Dialogue Management In a Virtual College

Martin D. Beer[1], Trevor J. M. Bench-Capon[1], and Andrew Sixsmith[2]

[1] Department of Computer Science, University of Liverpool,
Chadwick Tower, Peach Street, LIVERPOOL. L69 7ZF United Kingdom.
mdb@csc.liv.ac.uk and tbc@csc.liv.ac.uk
[2] Institute of Human Ageing, University of Liverpool,
PO Box 147, LIVERPOOL. L69 3BX United Kingdom.

Abstract. The development of distributed high-bandwidth data networks at the home and business level allows many new distributed information services to be developed, based on the co-operation of autonomous agents and distributed information sources. Agents collaborate by passing messages in an agent communication language, such as KQML. The validity and meaning of these messages depends critically on the context of the communication of which they form a part. The conversation class to which the communication belongs determines this context.

This paper describes some of the problems associated with conversation classes derived from a distance learning application. We discuss the mechanisms necessary to satisfy a number of complex scenarios within the application domain, and show how these conditions need to be specified in terms of the policies and strategies of particular agents.

1 Introduction

Many new applications are being considered which rely on effective high capacity communication between stakeholders of various types. One architectural model that is being considered for such applications is that of co-operative agents. The fundamental requirement of an information system for widespread non-technical use is ease of use. While the use of a Web browser such as Internet Explorer or Netscape is acceptable for relatively sophisticated users, it is far from ideal for large-scale general use. In particular, problems have been identified with navigation when the information space becomes large and complex, and when World Wide Web servers become overloaded. This has led to a relatively passive environment, in which users seeking information have to collate and apply it themselves. We are currently investigating the use of intelligent agent based Internet technology to provide a much more flexible level of response than is currently possible. Example applications have been developed for both Distance Learning [1] and Community Care [2]. The basic architecture is that of Wiederhold [3], and is based on communication using performatives specified in KQML (Knowledge Query and Manipulation Language) [4].

T. Bench-Capon, G. Soda, and A M. Tjoa (Eds.): DEXA'99, LNCS 1677, pp. 521-530, 1999.

The intention of this paper is to show how current research in distributed knowledge sharing and intelligent agents can be used to provide effective distributed services using agent-based techniques. The particular application considered is that of a 'Virtual College' environment [5]. The motivation for this study has been the design of an Occupational Therapy Internet School (OTIS), funded by the European Union [6]. If a single Internet School is to be developed over such a large geographical spread, effective communication mechanisms must be devised and implemented. Fortunately, for the pilot, the language of operation will be English, so that multi-lingual issues can be addressed at a later stage.

2 The 'Virtual College' Scenario

A basic Use-Case diagram is given in Figure 1, showing the principal actors involved, and some of the activities in which they are involved in the running of a course. In our model, a Course Director, who is responsible for ensuring its proper running and monitoring student activity, manages the course. Student support and assessment is undertaken by a number of Course Tutors, who each have a number of students assigned to them.

Courses are organised across institutions so the Course Director and the Course Tutors may well be in separate Institutions, using very different communications infrastructures. Our architecture allows them to use the systems with which they are most familiar, and to manage the necessary conversations through the use of wrappers. Wrappers provide the conversion facilities necessary to allow processing within the agent to be undertaken in whatever internal format is most appropriate, while communication between agents is in a common format (in our case KQML). Students will wish to communicate not only with the staff, but also with other students studying the same course.

3 System Architecture

All resource agents are required to *advertise* to the facilitator before they can accept communications of a particular type from other agents. This advertisement remains in force until the agent issues a corresponding *unadvertise*. For the purposes of this study, the message can be considered to be routed directly from the requesting to the responding agent. The facilitator simply acts as a name service. The User Agent acts as the interface for all actors, providing a common interface to all activities, no matter how and where they are provided.

A facilitator is first contacted to obtain the address of a suitable service mediator, to which an appropriate message is sent. This mediator directs the message to a suitable service provider after possibly adding additional information from the service database. Examples of information that may be added at this stage, are the validity of the student's registration, or the Course Tutor to whom the student is assigned. The service provider's mediator determines the appropriate course of action and forwards the message.

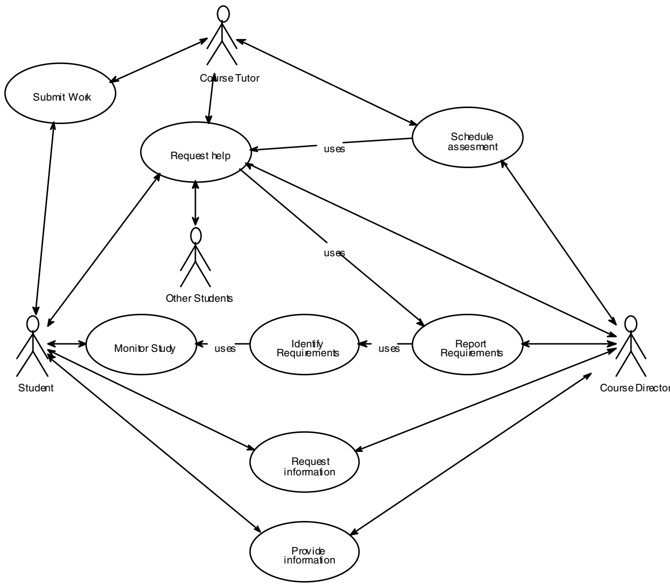

Fig. 1. A Basic Use-Case Analysis of managing a Virtual College Course

The facilitator therefore manages the routing of messages depending on the capability of the responding agent, whereas the mediator manipulates messages using its 'application domain knowledge' to decompose the problem and determine the most appropriate service to provide the necessary information. Once the requested information has been collected, the appropriate reply is formulated and the conversation is completed by its receipt by the requesting agent.

One of the main features of the work described is the specification of the conversational abilities of agents. In line with KQML [4], these are determined by a set of speech acts which that agent can perform, and another set to which it can respond. A speech act comprises a performative, indicating the illocutionary and intended perlocutionary force of the act, together with an argument, typically a proposition [7].

The semantics for the speech acts are as follows, following Labrou [8]:

1. a set of preconditions for the performance of the act,
2. a set of post-conditions to be enforced immediately on performance of the act, and
3. a set of completion conditions which are to apply when all the intentions associated with the act have been finally satisfied.

This means that the completion conditions may be achieved at some time after the communication being described has taken place, following some further conversation. Acts with such conditions can provide a context for later utterances.

Although performatives provide the building blocks for conversations between agents, they must be co-ordinated in a way that is appropriate to the particular conversation. The need for this co-ordination is shown by Barbuceanu et al [9]. It is possible to extend Labrou's method by representing the conversation rules as additional preconditions, postconditions and completion conditions on the speech acts they use. Experimental use of this approach has been reported in [10], which describes the specification of conversation rules for several dialogue games. In our particular application it is necessary to customise the act to particular agents. It is possible, for example, for confidentiality reasons, for an agent to be required not to respond to questions from some class of agents with a particular content. It may be that there is a rule that work for assessment can only be submitted to the appropriate course tutor, and the results returned only to the student who submitted it. We therefore need to impose further conditions on the performatives of a given agent which will be individual to the agent rather than derived from the conversation class.

4 Some Specific Instances of Conversation Classes

This section will show some of the situations that may be used to give preconditions, postconditions and completion conditions for the tell performative. We choose to express these conditions in terms made explicit within the context of the conversation itself.

4.1 Conditions Derived from the Conversation Classes

Within the context of a distance learning application, the conversations maintained between the agents in Figere 2 would be typical and fit well with the model that we have adopted. The student asks a question, which is to be answered by a Course Tutor. It is routed via a Mediator, which uses the information in a Student Registration Database to determine a list of appropriate Course Tutors to contact. This may be controlled by a number of factors known to the Mediator, for example:

1. Pass the query to the Course Tutor allocated to that student if the tutor is available
2. Otherwise it is passed to an appropriate Course Tutor, who is currently available to answer the query.

The Course Tutor receives the query and answers it. The answer is then passed back to the student along its original route. It is an implicit condition that the Course Tutor will only answer queries only from a student. This means that a course tutor can issue a in reply to an *ask* in this conversation class.

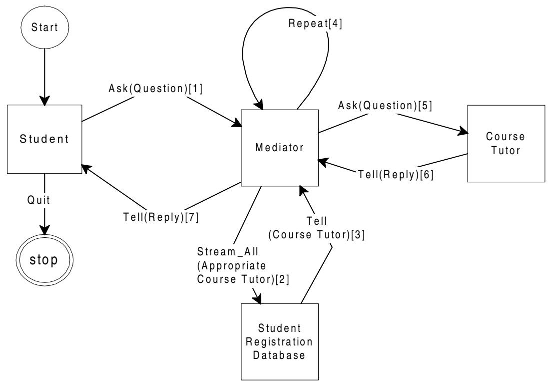

Fig. 2. The Conversation initiated by a Student Asking a Question

A set of preconditions and postconditions for each major stage (those in square brackets in Figure 2) in the conversation can be established as follows:

Once registered, a student can *ask* a question at any time. The student's Interface Agent passes the question to an appropriate Mediator, which will forward it to an appropriate Course Tutor [1]:
  Precondition: X is a registered student?
  Postcondition: create conversational instance and update records
  Completion: Help request has been sent (*tell* [5] issued)
The Mediator obtains the names of appropriate Course Tutors from the student Registration Database [2]. Additional information, such as the identity of the tutor to whom the student is allocated can also be included, so that the question can be directed to the most appropriate tutor. The conditions for initiating this request are:
  Precondition: A Question has been *ask*ed and no appropriate Tutor is known
  Postcondition: Tutor I is appropriate to answer this question
  Completion: Tutor I has answered the question satisfactorily[5]
And for the *tell*s giving appropriate tutors [3]:
  Precondition: A list of appropriate tutors has been requested
  Postcondition: An appropriate Tutor I has been found
  Completion: none
The database provides the identities of different course tutors singly, in an appropriate order of priority until a suitable Tutor is contacted, and the communication is terminated, or the list is exhausted [4]:
  Precondition: No replies from a previous *ask* with same content
  There is a suitable Course Tutor available
  Postcondition: Records updated
  Completion: Reply [6] received

The Mediator forwards the question to a Course Tutor, starting with the most appropriate [5]:

Precondition: Question q has been *ask*ed and no *tell(q)* received

Postcondition: Tutor I has been *ask*ed q [5]

Completion: A satisfactory reply [6] has been received

The Course Tutor replies to the question, which is passed back to the Mediator [6]:

Precondition: Tutor has received an *ask* from the mediator

Postcondition: Tutor is committed to the reply

Completion: none

The Mediator passes the reply back to the student [7]:

Precondition: Mediator has received an *ask* from the student [1] and tutor is committed to the reply[6]

Postcondition: The mediator is committed to the reply

Completion: A reply is transmitted.

The query will often raise questions which the tutor considers to be of a wider interest, and which are best answered by telling all appropriate students the information given in the reply. An appropriate description of such a conversation is given in Figure 3. Here the problem is that the reply is being sent to a number of students who have not asked the appropriate question. This is handled by using the insert performative, which does not require the responding agent to have been asked before it can provide information to another agent.

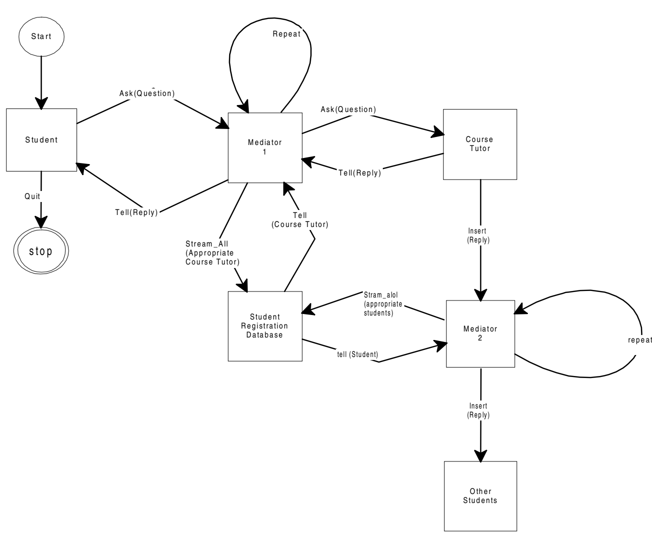

Fig. 3. Replying to a Student and Informing all Others of the Reply

The difficulty remains that the other students have not *ask*ed for the information being given, and have no basis on which to base their belief in the efficacy of the information included in it. Both inserts have the same properties:

Preconditions: A question has been *ask*ed and a *tell* has been issued
Postconditions: The sender is committed to the information contained in the message
Completion conditions: none

It will be seen that since the message is unsolicited, no preconditions or completion conditions apply to the receiving agent, and it is therefore always able to receive them. In this case, there are no difficulties with the other students believing the information as the students can be expected to believe implicitly any information provided by Course Tutors.

Another Conversation Class that poses interesting problems is that which controls the assessment process. The scenario in Figure 4 is that the Assessor provides model answers to Course Tutors who compare these with the answers submitted by the students, and enter their assessments in the appropriate database. Students have access to this database in some form, and can use this knowledge to allow them to progress to another module, or repeat the assessment if they have failed. If we follow the direction of the conversation, it will be seen to flow through a number of stages, with no agent replying directly to messages sent to it from another agent. This is because the Tutor is attempting to determine what the student believes to be the correct answer to the assessment. The Tutor, when marking the student's work, believes the Course Director's model answer is correct and assesses the student's work accordingly.

4.2 Conditions Derived From Policies and Heuristics

It is now necessary to consider the issues raised by the imposition of additional conditions required to customise the behaviour of agents in a given conversation class.

Preconditions Consider first a precondition for *tell*, found in Labrou's specification `believes(S,X)` If this is strictly imposed, an agent will only be able to commit to a proposition that it believes to be true. Adding this precondition thus gives an honest agent.

Take the case of the assessment scenario previously discussed. The course Director Agent sets an assignment to discover what the Student Agent 'believes'. The Course Tutor Agent, which could be a person or some form of automated agent (for example [11]), then compares the student's work against a model answer 'believing' that the model answer is the closest to the ideal. An assessment mark is determined by some measure of 'closeness' to this ideal, and this is entered into the result depository. The Student Agent 'believes' the accuracy of the results once they are stored in the Student Record. This relies on a number of specific 'beliefs' which are determined by the actions specified by the conversation class, rather than the form of the performatives used.

Another situation to be considered is where the agent has sensitive information, which can only be issued to a restricted group. A good example of this is the transmission of the model answer from the Course Director agent to the various Course Tutor agents. It is essential that this communication is received by the agents to whom it is addressed, and not intercepted by Student Agents. They could then subvert the assessment process by using this knowledge to provide an answer that does not represent what the agent 'believes' to be the correct answer, but what it 'believes' the assessment agent will accept as the correct answer.

Postconditions Postconditions help us to specify the effect of a performative. This will very often depend on the stance of the recipient. For example, a strongly credulous agent can be defined similarly, by adding the postcondition `believes(R,X)` This would cause the receiver to be permanently updated and any previously held beliefs of this form are discarded. This is very similar to committing the result at the end of a database transaction. Final reporting of the assessment mark is an activity of this type. It should be noted that these postconditions are entirely independent of the conversation class. The other agent need not be aware of what use is being made of the information, and only the receiving agent can be in a position to know what use is appropriate.

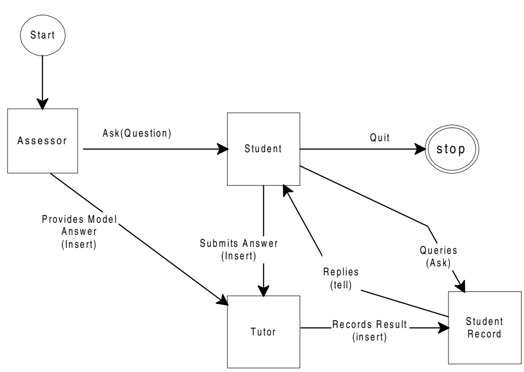

Fig. 4. The Assessment Conversation

Completion Conditions Policies and heuristics are less fruitful as a source of completion conditions, since a performative is complete once something has been agreed between the participants in the conversation. It should therefore be expected that these conditions derive always from the conversation class, and cannot, unlike preconditions and postconditions, be personal to an agent.

In our case completion conditions for the conversation not the acts, would be in the form:

For the student query scenario - that the query has been successfully answered (possibly after several supplementary questions), and that all other students have been informed, if necessary.

For the assessment scenario - that the work to be assessed has been received and the results to the student.

In the first case it may take several supplementary queries to ensure that a satisfactory reply has been given. Further communication with all other relevant students may also be required before completion can be achieved. In the second case, there may well be a rule that no assessments can be accepted after the first result has been returned. This may be enforced either by setting a latest hand-in date, or by waiting until all assessments expected have been received.

5 Discussion

The motivation for the analysis described here was to show that the communication between agents is characterised by the nature of the conversation between them, rather than the properties of the agents themselves or individual performatives consider out of context [12]. In the context of the distance learning scenario, the 'agents' include people, databases and expert systems. It does not matter which are to communicate, just the mode in which this communication is to take place. This is defined by the conversation class, within which performatives are defined to meet the overall requirements. In the context of different conversations a given performative may have different conditions associated with it. The conversation class says how a performative can be used, and sharing the conversation class harmonises the use of the particular set of performatives across agents. In order for effective communication to be possible, the communicators must operate within the same conversation class, so that they can share the conventions governing the use of the performatives.

6 Conclusions

This paper shows that by the adoption of appropriate conversation classes, it is possible to provide a wide range of services robustly and securely. One of the most important of these is ongoing educational activities, as it is increasingly required to update skills on a regular basis throughout life. It is an area where current Web based technology has been used effectively, but has proved not to be scalable when the number of students involved grows beyond certain limits. Simply adding additional servers is not a full answer, as effective distance learning systems require access to certain major information pools. These contain information on:

1. The course materials to be delivered
2. The students registered to take the course
3. Staff responsibilities and availability in the roles that they are expected to undertake
4. Student progress and assessment records

The definition of conversation classes between multiple agents allows these information sets to be separated. Adequate resources can then be allocated to the management and delivery of each. When a single agent is unable to handle the traffic directed to it, it can be replicated. The new agent needs to commit to the same conversation classes in order to perform all the necessary tasks, but does not need to be a clone of the original agent. This is because it does not have to commit to the policies and heuristics of the original to join the same conversation classes. The highly distributed nature of information delivery reduces the likelihood of bottlenecks as any agent that becomes overloaded can be replicated quite simply, without disturbing the stability of the rest of the system.

References

1. Wabi, J. (1998), Communication Requirements in Distance Learning Environments, MSc Dissertation, University of Liverpool.
2. Shin, A. P.J. (1998), INAS - Intelligent Alarm System for the Elderly, Honours Project Dissertation, University of Liverpool.
3. Wiederhold, G., (1992), Mediators in the Architecture of Future Computing Systems, IEEE Computer, 25(3), 38-49, March 1992.
4. Finnin, T., Labrou, Y. and Mayfield, J. (1997), KQML as an Agent Communication Language, MIT Press, Cambridge MA
5. Beer, M. D. (1997), Developing Distributed Learning Environments for the Workplace, ECSCW'97 Workshop on Network Communities: Supporting Community on the Net, Lancaster, UK, September 1997.
6. University of Liverpool, (1998) Occupational Therapy Internet School - OTIS, Submission to the European Union TEN Telecom Programme, June 1998.
7. Searle, J. R., (1969), Speech Acts: An Essay in the Philosophy of Language, Cambridge University Press
8. Labrou, Y., (1996), Semantics for an Agent Communication Language, PhD Thesis, University of Maryland, Baltimore.
9. Barbuceanu M and Fox M.S., (1995) COOL: A language for co-ordination in multi-agent systems. Procedings, 1st International Conference on Multi-Agent Systems, MIT Press, Cambridge. Mass.
10. Bench-Capon, T.J.M., (1998) Specification of Communication between Information Sources, Proceedings of DEXA98, Lecture Notes in Computer Science, 1460, Springer.
11. Benford, S. D., Burke, E. K., Foxley, E., Gutteridge, N. H. and Zin, A. M. (1994), Ceilidh as a Course Management Support System, The Journal of Educational Technology Systems, 22 (3) pp 235-250.
12. Carlson L. (1983), Dialogue Games: an approach to discourse analysis, Reidel, Dordrecht.