CONVERSATIONAL AGENTS

Michael N. Huhns • University of Southern Carolina • huhns@sc.edu
Munindar P. Singh • North Carolina State University • singh@ncsu.edu

When you build or buy an agent for the Web, you want it to perform as well as possible. Increasingly, this means your agent should take advantage not only of the Web's information resources, but also of all the other agents that might be operating there.

And soon there will be a lot of them. The computational architecture that seems to be evolving out of an informationally chaotic Web consists of numerous agents representing users, services, and data resources. A typical pattern of usage shows resource agents advertising to the services, user agents using the services to find resource agents and then querying the resource agents or the information needed.

Agents representing different users might collaborate in finding and fusing information, but compete for goods and resources. Similarly, service agents may collaborate or compete with user, resource, and other service agents. Whether they are collaborators or competitors, the agents must interact purposefully with each other. Most purposeful interactions—whether to inform, query, or deceive—require the agents to talk to one another, and talking intelligibly requires a mutually understood language.

LINGUA FRANCA

Agent projects investigated languages for many years. Early on, agents were local to each project, and their languages were mostly idiosyncratic. Now the challenge is to have your agent talk to any other agent, not just to your own. The obvious solution is a lingua franca—ideally, all the agents that implement the (same) lingua franca will be mutually intelligible.

What should such a lingua franca be like? It needs an unambiguous syntax, so the agents can all parse sentences the same way. It should have a well-defined semantics or meaning, so the agents can all understand sentences the same way. It should be well known, so different designers can implement it and so it has a chance of encountering another agent who knows the same language. And it should have the expressive power to communicate the kinds of things agents may need to say to one another.

As you can see, this is a nontrivial list of requirements. Coming up with an unambiguous syntax is the easiest. Being well known is not so much a technical as a political requirement on a language—committees and consortia are expected to ensure that their results will be widely adopted. Ensuring the expressive power of a language is potentially very difficult, but we can borrow a lot of good ideas from the study of human language.

That leaves the question of meaning—no one has quite figured that out, and we will soon explain why.

THE CURRENT CHOICES

So what language should you give your agent (or teach it—the subject of a future column) so that it will understand and be understood? At the moment, there are two main choices: Knowledge Query and Manipulation Language (KQML)* and the Foundation for Intelligent Agents (FIPA)* specification. Both are attempts to separate the domain-dependent part of a communication—the content—from the domain-independent part—the packaging—and then to provide a standard for the domain-independent part.

Both are based on "speech acts" (see the sidebar). Neither has been completed, although KQML was begun earlier and is more mature. Both are being formalized, so that independently developed agents conforming to the standard will understand each other. Both have prototype implementations, initial applications, and their own adherents.

KQML

The Knowledge Query and Manipulation Language (KQML) was defined under the DARPA-sponsored Knowledge Sharing Effort.[1] KQML assumes a layered architecture with the functionality for message transport or communication at the bottom and, at the top, the content to be specified by the applications—typically in some formal language, such as the Knowledge Interchange Format (KIF)* or Structured Query Language (SQL) for databases. Between these two layers are the primitives with which agents can exchange meaningful message. Basically, KQML provides a way to structure the messages, but lets the agent designers worry about what is in them.

In a slight misuse of terminology (see the sidebar), we say that KQML provides between 31 and 41 "performa-

SPEECH ACTS

Speech acts have to do with communication—they have nothing to do with speech as such, except that human communication often involves speech.

Speech act theory was invented in the fifties and sixties to help understand human language. The idea was that with language you not only make statements, but also perform actions.[1] For example, when you request something, you don't just report on a request; you actually cause the request. When a justice of the peace declares a couple man and wife, she is not reporting on their marital status, but changing it.

The stylized syntactic form for speech acts that begins "I hereby request . . ." or "I hereby declare . . ." is called a *performative*. With a performative, literally, saying it makes it so. Verbs that cannot be put in this form are not speech acts. For example, "solve" is not a performative, because "I hereby solve this problem" just doesn't work out—or math students would be a much happier lot!

Several hundred verbs in English correspond to performatives. This obviously calls for classifications, and many have been given. For most computing purposes, speech acts are classified into:

- assertives (informing),
- directives (requesting or querying),
- commissives (promising),
- prohibitives,
- declaratives (causing events in themselves as, for example, the justice of the peace does in a marriage ceremony), and
- expressives (expressing emotions).

In natural language, it is not easy to determine what speech act is being performed. For example, if Mike says "It's cold here," he might be telling you about the temperature, or he may be requesting you to turn up the heat. This is one of the reasons why natural language is tricky.

In artificial languages, we do not have this problem. However, the meanings of speech acts depend on what the agents believe, intend, and know how to perform and on the society they live in. It is difficult to characterize meaning because all of these things are themselves tricky.

REFERENCE

1. J.L. Austin, *How to Do Things with Words,* Clarendon Press, Oxford, UK, 1962.

tives."[2] Although varied, KQML's performatives are all assertives and directives. They fall into a few major classes. One class—which includes "tell," "evaluate," and "subscribe"—is geared toward the actual communication of content. Another class includes primitives to control the flow of information, for example, by sending an explicit "next" each time another answer is wanted from a source agent. A third class supports recruiting agents and performing other brokering and facilitating functions.

KQML assumes the message transport is reliable and preserves the order of messages, but does not guarantee delivery times. For this reason, the underlying paradigm of communication is asynchronous. At the application level, the effect of synchronous communication is achieved by tagging messages to relate them. For example, you can link responses to queries. In this way, KQML supports some elementary interaction protocols, although more sophisticated protocols must be built external to KQML.

The KQML semantics is given informally, although formalizations are under way. KQML agents are assumed to have a virtual knowledge base (VKB) containing beliefs and goals. They can communicate about their own and others' virtual knowledge bases. Thus, a "tell" reports on the contents of the sender's VKB, and an "evaluate" directs the recipient to report on its VKB.

FIPA

The Foundation for Intelligent Physical Agents (FIPA), a consortium with a number of European and Asian, and some American members, proposed an alternative speech-act-based language. FIPA specifies a smaller set of performatives than KQML—just six—but they can be composed to enable agents to express more complex beliefs and expectations. For example, an agent can request to be informed about one of several alternatives. The performatives deal explicitly with actions, so requests are for communicative actions to be done by the message recipient.

The FIPA specification comes with a formal semantics. This, in general, is a strong point. For example, it guarantees that there is only one way to interpret an agent's communications. Without this guarantee, agents (and their designers) would have to choose among several alternatives, leading to potential misunderstandings and unnecessary work.

However, the FIPA semantics is specific to a certain kind of agent, behaving in a certain manner. The agents must be sincere and must not make assertions

✶URLs from these pages:
KQML • www.cs.umbc.edu/kqml/kqmlspec/spec.html
FIPA • drogo.cselt.stet.it/fipa/
KIF • hpdce.stanford.edu/newkif.html
JAT • cdr.stanford.edu/ABE/JavaAgent.html

unless they are absolutely certain about their belief. For example, FIPA restricts the agents regarding when they can inform another agent (when they know something and the other party does not). This would eliminate broadcasts, because an agent would not know that every listener did not know or was uncertain about its assertion. Moreover, an engineer agent could not discuss design possibilities, and a politician agent might only make assertions when he is sure his audience already believes what he is saying! This overly legislates the society in which the agents live. We know which is the right behavior for our agents—it is not the function of the protocol to determine the behavior.

EVALUATION

So which should you choose? KQML still suffers from poorly defined semantics. As a result, each of the many KQML implementations seems unique. This makes communication difficult, and your KQML agent might not be understood. Nor has security been addressed in KQML. There are no provisions to authenticate agents or guarantee the integrity of KQML messages.

The FIPA specification, by contrast, attempts to formalize the semantics and provides a security model. However, in view of its recency, it has not been widely tested or adopted. As a result, your FIPA agent might not find anyone to communicate with.

The essential semantic difference is that with the FIPA specification, Agent A tells Agent B something, because A wants B to believe it, whereas with KQML, Agent A tells Agent B something, because A wants B to know A believes it. KQML is, in this sense, more cautious.

If you don't have the option of waiting for a consensus to emerge, we suggest that you choose KQML because of its current lead in market share. Then you can hope for continued effort in standardizing its semantics. And when dealing with agents designed by others, be sure to use the same dialect!

SYSTEM OF THE BIMONTH

There is a shell available for download that lets you construct Java applets that can converse in KQML. It's called the Java Active Template.* Check it out! ■

REFERENCES

1. R.S. Patil, et al., "The DARPA Knowledge Sharing Effort: Progress Report," *Proc.Third Int'l Conf. on Principles of Knowledge Representation and Reasoning.*
2. J. Mayfield, Y. Labrou, and T. Finin, "Evaluation of KQML as an Agent Communication Language," in *Intelligent Agents II: Agent Theories, Architectures, and Languages*, J.P. Muller and M. Tambe, eds, Springer-Verlag, Berlin, 1996.