Resolving conflicts in collaborative human-computer interaction (extended abstract)

Robert St. Amant
Department of Computer Science
North Carolina State University
Box 7534
Raleigh, NC 27695-7534
stamant@eos.ncsu.edu

In this extended abstract we discuss some of the issues surrounding conflict resolution in the interaction between software agents and human agents. Our discussion is drawn mainly from research in human-computer interaction (HCI) and intelligent user interfaces (IUI.) We describe some simple conflict management strategies common in these fields, and some of the mechanisms collaborative systems use to reduce the complexity of the process. We hope that this informal work will identify useful implications for conflict resolution in autonomous agents research.

Managing conflicts in human-computer interaction poses a set of challenges beyond those encountered in dealing strictly with software agents. Some familiar issues arise, such as asymmetry in agent capabilities and responsibilities, distributed processing and information storage, limited resources, and a high cost of exchanging information. Human-computer interaction also has some specialized characteristics:

- Lack of a common, shared representation. Software agents must communicate with users through graphics, natural language, and other media that are rich in implicit connotations, in addition to the direct information conveyed. The information picked up by a user from the interaction may differ significantly from that intended by the software agent, and vice versa.
- Unstructured knowledge. Much of what the user contributes to an exchange is difficult or impossible to encode in a formal representation, practically speaking. This may mean, for example, that some user decisions, whether correct or incorrect, cannot be justified in formal terms by any participant in the collaboration.
- Limited mutual comprehension. A software agent can model a human agent only to a limited and unreliable degree, and the reverse is usually also true. This can make it difficult for an agent to predict responses to its actions or even to communicate effectively.

These and related complexities have led to research methods in HCI that deal with abstractions at a higher level than the concept of conflict. HCI researchers commonly work with interaction strategies that implicitly incorporate static methods for dealing with conflict. Fortunately, problems of conflict management can be offset to some extent by the properties of intelligent interactive environments. To see this we will need to distinguish two types of interactive systems, both relatively common in intelligent user interfaces research.

On one side we find the familiar agents approach, in which a system with relatively high autonomy observes and communicates with the user as an assistant or even a peer. This is where conventional AI agents fit, able to make decisions and construct solutions to problems with little guidance necessary from a human user. On the other side are systems following the *model world* approach, derived from the direct manipulation paradigm associated with graphical user interfaces. A model world provides the user with an intelligent environment in which to work. The active components of the model world can be treated as tools, with little or no autonomy, though the system as a whole may have considerable autonomy in structuring the environment to facilitate problem-solving. Interaction based on the model world approach incorporates a number of design heuristics common to direct manipulation interfaces in general. These include the following, which may hold over some episodes (but not the entirety) of the interaction:

- The user and the system assume different roles, in which the user is able to command the system directly and can exercise extensive direct control over objects and actions in the environment.
- The system's behavior (and to some degree its internal state) is constrained to be predictable by the user.
- The viewer does not regard the system as a decision-making agent. Although the system may make decisions internally, it conveys these decisions implicitly via changes to the environment.
- Information relevant to the interaction is given in graphical and textual summaries of the current state of the environment; in other words, relevant shared context is largely made explicit.

The agent approach and the model world approach are not necessarily disjoint; most intelligent user interfaces have characteristics of both.

The benefit of adopting the model world approach in a collaborative system is that the system can manage the problem-solving environment so as to constrain and simplify conflict detection and resolution.

Let's take as an example the appointment scheduling problem. (Multi-user/multi-agent interfaces are beyond the scope of this abstract.) By imposing structure on its interaction with a user, an agent that relies on a model world can heuristically simplify or avoid many of the problems of conflict identification and resolution.

These simplifications are not without cost; it is very common for the system simply to abdicate responsibility to the user on detecting any sort of problem. Consider, for example, a "wizard" interface for making appointments. At each step the user is given relevant information about a potential conflict, such as overbooking a single time slot. If a problem arises the user is free to cancel the procedure, but the system takes no action on its own to repair the problem. This is a trivial one-shot example of conflict resolution.

Other, slightly more sophisticated strategies are possible, however, that remain within the strictest limitations of the model world approach. We can treat the agent interacting with user (or rather, the agent acting through the model world environment) as interactive search through a space of decisions, where some are made by the user and some by the system, possibly without the user's direct observation. The system can apply various mechanisms to reduce the probability of conflicts occurring.

Structure the interaction such that the probability of conflicts arising is minimized. In the wizard interface suggested above, the system might suggest time slots for a desired appointment, waiting on the user's approval. The suggestions might be selected using some best-fit heuristic to delay the occurrence of conflicts.

Structure the interaction such that the cost of dealing with potential conflicts is minimized. The goal is similar to the delaying tactic above, but here actions might be taken that reduce overall cost.

The system can also reduce the cost of resolving conflicts when they arise.

On detection of conflict, present justification for a given action or belief. This is commonly seen in decision support and expert systems, which may present a trace of rule firings or a network of relations. The system may or may not have be flexible in its response, deferring to the user or simply continuing without change. While simplistic, this strategy is often sufficient to resolve conflict by forcing the user to change his or her belief.

On detection of conflict, present relevant environment information. This is a comparable but distinct strategy from the above, in which the system lays out relevant features of the problem (rather than its solution in more detail.) Again, while simplistic, this often serves to resolve conflict.

Structure the interaction such that the user is presented with a restricted set of actions, eliminating those likely to lead to a conflict.

Structure the interaction such that the perceived cost of taking an action is reduced if it is less likely to lead to a conflict.

These are not simply abstractions tailored to the concept of conflict; we see such mechanisms in existing systems aimed directly at facilitating collaboration between the user and the system. In our presentation these basics and related issues will be expanded and discussed in more detail.