	Avg. Time	88.33	111.50
	Avg. clicks	10.33	13.67
Play a DVD and a CD at the same time	Avg. Time	171.33	179.83
	Avg. clicks	19.00	32.83
Record the piano	Avg. Time	202.17	444.00
	Avg. clicks	23.00	59.33

7. 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.

7.1 Interfaces to Consumer Electronics Devices

De Ruyter created a context-aware remote control where the state can be changed in response to the input of home sensors. The user can modify its look-and-feel and contextual rules. His work recognizes that a new programming metaphor needs to be developed [2].

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 nor any provision for planning or debugging. It is easy to imagine expanding the current Roadie functionality to allow correction of the rules while the system is in use.

One popular approach is to build universal remote controls, able to control every single device [17]. Logitech's Harmony remote control represents the state of the art, but since it is based on IR interfaces, it cannot read the state of the devices, such as what channel a television is tuned to, or what FM frequency a radio is receiving.

There are many "smart home" projects, such as MIT's House_N, [5], Georgia Tech, Philips, the University of Texas, and others. These houses contain appliances such as washing machines and microwave ovens that could be targets for our approach. But these projects emphasize sensor technology and have not focused on the user interface for controlling and debugging sets of devices. We have already explored kitchen applications in the section describing our work with Jackie Lee on KitchenSense.

PRECISE is able to translate *semantically tractable* sentences to SQL queries and allow users to control devices. This allows controlling a house thermostat by saying, "Increase the temperature 5 degrees" [16]. This requires the user and the system to share the same vocabulary. Roadie permits more open-ended sentences like "It is too cold here", inferring that the user wants to increase the temperature. When users do comply with PRECISE's semantic tractability requirements, Roadie could use their approach.

7.2 Mixed-Initiative Interfaces

Collaborative or *mixed-initiative* interfaces are software agents that cooperate with the user to satisfy goals. The outstanding system within this paradigm is Collagen [12]. 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. Task models are used to map high-level goals into concrete device operations. We were inspired by many of Collagen's interaction features in the design of Roadie.

The Collagen architecture has been applied to consumer electronics [13], in a system called DiamondHelp. DiamondHelp is a help system that explains procedures that accomplish high-level tasks, and also provides a virtual on-screen device interface. Roadie's use of the Commonsense knowledge base and natural language input allow it to handle a wider range of goals and allow the user to communicate intent more flexibly. DiamondHelp generally treats procedure one step at a time rather than producing the overview that Roadie offers, and it is more oriented toward the normal operation of the device rather than debugging scenarios.

7.3 Goal-oriented and Commonsense Interfaces

Our group at the MIT Media Lab has been working on a wide variety of interfaces using Commonsense computing, surveyed in [7]. Many of these interfaces share Roadie's approach of using Commonsense to infer goals from concrete actions. An early example is Hugo Liu's Goose, a goal-oriented search engine. This search engine goes beyond 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"* [10].

7.4 Self-explanatory Interfaces

The ability of systems to introspect their state and change it is called *reflection*, a necessary capability for systems that provide explanation and debugging help. The most significant system is EXPECT, a knowledge acquisition and reasoning tool. 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 [4].

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, also it allows to ask why and how something happened, and it can tag the data as successful or unsuccessful [8].

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.

8. Expanding Roadie's Capabilities

8.1 Learning from 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's temporal traces, the user should be able to mark the output links that are incorrect. A learning facility would also allow us to streamline the interface in the case that the user wants to perform simple tasks that they already know how to do and the system behaves as expected.

8.2 Allowing the User to set Custom Goals

Roadie's goal recognition depends on having either pre-programmed goals or being able to deduce the goal from EventNet. We would like to include the possibility for the user to program new goals themselves. This could be accomplished by adding Programming by Example [9] techniques.

9. DISCUSSION

9.1 Implications of Goal-oriented Interfaces

Raskin [11] 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 distinction gets blurred. Since the system can map high-level goals to low level actions, there is no need to keep the core functions small.

9.2 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.

10. CONCLUSIONS

In this paper we show an interaction schema for consumer electronics. This schema uses a plan recognizer from the Openmind Commonsense corpus to find what are the users intentions and propose relevant devices actions. In addition, it us a planner to control and manipulate the devices avoiding the users the trouble of dealing with low level configuration and helping them to debug the devices when something does not goes as planned.

11. ACKNOWLEDGMENTS

Our thanks to all the Media Lab sponsors for their support of this project, and to Rada Mihalcea of the University of North Texas for help us to mine the data from the Web.

12. REFERENCES

[1] 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.

[2] 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).

[3] 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)

[4] 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.

[5] 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.

[6] Chia-Hsun Jackie Lee, Leonardo Bonanni, Jose Espinosa, Henry Lieberman, Ted Selker. KitchenSense: Augmenting Kitchen Appliances with Shared Context using Knowledge about Daily Events. *Conference on Intelligent user interfaces* (in submission)

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

[8] Henry Lieberman, Earl Wagner: End-User Debugging for Electronic Commerce, *Proceedings of the 8th international Conference on Intelligent user interfaces*, ACM Press, 2003.

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

[10] 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.

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

[12] 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

[13] 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.

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

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

[16] 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.

[17] 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.