Proceedings of ICSLP'00, Vol. II, pp. 142–145, Beijing, China, October, 2000.

ORION: From On-line Interaction to Off-line Delegation

Stephanie Seneff, Chian Chuu and D. Scott Cyphers[*]

MIT Laboratory for Computer Science
`{seneff, cchuu, cyphers}@lcs.mit.edu`

Abstract. This paper introduces ORION, a conversational system that performs off-line tasks and initiates later contact with a user at a pre-negotiated time. Orion has two major episodes of activity: the enrollment of new tasks and the execution of pending tasks. The task manager periodically checks the pending tasks and updates their status, sending off requests to other servers for information and possibly launching a phone call when a particular task has reached its trigger time. A separate user interface engages in a dialogue with a user to enroll new tasks and/or update existing tasks. ORION is still in an early stage of its development cycle, but it has introduced several interesting new research issues, such as continuous state maintenance and contact verification.

1 Introduction

For more than a decade, the Spoken Language Systems (SLS) Group at the MIT Laboratory for Computer Science has been conducting research leading to the development of conversational interfaces: interfaces that will enable naive users to access and manage information using spoken dialogue. To realize such systems, several language-based input and output technologies, including speech recognition/synthesis and language understanding/generation have been developed and integrated. Typically, these systems engage the user in a dialogue to retrieve information from databases.

[*] This work was supported by DARPA under contract N66001-99-1-8904 monitored through Naval Command, Control and Ocean Surveillance Center.

Until now, all of our systems have assumed that each task is completed as soon as the user hangs up the phone. However, it seems reasonable to suppose in principle that a conversational system could perform certain tasks *off-line*, i.e., that the user could *delegate* information-dependent activities to the system, which would later inform them of the outcome, either through e-mail or through system-initiated telephone contact.

This paper describes our newest conversational domain, called ORION, which for the first time begins to address the idea of delegation. The user can request a task to be executed at some later time, and ORION must then initiate a follow-up contact with the user, once the assigned task has been completed. Some of the tasks involve monitoring a dynamically changing database over time. Others involve a single look-up at the designated call-back time.

The Orion system is configured using the Galaxy Communicator architecture, in which all servers communicate via a common hub. ORION consults other domain servers to retrieve critical information. The Orion server plays two distinct roles, each implemented as a separate stream. One is devoted to the enrollment of new tasks and the other is concerned with the execution of existing tasks. Users first enroll by providing critical information about their name, appropriate phone numbers, and e-mail address. To edit existing tasks or add new tasks, the user interacts with ORION at a Web site, in conjunction with conversation through speech over the telephone, or through typing at the GUI interface. Pending tasks are displayed in the graphical interface, and ORION engages the user in a mixed-initiative conversation until a new task is fully specified, or a pre-existing task is appropriately modified. If a new task needs to be executed today, it is sent to the agent stream for an immediate update. ORION can currently handle a number of types of requests, as suggested by the examples given in Figure 1.

Call me at 4 p.m. tomorrow to remind me to pick up my son at soccer practice.
Call me every weekday morning at 6:30 a.m. and tell me the weather in Boston.
Call me an hour before American flight 93 lands in Dallas.
Call me at work between 5 and 6 p.m. if the traffic on route 93 is at a standstill.

Fig. 1. Examples of the types of tasks that Orion can currently handle.

A unique aspect of ORION is that it behaves like a user in requesting information from other domains. ORION finds it convenient to make these requests using standard English as a natural and intuitive scripting language.

The notion of off-line delegation is a radical departure from our prior research activities, and it opens up a number of interesting research issues, such as continuous maintenance of state information, dealing with the termination of or corrections to previously assigned reiterant tasks, and verification that the call-back was successful. A domain of the degree of complexity of ORION would not have been feasible for us to develop prior to the introduction of the GALAXY Communicator architecture [2].

In the remainder of this paper, we will first describe the task negotiation phase, followed by the task execution phase. We will then discuss a number of issues that have emerged as a consequence of our explorations with ORION. We conclude with a discussion of some of our future visions.

2 Task negotiation phase

The first step in task negotiation is to enroll a new user. This is done at a graphical interface where ORION asks a series of questions and the user types the responses into a GUI type-in window integrated into a Web browser. ORION seeks three phone numbers (home, work, and cell) as well as the user's name and e-mail address. Once the user has enrolled, they can type, "call me now at work," to initiate an immediate call-back for spoken interaction, or they can continue to interact through typed queries.

ORION then launches a mixed initiative dialogue to invite the user to specify a task, or to edit any existing tasks, should this be a follow-up interaction. In the latter case, all pre-existing tasks are displayed to the user in the graphical interface. The user could, for example, click on a given task and say, "Edit this task."

If the user intends a new task, they can simply specify it through speech. They might say, for example, "Call me to remind me to call my sister." ORION will prompt the user for a date, time, and phone number, where "work," "home," and "cell" are all possible, in addition to a unique newly introduced task phone number. Figure 2 shows an example dialogue between a user and ORION, where the user is requesting a wake-up call the following morning.

If the user cannot remember the particulars of a task, he/she can ask to talk to one of the other domains. The user might, for example,

Orion: Welcome to the Orion planning system. Please type your user name, or, if you have not enrolled, type 'new user'.
User: Stephanie Seneff

Orion: Here is your personal information in our database, Stephanie Seneff. What is the task that you have in mind? (wakeup call,reminder, flight status, or traffic update) **<shows user profile and list of prior tasks>**
User: Call me at six thirty tomorrow morning and tell me the weather in Boston.

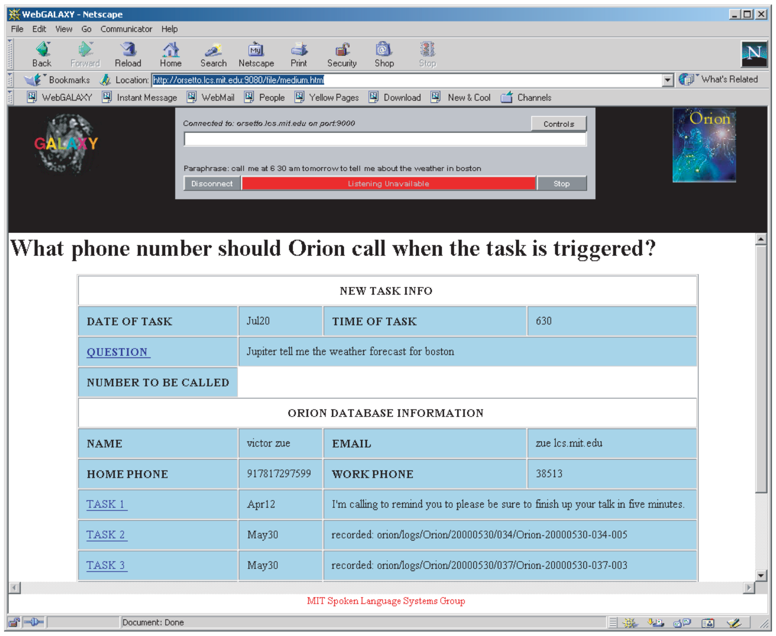
Orion: What phone number should I call you at?
User: Call me at home.

Orion: You have entered all the necessary data, stephanie seneff. Is the following information correct? **<shows task>**
User: Yes.

Orion: You will receive a call at the appropriate time. Thank you for using Orion.

Fig. 2. Example dialogue between a user and Orion negotiating a wake-up call, in the context of a graphical interface.

find out from the PEGASUS flight status domain or the MERCURY flight reservations domain [3] the flight number of a flight they are interested in monitoring. In the same phone conversation, the user can return to ORION, and ORION will be able to obtain directly from MERCURY's dialogue state all the information concerning the flight in focus, which ORION then assumes is of relevance to the conversation.


WebGALAXY - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Location: http://orsetto.lcs.mit.edu:9080/file/medium.html What's Related
WebGALAXY Instant Message WebMail People Yellow Pages Download New & Cool Channels
GALAXY
Controls
Disconnect
Listening Unavailable
Stop
Orion
Document: Done

Connected to: orsetto.lcs.mit.edu on port:9000

Paraphrase: call me at 6 30 am tomorrow to tell me about the weather in boston

What phone number should Orion call when the task is triggered?

NEW TASK INFO			
DATE OF TASK	Jul20	TIME OF TASK	630
QUESTION	Jupiter tell me the weather forecast for boston		
NUMBER TO BE CALLED			
ORION DATABASE INFORMATION			
NAME	victor zue	EMAIL	zue lcs.mit.edu
HOME PHONE	917817297599	WORK PHONE	38513
TASK 1	Apr12	I'm calling to remind you to please be sure to finish up your talk in five minutes.	
TASK 2	May30	recorded: orion/logs/Orion/20000530/034/Orion-20000530-034-005	
TASK 3	May30	recorded: orion/logs/Orion/20000530/037/Orion-20000530-037-003	

MIT Spoken Language Systems Group

Fig. 3. The graphical interface for Orion, showing the task under negotiation, the user's profile, and a list of three previously specified tasks.

Figure 3 shows a GUI interaction with ORION. Once the new task is fully specified and confirmed, ORION updates the user's personalized file of pending tasks. If the task is scheduled for today, ORION wakes up the agent stream via a special hub-mediated operation devoted to newly introduced tasks, as will be more fully described in the next section.

If the user asks to be called at some future time, but does not mention a particular task, ORION by default invites them to pre-record a message to be played back at the designated time. This is a very general device with wide utility, that allows the user to say anything at all in the recording, subject to a length constraint.

QUERY: ”Call me at six a m and tell me the weather in Boston”

LINGUISTIC FRAME:

```
{c call_me
   :domain "Orion"
   :reason
     {c inform
        :topic {q weather
                  :quantifier "def"
                  :pred {p in
                           :topic
                             {q city
                                :name "boston"
                             }
                         }
                }
      }
   :pred {p at :topic {q time :hour 6
                              :xm "am" } } }
```

E-FORM:

```
{c eform :action “call_me” :time “6:00 am” :clause “inform”
:task_label “weather” :in “boston” }
```

Fig. 4. Example linguistic frame and *e*-form for a user query in the ORION domain.

In order to plan its interactions with the user and the other servers, ORION makes use of a dialogue control table, as described more fully in [3]. The GENESIS-II [1] generation server is invoked to paraphrase users' requests into an appropriate electronic form (henceforth *e*-form) format, which is used to initialize the dialogue state at each new turn. Figure 4 shows the query, the semantic frame, and the derived *e*-form obtained via GENESIS-II for the sentence, “Call me at six a m and tell me the weather in Boston.” The dialogue state is consulted to determine which functions to call to carry out the task-specification phase, using dialogue control procedures that are common to all of our domain servers.

GENESIS-II is also responsible for altering the pronominal references from first to second person, so that "call me to remind me to call my sister" becomes "This is ORION calling to remind *you* to call *your* sister."

3 Agent activities

The agent stream currently maintains information about all of its users in data structure files that are reloaded every day at midnight. It first determines which tasks are possible on the current day, including reiterant tasks (e.g., "every weekday"). It then creates an internal data structure for each pending task that includes slots for various temporal events such as the next update time or the estimated (or exact) *trigger* time (time to contact the user).

To execute its various plans, the system consults a second dialogue control table [3], distinct from the one used to plan new tasks. It iterates through the dialogue table repeatedly until all tasks are completed as fully as possible. At midnight, this would typically include consulting the MERCURY flight information server to determine the scheduled departure and arrival times of any flights being monitored.

Once all tasks have been initiated, the ORION agent stream goes into a sleep cycle until the minimum next update time of its set of pending tasks. However, it could be awakened at any time by a new request from a user who has just finished specifying all the particulars of this request, or who may have altered a previous request that now needs immediate attention. These interruptions are implemented as a separate operation, invoked by a rule in a hub script, according to the GALAXY Communicator design [2].

When the next update time has been reached, ORION consults a dialogue control table to determine which functions to call. Functions may involve module-to-module subdialogues with other domains to look up information about weather [5], flights [3], or traffic. These subdialogues are controlled by a separate hub program appropriately named *orion*, and they involve calls to the natural language server to parse ORION's request, and calls to the designated turn manager to fully interpret the parsed frame and produce a reply frame. This is accomplished via a database request initiated in a nested module-to-module subdialogue (through normal channels used for all user requests). Selected rules from the *orion* program are shown in Figure 5.

```
PROGRAM: orion

RULE:     :call_request → dispatch_to_main
IN:       (:input_string :call_request) :domain :recording

RULE:     !:request_frame & :input_string → create_frame
IN:       :input_string :domain
OUT:      :request_frame :domain

RULE:     :request_frame & :domain Jupiter → jupiter.turn_management
IN:       :request_frame
OUT:      :reply_frame

RULE:     !:reply_string & :reply_frame → paraphrase_reply
IN:       :reply_frame :domain (:out_lang english)
OUT:      :reply_string :reply_frame
```

Fig. 5. Example rules from the *orion* program used by the GALAXY hub. Note: "!" stands for "NOT", and "&" is a logical "AND."

For example, if ORION has been requested to call the user at some designated time relative to the arrival of a flight, it monitors the flight at appropriate intervals throughout the period of time that the flight is in the air. Its algorithm is to cut the time by half between the current time and the currently estimated arrival time to determine when to next check the flight status. Checking the status involves a contact with the PEGASUS flight status domain. As soon as the estimated callback time is reached, ORION delegates to PEGASUS the task of actually calling the user.

When a trigger time is reached, ORION executes the function named "call_user_now," but only after it has prepared a frame containing an appropriate welcome message. If the task involves a particular domain of expertise, ORION currently delegates the phone call to the appropriate domain server[1]. From ORION's standpoint, this is easy to do: it just prepares a query of the form "<domain_name> call me at <task phone number>." The *orion* hub program dispatches the query to the main program[2], along with the welcome message, which might be prerecorded. It follows normal channels, modelled after the pre-existing

[1] Although the user can ask to speak to ORION should they have further tasks in mind.

[2] see the dispatch_to_main operation in the example rules of Figure 5.

feature that a user can type a call-back request into a GUI interface. For reminders that don't invoke any of the other domains, e.g., "call me to remind me to pick up my son," ORION itself initiates the call, and is then prepared to negotiate further tasks if the user so desires.

4 Unresolved issues

ORION is an early prototype of a conversational intelligent agent (IA) that interacts with the real world. We have barely tapped into the large body of research in IAs at this time [4]. Special considerations are required when IAs are exposed to the real world, several of which are discussed here.

Security ORION must be protected from hackers who could potentially use it to place unwanted calls, change other users' tasks, etc. We must devise secure mechanisms for user authentication, and extend existing access control mechanisms to conversational systems.

Reliability ORION must be dependable. Tasks must survive system crashes. Infrastructure must be developed to reduce the complexity of our existing system development and support the new capabilities required by ORION.

Efficiency ORION currently uses a polling mechanism to track actions. Polling is complex and innefficient, and will not scale as the number of users and tasks increases.

In the future, we expect GALAXY to make use of daemon agents to update system data such as weather and flight information as we receive it from our data sources. For example, a user agent can tell the flight information update daemon to notify it whenever flight information about United flight 805 changes, and then sleep until thirty minutes before arrival. If the flight information changes, the daemon agent notifies the user agent which wakes up, reschedules its future wakeup, and goes back to sleep. This approach reduces complexity because it allows the same user agent code to be used for weather and flight notifications, and it increases efficiency because the agent only runs when it has something to do.

Resource Management To satisfy ORION's future needs, GALAXY will need to support an additional layer of distributed operations that provide common services like speech recognition, virtual device management (such as a pool of phone lines), and persistence to agents. Systems such as JUPITER and MERCURY will be collections of related agents that ORION interacts with.

Common Sense If a user asks for a wakeup call every day at 6:00 a.m., ORION should be aware that this means local time for the user.

If ORION knows a particular user will be travelling during a certain time interval, it should be able to pro-actively enquire whether the call should still happen, and, if so, perhaps suggest calling on the cell phone. Also, how should ORION negotiate day by day monitoring of such reiterant tasks? If, on Friday, the user says, "Don't call me Monday," should ORION then enquire about Tuesday, or assume that Tuesday is the next time to call?

Customization We expect that in the future ORION will become increasingly capable of managing an experienced user's tasks. ORION should be able to accumulate information on patterns based on past experience, such that it will be increasingly able to take the initiative in proposing or even executing specific plans. Thus, for example, in the future, an expert user might be able to ask ORION to make travel arrangements to Chicago, and ORION would be able to make decisions about the flights, hotel, and rental car, calling back only to specify a fully executed solution.

Verification If the user fails to answer the phone when ORION calls, then ORION would need to recognize that the call is incomplete, and perhaps try again ten minutes later, and/or send e-mail informing the user of the failed contact. A more serious problem is a pick up by an answering machine. How should ORION be made aware of this situation, and what should it do? Another problem is the recognition of the voice of the user, or a verbal verification that the person answering the phone is indeed the intended user.

Social Users will be reluctant to depend on a system like ORION for important tasks, particularly if the agents will be searching for information, making decisions, or spending the user's money. We will need to provide ways for users to check the reasoning process and to verify that ORION is really planning to do what they think it will do.

5 Summary and future plans

This paper introduces ORION, a system devoted to off-line delegation as a new paradigm in conversational systems. ORION was made feasible as a consequence of the powerful capabilities of the GALAXY Communicator architecture. We found that it was straightforward to leverage off of pre-existing mechanims with only minor modifications to carry out both the server initiated queries to other domains and the user call-back procedures. Thus far ORION has been used only by researchers in our group and their families, as well as in live demonstrations. We have not yet begun the task of data collection from naive users, as the system is not sufficiently mature.

We expect that ORION represents the early stages of an ambitious long-term project. In the future, we envision that people will be able to take advantage of available on-line information systems such that routine tasks can be delegated to the computer as much as possible, thus freeing humans to attend to tasks that truly need their attention. We will explore how we can build systems that can easily customize and adapt to the users' needs and desires. In the process, we will also examine how a system such as ORION could be made more intelligent by incorporating into its decision-making process all available information at its disposal, both about the domain and about the individual user. We believe that ORION has tremendous potential, although we are still working out the details of some of the more difficult aspects of its design, including user identification, customization, resource allocation, and task completion verification.

References

1. L. Baptist and S. Seneff. GENISIS-II: A versatile system for language generation in conversational system applications. In *Proceedings of ICSLP'00*, Beijing, China, 2000.
2. S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid, and V. Zue. GALAXY-II: A reference architecture for conversational system development. In *ICSLP '98*, pages 931–934, Sydney, Australia, December 1998.
3. S. Seneff and J. Polifroni. Dialogue management in the MERCURY flight reservation system. In *ANLP-NAACL 2000*, Seattle, WA, May 2000.
4. Yoav Shoham. An overview of agent-oriented programming. In Jeffrey M. Bradshaw, editor, *Software Agents*. AAAI Press/The MIT Press, Cambridge, Massachusetts, 1997.
5. V. Zue, S. Seneff, J. Glass, J. Polifroni, T.J. Hazen C. Pao, and L. Hetherington. JUPITER: A telephone-based conversational interface for weather information. *IEEE Trans. on Speech and Audio Processing*, 8(1):85–96, January 2000.