From: AAAI Technical Report SS-98-02. Compilation copyright © 1998, AAAI (www.aaai.org). All rights reserved.

Towards Rapidly Deployable Intelligent Environments

Sandeep Chatterjee

Laboratory for Computer Science
Massachusetts Institute of Technology
545 Technology Square, NE43-505
Cambridge, MA 02139
sandeep@lcs.mit.edu

Abstract

This paper presents a novel framework and paradigm for creating rapidly deployable intelligent interactive environments, and in particular, homes. We argue that common household information appliances, such as televisions and stereos, present a seamless and non-intrusive point from which to "observe" and interact with members of a household. To this end, we describe the framework of our distributed software agent system and the underlying hardware infrastructure. We also briefly describe applications that can be built with our system.

Introduction

There is currently much interest in developing intelligent interactive environments, and in particular, homes. However, for the most part, these efforts are intrusive and are not rapidly deployable. For example, neither embodied robots nor video camera (observation) based approaches would be readily accepted into typical homes. Although it is clear that these approaches have their place in research and may one day be embraced by society, this paper presents a novel framework and approach for creating *rapidly deployable* interactive homes based on autonomous software agents (Maes 1991) within information appliances.

We propose a framework based on adding computational intelligence to common household information appliances, such as televisions, stereos, and telephones. Our approach is motivated by the hypothesis that information appliances provide a set of essential points from which to "observe" members of a household.

From the moment that we are awakened by our alarm clocks to the moment that we reset them for the following morning, we are constantly interacting with information appliances. These interactions are important because they mold and reflect our emotional state, which affects our actions. For example, when we watch a television news report of a crime, our emotional state may become somber or introspective. And, when we are somber, we may be more likely to tune into a particular television program, such as a comedy show. Of course, each individual's reaction to the information presented by an information appliance will differ, and as such, the embedded agent must be sufficiently intelligent to learn from and adapt to its users.

Conceptually, the software agent associated with each information appliance monitors these user-to-appliance interactions in an effort to learn about the user's frame-of-mind and to discover behavioral patterns. These "observations" may then be used to automate routine tasks (e.g., immediately switch incoming telephone calls directly into voicemail when the evening news is on) and to suggest activities for the user (e.g., automatically turn on the radio to a soothing station when an agent believes the user is under heavy stress). Our approach turns the passive base of home information appliances found in typical homes into an intelligent and interactive segment of a household.

The next section of this paper describes our hardware platform and its underlying motivations. After this, we present our distributed software framework for adding intelligence and interactivity to information appliances. Finally, we conclude with a discussion of applications that can be built upon our framework and touch on future work.

The Hardware Infrastructure

From the most basic alarm clocks to the most sophisticated television sets, today's information appliances embody a substantial amount of computation. Unfortunately, however, these appliances, and in particular, their embedded micro-computers and controllers, are "stranded" in the sense that their software, memory, and computational capabilities are fixed, and they are not connected to a bi-directional network.

Stranded information appliances present many obstacles to creating intelligent and interactive home appliances

· Copyright © 1998, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

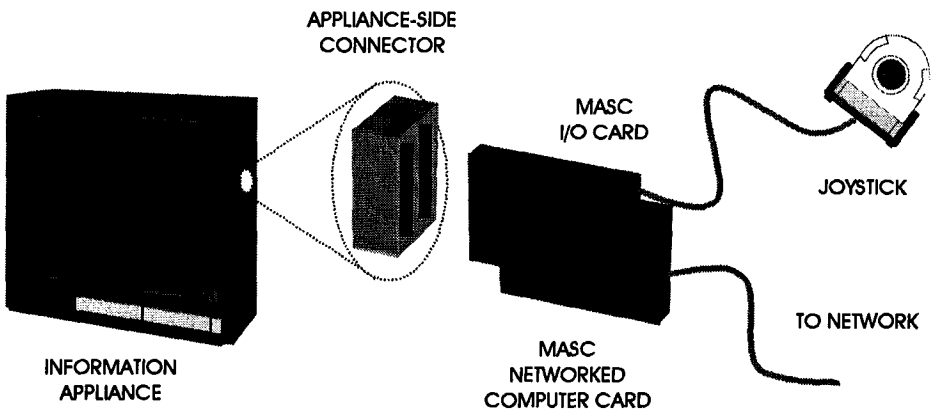

Figure 1: The MASC System.

based on software agents:

- Stranded appliances do not provide a means to economically upgrade the computation and communications hardware platform, which is often necessary for software upgrades. This is important because as the area of autonomous software agents is relatively new, a *rapidly deployable* approach such as ours must provide provisions for upgrading to newer algorithms and agent systems. As these agents become increasingly complex, it may be required to upgrade to a faster processor or to increase the amount of memory in order to deliver sufficiently low interactive latencies.
- Stranded appliances do not provide a means to communicate with other appliances; essentially, they are not connected to a bi-directional network. This is important because, as we will show in the next section, in order to create applications of any import, the agents of different information appliances must communicate with one another.

Un-stranding information appliances requires the addition of a computer and a network back-end. This can be achieved by either a centralized unit connected by a network to each appliance (e.g., a server) or by a set of distributed units connected by a network to one another. We believe that the peer-to-peer distributed approach is preferable for two reasons, both of which are based on lowering the overall cost of ownership of the system. First, a computer unit associated with each information appliance allows a smaller granularity of upgrades; that is, the hardware platform for a single appliance can be upgraded independent of the others. Second, a local computer for each appliance decreases network traffic which may allow a less expensive network to deliver the required application execution latencies (because data and commands from the appliances do not have to flow to a central computer unit via the network). Furthermore, a local unit for each appliance allows households to incrementally add intelligence to each of their appliances, rather than having to invest in a powerful computer unit for the entire base of home appliances at one time.

Having discussed the requirements for a computation and communications hardware platform for information appliances, we next describe one incarnation of these ideas.

The MASC Platform

The MASC (Modular Appliance Super Computer) architecture addresses the need for a modular computation and networking device for information appliances (Chatterjee 1997). As shown in Figure 1, MASC is comprised of the five following components:

- MASC networked computer cards
- Information appliances with MASC appliance-side connectors
- Software running atop MASC cards
- Internal home network
- Gateway to external networks

MASC cards form the computation and communications hardware platform of our approach. Each card synthesizes the properties of "smart cards" and PCMCIA cards (PC cards), and contains within it a microprocessor, a network adapter, main memory, and non-volatile storage, and implements a common interface with which to

communicate with all host appliances[1].

Microprocessor. Any standard microprocessor can be used as the computation engine for MASC. Furthermore, different implementations may emphasize different performance characteristics, e.g., speed versus power consumption, to suit different applications.

Network Adapter. Any network can be used as the vehicle for communication between different MASC cards. Accordingly, different implementations may represent different combinations of processor and internal network architecture. MASC's network adapter can be used to provide connectivity between different appliances, and to download new software onto the card.

Memory and Non-volatile Storage. Each MASC card contains memory for use by the processor and some local non-volatile storage. Again, the type and amount of memory and storage can be tailored to meet the demands of different applications. The non-volatile storage allows application programs and user data to be saved locally. Additional application programs can be downloaded over the network and executed in a manner similar to that used in a network computer.

Interface Block. The interface block allows communication between the MASC card and the appliance-side connector. When a MASC card is connected to an appliance, the interface block deciphers the type of the appliance and how to communicate with it. After this, its primary task is to send data and commands between the MASC card and the appliance socket.

As newer computation engines and home networks become available, the user can simply remove the MASC card from the appliance and replace it with a new one. Essentially, the MASC architecture allows three *degrees of freedom* (in choosing and upgrading 1. the processor, 2. the network and 3. the software environment) which can be exploited to take advantage of the areas of greatest technological change and improvement. MASC's modular and user-swappable cards are essential because software upgrades often first require hardware upgrades. As software becomes increasingly complex, the most common hardware components for upgrading are processors and memory (including L2 cache).

Essentially, MASC offers a flexible and general-purpose computation and communications platform on which to place software for the host appliance. This hardware platform may be used to move appliance functionality from dedicated hardware to more flexible and dynamically upgradable software. This provides a mechanism to add value to the established base of household information appliances by leveraging software and network "externalities"[2].

In addition to application software for augmenting an appliance's base functionality, MASC provides an excellent platform on which to run intelligent agents which "observe" user-to-appliance interactions. We describe our distributed software framework and agent prototype next.

Software Infrastructure

Our software framework is a complete peer-to-peer distributed agent system. Local agents, executing atop local MASC cards, observe all interactions between their host appliances and the user. These observations are used to discover routine interaction or behavior patterns, and, in conjunction with analysis of the host appliance information stream, e.g., television program content, to form a *local opinion* of the user's frame-of-mind. The local opinions of agents for different appliances can be combined at any time to form a *global opinion*. Local interaction behavior, local opinions, and the global opinion may be used to automate routine tasks and to suggest activities for (i.e., interact with) the user.

SANI

In order to demonstrate our approach, we are developing the SANI (Seamless and Non-Intrusive) agent system for household information appliances.

Our first prototype implementation consists of three SANI agents—one for a television set, one for a telephone, and one for a stereo. Each agent analyzes its local user-to-appliance interaction stream and tries to find correlations between different interactions in an effort to automate routine tasks. For example, the stereo agent, realizing that the user turns the stereo on, tunes it to 106 FM, and sets the volume to 55% at 5PM every Friday evening, may choose to automate these set of actions.

Applications of more import can be achieved when the different agents are allowed to communicate with one another. An interaction segment that would be made possible by a set of communicating SANI agents is shown below. SANI 1, SANI 2, and SANI 3 are the local agents for the television, stereo, and the telephone, respectively.

User: [Watching television. Time is 9:34 PM]

SANI 1: [SANI 1 knows that the TV is typically turned off at 9:30 PM. Contacts other agents to find out what happens at that time. Learns that user listens to the radio from 9:30-10:30 PM.] The

1 The network adapter and non-volatile storage may be placed onto separate cards in order to enhance the modularity and flexibility of the approach.

2 "Network externality" is the synergistic value derived from networking multiple appliances together. Similarly, "software externality" is the value derived from application software written to complement or augment an appliance's base functionality.

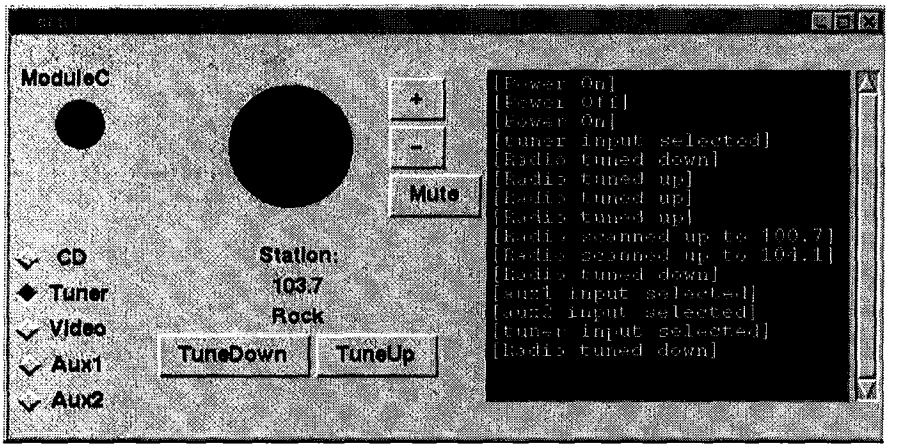

Figure 2: Stereo graphical front-end.

upper right corner of the TV displays: *You usually listen to 106 FM on the radio from 9:30-10:30pm. Press Enter on the TV remote to cancel, Power to turn off TV, or 1 to route radio stream to TV speakers.*

User: Presses 1 on the TV remote control.

SANI 1: [Contacts SANI 2 and asks it to tune to 106 FM and to route the information stream to the television, e.g., the television's MASC card.]

SANI 3: [Phone call in-coming. But the local agent knows that phone calls are not answered from 9:30-10:30 PM, so it immediately switches the phone call into voicemail and suppresses the phone ringer. Asks SANI 2 to alert user to voicemail after radio show is over at 10:30 PM]

SANI 2: [Realizes that the radio signal has stopped, possibly because of an external network outage.] Asks SANI 1 to display on TV: Radio outage. You have voicemail. There is a similar program on channel 8 on the TV. [Turns on TV to channel 8]

The above interaction segment provides a glimpse into the value of SANI. The system may potentially have tremendous utility while being both seamless and non-intrusive. That is, from the user's point of view, SANI-enhanced appliances are exactly identical to those with which he has interacted in the past and is comfortable. However, SANI adds intelligence and tremendous value to the base of home information appliances.

Currently, our prototype system uses neither the MASC hardware platform nor actual information appliances.

Figure 2 shows an example of a Tcl/Tk front-end for a stereo that we are using to simulate interacting with the underlying SANI agent. Each button press on the graphical stereo component is passed through to the SANI agent. As a result, SANI may instruct the appliance to perform a set of actions which are then registered with the graphical front-end. The right-most window within the Tcl/Tk window shows data and commands that are passed through to SANI and those that are passed up from SANI.

In the next sections, we describe the software architecture of the SANI system in more depth.

SANI Software Architecture

Each SANI agent has the following responsibilities:

- Find correlations and patterns in user-to-appliance interactions, both locally and globally.
- Analyze current appliance information stream and user interaction behavior to formulate a local opinion of the user's frame-of-mind.
- Gather remote local opinions and form a global opinion when needed.
- Communicate with the host appliance to instruct it to perform particular tasks.
- Serve the requests of other agents.

Figure 3 shows a high-level depiction of SANI's software architecture. Each user-to-appliance interaction that is passed through from the host appliance is first placed into a packet. The packet includes the appliance component name, the component value (e.g., volume is at 55%), and the current date and time. The date and time are used as a context in which to interpret the interaction. If the packet matches any learned patterns, the host appliance is instructed to perform those tasks. The packet is then placed into the agent's existing interaction stream and analyzed for new patterns.

A local opinion is formed by analyzing the current information stream from the host appliance. The content of the information is classified into different emotional categories. This classification is then merged together with the existing local opinion and interaction stream to form a new local opinion. The global opinion is formed in a similar fashion, but by combining the local opinions of

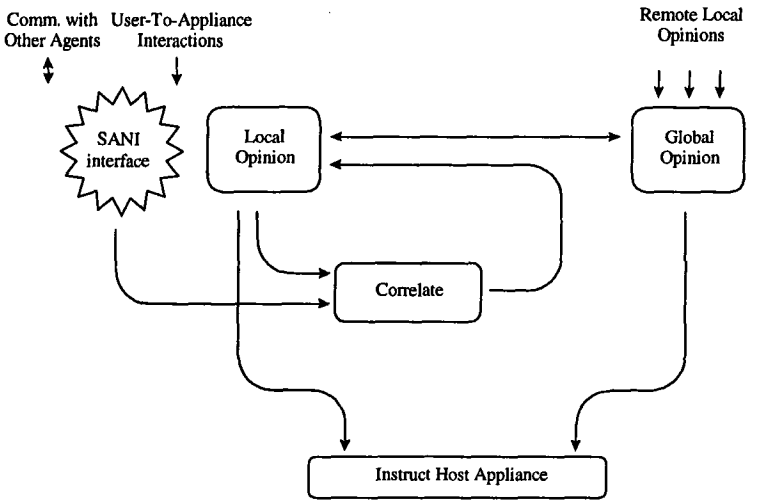

Figure 3: The software architecture of the SANI system.

different agents together.

Future Work

In future prototypes of our system, we will incorporate the MASC hardware platform and use actual information appliances, instead of simulations thereof. The upgradable MASC platform will also provide us with a mechanism to incorporate alternate user interfaces, such as voice. A voice-based interface to household appliances will not only allow a more natural method of interaction but will also provide more information to the SANI agent system. For example, voice "footprints" can be used to identify individual members of a household which then can be used to automate and suggest tasks based on physical presence.

We also plan on improving the SANI agent system. In particular, its correlation and pattern recognition algorithms can not only be optimized for space and time and but also for the type of correlations it can make. The overhead of cross-agent communication can be reduced by limiting when an agent communicates with one another. Further research is required to understand the optimal time and granularity for such global communications.

Conclusions

This paper presented a novel framework and paradigm for creating *rapidly deployable* intelligent interactive homes based on autonomous agents running within common household information appliances. We argued that information appliances, such as televisions, stereos, and telephones, provide a set of seamless and non-intrusive points from which to "observe" and interact with members of the household. To this end, we described the software and hardware architecture of the SANI system. Furthermore, we showed that applications of significant import can be created with our framework.

Acknowledgements

The research described herein was supported in part by the Advanced Research Projects Agency under contract No. DABT-63-95—C-0060 (monitored by U.S. Army, Fort Huachuca).

References

Chatterjee, S. 1997. The ModuleC Network Architecture: A Novel Paradigm of Computing Through Information Appliances. In Proceedings of IEEE International Symposium on Consumer Electronics, Singapore.

Maes, P. 1991. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back.* Cambridge, MA: MIT Press.