MP 06W0000081

MITRE PRODUCT

The EMMA System: Enabling Novel User Interfaces to Integrated Conferencing Systems

September 2004

Lisa D. Harper
Abigail S. Gertner
Paul M. Herceg
Thomas T. Hines
Edwin H. Kemon
David E. Mireles
Michael E. Shadid
James A. Van Guilder

MITRE
Corporate Headquarters
McLean, Virginia

The EMMA System: Enabling Novel User Interfaces to Integrated Conferencing Systems

L. Harper, A. Gertner, P. Herceg, T. Hines, E. Kemon, D. E. Mireles, M. Shadid, J. Van Guilder

The MITRE Corporation
1820 Dolley Madison Blvd
McLean, VA 22012 USA
+1 703 883 5241
[lisah,gertner,pherceg,tomhines,ekemon,mireles,mshadid,jamesv]@mitre.org

ABSTRACT

We discuss the infrastructure and design of an experimental conference room designed for the purpose of integrating emergent technologies in an operational setting. This system is a replica of standard corporate conference facilities at The MITRE Corporation in McLean, Virginia. Our environment is currently used by a small number of volunteer participants in order for us to accommodate user input early and frequently in the development cycle. Most commercial conference facilities rely on interfaces designed and built by hardware vendors or system integrators. We have created an abstraction and interface to our facility's controller that enables the development of new kinds of room interfaces. We describe our system EMMA and also user interface development for this system.

Keywords

Smart Room, VTC, User Interfaces, Dialogue, Multimodal, Infrastructure, Intelligent Assistant, Context-aware

INTRODUCTION

Audio-video (AV) environments, including state-of-the-art conference rooms rely on control systems to bind system components into an integrated system. These systems typically have three components: a CPU, various communication boxes or modules and a controller. The controller provides tools for creating graphical user interfaces programmed by developers who are familiar with a proprietary language supplied by the control system vendor. There are, in fact, two vendors that consume about 95% of the control system market: AMX and Crestron.

The control system interface provides a single user interface to room devices through a single control point. A primary design function is to provide an intuitive user interface that any user can learn through minimal or no training. Furthermore, the control system interface should be easily accessible to room users during the course of a meeting. Even though room devices may be stored in a cabinet in the back of the room, users should not have to get up from the table to operate them. The paradigm for such systems is that everything in the room (i.e. lights, audio, screens, PCs, etc.) should be controllable from the central control system and interfaced by the user through a touch panel or hand-held remote.

MITRE AV facilities are designed with an AMX controller. Therefore, we will discuss how we have instrumented this controller to enable novel kinds of interface development. We will describe several sorts of interfaces that we are developing.

MOTIVATION

Our organization hosts a network of state-of-the-art VTC rooms. Room equipment typically includes a pan-tilt-zoom camera (or PTZ), microphones and speakers, an audio mixer/switch, a video switch, a codec hosting an H.323 endpoint, multiple displays/projectors, motorized projection screens, a satellite television receiver, a VCR, a lighting control system, motorized window shades, and more. Multipoint meetings are switched via a multi-function controller (MFC) video bridge in the network. An ISDN service enables videoconference connections with outside organizations. The analog telephone line is used for a connecting a single party or for connecting multiple participants via an audio bridge.

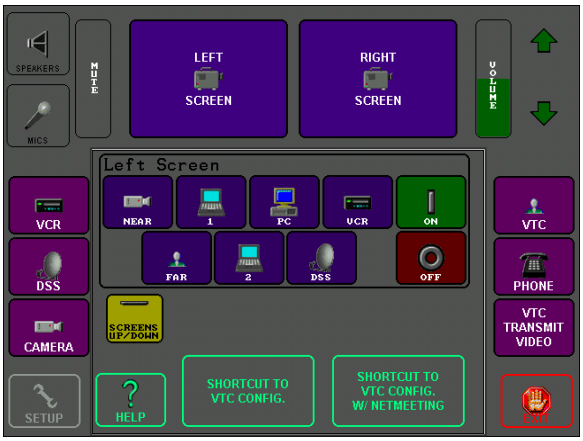

Figure 1. Complex AV Control System Interface

The *AV Control System* (i.e., the AMX) centrally controls all standard Team Room devices including VTC and telephony connections, plasma screens, and camera, overhead lights and window shade settings. Essentially, it

is a proxy because it acts on behalf of other devices in servicing requests. A standard *AV Control System Interface* – the user interface (see Figure 1) – is a touch-panel screen with a collection of button-panels, which are tailored to specific devices and functions. Over time, the number of Team Room devices and features at MITRE has expanded, increasing the complexity and quantity of the panels. The standard panel interface has seven unique screens and over a hundred buttons. Users find the interface confusing and often rely on telephone support from corporate communications to setup a meeting. It is impractical to train all employees on this user interface – in actuality, panels vary slightly from room to room since not all Team Room configurations are precisely identical. However, the lack of user training and sporadic frequency of use can account for meeting delays of up to 10 to 20 minutes. Although, we have specific hypotheses about features that we believe will reduce or eliminate these problems, in order to implement these ideas we needed to design an infrastructure that would support various types of user interfaces and additional devices and software processes that would support these interfaces.

Our goal, ultimately, is to reduce the cognitive burden on humans allowing them to focus their attention on decision-making and social interaction necessary to achieve complex team tasks. These goals are discussed in more detail in [6]. To test this we constructed an experimental AV facility with user-observation capabilities – the *Experimental Team Room (ETR)* – and we developed an infrastructure and experimentation environment to support further interface development. By leveraging our own operational mission and corporate expertise, we have a tremendous opportunity to study real user populations over an extended period of time.

FACILITIES DESIGN

The ETR facility was constructed as a split facility with both a meeting area and observation / equipment room. Two panes of 3' x 6' one-way glass were installed to separate the rooms and enable meetings to be observed from the back room. A three-inch raised floor was added for cabling between the rooms. Air conditioning and high density power (i.e., several 20 amp circuits) were configured for generator backup. Electrical outlets and cable trays were installed above the drop ceiling in anticipation of camera and projector equipment. And standard six-connector network jacks were installed every 4 feet.

In the front room, users view four systems: the VTC connections, user interface, the Windows PC for user desktop applications, and the AV Controller Interface. Wireless keyboards and mice are used as needed for the desktop applications and user interface. The PCs and AV equipment rack reside in the back room.

Since the facility and equipment could not be purchased with limited research funding, it was constructed as a corporate resource with the agreement that it would be available for employees' use twice per week. Although seemingly incompatible, we manage both operational and development systems and provide a switching mechanism between the two. We will be configuring the wireless keyboards/mice, microphones, and AV system to be switched bi-weekly between test and development systems.

APPROACH

Our system design is driven by the following requirements:

- Foremost, we must make this environment accessible to real users for incremental feedback during our development cycle.
- We need to support logging of user interactions so that we may evaluate user performance.
- The system should have access to contextual information such as a history of user interaction, device and room state, and also meeting context (e.g., time and date, participant names, slides, etc).
- They system must be accessible by both remote and local users; and, preferably, local users can access the system from multiple locations in the room.
- The system must support both naïve and experienced users.
- The system must support mixed-initiative interaction.
- Users should be able to switch between interaction modalities or communicate to the system using multimodal, or synchronous modes, of interaction.
- We must be able to easily add new room devices and capabilities.
- We must be able to support and experiment with, essentially, plug-and-play user interfaces since we may want to support multiple paradigms.

It is on this last design tenet that we focus this paper.

ETR USER INTERFACES

During the course of development, we have designed and implemented several sorts of user interfaces for the room. The majority of this paper will focus on the EMMA system, which includes an embodied agent interface to the room. We are in the process of endowing EMMA with various perceptive capabilities such as recognizing user identity via face recognition and / or badge ID. Our goal is to gradually increase the amount of perceptual and contextual information that she has access to in order to make her more aware of room state and meeting activities. However, in the course of development, we have developed other interaction paradigms which may be of use to A/V systems integrators in the near term.

Speech-based Interfaces

There are two styles of speech interfaces that we will briefly describe.

"What Can I Say?" GUI

The first speech-based interface that we have used is actually mixed-modal. This is the "What Can I Say" speech-based GUI interface associated with the Mitsubishi Electronic Research Laboratories (MERL) Collagen system described later in a later section of this paper. The GUI is essentially a list of utterances that are valid (according to the task model) at any particular point. Because this can potentially be a rather large number of possibilities, we've come up with other UI designs that still take advantage of the Collagen Dialogue manager but aren't strictly "utterance"-based. This GUI is used in association with an Emma avatar and push-to-talk speech recognizer which will be described later in the paper.

Speech Prompt

This interface is purely speech based and reminiscent of call center telephone speech interfaces. In fact, since one possible means of interacting with our system is via telephone, we believe this to be useful for remote users who are communicating with the room over the telephone. However, this may not be a preferred means for interfacing with the room when the user is actually inside the room.

GUI-based Interface

We designed a method for rapid development of user interfaces to the AV Control System. We developed Java-based widgets (i.e., GUI building blocks) that are associated with a focused set of AV Control System functions. Using a library of widgets, developers compose novel GUIs by experimenting with screen placement and different widget combinations. This increases reuse of UI components and reduces development time, thereby increasing the number of UIs with which we can experiment. These widgets can be used for GUI only interface development or also as a part of a multimedia interface such as the one described below.

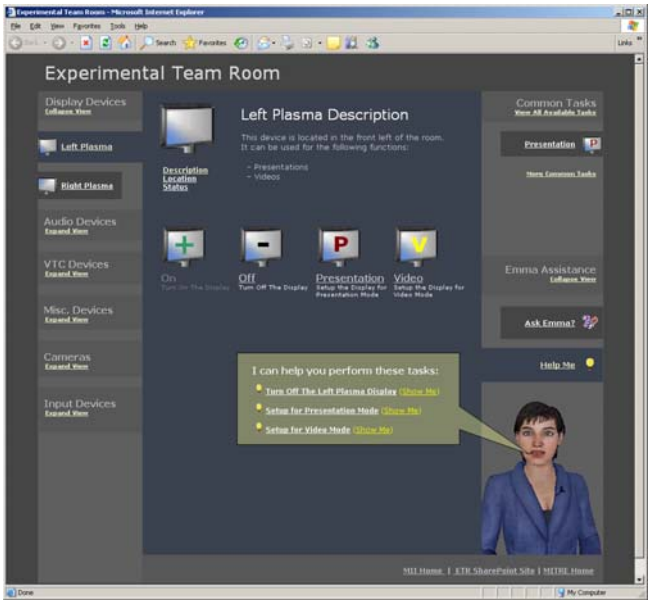

Figure 2. Intelliprompt Multimodal Interface.

IntelliPrompt Multimodal Interface

The IntelliPrompt system interface, currently under development, is designed to overcome shortfalls in the "What Can I Say?" GUI design as well as shortcomings in the current GUI-based interface. This GUI runs in a web browser and is composed of the EMMA embodied agent (who can gesture and refer to other elements in the GUI), speech GUI components and GUI elements that show user interaction history as well as *task* and *device* views of the interaction (see figure 2). We recognize that while a task-level systems interaction (e.g., establish remote connection) may be useful at the start or end of a meeting, it may be that users only want access to particular devices in a device view interaction during the course of a meeting. The IntelliPrompt GUI is discussed further in the system design section below.

SYSTEM DESIGN

EMMA was designed to be a modular and distributed research system for the exploration of Team Room UIs. Following are the subsystems that compose the most recent version: the AV Controller, the Agent Controller, the Macro Player, the Speech Recognizer, the Intelliprompt GUI, the Avatar, the Biogate, and the Context Model. Figure 2 shows the typical communication paths between some of the subsystems. For new development we use a service-oriented infrastructure approach, which facilitates the addition of new capabilities. This approach enables us to easily add new subsystems, which is vital to the iterative nature of our research and development. It also enables us to easily add more CPUs to meet growing processing requirements.

AV Controller

We instrumented the commercial AV Control System (i.e., AMX) into the EMMA infrastructure via the *AV Controller* subsystem. This enabled the perceptive assistive agent to control room devices via a single telnet connection. *Commands* were sent as strings, and *event* strings were monitored via the same telnet connection.

Enhancements followed to improve scalability. Communication with the AV Controller subsystem was asynchronous, providing no correlation of acknowledgement events to the commands sent. Also the logic to parse and correlate events resided in the client, which limited control of the room to the agent subsystem. Therefore, a Java system layer was added in front of the telnet interface to enable multiple subsystems to control room devices simultaneously (i.e., multiplexing). It also synchronously correlates commands and acknowledgement events, and enables asynchronous monitoring of events (i.e., listening) – complex business logic which had previously resided in the client subsystem.

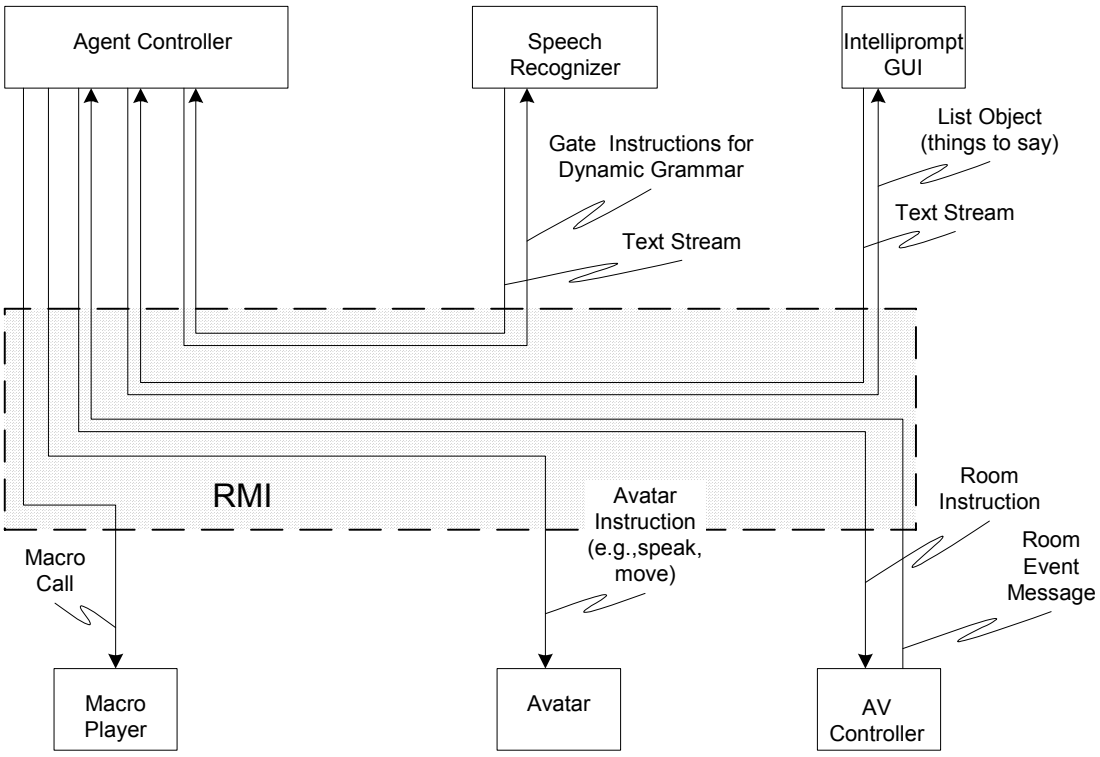

Figure 3. Typical communication paths in the EMMA system.

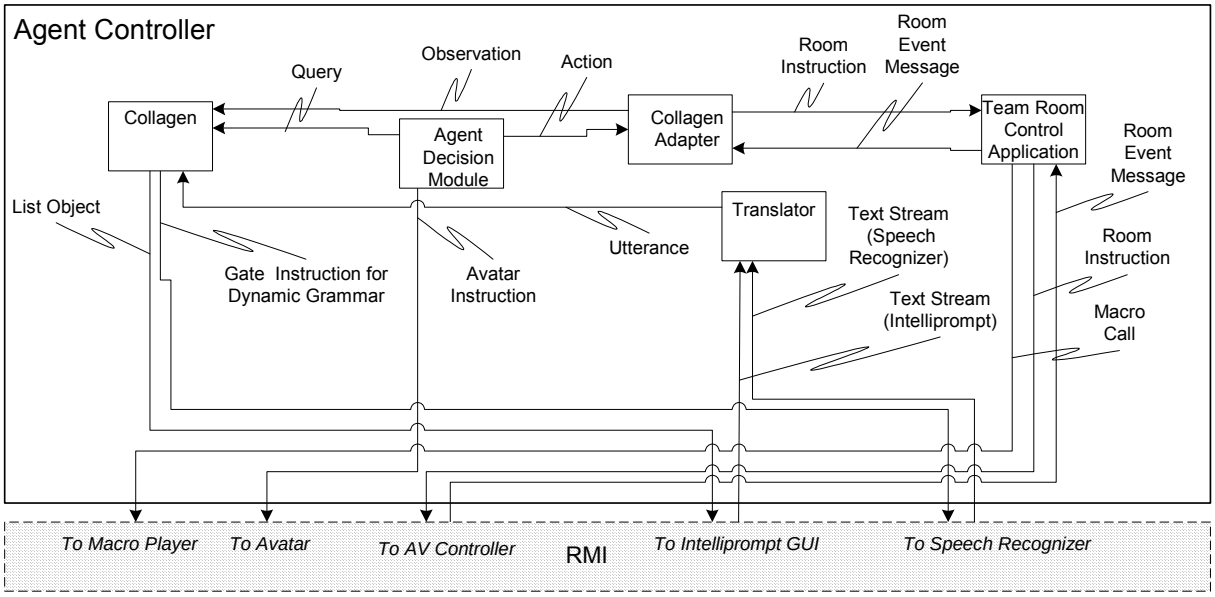

Figure 4. Agent Controller design.

As a result, all devices in the Team Room can be controlled by calling AV Controller methods.[1] This central development enables us to more rapidly prototype and test new user interfaces using a common set of methods.

[1] Developers do not have to be directly connected to the ETR to develop new capabilities for EMMA. We developed an AMX emulator that actually simulates the way the AMX operates in the AV environment.

The Agent Controller

In the ETR, meeting participants collaborate with EMMA to setup the room and make changes in room configuration. We created a dialogue manager using the Mitsubishi Electronic Research Laboratories (MERL) Collagen system. The dialogue manager is responsible for understanding and generating conversation. It relies on discourse plans and a task model to do this. Rather than reiterating the Collagen design and approach in this paper,

we refer the reader to [13] and [14]. We also use the terminology found in these works, such as collaboration, collaborative discourse, recipe, discourse state plan tree, etc.

In the ETR, participants have the following options: controlling devices manually, controlling them via a GUI panel, or controlling them in a mixed-initiative interaction with EMMA. We believe that many users will prefer a mixed-initiative interaction where they take some actions themselves but also delegate some actions to EMMA.

Currently, EMMA intervenes only at the request of a user. EMMA is essentially modeled on a user-initiated interface. We have also experimented with a tutorial-based system-initiated approach and feel that eventually both conversational styles may be useful depending on the knowledge and experience of a particular user in performing particular tasks.

Figure 3 shows the Agent Controller in the context of the entire EMMA system. Figure 4 shows a decomposition of the Agent Controller. EMMA's task model consists of rules for achieving tasks in the domain. These rules, or recipes, are used to dynamically construct the discourse state plan tree as the discourse unfolds. At startup *Collagen* generates an initial plan tree and the resulting agent agenda and user agenda. It then sends a grammar and list objects to the Speech Recognizer and IntelliPrompt GUI, which presents the user agenda as a list of valid utterances or actions.

When the user speaks or indicates a valid utterance, Collagen receives via the *Translator* an utterance object. When the user operates a device directly, Collagen receives via the *Collagen Adapter* a corresponding observation object. Collagen determines how observations, utterances, and recipes contribute to goals in the current discourse state plan tree or to a new goal. It updates the discourse state plan tree, agent agenda, and user agenda, accordingly. Grammars and list objects are also updated, accordingly. The *Agent Decision Module* detects the Collagen update, then queries the agent agenda and selects a domain action (e.g., connect a VTC, show a PowerPoint briefing, dim the lights) or a discourse action (e.g., speak a proposal, move), as appropriate. Collagen monitors this agent behavior in the same manner as the user's behavior (i.e., with utterances and observations), and proceeds with the corresponding updates. This interaction continues – with the agent and user taking turns.

The *Team Room Application* enables the Agent Decision Module to perform actions to control the room, both using the AV Controller and a Windows-based *Macro Player* that we developed as a part of this system. The Team Room Application also receives room event messages reporting any changes to the AV environment[2]. The *Collagen Adapter* converts room event messages to observation objects for consumption by Collagen.

The architecture we are using is configurable; we can replace the Collagen dialog manager with an application that lets a human masquerade as the agent. In this manner we have performed several user evaluations known as "Wizard-of-Oz" (WOZ) experiments. These experiments are designed to help us understand more about how users naturally desire to interact with our system. Experiment subjects speak to EMMA using a hold-to-talk button and are given only brief instructions on the kind of language that they should use. We convey that they should speak in short utterances and that her speech recognition is limited to a relatively small domain. A human wizard does the actual language understanding and response generation (through the EMMA avatar) but uses the EMMA system for carrying out actions. These experiments have been useful in informing our system design and also for testing and bug fixes.

The Agent Decision Module uses the task model reflected in the recipe library as its domain knowledge. It uses it to generate and interpret observations. Initial user evaluations have revealed patterns in task performance that span across users familiar with Team Room operation. Designing the task model has been an iterative process of developing recipes and testing system behavior in interaction with users.

The Macro Player

EMMA controls the ETR's Windows PC in addition to other team room devices. As with the AV Controller, a Windows PC is shared between meeting participants and EMMA[3]. Therefore we developed a Windows-based *Macro Player* that could take parameters; for example, executable names or user names[4]. We also decided that non-programmers would need to write the macro scripts, so a corresponding scripting language was developed. The macro scripting language is simple, consisting of primitive *actions* and *conditions*, which can be combined into *sequences*. At runtime, the macro player parses the macro scripts to instantiate macro objects that are available to the Team Room Application. As a result, the Macro Player enables EMMA to bring up PowerPoint presentations, conduct web and intranet searches[5], or novel actions developed by the Macro Player scripter.

Speech Recognizer and IntelliPrompt GUI

Interaction with EMMA is currently mixed modal – via speech (i.e., audio) or a GUI (i.e., point and click). We are also in the process of extending her ability to recognize a small set of arm gestures as well as location relative to

[2] Participants can use the AV Controller Interface to monitor changes to the environment and to perform actions simultaneous with EMMA.

[3] This PC hosts typical desktop applications, including a collaborative white board.

[4] The commercial products we reviewed did not support such parameters.

[5] Participants have the option of using the applications directly.

room zones or devices so that users can point to a display device and request a set of specific actions. At each point in the discourse, there is a list of appropriate phrases, or utterances, according to the current discourse state plan tree. Collagen sends instructions to the speech recognizer to enable and disable parts of the grammar; this tailors the grammar to recognize only phrases appropriate for the current user agenda. Simultaneously, it sends a list of the same phrases to the IntelliPrompt GUI (web-based interface currently under development). Each phrase is displayed as a separate button in the GUI. The user can speak an utterance or click on an utterance to advance the dialog. Hence with each user utterance, the change in agenda affects a corresponding change in the speech recognition grammar and IntelliPrompt GUI. In addition to displaying user utterances and actions, the IntelliPrompt GUI has an embedded push-to-talk button and display space for our half-body avatar.

For speech, we have experimented with both array microphones and a lavaliere microphone. We found speech recognition in the room to be more consistent with the close-talk lavaliere microphone. Also, the lavaliere microphone can be muted. The Array microphones are always on and lead to an abundance of false positive conditions (e.g., a false positive "dim lights" leaves participants in the dark). However there are benefits to the mobility offered by the array microphones.

Avatar

The Avatar "talking head" was initially developed as an API to a text-to-speech engine and a collection of approximately 100 face images. Actions included smile (soft or full), blink, speak-a-phrase, look-right or look-left, turn-right or turn-left[6], nod or shake. For each action a different image was displayed. An XML-like format was used for commands, for example, <SMILE DEG=1 TIME=20>, <SPEAK TEXT="welcome to the experimental team room">, <EYESMOVE DIR=LEFT TIME=10 DEG=10>. The agent controller issues such avatar actions as needed to speak or gesture with participants in a human-like manner.

Figure 5. EMMA

We used Boston Dynamic's DI-Guy SDK to revise the EMMA head and provide a half-body for arm gestures and body poses (see Figure 4). This interface was an improvement since we were able to synchronize actions (i.e., parallel actions such as smile and speak). We are currently in the process of migrating to a Haptek API for creating a web-based EMMA that is embeddable as an ActiveX component in the IntelliPrompt GUI. DI-Guy is better suited for distance and crowd behaviors as opposed to the close-in interaction that we need for a web-based EMMA.

Inter Process Communication

As mentioned above, EMMA is primarily written in Java. Java Native Interface wrappers are used where components are written in C. Therefore, we used Remote Method Invocation (RMI) to distribute the EMMA system across several platforms in the form of clients and servers. Planned for the next release, a common application class was developed to standardize the way each module plugs into the distributed system. The same class provides object management functions: registration, start, stop, enable, disable, etc. These enhancements establish a framework for extending and maintaining the EMMA system. This framework will also be used to host pervasive personal systems.

Biogate

We developed the *biogate* as a place for participants to register into the room and pick up devices to be used during the meeting (e.g., a microphone, a location tracking badge). Upon entry to the front room participants register by undergoing face recognition. The biogate displays the planned and present participants. Once integrated with the larger system, this component will report to EMMA the device-to-user associations for the purpose of inference. An overhead camera enables remote observation and recording.

As with most face recognition systems, the users must undergo an enrollment process prior to recognition. The commercial system in the ETR captures approximately 100 face images during enrollment. Enrolling multiple times increases the number of images and recognition accuracy.

Context Model

One of the challenges in developing smart room technology is providing an environment that will respond appropriately and timely to users' needs. Prior research has recognized that intelligent environments can benefit by using models to maintain and evaluate the context in which actions in the environment occur [7],[9], [11]. As part of our continuing work we are integrating a model to maintain contextual information about meetings, participants, and conference rooms. EMMA and devices in the environment will make use of this model to provide more context-appropriate responses to users' requests and actions. As similarly shown in other research [2], [4] this model will also contain logic to maintain user preferences, authorizations and permissions.

[6] Six settings between 0 and 90 degrees

The context model consists of three primary components: a knowledge-base, a reasoning engine, and an agent interface. The knowledge base was implemented in Protégé [12] as a database-backed OWL [3] ontology. The ontology characterizes the concepts and relations of objects specific to a meeting, users, and the environment (sensors and other room devices). We are currently testing reasoning engines that have been developed as plug-ins for Protégé, and are initially using the Algernon [8] plug-in. The agent interface is managed using the Java Management Extensions (JMX) [16], which provides a flexible distributed access to the knowledge base and reasoning engine using any of a number of protocols such as, SOAP, HTTP, RMI, or CORBA. EMMA, infrastructure systems, and external calendaring systems will interface to the context model to maintain and access room state information as well as user and meeting specific information.

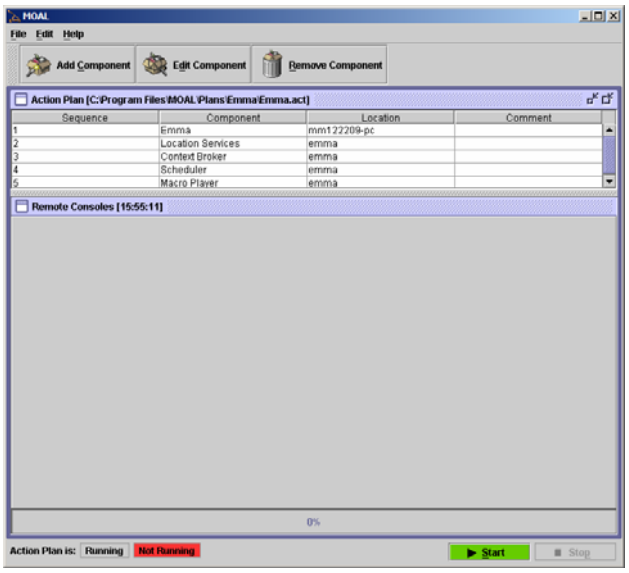

Figure 6. An action plan in MOAL.

Location Tracking

Making EMMA aware of participant identity and location enables the execution of zone-based actions. As an individual enters the "presenter's" zone, EMMA can display the individual's profile to the meeting attendees. We've investigated various ad-hoc and infrastructure based system location tracking systems. The Ubisense ultra wideband (UWB) based solution was selected for our implementation. The Ubisense system uses a network of ceiling-mounted sensors in conjunction with a central server. The associated Ubitags, attached to individuals or assets, are in constant communication with the network of sensors. Time of flight calculations are used to determine each Ubitag location. Since the system has an accuracy to within 5cm (95% of the time), we hope to also use it for gesture detection.

Component Launcher

Managing the various distributed systems in the ETR requires extensive user interaction. To launch the systems' distributed applications, the administrator must start processes on a number of different computers and in a specified sequence.

Mother of All Launchers (MOAL) was developed to allow the administrator to interact with a single computer to configure and launch applications across multiple computers. All software setup and configuration is managed by the MOAL in an *action plan* to enable rapid execution (or reproduction) of a particular distributed application. By automating the launching of distributed applications, human error is avoided and setup time is significantly reduced. This improves configuration management via a remote management interface, which gives the administrator centralized access to each of the application consoles.

CONCLUSIONS

In this paper we have discussed how our novel plug and play infrastructure enables experimentation with a number of intelligent UI designs. We presented an innovative instrumentation of the AV Control System (i.e., AMX) which has enabled UI research in the AV domain. The EMMA infrastructure has become a framework for exploring new interfaces to rich AV environments.

This system was originally created to test the hypothesis that a perceptual, human-like agent can streamline room operation and increase meeting productivity [6]. Therefore our next step is to conduct a series of usability comparisons between the commercial AV Control System and alternative EMMA interfaces. These experiments will be conducted with corporate volunteer users.

Future plans for development of context-aware capabilities include integrating the UbiSense ultra-wideband (UWB) location tracking system to locate people and room objects in real-time. UbiSense badges will be assigned at the biogate when users first enter the facility. In addition, we will extend the AV Controller to monitor microphone audio levels and assign lavaliere microphones to participants as they enter at the biogate. By doing so, we can extend EMMA to correlate participant audio level, audio duration, and participant location, in order to select the best camera preset for remote VTC participants (similar to the intelligent cameraman in [15]). We are also integrating EMMA with the corporate scheduling system, which will enable her to anticipate meetings and have access to presentation files to load into PowerPoint. These and others are opportunities to have EMMA correlate information similar to [4].

Continued enhancement of the EMMA infrastructure will ensure its scalability and extensibility. This will increase the room's ability to host concurrent and follow-on research in human-computer interaction.

ACKNOWLEDGMENTS

We thank our colleagues Robert Simmers, Chief Engineer of the MITRE Information Infrastructure, and Marnie Salisbury, Chief Engineer of MITRE's Center for

Innovative Computing and Informatics, for their vision and support in the design and development of the Experimental Team Room.

REFERENCES

1. Addlesee, M., Curwen, R., Hodges, S., Newman, J., Steggles, P., Ward, A. "Implementing a Sentient Computing System", IEEE Computer, August 2001.
2. Al-Muhtadi, J., Ranganathan, A., Campbell, R., and Mickunas, M.D. Cerberus: A Context-Aware Security Scheme for Smart Spaces. In 1st Annual IEEE International Conference on Pervasive Computing and Proceedings of the 1st Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'03), March 2003.
3. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A., OWL Web Ontology Language Reference, W3C Recommendation 10, February 2004.
4. Chen, H., Finin, T., Joshi, A. "A Context Broker for Building Smart Meeting Rooms" Proceedings, Proceedings of the Knowledge Representation and Ontology for Autonomous Systems Symposium, 2004 AAAI Spring Symposium, March 2004.
5. Hanssens, N., Kulkarni, A., Tuchinda, R. and Horton, T. "Building Agent-Based Intelligent Workspaces". In ABA Conference Proceedings. June 2002
6. Harper, L., Gertner, A., Van Guilder, J. "Perceptive Assistive Agents in Team Spaces", In IUI'04, Jan. 13-16, 2004.
7. Hattori, M., Cho, K., Ohsuga, A., and Isshiki, M. Context-Aware Agent Platform in Ubiquitous Environments and its Verification Tests. In 1st Annual IEEE International Conference on Pervasive Computing and Proceedings of the 1st Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'03), March 2003.
8. Hewett, M. Algernon-J Rule-Based Programming. http://algernon-j.sourceforge.net/
9. Judd, G., and Steenkiste, P. Providing Contextual Information to Pervasive Computing Applications. In 1st Annual IEEE International Conference on Pervasive Computing and Proceedings of the 1st Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'03), March 2003.
10. Lyytinen, K., Yoo, Y. "Issues and Challenges in Ubiquitous Computing", Communications of the ACM. December 2002, Volume 45, Issue 12, pp. 62-65.
11. Peters, S., and Shrobe, H. Using semantic networks for knowledge representation in an intelligent environment. In 1st Annual IEEE International Conference on Pervasive Computing and Proceedings of the 1st Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'03), March 2003.
12. Protégé-2000. The Protégé Ontology Editor and Knowledge Acquisition System. http://protege.stanford.edu/
13. Rich, C., Lesh, N., Rickel, J., Garland, A. "A Plug-in Architecture for Generating Collaborative Agent Responses", International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), ISBN: 1-58113-480-0, pp. 782-789, July 2002 (ACM Press).
14. Rich, C., Sidner, C. L., Lesh, N. B., "COLLAGEN: Applying Collaborative Discourse Theory to Human-Computer Interaction", Artificial Intelligence Magazine, Winter 2001 (Vol 22, Issue 4, pp. 15-25)
15. Shi, Yuanchun, Xie, W., Xu, G., Shi, R. Chen, E., Mao, Y., Liu, F. "The Smart Classroom: Merging Technologies for Seamless Tele-Education". IEEE Pervasive Computing, pp. 47- 55, Vol. 2, Issue 2,April-June 2003.
16. Sun Microsystems, Inc. Java™ Management Extensions Instrumentation and Agent Specification, v1.2, 2002.
17. Want, R., Perring, T., Borriello, G., Farkas, K. "Disappearing Hardware" Pervasive Computing, IEEE. Volume 1, Issue 1, Jan.-March 2002, pp. 36-47.