Design Principles of a Reactive Behavioral System for the Intelligent Room

Ajay Kulkarni

Artificial Intelligence Laboratory
200 Technology Sq. Rm 800
Cambridge, MA 02139
kulkarni@ai.mit.edu

The *Intelligent Room* (IRoom), an Intelligent Environment (IE) at the MIT Aritificial Intelligence (AI) Laboratory (Coen, 1998), is an experiment in human-computer interaction (HCI). Most of our research centers on a particular implementation of the IRoom, a conference room enhanced with cameras, microphones, and other sensors. Part of our HCI research is in enabling the Intelligent Room to "intelligently" react to human user behavior. This paper describes several design principles for an "intelligent" system to help fill that role. Such a system is called a *reactive behavioral system*.

The *Intelligent Room* (IRoom), an Intelligent Environment (IE) at the MIT Aritificial Intelligence (AI) Laboratory (Coen, 1998), is an experiment in human-computer interaction (HCI). Most of our research centers on a particular implementation of the IRoom, a conference room enhanced with cameras, microphones, and other sensors. Part of our HCI research is in enabling the Intelligent Room to "intelligently" react to human user behavior. This paper describes several design principles for an "intelligent" system to help fill that role. Such a system is called a *reactive behavioral system*. At the heart of an intelligent reactive behavioral system is a two-step process designed to assist users with their daily tasks: first, to understand the context of the current user- IRoom interaction; next, to adapt the IRoom's response to that context.

While this process requires a variety of information describing a user's behavior, this system need not capture this data itself. It can instead take advantage of existing Intelligent Room software and hardware for sensory input. Consequently, the design principles presented in this paper assume that the system is written in *Metaglue*, the Multi-Agent System of the Intelligent Room (Phillips, 1999; Coen et al., 1999). This paper first examines the previous reactive system in the Intelligent Room, establishing a need for a new, better system. For guidance in developing this new system, this paper then looks at reactive systems in other areas of research. Finally, this paper proposes a set of criteria for a new, better reactive behavioral system.

2. Previous Research

Examining the limitations of the previous IRoom reactive system will show us why the IRoom needs a better system. Reviewing previous research in reactive systems will help us articulate what qualities a reactive system should have. This section begins with a critique of the existing system, and then moves on to reactive systems in other research areas: mobile robots and self-configuring space systems.

2.1 Limitations of the Existing Reactive System

The previous reactive system in the Intelligent Room provided a fairly simplistic method of creating IRoom reactions. Basic reactions were encoded in JESS (Friedman- Hill, 2001), a rule-based programming language written in Java. Rule-based programming languages are based on a database of *facts* and a set of *rules*. Rules are triggered on the presence (or absence) of certain rules. In the Intelligent Room, various conditions about the environment and users were encoded as facts. Similarly, an IRoom reaction to those conditions was encoded as a rule. These facts and rules are often organized into different files called *scripts*. The basic deficiency of the rule-based system is its lack of flexibility. Imagine the following scenario:

- User enters IRoom;
- IRoom detects entry through visual or other sensory cues;
- IRoom turns on the lights.

If the IRoom was empty before the user entered, then this behavior is correct. Now imagine that before the user entered, a group of users were watching a movie or a presentation with the lights down. In this situation, turning on the lights would disrupt the activity of the users already present; the behavior is incorrect.

One solution to this problem would be to require the “lights” rule to rely on the “empty room” fact. But let’s imagine this process for another problem: let’s say that the user also wants the lights to turn on whenever his supervisor enters, regardless of what’s occuring in the IRoom. To solve this problem, he’d have to add another “not supervisor” fact to that same rule. These examples should highlight that this appraoch only solves one appraoch at a time. A better approach would understand that certain reactions apply in some contexts but not others.

The source of this problem is the rule-based system’s inability to force the individual designing the reactoin to organize his or her code. As the case-by-case approach is used to allow the IRoom to handle more situations, more rules and facts are added without any enforced systematic regulations. As the IRoom becomes more robust, this approach makes the reactive system cluttered. To understand a given reaction, the user would have to trace through all the rules and facts, which may be in different scripts.

In addition, transferring the reactions across different implementations of the Intelligent Room becomes difficult. Two different IRooms will share some reactions, but as one IRoom may serve as a conference room and the other as a living room, many of their reactions will vary. For example, the “Intelligent Conference Room” would require the ability to react to someone giving a presentation, while the “Intelligent Living Room” would require the ability to react to someone hosting a party. In creating reactions for the different IRooms, it would be helpful for the developer to be able to create separate “Presentation” and “Host Guests” reaction modules.

These limitations of Jess, the rule-based system used for IRoom reactivity, make evident the need for a better reactive behavioral system for the Intelligent Room. In particular, this system should possess the following characteristics: an understanding of the context of user actions, a structure that enforces a systematic method of organizing IRoom reactions, and an ability to allow the creation of separate reaction modules for different activities.

2.2 Sources of Inspiration

Existing research in other areas can inspire us in our creation of a better reactive behavioral system for the Intelligent Room. In this paper, successful reactive systems in Mobile Robots and Self-Configuring Space Systems serve as good sources of inspiration. While the details of these systems are specific to their own areas, their design choices and overall thought processes are very valuable.

2.2.1 MOBILE ROBOTS

If we look at the Intelligent Room as an “immobile robot,” the similarities between reactive systems in Mobile Robots and in the IRoom become clear. The Intelligent Room, like a mobile robot, is subject to external influences to which it must react. However, as size and power consumption (among other differences) are important in the design of a mobile robot (but not in the IRoom), this analogy can only be partially extended. In the area of Mobile Robots, we look at *Subsumption*, a layered architecture developed by Rodney Brooks (Brooks, 1986). Brooks is now Director of the Artificial Intelligence (AI) Lab at MIT. The Subsumption architecture manages to allow a robot to safely meander through an unfamiliar surrounding. The control system is achieved by various *levels of competence*, where a level of competence is “an informal specification of a desired class of behaviors for a robot over all environments it will encounter.” In addition, a higher level can override, or *subsume*, a lower level by inhibiting its outputs. By default however, lower levels will continue to respond even when higher levels are added.

By segmenting behaviors into different layers, with higher priority layers overriding those with lower priority, Subsumption creates a mobile robot with a robust and complex behavioral pattern. A similar approach for the Intelligent Room could provide it with a robust and complex reactive behavioral system. This segmentation would create a forced organization of behaviors that the earlier rule-based system lacked. A segmentation based on activity would allow for activity-based reaction modules. Subsuming layers of reactive behaviors could help integrate many reaction modules into a coherent response.

2.2.2 SELF-CONFIGURING SPACE SYSTEMS

The "immobile robot" analogy also makes it easy to compare reactive system research in the Intelligent Room to that in Space Systems. Researchers in field of Self-Configuring Space Systems have already considered their control systems as immobile robots (Williams & Nayak, 1996a). According to Brian Williams, also currently at the MIT AI Lab, "the focus of the attention of immobile robots is directed inward, toward maintaining their internal structure, as opposed to traditional robots, whose focus is on exploring and manipulating their external environment."

These control systems are in charge of goal-directed planning for a spacecraft system. A spacecraft system requires reactive behaviors that help withstand the hostile space environment. It needs to react to temperature changes, collisions from small particles, malfunctioning internal components, and other similar problems. When a problem arises, due the inaccessibility of space, human intervention is fairly impossible; a self-monitoring system is necessary. Fault detection and replanning are important aspects of these systems.

Livingstone, one such control system, uses a mode transitions model to address this problem (Williams & Nayak, 1996b). By understanding the mode that the system is currently in, the system can accordingly adapt its actions. For example, say that a fault F is more likely in mode A or mode B. In accounting for fault F, the system would then take into account which mode it is currently in.

This mode-transitions model forces the control system to recognize the context of the current failure. Applied to the Intelligent Room, this idea introduces the idea of using context to react to a user's actions. Maintaining *activity context* –information about the current ongoing task, or activity, performed in the IRoom–is consistent with the activity-based reaction modules inspired by Subsumption.

3. Design Principles

Previous research in Mobile Robots and Space Systems highlight several key requirements of a reactive behavioral system for the Intelligent Room: context maintenance, modular reactions, and a layering structure. This section also introduces three more requirements: an ability to customize reactions, a simplicity in designing new reactions, and an ease in adding reactions to a system. Following are six design principles that elaborate on these six requirements.

1. React to user behavior within the current activity context

As discussed earlier, the main problem of the rule based reactive system was its inability to understand that certain reactions apply in some contexts but not others. A better reactive behavioral system takes into account what the user is currently doing. In other words, a reactive behavioral system should maintain activity-context. For example, the system should understand that if a user enters an empty room, it should turn on the lights, but if the user enters a room where a presentation is being given, then it should not affect the lights.

An activity-context representation also needs to take into account the order of the current activities. When a user performs several tasks at the same time, he may be starting one within another. For example, a user may play a movie after entering the IRoom; or, he may enter the IRoom, start a meeting with other users, and then play the movie. The system should be able to react differently in these two scenarios. This principle is aided by the next two design principles.

2. Each reaction should be encoded in a *behavior*, an activity-based module

The Intelligent Room is implemented in different spaces; a reactive behavioral system should not be specific to one particular space. Different spaces would share reactions. For example, the "Intelligent Living Room" and the "Intelligent Conference Room" might share a reaction that turns on the lights when someone enters an empty room. Writing the same reaction for each of these rooms would be unnecessarily redundant.

If instead, that shared reaction was encapsulated into an object that could be inserted into the reactive system of each room, the redundancy would be eliminated. We call these modular objects *behaviors*. A behavior could be developed independent of any particular space, and then be included into any Intelligent Room implementation. Users would choose which reactions they want their own room to exhibit.

Forcing users to select individual reactions among all possible reactions is perhaps unreasonable. Users are most likely to select certain groups of reactions at one time. For example, users of the Conference Room are likely to select reactions particular to a meeting, while those of the Living Room would not. If the "meeting" reactions could be grouped into a "Meeting" behavior, then inserting (and deleting) these reactions becomes much easier.

The same argument applies to reactions particular to a presentation, a movie, and any other activity. We argue that reactions should organized into activity centric behaviors.

This organization also helps in maintaining activity context. As each activity is started, the system should activate the behavior–i.e., the set of reactions–corresponding to that activity. Only when a behavior is activated, can its reactions take place; the system reacts within the context of that activity.

3. The system needs an organizing structure for all behaviors, and a layering one for active behaviors

This principle, describing the need for an organized behavior architecture, is inspired by the Subsumption architecture. One of the problems with the rule-based reactive system was the lack of an enforced structure for organizing reactions. As more reactions–i.e., rules–were added, the system became cluttered and difficult to predict. A reactive behavioral system needs an organizing structure.

Organizing reactions into activity-centric behaviors, as described in the previous design principle, helps to create this structure. As each behavior is activated, its reactions start to respond to user actions.

This principle also specifies how to structure all the activated behaviors. In Subsumption, higher levels of behavior override lower levels. If we apply this idea to our "behaviors," we have a method of organizing the activated behaviors. As each behavior is activated, it should be layered on top of the older behaviors. In other words, as an activity is started, the reactions particular to that activity should take precedence over the reactions of older activities. For example, if a user starts showing a movie within a meeting, then the reactions in the "Movie" behavior should take precedence over those in the "Meeting" behavior.

Layering also helps in resolving conflicts between behaviors. For example, let's imagine an "Empty Room" behavior that specifies that, on user entry, the lights should turn on. Now, imagine a Movie behavior that specifies that the lights should not be affected if a movie is being watched. Then, when a user enters the room and starts a movie, the "Movie" reaction is in conflict with the "Empty Room" reaction. But with the layering, the Movie reaction takes precedence over the Empty Room reaction. In this way, this principle helps the first design principle: it allows the reactive system to maintain the order of activities in its representation of activity-context. Intertwined by the theme of activity-context, these first design principles are central requirements of the reactive system for the IRoom. The next three principles, while less important, complete the list of requirements.

4. The reactive system should be customizable by a user

Although the first three design principles help in mirroring user expectations, the system may still fall short. This problem is one of the main weaknesses of any reactive system. For example, a room that turns on the lights when it should not, would irritate users instead of helping them.

To avoid this problem, the reactive behavioral system should be customizable by users. In our example, a user should be able to tell the room not to turn on the lights. Perhaps this could be achieved by having a new "User" behavior that overrides the other active behaviors.
Asking a user to create his or her own behavior may be too much. A better solution would allow reactions to learn from user feedback. If a user consistently turns off the lights during a specific activity, then the behavior

corresponding to that activity should incorporate that feedback.

5. Designing a new set of reactions should be easy

If users need to design new behaviors to customize the room, as suggested by the previous principle, then designing behaviors needs to be made as simple as possible. Even without this requirement, extending the reactive system to incorporate more reactions and contexts should be easy.

Making the design of new behaviors easy for all users is a daunting task. But making the design of new behaviors easy for those users comfortable with programming–say, other Intelligent Room developers–is much easier. Learning how to program new behaviors in Metaglue, the language of IRoom software, should not be harder than basic Metaglue programming.

6. Adding a new behavior should not interrupt the system's operation

Allowing users to easily insert behaviors into their own room is important. A user should not be required to stop and restart her reactive behavioral system to add a new behavior. Just as she can literally plug in a new appliance into an outlet for power, she should be able to "plug" in a new behavior.

4. Conclusion

Currently, I am using these design principles to implement a reactive behavioral system for the Intelligent Room. A brief description of the current state of the system can be found in (Hanssens et al., 2002). While the true evaluation of these principles will occur as each is incorporated– successfully or unsuccessfully–into the system, a few potential problems are identified below. First, significant difficulty may arise in decomposing user actions into multiple "activities," or even in determining at what resolution to break an activity down into its corresponding "behaviors." For example, should there be a Movie behavior, or VCR and DVD behaviors? A Meeting behavior, or Presentation, Demo, Brainstorming, and Group Meeting behaviors? Adapting system reactions to user feedback will also be difficult to accomplish. How should the system gauge user happiness? How should the system resolve conflicting user preferences? Gauging user happiness, if possible, will provide true "artificial intelligence" to an intelligent space. Just like a good human secretary or servant, a digital one should not only respond to what you ask it to do, but over time it should learn what you like to do, and start doing it before you even ask.

References

Brooks, R. (1986). A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation, RA-2*.

Coen, M. (1998). Design Principles for Intelligent Environments. *Proceedings of AAAI'98*.

Coen, M., Phillips, B., Warshawsky, N., Weisman, L., Peters, S., & Finin, P. (1999). Meeting the Computational Needs of Intelligent Environments: The Metaglue System. *Managing Interactions in Smart Environments*. Dublin, Ireland.

Friedman-Hill, E. J. (2001). Jess, The Java Expert System Shell. http://herzberg.ca.sandia.gov/jess.

Hanssens, N., Kulkarni, A., Tuchinda, R., & Horton, T. (2002). Building Agent-Based Intelligent Workspaces. *Agents for Business Automation Conference Proceedings*. To appear.

Phillips, B. (1999). Metaglue: A Programming Language for Multi-Agent Systems. M. Eng Thesis, MIT, Cambridge, MA.

Williams, B. C., & Nayak, P. P. (1996a). Immobile Robots: AI in the New Millennium. *AI Magazine*.

Williams, B. C., & Nayak, P. P. (1996b). A Model-based Approach to Reactive Self-Configuring Systems. Recom Technologies, NASA Ames Research Center, MS 269-2.