Towards Socially Intelligent Agent-Building

Phoebe Sengers
Department of Computer Science and Program in Literary and Cultural Theory
Carnegie Mellon University
Pittsburgh, PA 15213 USA
phoebe.sengers@cs.cmu.edu

Abstract

If agents are going to interact socially with humans, they can not be simply correct; they must also be comprehensible. This article describes my thesis work-in-progress, which focuses on designing agents that can effectively express to users the goals and activities their designer has chosen for them. In order to build agents that express clearly the intentions of their designers, I am creating tools that allow a designer to build agents with respect, not merely to their internal goals and behaviors, but also to the *signs* that agents communicate to their audiences. By focusing on the agent within a social, communicative environment and explicitly considering the narrative structure of the agent's behaviors, builders can construct agents whose goals and intentions are clearly, effectively, and explicitly communicated; rather than, as has been done in the past, leaving the communication of these essential aspects of their agents over to a chance side-effect of their agents' internally-defined behaviors.

My thesis is that it is not only the agents themselves that must become social; the process of *designing* agents must become socially intelligent as well - by putting agents in the context of their designer and audience, we can allow designers to build agents that function more effectively in the social and cultural environment within which the agents will be inserted.

Motivation

In 1992, the Oz Project built the Woggles (Loyall & Bates 1993) (Figure 1), a system containing small, social, emotional agents that interact with each other and with the user. While building the agents, we took care to include a wide variety of behavior, which ranged from simple behavior like sighing and moping to relatively complex social behavior like follow-the-leader and fighting. At the same time, we made sure that the agents did not blindly follow the user but had a 'life of their own;' we hoped that this would make them more compelling personalities to get to know.

At the time, we believed that the individual behaviors of the agents were reasonably clear. After all, we (their builders) could usually tell what they were doing ("A-ha! It's small and flat! That means it is moping!"). Soon, however, we found that it was difficult for other people to be able to understand the behaviors and emotions we were trying to communicate through the Woggles. Users were at

Figure 1: The Woggles

a disadvantage because (among other things) they did not actually have the code memorized while they were watching the agents. Because we - the builders - thought in terms of the underlying behavior names in the code, we had thought the agents' behavior was clear. This had led us to neglect to some extent the *external* behavior of the agents. This problem was compounded because of our decision to make the agents more 'independent' of the user; the agents, in effect, often ignored the user, increasing the user's feeling of being left out of an alien domain. In the end, the problem with the Woggles' comprehensibility could be traced, at least in part, to the fact that we thought of our agents as separate from the audience's *experience* of the agents, and that we did not realize the effect of the biases we incurred from our own role in the construction of the agents.

In order to be able to explain the agents' behavior more easily, we added a display that gave the user a view into the internals of the agent by listing the name of the behavior in which each agent was engaging and its primary emotions. While this worked, it was clearly a stop-gap measure; it would be far better if the agents themselves could be designed to display their emotions, behaviors, and intentions clearly enough that no 'internal display' would be necessary. As work in agents intended for a general audience has progressed, making agents truly expressive has become a pressing problem (Blumberg 1996). The goal of the work presented here is to develop an approach to the construction of agents that takes into account the designer's role in constructing and audience's role in interpreting agents, and to

create tools based on this approach that will allow designers to develop agents that directly communicate their behaviors, intentions, and personality clearly and effectively to the audience.

Approach

The internalist perspective we had on the Woggles is only one example of a general trend in AI. For the past several decades, AI has had the luxury of operating within a relatively homogeneous segment of humanity, i.e. AI researchers. These researchers, mostly scientists, have often thought of agents basically as problem-solvers or tools for getting work done. Sharing a technical point of view on agents also means sharing a perspective on the proper way to evaluate agents. In order to evaluate these agents, one looks at the theoretical properties of the agent architecture, or runs experiments to see if the agent can solve the problems it is given accurately and perhaps efficiently. While there are clearly debates about which aspects of agents are most important to model (e.g. (Brooks 1991; Vera & Simon 1993)), in general computer scientists could agree on the way in which one should reason about agents: look at the code, or count the number of times the agent Does the Right Thing.

These days, this point of view is becoming less and less tenable. With the explosion of powerful personal computing and the Web and the accompanying popularization of high tech, there are many more people coming into contact with AI agents like Julia, Ahoy!, and Firefly as well as more and more advanced personal computing software such as Dogz, Creatures, and the Japanese Tamagotchi. These people cannot be expected to evaluate agents in the same way as AI researchers; they bring their own values and expectations to the agent-interaction experience. Rather than being focused on how rational an agent is, or on the intricacies of its construction, the general public is more likely to be interested in the overall impression of an agent and in how that agent fits into their lives. If old AI is chess players, shop floor schedulers, and planners only a scientist could love, new AI is agents with which non-expert persons come into contact, that have social effects, that can communicate with users and that may even be fun to have around.

If these programs are to be built effectively, they cannot simply solve mathematically formalized problems in the classically scientific ideal of a rational, if not outwardly understandable, manner. Rather, they must be able to communicate their goals, actions, and perhaps endearingly irrational emotions in a way that is designed to align with the social and cultural norms and expectations of the target audience. Given that those norms and expectations may be different from those of the builders, such agents may be built more effectively if the social and cultural environment for which the agent will be built is explicitly taken into account.

Socially Situated AI

My research program focuses on 'socially situated AI' (Sengers 1996a), i.e. methodologies for building agents that are situated not only in a physical environment but also in a social and cultural one. I believe on the basis of painful experience that it is not enough to build agents that try to be social, but that *the process of agent-building itself must become 'socially intelligent'* by being aware of the contexts into which agents will be inserted. Agent builders must not only design the internal structure of their agents; they must also design the interactive experience through which other people will come to know their agents.

Socially situated AI sees agents not as beings in a vacuum, but as representations which are to be communicated from an agent-builder to an audience. This point of view is inspired by recent work in believable agents such as (Reilly 1996; Loyall 1997; Wavish & Graham 1996; Blumberg & Galyean 1995), which focus more and more on the audience's perception of agents, rather than on an agent's correctness per se. By making the commitment that 'agentiness' is meant to be communicated, we can explicitly communicate to the audience what the agent is about, rather than assuming (perhaps incorrectly) that this will happen as a side-effect of the agent "doing the right thing." By building agents with an eye to their reception, builders can tailor their agents to maximize their effectiveness for their target audience. In this sense, agents built for social contexts can be more correct than purely rational, problem-solving style agents; they may actually get across the message for which they have been designed.

The rest of this paper will be a case study, taken from my thesis work (Sengers 1996b), of the application of these ideas to a specific technical problem, action-selection for autonomous agents. In this domain, I will identify a number of important technical concepts that arise when considering action-selection within a social context. Here I hope to demonstrate that taking the socially intelligent approach leads to changes in the entire way the technical problem of action-selection is constructed. Taking the band-aid approach of building social interaction on top of an already working system turns out not to be enough; the socially intelligent approach requires rethinking technical problems from the bottom up. While this work is not yet completed, I will attempt to make plausible that building agents using a socially intelligent version of action-selection, while requiring more thought, may pay off by leading to agents that are more comprehensible to the user.

Case Study: Socially Intelligent Action-Selection

The action-selection problem of behavior-based AI is traditionally framed in the following manner (e.g. (Blumberg 1994; Maes 1989)): *how can an agent, interacting with a changing environment, at every point choose an action that best fulfills its internal goals*? Once a particular algorithm is selected, an agent is programmed to continuously consider its range of actions, repeatedly selecting new actions and behaviors based on the agent's current drives and the world state. While this can deliver a reasonable quality of behavior in terms of fulfilling the agent's pre-programmed goals, it is *not* so good for communicating to the user what the agent is up to. The agent's behavior may in fact be quite confusing

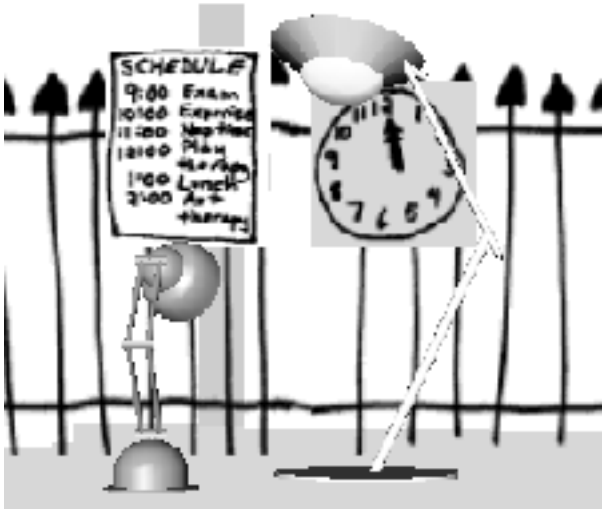

Figure 2: Patient and guard in the Industrial Graveyard

to a user, since the agent is continuously switching from one activity to another. While the user, when fortunate, may be able to identify each of the activities of the agent, these activities often seem to be randomly strung together, making it hard for the user to figure out the overall goals of the agent and why it is selecting the activities that it chooses.

Seen in a social context, the very definition of the action-selection problem, which focuses purely on the internals of the agent, is problematic. The action-selection problem can be more effectively redefined as what Tom Porter terms the 'action-expression' problem (Porter 1997): *what should the agent do at any point in order to best communicate its goals and activities to the user*? Instead of focusing on behavioral correctness per se, the action-expression problem is interested in increasing the quality of the agent's behavior and its comprehensibility for humans with which the agent will interact. While the action-selection problem is often addressed by building more and more complex decision-making algorithms into the agent's mind, i.e. selecting the Right behavior, the action-expression problem is less focused on *what* the agent does and instead interested in *how* the agent does it, i.e. engaging in and connecting its behaviors in an effective way.

In my thesis system, the action-expression problem is addressed by providing agent-builders with the following tools:

1. A *sign-management system* allows the agent (and builder) to keep track of what has been communicated to the user.
2. Behaviors are connected in a narrative sequence using *behavior transitions*, which explain to the user why an agent is changing from behavior to behavior, thereby making the overall goals of the agent clearer.
3. Behaviors are given *meta-level controls*, which makes behavior transitions easy to construct. In addition, it allows them to express to the user parts of the agent architecture that were formerly implicit (and therefore invisible).

Behavior: *Harass patient to follow scheduled activity*

1. Go to schedule
2. Read schedule
3. Look at clock
4. Look at schedule
5. Look at patient
6. Wait a moment for patient to comply
7. Look at schedule
8. Look at patient
9. Shake head
10. Approach patient menacingly

....

Figure 3: Example of a behavior and its signs

These tools are being implemented as changes to Hap (Loyall 1997; Loyall & Bates 1991), the Woggles' underlying agent architecture, and are being tested as part of a virtual environment, the Industrial Graveyard (Figure 2). In this world, a discarded lamp ekes out a marginal existence in a junkyard while being overseen by a nurse/guard from the Acme Sanitation and Healthcare Maintenance Organization. The goal of the implementation is to have the agents combine a variety of behaviors while making their behaviors, goals, and emotions clear to the user of the system, who takes on the role of an auditor overseeing the efficiency of the Acme-run junkyard.

Sign Management

The action-selection problem sees the behaviors with which the agent is programmed as activities which allow the agent to achieve its goals. In terms of the action-expression problem, behaviors are better thought of as 'activities to be communicated to the user.' This means that the fundamental units of behaviors are not physical actions that have effects in the world, but *signs* that have effects on the user. Figure 3 shows an example of a high-level behavior and the signs that are emitted during it.

These signs look somewhat like low-level actions, but there are important differences. Rather than corresponding to simple movements an agent can engage in, a sign corresponds to a set of such movements that carries meaning to a user. The "reading" sign, for example, combines a *set* of low-level actions as the lamp's head moves from left to right across each line of the schedule. More fundamentally, signs are different from both actions and behaviors in that they focus on what the user is likely to interpret, rather than what the agent is "actually" doing. When "reading," for example, the agent does not actually read the schedule at all (the locations of the lines and their contents are preprogrammed); it merely needs to give the *appearance* of reading.

Given that such signs are a basic unit of expressive behavior, an important component of action-expressive agents is a sign-management system that keeps track of the signs the agent has communicated. Using a sign-management system, the agent can make decisions about what is best to do based on what the user has seen the agent do, rather than on what the agent thinks it has done. In my thesis system, the sign-management system allows behaviors to post signs that have been expressed, and allows matching on arbitrary sequences of signs in subsequent behaviors, so that the signs the agent expresses can be used just like environmental stimuli and internal drives to affect subsequent behavior.

The sign-management system is designed to improve not only the agent's behavior but also the agent-builder's! By noting every time a sign is supposed to have been communicated by a behavior, builders' attention is focused on the problem of breaking a behavior into signs and then making sure that those signs are expressed - rather than merely assuming that a behavior that is called "follow-the-leader," which includes follow-the-leader-y actions, will also look like follow-the-leader to the user. The structure of the sign-management system encourages them to think about their behavior in terms of signs, and to construct appropriately expressive low-level behaviors to display those signs.

Behavior Transitions

When building an agent to be comprehensible, it is not enough to make the behaviors that the agent engages in clear. The user should also be able to understand *why* the agent is engaging in a behavior. The context of a behavior and the reasons an agent decides to engage in it have a great effect on the way in which a user will interpret the agent's personality and situation.

For example, suppose the agent is currently napping but decides to start exercising. This could be for various reasons:

1. It could be well-rested and ready for something strenuous.
2. It could feel guilty about napping because it was trying to stay in shape.
3. It could be engaging in an exercise marathon, but just work up after accidentally falling asleep in the middle of the marathon.
4. It could be threatened by another agent, who is forcing it to exercise against its will.

In each of these cases, the starting and ending behavior are the same, but the *connection* between the behaviors is vastly different and displays something about the agent's personality and situation. If the user is to get a complete conception of what the system is about, they need to understand not only the agent's behaviors but also how they are connected with each other.

Unfortunately, expressing the relationships between behaviors is not well-supported in most behavior-based systems (a complaint also raised in (Reilly 1996)). While these architectures do provide support for clear, expressive *individual* behaviors, they have problems when it comes to expressing relations *between* behaviors.

This is because a typical behavior-based system (e.g. (Blumberg 1994; Agre & Chapman 1987; Brooks 1986; Maes 1989)) treats each behavior separately; behaviors should refer as little as possible to other behaviors. Because of this design choice, a behavior, when turned on, does not know why it is turned on, who was turned on before it, or even who else is on at the same time. It knows only that its preconditions must have been met, but it does not know what other behaviors are possible and why it was chosen instead of them. Fundamentally, *behaviors do not know enough about other behaviors to be able to express to the user their interrelationships*. As a result, the agent switches abruptly between behaviors, causing them to seem to be randomly strung together, and leaving the interpretation of the *reasons* for the behavioral changes up to the often sadly overburdened imagination of the user.

In my thesis system, connections between behaviors are explicit, and are represented by special behaviors called *behavior transitions*. Behavior transitions function to explain why the agent's behavior is changing and what its intentions are. In the above examples, the agent could engage in the following transitions:

1. Yawn, stretch, bounce around, start exercising.
2. Wake up slowly. Look guilty. Sigh. Look at body. Sigh again. Start exercising slowly.
3. Wake with a start. Look around to see if anyone caught it napping. Tiredly start exercising again.
4. Wake slowly, then jump back upon seeing other agent. Quickly start exercising, tapering off as the other agent leaves.

Each of these behavior transitions sets the stage for the following behavior while making clear how the agent feels about what it is doing. Instead of jumping from behavior to behavior, the agent expresses the reasons for its abandoning the old behavior and anticipates the new behavior. By explicitly connecting behaviors with transitions, the behaviors are no longer randomly jumbled together, but organized into a story where one behavior naturally follows from another. Instead of simply engaging in apparent stimulus-response activity, the agent shows that there are reasons for its behavioral decisions, thereby giving the user more insight into its motivations. Behavior transitions show that the agent is truly conscious, a thinking being that considers, however briefly, before it acts.

Implementation: Meta-Level Controls

While the idea of behavior transitions is relatively straightforward, their implementation in a behavior-based system is not as simple as one might hope. This is because, as mentioned above, behaviors in these architectures are distinct entities which do not have access to each other. In order to handle conflicts between behaviors, the agent architecture will typically have an underlying (perhaps dis-

Transition behavior: *Reading to exercising*
Precondition: **Reading behavior is active**
Overseer has approached

1. **Delete reading behavior**
2. Look at Overseer
3. Look at sign
4. Show sudden shock reaction
5. Look at Overseer again
6. Do some quick, sloppy exercises
7. **Spawn** exercise behavior **with affect** $< frantic >$
8. **Add "Watch Overseer" subbehavior to exercise**
9. When Overseer leaves, **tell exercise behavior to be** $< lazy >$

Figure 4: Example of a transition behavior

tributed) action-selection mechanism whose sole responsibility is handling inter-behavior interactions. Previous behavior-based systems have, in fact, included more and more elegant, subtle, and refined action-selection mechanisms as part of their construction (see (Blumberg 1996) for a beautiful example). While deciding what to do, an agent built in these architectures may be able to consider a host of environmental and internal factors, weighing its previous use of a behavior vs. the likelihood of it succeeding vs. how well a behavior fulfills various internal goals of various importances, etc. Clearly these improvements have the chance of substantially improving an agent's intelligence with respect to fulfilling its goals in an uncertain environment. Sadly, however, much of the power of these improvements may be lost on the user: *because the mechanisms by which the agent decides what to do are part of the implicit architecture of the agent, they are not directly expressible to the user*.

The solution in my thesis system is to allow behaviors, when necessary, to affect one another directly, rather than having inter-behavior effects be implicit in the design of the underlying agent architecture. Behaviors are given *meta-level abilities* by which they can have access to more information about who has been selected when, and with which they can communicate information to each other. Specifically, behaviors are given the ability (1) to query which other behaviors have recently happened or are currently active, (2) to delete other behaviors, (3) to propose new behaviors, (4) to add new sub-behaviors to other behaviors, and (5) to change the internal variables that affect the way in which other behaviors are processed. Using these meta-level abilities, behavior transitions become simple to implement. An example transition behavior is in Figure 4; meta-level abilities are annotated in bold face.

Meta-level abilities are not only for creating behavior transitions, however; they give the agent builder more power to expose the inner workings of the agent by letting them access and therefore express aspects of behavior processing that other systems leave implicit. Rather than being a set of autonomous behaviors that each proceed independently with no understanding of how they fit into the big picture, behaviors in this system can check on and coordinate with each other. Because behaviors are no longer completely independent, they can coordinate to express a coherent story to the user, rather than each expressing something independently that they pray the other behaviors will not contradict.

Conclusion

In order to be able to function effectively in a social milieu that includes its users, a social agent will need to be able to communicate its intentions effectively to a user and to fulfill particular, culturally situated human social norms. To give an agent these kinds of abilities, it may be helpful for the agent designer to consciously design an agent with an eye to the way in which the agent will be interpreted. Socially situated AI is intended to help designers reach their audiences more effectively by providing them with tools to design not just the agent itself, but the audience's *experience* of the agent. By designing an agent with respect to the signs it emits, the author may be more certain that the audience for which s/he has designed the agent will be able to interpret the agent's behavior correctly. By connecting the agent's behaviors with transitions that explain the reasons the agent has chosen a particular behavior, the author can make the agent's intentions clear and reveal more about the agent's personality and values. By using meta-level controls, the author can coordinate the various behaviors s/he has designed so that they present a coherent overall picture to the user.

Experience with socially situated AI suggests that building agents that can function effectively in a social context is unlikely to be a simple add-on functionality but may affect the entire structure of an agent; even parts previously considered to be pure problem-solving may need to be altered to allow the abilities and goals of the agent to be clear to a user and to allow the agent to work effectively in an environment including specific human social norms. Action-selection is an example of a problem-solving algorithm whose very premises need to be questioned when thinking of the agent as a representation to be communicated to an audience; while action-selection is still important, action-expression - or building an agent to show clearly what it does and why it does it - turns out to be just as important for agents in a social context.

Sign management, behavior transitions, and meta-level controls are three ways in which socially aware agent building may allow agents to become more understandable to the user. In general, making agent-building socially aware means designers will be encouraged to think about how the behavior of their agents will be received by their target audience. Instead of merely hoping the audience will interpret the agent's behavior correctly, designers are given tools that allow them to express the agent's goals and intentions directly to the audience. This way, designers may be more likely to build agents that can address their audience and respect their audience's social conventions, rather than being

pure problem solvers who cannot care or reason about what their social partners think of them.

Acknowledgments

This work was done as part of Joseph Bates's Oz Project, and was funded by the ONR through grant N00014-92-J-1298. I have also benefited greatly from conversations with Camilla Griggers, Jill Lehman, Bryan Loyall, Michael Mateas, and Simon Penny; all opinions expressed, however, are mine.

References

Agre, P. E., and Chapman, D. 1987. Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*.

Blumberg, B., and Galyean, T. A. 1995. Multi-level direction of autonomous creatures for real-time virtual environments. In *Proceedings of SIGGraph*.

Blumberg, B. 1994. Action-selection in hamsterdam: Lessons from ethology. In *Proceedings of the 3rd International Conference on the Simulation of Adaptive Behavior*.

Blumberg, B. 1996. *Old Tricks, New Dogs: Ethology and Interactive Creatures*. Ph.D. Dissertation, MIT Media Lab, Cambridge, MA.

Brooks, R. 1986. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* RA-2:14–23.

Brooks, R. A. 1991. Intelligence without reason. Technical Report AI Memo 1293, MIT AI Lab.

Loyall, A. B., and Bates, J. 1991. Hap: A reactive, adaptive architecture for agents. Technical Report CMU-CS-91-147, Carnegie Mellon University.

Loyall, A. B., and Bates, J. 1993. Real-time control of animated broad agents. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*.

Loyall, A. B. 1997. *Believable Agents: Building Interactive Personalities*. Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh. CMU-CS-97-123.

Maes, P. 1989. How to do the right thing. AI Memo 1180, MIT AI Laboratory.

Porter, T. 1997. Depicting perception, thought, and action in *toy story*. In *First International Conference on Autonomous Agents*. Invited Talk.

Reilly, S. N. 1996. *Believable Social and Emotional Agents*. Ph.D. Dissertation, Carnegie Mellon University. CMU-CS-96-138.

Sengers, P. 1996a. Socially situated ai: What it is and why it matters. In Kitano, H., ed., *AAAI-96 Workshop on AI / A-Life and Entertainment*. Menlo Park, CA: AAAI Press. AAAI Technical Report WS-96-03.

Sengers, P. 1996b. Symptom management for schizophrenic agents. In *AAAI-96*, volume 2, 1369. Menlo Park, CA: AAAI Press.

Vera, A., and Simon, H. A. 1993. Situated action: A symbolic interpretation. *Cognitive Science* 17:1–6.

Wavish, P., and Graham, M. 1996. A situated action approach to implementing characters in computer games. *AAI* 10.