Dialogue Systems Require a Reactive Task Architecture

Will Fitzgerald
R. James Firby
I/NET, Inc.
AAAI Spring Symposium 2000

July 2001 Printing

Abstract

Natural language dialogue is one type of planning and task execution. Task execution typically requires that an agent be able to act reactively in its dynamic, unpredictable world. This is all the more true in dialogue planning and execution, in which agents' goals and beliefs are often changing or opaque. Dialogue planning and execution is reactive in nature. Therefore, systems for dialogue interaction should have at their core a reactive task execution architecture.

This paper was originally entitled "Dialog is task execution; task execution is best done reactively; therefore, dialog systems call for a reactive task execution architecture"

Dialogue is task execution

Throughout the history of natural language dialogue research, taking part in dialogue has been seen as a planful activity; that is, engaging in dialogue has been viewed (from various research perspectives) as an activity which involves agents' goals, and plans to achieve those goals. Engaging in a dialogue is engaging in a particular kind of task.

Dialogue execution shares many of the same requirements with other kinds of task execution. For example:

- Agents can have multiple goals active simultaneously,
- Agents can be attempting to achieve these goals simultaneously,
- The world in which agents act is often unpredictable and often changing,
- Tasks and goals can be described hierarchically,
- Resources available for achieving goals vary,
- Goals can have differing priorities, which may vary dynamically,
- Tasks will sometimes fail to achieve their goals,
- Multiple agents will be attempting to achieve their goals simultaneously, perhaps collaboratively,

and so on.

Although it is true that engaging in natural language dialogue has different qualities from non-dialogue tasks (for example, natural language is rife with vague descriptions whose meanings must be negotiated; dialogue almost immediately requires one to address issues of other agents' beliefs, goals and intentions), it is equally true that many non-dialogue tasks share these same qualities. For example, we have argued that descriptions have value for tasks beyond natural language dialogue (Fitzgerald and Firby 1996). In the end, dialogue planning and execution is task planning and execution.

Task execution is best done reactively

It is clear that task execution must be done reactively; that is, as an agent executes tasks, the agent must be able to react to changes in the world, changes in the agent's goals, new information gained (perhaps as a result of achieving information-gathering goals).

A computational architecture for task execution should have a clear model of the reactive nature of task execution. The Reactive Action Plan (RAP) System (Firby, 1989) is one such system. The RAP System has been successfully used to control task execution in a number of complex domains; the RAP System is used as part of the 3T (three-tier) architecture at use in NASA's Johnson Space Center. The RAP System provides the "middle tier" in the 3T architecture; with the "top tier" being a planning system, and the "bottom tier" being the skill system controlling and receiving input from the real world.

Dialogue systems call for a reactive task execution architecture

Dialogue systems, as instances of task execution systems call for a reactive task execution architecture. The multi-agent nature of dialogue makes this imperative, as an agent's interlocutor is not under the agent's control, and the interlocutor's beliefs, goals, and actions are especially opaque, and the agents are attempting to achieve their goals simultaneously, with only loose synchrony.

In addition, an agent engages in dialogue to help achieve other goals, not just to engage in dialogue. Hence, dialogue tasks must take place in the context of other, non-dialogue, tasks, sharing state, and "cognitive" and physical resources.

The Dynamic Predictive Memory Architecture

We have built a computational architecture, called the Dynamic Predictive Memory Architecture, or DPMA (Fitzgerald and Firby, 1998). DPMA is comprised of:

1. The RAP system, which controls all task execution, including dialogue.
2. The Conceptual Memory system, a descriptive logic system in the KL-ONE tradition (Fitzgerald, 1997),
3. The Conceptual Memory Parser, a semantically oriented parser in the DMAP tradition (Fitzgerald, 1997; Martin 1993).

and, of course, the interactions among these subsystems, skill systems, and planning systems.

One interesting aspect of the DPMA Conceptual Memory Parser is that it essentially acts as a skill-subsystem whose job is to take natural language input and pass, to the RAP task execution system, descriptions of utterances. It is the responsibility of

plans written in the RAP System language to use these descriptions for disambiguation, task action, etc. Descriptions resulting from parsing are handled like other forms of input.

The DPMA AERCam Testbed

AERCam is a free-flying robotic space camera developed at NASA's Johnson Space Center. We developed the DPMA AERCam Testbed as an environment to test and enhance DPMA. In the DPMA AERCam Testbed, a human operator can interact with a DPMA-enabled control station to fly the (simulated) camera around a space shuttle. It also allows the operator to control the camera via joystick operations. DPMA proved to be a robust architecture for building a flexibly autonomous system that can engage in dialogue with a human. The DPMA AERCam controller can understand commands for translation and rotation, commands to move to various locations around the space shuttle, query the camera's internal state and the state of the world around it, control the camera's other devices (lights), monitor and warn for important states, provide for shared control, do object disambiguation, and so on. Here is a typical dialogue:

Astronaut:	Go to the inboard elevon.
AERCam CS:	There are two inboard elevons. Which one?
Astronaut:	The port one.
AERCam CS:	[Commands the camera to go to the port, inboard elevon.]
Astronaut:	Look at the outboard elevon.
AERCam CS:	[Commands the camera to rotate, so its camera points at the port, outboard elevon.]

Although "outboard elevon" is, strictly, ambiguous, the AERCam control station's plans for disambiguation take physical proximity into account. Note, too, the smooth interleaving of dialogue tasks and movement tasks.

Conclusions

It is an important insight to realize that dialogue planning and execution are not essentially different from other planning and task execution domains. It is true that building dialogue systems immediately bring to the fore such issues as other agents' beliefs, desires and intentions and ambiguity/vagueness. On the one hand, however, these issues are important for other domains such as story understanding, building educational systems, and so forth. On the other hand, the same issues that arise from executing tasks in a dynamic, unpredictable world arise in dialogue planning and execution. Ambiguity and vagueness arise in other sensor modalities, too. For example, an agent may have to change its point of view to eliminate ambiguity in a visual scene. This strongly argues for a reactive task execution architecture for dialogue systems. We have built an architecture, the Dynamic Predictive Memory Architecture, which unifies task execution, descriptive logic and conceptual parsing in a robust dialogue system that cleanly interacts with an agent's other tasks. Any architecture, though, will have to address the issues that reactive task execution systems are designed to solve.

Bibliography

Firby, R. J. 1995. Adaptive Execution in Complex Dynamic Domains, Yale University Technical Report YALEU/CSD/RR #672.

Fitzgerald, W. 1997. The Conceptual Memory Manual, I/NET Technical Report, October, 1997.

Fitzgerald, W. 1997. The Conceptual Memory Parser Manual, I/NET Technical Report, October 1997.

Fitzgerald, W. and Firby, R. J. 1998. The Dynamic Predictive Memory Architecture: Integrating Language with Task Execution. Proceedings of IEEE Symposia on Intelligence and Systems (SIS '98), May 21-23, Washington, D.C.

Fitzgerald, W and Firby, R. J. 1996. Item descriptions add value to plans. Proceedings of The Workshop on Plan Execution: Problems and Issues of the AAAI Fall Symposium, San Mateo, CA: Morgan Kaufmann.

Martin, C. 1993. Direct Memory Access Parsing. Ph.D. diss., Dept. of Computer Science, Yale Univ.

I/NET's Conversational Interface: http://www.inetmi.com/ci/ci.html.