Collaborative Activities and Multi-tasking in Dialogue Systems

Towards natural dialogue with robots

Oliver Lemon — Alexander Gruenstein — Stanley Peters

Center for the Study of Language and Information (CSLI),
Stanford University,
210 Panama Street,
Stanford, CA 94305

lemon,alexgru,peters@csli.stanford.edu

RÉSUMÉ. *Nous concentrons notre recherche sur le management du dialogue en vue d'activites collaboratives qui impliquent des tâches multiples et simultanées. Le contexte conversationnel pour les activités simultanées est représenté en utilisant un "Arbre de Mouvement de Dialogue" et un "Arbre d'activites" qui soutiennent les fils multiples et "interleaved" du dialogue au sujet des differentes activites et de leur plan d'execution. Nous décrivons aussi la sélection cumulative du message, l'agrégation, et la méthode de génération employées dans ce contexte. Le systeme de génération doit aussi etre capable de contrôler des contextes où il y a de multiples sujets coordonnes par la conversation. Nous montrons comment utiliser les représentations dynamiques de contexte pour l'interprétation flexible et naturelle de contributions de dialogue dans de telles applications. Nous démontrons que ces techniques sont viables dans un système de dialogue pour les conversations avec des robots semi-autonomes et mobiles.*

ABSTRACT. *We focus on dialogue management for collaborative activities, involving multiple concurrent tasks. Conversational context for multiple concurrent activities is represented using a "Dialogue Move Tree" and an "Activity Tree" which support multiple interleaved threads of dialogue about different activities and their execution status. We also describe the incremental message selection, aggregation, and generation method employed in this context. The generation component must also be able to handle contexts where there are multiple topics being co-ordinated by the conversation. We show how to use dynamic context representations for flexible interpretation and natural generation of dialogue contributions in such applications. We demonstrate that these techniques are viable in a dialogue system for multi-modal conversations with semi-autonomous mobile robots.*

MOTS-CLÉS : *Dialogue, le Multi-tasking, le Multi Enfiletage, la Production de Message, l'Activité Commune.*

KEYWORDS: *Dialogue, Multi-tasking, Multi-threading, Message Generation, Joint Activity.*

1. Introduction

This paper describes a multi-modal dialogue system for human-robot interaction, and a generic architecture which implements a dynamic information state model of dialogue (e.g. [BOH 99, COO 98, GIN 96a, GIN 96b, LEM 01a]). We will focus on the particular aspects of the system which allow it to handle dialogues about multiple concurrent tasks in a coherent and natural manner. Many conversations between humans have this property. For example, while driving two people could be talking about directions to a location, and also about a plan of action to be executed when they get there. Here, material from one thread (giving directions) can intervene between, say, a question and answer pair in the other conversational thread (planning future actions). Throughout the paper we use the term "conversational thread" to refer to the set of utterances which serve a particular dialogue goal. For example, all the utterances used to specify, clarify, revise, and monitor a particular activity (e.g. search for a red truck) form a thread. These threads can be picked up and dropped spontaneously, but further acknowledgements, reiteration, and dialogue may be required to robustly establish interpretation. Another example of interleaved dialogue is where "meta-dialogue" such as "let's talk about kayaking tomorrow" is used to direct a conversation (i.e. to start a new thread, to change between threads, or to postpone a thread).

1.1. *Multi-tasking and interleaved dialogues*

Most researchers in the field study *embedded* dialogues, such as clarification subdialogues, but little attention has been paid to dialogues which are *interleaved*. Research by [LEV 98] involves a system for tagging telephone dialogue data[1] where "interleaved games" are recognised conversational structures, as well as the more standard nested games or embedded dialogues. Work by [ROS 95] discusses multiple dialogue threads in conversations where two speakers attempt to schedule a meeting.

In interleaved dialogue, although material from one dialogue thread intervenes between moves in another dialogue, that material does not have to 'close' in order for the other dialogue to be resumed (e.g. in the above driving example, we may go on planning our activities at the destination, long after the direction-giving dialogue is over, or vice versa). Another difference is that, in general, no one dialogue need be thought of as the 'parent' of the other, or as having priority. Of course, not any two dialogues can sensibly be interleaved – discovering constraints on this process is a research issue – and there are psychological questions regarding human competence in this area.

Everyday experience leads us to believe that humans are competent at carrying out conversations with at least a small number of multiple threads, or topics, and that this capability enables fluid and efficient communication, and effective co-ordination of multiple activities. We believe that the ability to interleave communication threads is

1. The CallHome and CallFriend conversation databases used in the CLARITY project.

at least a useful, if not essential, property of conversational interactions. If it were not present, conversations would be restricted to those concerning sequences of topics, with attendant embedded dialogues.

Dialogues between humans and semi-autonomous devices will have these "multi-tasking" features in as much as the device is able to carry out activities concurrently and to plan future activities. We will show how to endow a dialogue system with some of these capabilities.

The main issues which we address in this paper are :

– Representation of dialogue context such that collaborative activities and multi-tasking are supported.

– Dialogue management methods such that free and natural communication amongst several conversational topics is supported.

– Representing and reasoning about multiple collaborative activities in dialogue.

– Natural generation of messages in multi-tasking collaborative dialogues.

1.2. *Outline*

In Section 2 we discuss the particular challenges raised by conversational interaction with autonomous systems such as robots. Section 3 describes the robot application with which our current dialogue system interacts, and the architecture of the dialogue system. In Section 4 we introduce the "joint activities" and Activity Models which form an interface layer between the dialogue system and autonomous devices. Section 5 presents the dialogue modelling and management techniques used to handle multiple topics and collaborative activities. Section 6 explains the message selection and generation component of the system.

2. Dialogues with robots

A useful dialogue system for interaction with robots will enable collaboration between a human and a robot in the planning and completion of tasks (see e.g. [CRA 94, CRA 97]). Dialogue will be used to specify and clarify instructions and goals for the robot, to monitor its progress, and also to solve problems jointly. Before we deal with such issues in detail, we note that robots also have the following properties which are relevant from the point of view of dialogue management :

– Robots exist within dynamic environments, where new objects appear and are available for discussion. Their sensors may give rise to new information at any time, and this may need to be communicated urgently.

– Robots perform multiple concurrent activities which may succeed, fail, become cancelled, or be revised. These activities can be topics of conversation.

[ALL 01] present a taxonomy of dialogue systems ranging from "finite-state script" dialogues for simple tasks (such as making a long-distance call) to the most complex "agent-based models" which cover dialogues where different possibilities, such as future plans, are discussed. Within this taxonomy, a useful dialogue system for interaction with autonomous robots must be located at or near the "agent-based" point since we wish to communicate with robots about their possible actions, their plans, and the tasks they are currently attempting. For these reasons we built a dialogue manager that represents (possibly collaborative) activities and their execution status, and tracks multiple threads of dialogue about concurrent and planned activities.

Command and control tasks are typically carried out by trained personnel using a restricted command language, to guarantee reliability and robustness. Our interface is designed to support usability by non-experts using standard English, while retaining and in some senses enhancing reliability. One requirement for reliability is that translation from human commands to robot actions is consistent and transparent. In our system, the user can employ many different linguistic constructions to effect the same action (e.g. "go the tower", "fly here" [click on tower]), and spoken feedback is always given to ensure that the user is appraised of how the robot understood their command. On the speech output side, the robot uses variation and anaphora only in as much as is required for naturalness which avoids misunderstanding (see Section 6). For instance, the robot will refer to a red car that the user has spoken about either as "it" or "the car". If the robot were always to use the phrase "the red car", users could be misled into believing that a new red car is being spoken about.

As well as considerations generated by multi-tasking with robots, the general flexibility of conversational interaction with autonomous systems places the following requirements on a dialogue manager (see also [CLA 96]) :

1) *Asynchronicity* : events in the dialogue scenarios can happen at overlapping time periods (for example, new objects may enter the domain of discourse while the operator is giving a command).

2) *Mixed task-initiative* : in general, both operator and system will introduce issues for discussion.

3) *Open-ended* : there are no clear start and end points for the dialogue and subdialogues, nor are there rigid pre-determined goals for interchanges.

4) *Resource-bounded* : participants' actions must be generated and produced in time enough to be effective dialogue contributions.

5) *Simultaneous* : participants can produce and receive actions simultaneously.

For these sorts of reasons it is clear that finite state networks ("dialogue graphs"), and form-filling or data-base query style dialogues (e.g. the CSLU Toolkit [MCT 98]) will not suffice here (see [ROY 00, ELI 99, ALL 01] for similar arguments).

3. The WITAS Dialogue System

In our current application, the autonomous system is the WITAS[2] UAV ('unmanned aerial vehicle'), a small robotic helicopter with on-board planning and deliberative systems and vision capabilities (for details see e.g. [DOH 00]). This robot helicopter will ultimately be controlled by the dialogue system developed at CSLI, though at the moment we interact with simulated[3] UAVs. Mission goals are provided by a human operator, and an on-board planning system then responds.

While the helicopter is airborne, an on-board active vision system interprets the scene or focus below to interpret ongoing events, which may be reported (via NL generation) to the operator (see Section 6). The robot can carry out various "activities" such as flying to a location, or following a vehicle, or landing. These activities are specified by the user during dialogue, or can be initiated by the UAV's on-board AI. In any case, a major component of the dialogue, and a way of maintaining its coherence, is tracking the state of current or planned activities of the robot.

A further issue is the co-ordination of "joint-activities" between an autonomous system and a human operator. These are activities which the autonomous system cannot complete alone, but which require some human intervention. In our current scenarios, the UAV's vision system is not good enough to determine whether a particular vehicle is the one sought-after, and only the human operator has the authority to determine this, so that human and robot must collaborate in order to find and track a vehicle. The dialogue in Figure 1 shows how a typical interaction works[4] (other capabilities, such as clarification subdialogues, are covered in [LEM 01a]). Note here that the user is able to make explicit queries about the robot's activities (both current and future), that there are concurrent activities, and that conversational initiative centers around the joint activities currently being specified and executed.

3.1. *Dialogue System Architecture*

As argued above, robot interaction scenarios present a number of challenges to designers of dialogue systems. Such systems require a particularly flexible architecture – one which can coordinate multiple asynchronous communicating processes. For these reasons we currently use the Open Agent Architecture (OAA2, see [MAR 99]), with the following agents (see Figure 2) ;

2. See `http://www.ida.liu.se/ext/witas`
3. CSLI's UAV simulator is logic-based, using KIF statements under JTP (the Java Theorem Prover) to represent and non-monotonically update UAV state information. The Linköping UAV simulator, to which we can also interface, runs in Lisp.
4. The system runs in real time on a laptop computer under Windows 2000. Video footage of the system in action can be found at `http://www-csli.stanford.edu/semlab/witas/`

Multi-modal Utterances	**Dialogue Move**
Operator (O) : *Our job is to look for a red car*	Command (Joint Activity)
UAV (U) : *Ok. I am looking for one.*	Report (Confirm Activity)
O : *Fly here please [+click on map]*	Command (Deictic)
U : *Okay. I will fly to waypoint one*	Report (Confirm Activity)
U : *Now taking off and flying there.*	Report (Current Activity)
O : *Stop that. Go to the tower instead.*	Command, Revision
U : *I have cancelled flying to waypoint one. I will fly to the tower.*	Report (Activity status)
O : *What are you doing ?*	Wh-question (Current Activity)
U : *I am searching for a red car and flying to the tower*	Answer (Current Activity)
O : *What will you do next ?*	Wh-question (Planned Activity)
U : *I have nothing planned.*	Answer(Planned Activity)
U : *I see a red car on main street* [display on map, show video images], *Is this the right car ?*	Report, Yn-question (Activity)
O : *Yes, that's the right car*	Yn-answer (Positive)
U : *Okay. I am following it .*	Report (Current activity)

Figure 1. *A demonstration of the WITAS dialogue system (November 2001)*

1) *NL (natural language) :* a wrapper to SRI's Gemini parser and generator using a grammar for human-robot conversation developed at CSLI. Our current grammar has 166 grammar rules and 749 lexical entries.

2) *SR (speech recognizer) :* a wrapper to a Nuance 8 speech recognition server using a language model compiled directly from the Gemini grammar (with the consequences that every recognized utterance has a logical form, and that every logical form can be mapped to a surface string).

3) *TTS (text-to-speech) :* a wrapper to the Festival 1.4.3 speech synthesizer [TAY 98], for system speech output.

4) *GUI :* an interactive map display of the current operating environment which displays route plans, waypoints, locations of vehicles including the robot, and allows deictic reference (i.e. mouse pointing) by the user.

5) *DM (dialogue manager) :* co-ordinates multi-modal inputs from the user, interprets dialogue moves made by the user and system, updates and maintains the dialogue context, handles robot reports and questions, and sends speech and graphical outputs to the user (see Figure 4).

6) *Activity Layer :* translates commands and queries from the dialogue interface into commands and queries to the robot, and vice-versa for reports and queries received from the robot. Uses an Activity Model (see Section 4).

Variants of some of these components have been used in other dialogue systems, notably SRI's CommandTalk [STE 99b], the NASA Personal Satellite Assistant [RAY 00],

and the robot control system of [GUZ 96]. However, our system stands apart from these with respect to its Dialogue Manager and its support for multi-tasking and collaborative activities.

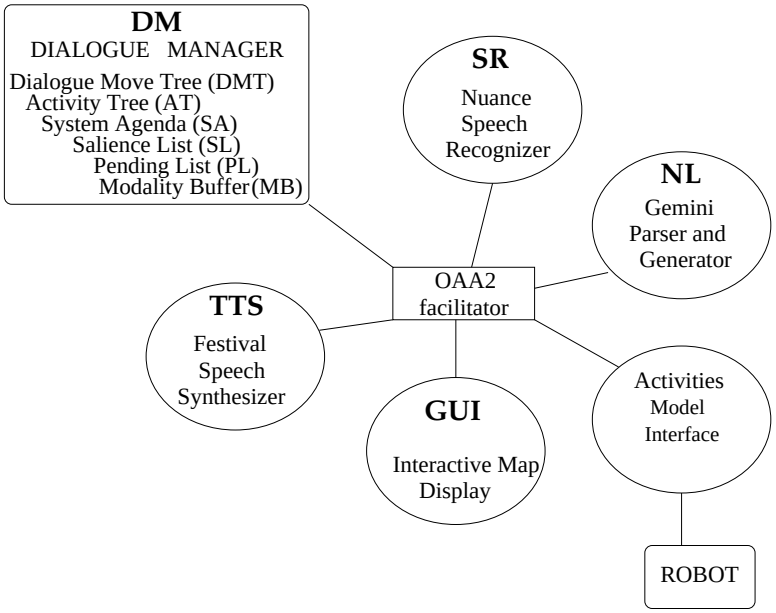

Figure 2. *The dialogue system architecture*

4. Activity Models in the CSLI Dialogue System

The idea of Activity Modelling in our system is the vision that dialogue systems can, in general, be built for 'devices' which carry out certain well-defined activities (e.g. switch lights on, record on channel n, send email E to X, search for vehicle v), and that an important part of the dialogue context to be modelled in such a system is the device's planned activities, current activities, and their execution status. This is similar to the motivation behind the "Pragmatic Adapter" idea outlined in [LUP 98]. We also share with [RIC 01] the idea that declarative descriptions of the goal decomposition of activities (COLLAGEN's "recipes", our "Activity Models") are a vital layer of representation, between a dialogue system and the device with which it interacts.

In general we assume that a device is capable of performing some "atomic" activities or actions (possibly simultaneously), which are the lowest-level actions that it can perform. Some devices will only know how to carry out sequences of atomic activities, in which case it is the dialogue system's job to decompose linguistically specified high-level activities (e.g. "record the film on channel 4 tonight") into a sequence of

appropriate atomic actions for the device. In this case the dialogue system is provided with a declarative "Activities Model" (see e.g. Figure 3) for the device which states how high-level linguistically-specified activities can be decomposed into sequences of atomic actions. This model contains traditional planning constraints such as preconditions and postconditions of actions. In this way, a relatively "stupid" device (i.e. with little or no planning capabilities) can be made into a more intelligent device when it is dialogue-enabled.

At the other end of the spectrum, more intelligent devices are able to plan their own sequences of atomic actions, based on some higher level input. In this case, it is the dialogue system's role to translate natural language into constraints (including temporal constraints) that the device's planner recognizes. The device itself then carries out planning, and informs the dialogue manager of the sequence of activities that it proposes. Dialogue can then be used to re-specify constraints, revise activities, and monitor the progress of tasks. At the very least, we propose that the process of decomposing a linguistically specified command (e.g. "vacuum in the main bedroom and the lounge, and before that, the hall") into an appropriate sequence of constraints for the device's on-board planner, is an aspect of "conversational intelligence" that can be added to devices by dialogue-enabling them.

We are developing one representation and reasoning scheme to cover this spectrum of cases from devices with no planning capabilities to some more impressive on-board AI. Both dialogue manager and robot/device have access to a single "Activity Tree" which is a shared representation of current and planned activities and their execution status, involving temporal and hierarchical ordering (in fact, one can think of the Activity Tree as a Hierarchical Task Network [ERO 94] for the robot). This tree is built top-down by processing verbal input from the user, and its nodes are then expanded by the device's planner (if it has one). In cases where no planner exists, the dialogue manager itself expands the whole tree (via the Activity Model for the device) until only leaves with atomic actions are left for the device to execute in sequence. The device reports completion of activities that it is performing and any errors that occur for an activity.

Note that because the device and dialogue system share the same representation of the device's activities, they are always properly coordinated. They also share responsibility for different aspects of constructing and managing the whole Activity Tree. Note also that some activities can themselves be speech acts, and that this allows us to build collaborative activities into the system. For example, in Figure 3 the ASK-COMPLETE activity initiates a speech act, generating a *yes-no question* to be answered by the user. In the latest version of the MURI tutorial dialogue system [CLA 01], developed at CSLI using this software, almost all of the activities are speech acts.

4.1. *An example Activity Model*

An example LOCATE activity model for the robot UAV is shown in Figure 3. It is used when constructing parts of the activity tree involving commands such as "search for", "look for" and so on. For instance, if the user says "We're looking for a truck", that utterance is parsed into a logical form involving the structure `(locate, np[det(a),truck])`.

The dialogue manager then accesses the Activity Model for LOCATE and adds a node to the Activity Tree describing it. The Activity Model specifies what sub-activities should be invoked, and under what conditions they should be invoked, what the postconditions of the activity are. Activity Models are similar to the "recipes" of [RIC 01]. For example, in Figure 3 the Activity Model for LOCATE states that,

– it uses the `camera` resource (so that any other activity using the camera must be suspended, or a dialogue about resource conflict must be initiated),

– that the preconditions of the activity are that the UAV must be airborne, with fuel and engine indicators satisfactory,

– that the whole activity can be skipped if the UAV is already "locked-on" to the sought object,

– that the postcondition of the activity is that the UAV is "locked-on" to the sought object,

– that the activity breaks into three sequential sub-activities : WATCH-FOR, FOLLOW-OBJ, and ASK-COMPLETE.

Nodes on the Activity Tree can be in one of the following states :

– active

– complete

– failed

– suspended

– canceled.

Any change in the state of a node (typically because of an action or report from the robot) is placed onto the System Agenda (see Section 5) for possible verbal report to the user, via the message selection and generation module (see Section 6).

5. The Dialogue Context Model

Dialogue management falls into two parts – dialogue modelling (representation), and dialogue control (algorithm). In Section 5 we focus on the representational aspects, and Section 5.2 surveys the main algorithms. As a representation of conversational context, the dialogue manager uses the following data structures which make up the dialogue Information State (*IS*) ;

```
Locate// locate is "find-by-type", collaborative activity.
  //Breaks into subactivities: watch_for, follow, ask_complete
  {ResourcesUsed {camera;}     // will be checked for conflicts
     PreConditions             //check truth of KIF statements
       {(Status flight inair)
        (Status engine ok)
        (Status fuel ok)     ;}
     SkipConditions // skip this Activity if KIF condition true
       {(Status locked-on THIS.np);}
     PostConditions// assert these KIF statements when completed
       {(Status locked-on THIS.np) ;}
     Children SEQ                  //sequential sub-activities
      {TaskProperties
        {command = "watch_for";   //Starts a basic robot action:
                   np = THIS.np;} // set sensors to search.
       TaskProperties
        {command = "follow-obj";//Starts a new complex activity
                    np = THIS.np;} //follow a candidate object.
       TaskProperties             // Collaborative dialogue:
          {command = "ask_complete";// ask user whether this is
             np = THIS.np;  }}}     //object we are looking for.
```

Figure 3. *A "Locate" Activity Model for a robot UAV, exhibiting collaborative dialogue*

– Dialogue Move Tree (DMT)
– Activity Tree (AT)
– System Agenda (SA)
– Pending List (PL)
– Salience List (SL)
– Modality Buffer (MB).

Figure 4 shows how the Dialogue Move Tree (DMT) relates to other parts of the dialogue manager as a whole. The solid arrows represent possible update functions, and the dashed arrows represent query functions. For example, the Dialogue Move Tree can update the Salience List, System Agenda, Pending List, and Activity Tree, while the Activity Tree can update only the System Agenda and send execution requests to the robot, and it can query the Activity Model (when adding nodes). Likewise, the Message Generation component queries the System Agenda and the Pending List, and updates the Dialogue Move Tree whenever a synthesized utterance is produced.

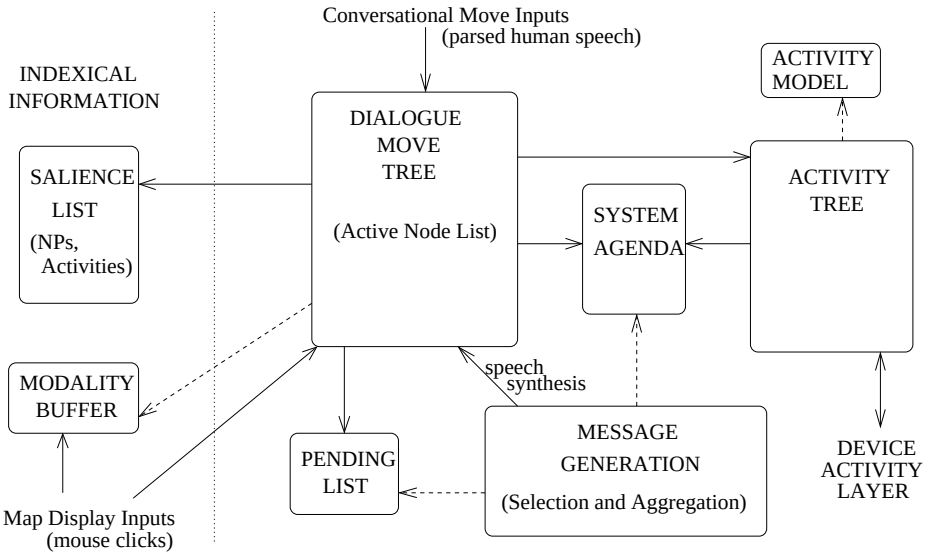

Figure 4. *Dialogue Manager Architecture (solid arrows denote possible updates, dashed arrows represent possible queries)*

Figure 5 shows an example Information State logged by the system, displaying the interpretation of the system's utterance "now taking off" as a report about an ongoing "go to the tower" activity (the Pending List and System Agenda are empty in this example, and thus are not shown).

Note that the Salience List is, in general, a list of "groups" of NPs with the same salience, ordered by primarily by recency. We use "groups" because some NPs have the same salience, for example in "go to the tower and the tree", [tower, tree] form a salience "group", since a follow-up utterance such as "drop medical supplies there" would refer to both locations. The Salience List is maintained via the update functions in dialogue moves when nodes are added to the DMT.

5.1. *Dialogue Moves and the Dialogue Move Tree*

Dialogue management uses a set of abstract dialogue move classes which are domain independent (e.g. *command, activity-query, wh-question, revision*, . . .). Any ongoing dialogue constructs a particular Dialogue Move Tree (DMT) representing the current state of the conversation, whose nodes are instances of the dialogue move classes, and which are linked to nodes on the Activity Tree where appropriate, via an *activity tag* (see below).

```
Utterance: ‘‘now taking off’’   (by System 11/7/01 4:50 PM)
Conversational Move:
report(inform,agent([np([n(uav,sg)])]),
                          curr_activity([command([take_off])]))

Dialogue Move Tree (position on ANL in parens [0=most active])
 * Root (1)
   Root
    o Command (0)
     command([go],[param_list([pp_loc(to,arg([np(det([def],the),
                       [n(tower,sg)])]))])])[[dmtask0] current]
        + Report
          report(inform,agent([np([n(uav,sg)])]),curr_activity(
                                        [command([take_off])]))[]
    o Report
      report(inform,agent([np([n(uav,sg)])]),confirm_activity(
                  [command([go],[param_list([pp_loc(to,arg(
                   [np(det([def],the),[n(tower,sg)],
                               )])])])])]))[[dmtask0] current]

Activity Tree
   * root
        o [dmtask0] current
          relation = SEQuential
          command = go
          pp = pp_loc(to,Args)
          np = np(det([def],the),[n(tower,sg)])
             + [sim3] current
               relation = none
               command = take_off
               pp = null, np = null

Salience List (least salient -- most salient)
   * [np(det([def],the),[n(tower,sg)])] (speech)
   * [np(det([def],the),[n(tower,sg)])] (speech)
```

Figure 5. *A snapshot of an Information State (from the HTML system logs)*

Incoming logical forms (LFs) from the parsing process are always headed with a dialogue move (see e.g. [GIN 01]), which precedes more detailed information about an utterance. For instance the logical form :

```
command([go],[param_list([pp_loc(to,arg([np(det([def],the),
                                              [n(tower,sg)])]))])])
```

corresponds to the utterance “go to the tower”, and is flagged as a *command*. A more complex example is :

```
report(inform,agent([np([n(uav,sg)])]),compl_activity
                                        ([command([take_off])]))
```

which corresponds to “I have taken off” – a report from the UAV about a completed ‘taking-off’ activity.

In general, we think of Conversational Moves as falling under the following definition.

> [Definition 1]
> A *Conversational Move* (CM) is a structure (*Input Logical Form*, *Activity Tag*, *Agent*), where *Input Logical Form* is the logical form of whatever utterance has just been made, *Activity Tag* is a (possibly empty) field specifying which activity on the activity tree generated the input, and *Agent* is the identifier of the speaker of the input.

Thus, an example conversational move is :

```
(report(inform,agent([np([n(uav,sg)])]),compl_activity
                                ([command([take_off])])), t1, uav)
```

The first problem in dialogue management is to figure out how these incoming Conversational Moves relate to the current dialogue context. In other words, what dialogue moves do they constitute, and how do they relate to previous moves in the conversation? In particular, given multi-tasking, to which thread of the conversation does an incoming utterance belong? We use the Dialogue Move Tree to answer these questions :

1) A DMT is a history or “message board” of dialogue contributions, organized by “thread”, based on activities.

2) A DMT classifies *which* incoming utterances can be interpreted in the current dialogue context, and which cannot be. It thus delimits a space of possible Information State update functions.

3) A DMT has an *Active Node List* (ANL) which controls the order in which this function space is searched.

4) A DMT classifies *how* incoming utterances are to be interpreted in the current dialogue context.

In general, then, we can think of the DMT as representing a function space of dialogue Information State update functions of the following form :

f : *Node* $\times$ *Conversational Move* $\rightarrow$ *Information State Update*

where *Node* is an *active node* on the dialogue move tree, a *Conversational Move* (CM) is a structure (*Input Logical Form*, *Activity Tag*, *Agent*) and an *Information State Update* is a function $g : IS \rightarrow IS$ which changes the content of the current *IS*.

The details of any particular update function are determined by the node type (e.g. *command*, *question*) and incoming dialogue move type and their contents, as well as the values of *Activity Tag* and *Agent*. We discuss this further below.

Note that this technique is a variant of "conversational games", also known as "dialogue games" [CAR 83, POW 79], "interactions" [HOU 86], and, in the context of task-oriented dialogues, "discourse segments" [GRO 86]. All these accounts rely on the observation that answers generally follow questions, commands are generally acknowledged, and so on, so that dialogues can be partially described as consisting of "adjacency pairs" of such dialogue moves. Our notion of "attachment" of dialogue moves (see Section 5.2) embodies this idea.

5.2. *Interpretation and State Update*

The central algorithm controlling dialogue management has two main steps, *Attachment*, and *Process Node* ;

1) *Attachment* : process incoming input conversational move c with respect to the current DMT and Active Node List, and "attach" a new node N interpreting c to the tree if possible (i.e. find the most active node on the DMT of which the new node can be a daughter, and add the new node at that location).

2) *Process Node* : process the new node N, if it exists, with respect to the current information state. Perform an Information State update using the dialogue move type and content of N.

When an update function g exists, its effects depend on the details of the incoming input c (in particular, on the dialogue move type and the contents of the logical form) and the DMT node to which it attaches. The possible attachments can be thought of as adjacency pairs, and each dialogue move class contains information about which node types it can attach.

Examples of different attachments available in our current system can be seen in Figure 6[5]. For example, the first entry in the table states that a *command* node, generated by the user, with activity tag t, is able to attach any system *confirmation* move with the same activity tag, any system *yes-no question* with that tag, any system *wh- question* with that tag, or any system *report* with that activity tag. Similarly, the rows for *wh-question* nodes state that :

5. Where Activity Tags are not specified, attachment does not depend on sharing of Activity Tags.

– a *wh-question* by the system with activity tag t can attach a user's *wh-answer* (if it is a possible answer for that activity)

– a user's *wh-question* can attach a system *wh-answer*, and no particular activity need be specified.

DMT Node			**Attaches**		
Node Type	**Activity Tag**	**Speaker**	**Node Type**	**Activity Tag**	**Speaker**
command	t	user	confirmation,	t	system
			y-n question,	t	system
			wh-question,	t	system
			report	t	system
confirmation	t	system			
report	t	system	command	t	user
wh-question	t	system	wh-answer	t	user
wh-question		user	wh-answer		system
yn-question	t	system	yn-answer	t	user
revision	t	user	wh-question	t	system
yn-answer	t	user	confirmation	t	system
wh-answer		user	confirmation		system
wh-answer		system	confirmation		user
root	n/a	n/a	command,		user
			question,		user
			revision		user
root	n/a	n/a	report		system

Figure 6. *Attachment in the Dialogue Move Classes*

These possible attachments delimit the ways in which dialogue move trees can grow, and thus classify the dialogue structures which can be captured in the current system. As new dialogue move types are added to the system, this table will be extended and generalized to cover other conversation types (e.g. tutoring [CLA 01, HOC 02]). Note that the system's dialogue moves appear on the DMT, just as the user's do – so that in some sense the dialogue system is an interloper, attempting to follow a conversation that it overhears. Of course, the dialogue system has no interpretation problem regarding the intentions behind its own utterances, so the symmetry is by no means exact.

It is worth noting that the node type created after attachment may not be the same as the dialogue move type of the incoming conversational move c, and that Figure 6 does not state what dialogue move type a new input is attached as. Depending on the particular node which attaches the new input, and the move type of that input, the created node may be of a different type. For example, if a *wh-question* node attaches an input which is simply a *command*, the *wh-question* node may interpret the input as an answer, and attach a *wh-answer*. These interpretation rules are local to the node to

which the input is attached. In this way, the DMT interprets new input in context, and the pragmatics of each new input is contextually determined, rather than completely specified via parsing using conversational move types.

6. Message selection, aggregation, and generation

Since the robot is potentially carrying out multiple activities at once, a particular problem is how to determine appropriate generation of utterances about those activities, in a way which does not overload the user with information, yet which establishes and maintains appropriate context in a natural way.

Generation for dialogue systems in general is problematic in that dialogue contributions arise incrementally, often in response to another participant's utterances. For this reason, generation of large pieces of text is not appropriate, especially since the user is able to interrupt the system. Other differences abound, for example that aggregation rules must be sensitive to incremental aspects of message generation.

As well as the general problems of message selection and aggregation in dialogue systems, this particular type of application domain presents specific problems in comparison with, say, travel-planning dialogue systems. An autonomous robotic device will, in general, need to communicate about,

– its perceptions of a changing environment,

– progress towards user-specified goals,

– execution status of activities or tasks,

– its own internal state changes,

– the progress of the dialogue itself.

For these reasons, the message selection and generation component of such a system needs to be of wider coverage and more flexible than template-based approaches, while remaining in real, or near-real, time [STE 99a]. As well as this, the system must potentially be able to deal with a large bandwidth stream of communications from the robot, and so must be able to intelligently filter them for "relevance" so that the user is not overloaded with unimportant information, or repetitious utterances. In general, the system should appear as "natural" as possible from the user's point of view – using the same language as the user if possible ("echoing"), using anaphoric referring expressions where possible, and aggregating utterances where appropriate. A "natural" system should also exhibit "variability" in that it can convey the same content in a variety of ways. A further desirable feature is that the system's generated utterances should be in the coverage of the dialogue system's speech recognizer, so that system-generated utterances effectively prime the user to speak in-grammar (see [HOC 02]).

Consequently we attempted to implement the following features in the message generation component of our dialogue system :

1) relevance filter

2) recency filter

3) echoing

4) variability

5) aggregation

6) symmetry and priming - using only “in grammar” utterances

7) real-time message generation.

Our general method is to take as inputs to the process various communicative goals of the system, expressed as logical forms, and use them to construct a single new logical form to be input to Gemini’s Semantic Head-Driven Generation algorithm [SHI 90], which produces strings for Festival speech synthesis [TAY 98].

The novel features of our message generation system derive from the rich dialogue context to which the generation module has access. We now describe how to use our notion of dialogue Information State to produce natural generation in multitasking contexts.

6.1. *Message selection*

Inputs to the selection and generation module are “concept” logical forms describing the communicative goals of the system. These are structures consisting of *context tags* (e.g. activity identifier, dialogue move tree node, turn tag) and a *content logical form* consisting of a Dialogue Move (e.g. *report, wh-question*), a priority tag (e.g. *warn* or *inform*), and some additional content tags (e.g. for objects referred to). An example input logical form (LF) is,

```
report(inform, agent(AgentID), cancel_activity(ActivityTag))
```

which corresponds to the report “I have cancelled flying to the tower” when AgentID refers to the robot and ActivityTag refers to a “fly to the tower” task.

Items which the system will consider for generation are placed (either directly by the robot, or indirectly by the Activity Tree) on the “System Agenda” (SA), which is the part of the dialogue Information State which stores communicative goals of the system. Communicative goals may also exist on the “Pending List” (PL) which is the part of the Information State which stores questions that the system has asked, but which the user has not answered, so that they may be re-raised by the system. Only questions previously asked by the system can exist on the Pending List. Note that the Pending List is merely a convenience, which caches calculation of unanswered system questions on the Dialogue Move Tree. Questions which have been re-raised three[6] times, but have still not been answered, are removed from the Pending List, and their corresponding node on the DMT is closed.

6. This number is arbitrary, and should be set adaptively in future systems.

Note that other items, for example confirmations which have not yet been spoken, are not placed on the Pending List, but instead go on the System Agenda. The System Agenda is used for items that are queued up to be said as soon as possible, while the Pending List is a list of items that need to be re-added to the System Agenda unless they are dealt with.

Due to multi-tasking, at any time there is a number of "Current Activities" which the user and system are performing (e.g. fly to the tower, search for a red car). These activities are topics of conversation (defining threads of the DMT) represented in the dialogue Information State, and the system's reports can be generated by them (in which case they are tagged with that activity label) or can be relevant to an activity in virtue of being about an object which is in focus because it is involved in that activity.

Some system reports are more urgent that others (e.g. "I am running out of fuel") and these carry the label *warning*. Warnings are always relevant, no matter what activities are current – they always pass the recency and relevance filters. If a warning is triggered while the system is formulating an utterance, the warning goes on top of the System Agenda. In fact, the probability of this happening is very low anyway, since messages only take in the order of a few milliseconds to formulate. In the more likely case where the warning comes in when the system is speaking, the system does not interrupt itself, but queues the warning at the top of the System Agenda.

Echoing (for noun-phrases) is achieved by accessing the Salience List whenever generating referential terms, and using whatever noun-phrase (if any) the user has previously employed to refer to the object in question. If the object is top of the Salience List, the generator will select an anaphoric expression.

6.2. *Incremental aggregation*

Aggregation [APP 85] combines and compresses utterances to make them more concise, avoid repetitious language structure, and make the system's speech more natural and understandable overall. Aggregation techniques on a prewritten body of text combine and compress sentences that have already been determined and ordered. In a dialogue system however, aggregation should produce similarly natural output, but must function incrementally because utterances are generated on the fly. In dialogue systems, when constructing an utterance we often have no information about the utterances that will follow it, and thus the best we can do is to compress it or "retro-aggregate" it with utterances that preceded it (see the example below). Only occasionally does the System Agenda contain enough unsaid utterances to perform reasonable "pre-aggregation".

Each dialogue move type (e.g. *report*, *wh-question*) has its own aggregation rules, stored in the class for that LF type. In each type, rules specify which other dialogue move types can aggregate with it, and exactly how aggregation works. The rules note identical portions of LFs and unify them, and then combine the non-identical portions appropriately.

For example, the LF that represents the phrase “I will fly to the tower and I will land at the parking lot”, will be converted to one representing “I will fly to the tower and land at the parking lot” according to the compression rules. Similarly, “I will fly to the tower and fly to the hospital” gets converted to “I will fly to the tower and the hospital”.

The “retro-aggregation” rules result in sequences of system utterances such as, “I have cancelled flying to the school. And the tower. And landing at the base.”

The end result of our selection and aggregation module is a fully specified logical form which is to be sent to the Semantic-Head-Driven Generation component of Gemini [DOW 93, SHI 90]. The bi-directionality of Gemini (i.e. that we use the same grammar for both parsing and generation) automatically confers a useful “symmetry” property on the system – that it only utters sentences which it can also understand. This means that the user will not be misled by the system into employing out-of-vocabulary items, or out-of-grammar constructions. Thus, as desired, the system’s utterances prime the user to make in-grammar utterances.

A final aspect to note is that the system is able to perform limited multi-modal generation using its map display (see Figure 7) and a video output window. Whenever an object is mentioned in the spoken dialogue, its icon is highlighted on the map. In some cases, questions (e.g. “where is the tower ?”) are best answered with multimodal output, in which case the System Agenda is given a simple answer to say (e.g. “here you are”) and the appropriate icon(s) are highlighted on the map display.

7. Summary

Our first task was to motivate our focus on interleaved or “multi-threaded” dialogue, and its usefulness in interaction with autonomous devices. We then explained the domain-general dialogue modelling techniques which we implemented in order to build a real-time multi-modal conversational interface to an autonomous mobile robot (the WITAS UAV). The novel issues tackled by the system and its dialogue model are that it is able to manage conversations about multiple tasks and collaborative activities in a robust and natural way.

We argued that in the case of dialogues with robots, a dialogue management mechanism has to be particularly robust and flexible, especially in comparison with finite-state or frame-based dialogue managers which have been developed for information-seeking dialogues, such as travel planning, where topics of conversation are predetermined. Another challenge was that conversations may have multiple open topics at any one time, and this complicates utterance interpretation and generation. Although more ambitious, a dialogue management system for autonomous robots is more general, and teaches us more about the nature of dialogue in a wider sense.

We discussed the dialogue context model and algorithms used to produce a system with the following features :

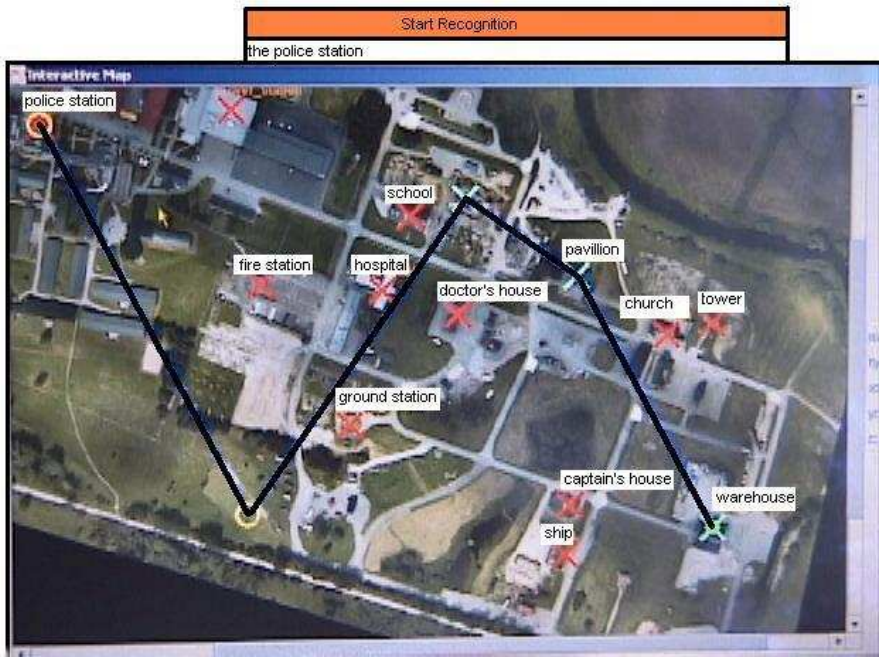

Figure 7. *Part of the Graphical User Interface, showing a flight plan*

1) a multi-threaded dynamic Information State model of dialogue,

2) supports multi-tasking, multiple topics, and collaboration,

3) support of commands, questions, revisions, and reports, over a dynamic environment,

4) multi-modal, mixed-initiative, open-ended dialogues,

5) Semantic-Head-Driven Generation [SHI 90] of system reports,

6) echoic and variable message generation, filtered for relevance and recency,

7) asynchronous, real-time operation,

8) interfaces to real-time UAV simulators.

7.1. *Future work*

The system described here is the prototype of more general dialogue system for collaboration with semi-autonomous agents. Recent work at CSLI includes the deve-

lopment of a tutorial dialogue system[7] using the same software base (i.e. this Dialogue Manager, OAA, Gemini, Nuance, Festival). In future work, we plan to evaluate how well our dialogue management architecture (the Dialogue Move Tree and Activity Models) handles tutorial dialogues, and the prospects for multi-tasking dialogue systems in interactive entertainment [LEM 02].

7.2. *Evaluation*

As part of the research described in [HOC 02], we have performed only a limited evaluation of the system thus far. Ten student volunteers, with no previous experience using the system, have each completed a session consisting of five tasks (an example task is "Search the area for a red truck. Follow it until it stops, then land back where you started"). Each subject was able to complete all the tasks[8], and data has been collected regarding completion time, steps to completion, and speech recognition error rates. All dialogues have been recorded, and the Information States logged as HTML files (see Figure 5). We plan a much larger evaluation effort in the near future. For instance, it would be interesting to discover how often the system gets the attachment of dialogue moves correct (i.e. recognition of user intention), compared to human attachment judgements.

Acknowledgements

This research was (partially) funded under the Wallenberg laboratory for research on Information Technology and Autonomous Systems (WITAS) Project, Linköping University, by the Wallenberg Foundation, Sweden. We wish to thank Anne Bracy for her work on the Gemini grammar while at CSLI, Alexis Battle for her work on the generation component, Patrick Doherty of WITAS, David Martin of SRI, Didier Guzzoni, David Traum of ICT, and John Dowding and Beth-Ann Hockey of NASA/RIACS. We also wish to thank the referees for raising many valuable questions.

8. Bibliographie

[ALL 96] ALLEN J. F., MILLER B. W., RINGGER E. K., SIKORSKI T., « A Robust System for Natural Spoken Dialogue », *Proceedings of ACL*, 1996.

[ALL 01] ALLEN J., BYRON D., DZIKOVSKA M., FERGUSON G., GALESCU L., STENT A., « Toward Conversational Human-Computer Interaction », *AI Magazine*, vol. 22, n° 4, 2001, p. 27-37.

7. Information on the tutorial dialogue system is available at `www-csli.stanford.edu/semlab/muri/`

8. Except for one case where the PC's soundcard broke.

[APP 85] APPELT D. E., « Planning English Referring Expressions », *Artificial Intelligence*, vol. 26, n° 1, 1985, p. 1 - 33.

[BOH 99] BOHLIN P., COOPER R., ENGDAHL E., LARSSON S., « Information States and Dialog Move Engines », *Electronic Transactions in AI*, vol. 3, n° 9, 1999, Website with commentaries : `www.etaij.org`.

[CAR 83] CARLSON L., *Dialogue Games : An Approach to Discourse Analysis*, D. Reidel, 1983.

[CLA 96] CLARK H. H., *Using Language*, Cambridge University Press, 1996.

[CLA 01] CLARK B., FRY J., GINZTON M., PETERS S., PON-BARRY H., THOMSEN-GRAY Z., « Automated Tutoring Dialogues for Training in Shipboard Damage Control », *Proceedings of SIGdial 2001*, 2001.

[COO 98] COOPER R., LARSSON S., « Dialog Moves and Information States », rapport n° 98-6, 1998, Goteborg University, Gothenburg papers in Computational Linguistics.

[CRA 94] CRANGLE C., SUPPES P., *Language and Learning for Robots*, CSLI Publications, 1994.

[CRA 97] CRANGLE C., « Conversational Interfaces to Robots », *Robotica*, vol. 15, 1997, p. 117 - 127.

[DOH 00] DOHERTY P., GRANLUND G., KUCHCINSKI K., SANDEWALL E., NORDBERG K., SKARMAN E., WIKLUND J., « The WITAS Unmanned Aerial Vehicle Project », *European Conference on Artificial Intelligence (ECAI 2000)*, 2000.

[DOW 93] DOWDING J., GAWRON J. M., APPELT D., BEAR J., CHERNY L., MOORE R., MORAN D., « GEMINI : a Natural Language system for spoken-language understanding », *Proc. 31st Annual Meeting of the ACL*, 1993.

[ELI 99] ELIO R., HADDADI A., « On Abstract Task Models and Conversation Policies », *Workshop on Specifying and Implementing Conversation Policies, Autonomous Agents'99*, Seattle, 1999.

[ERO 94] EROL K., HENDLER J., NAU D. S., « HTN Planning : Complexity and Expressivity », *Proceedings of AAAI*, 1994.

[FRY 98] FRY J., ASOH H., MATSUI T., « Natural Dialogue with the Jijo-2 Office Robot », *IEEE/RSJ International Conference on Intelligent Robots and Systems IROS-98*, Victoria, B.C., Canada, 1998, p. 1278-1283, (See `www-csli.stanford.edu/semlab/juno`).

[GIB 00] GIBBON D., MERTINS I., MOORE R., *Handbook of Spoken and Multi-modal Dialogue Systems*, Kluwer, 2000.

[GIN 96a] GINZBURG J., LAPPIN S., Ed., *Interrogatives : Questions, facts and dialogue*, chapitre The Handbook of Contemporary Semantic Theory, 1996.

[GIN 96b] GINZBURG J., « Dynamics and the semantics of dialogue », SELIGMAN, WESTERSTAHL, Eds., *Logic, Language, and Computation*, 1996.

[GIN 01] GINZBURG J., SAG I. A., PURVER M., « Integrating Conversational Move Types in the Grammar of Conversation », *Bi-Dialog 2001—Proceedings of the 5th Workshop on Formal Semantics and Pragmatics of Dialogue*, 2001, p. 45–56.

[GRO 86] GROSZ B., SIDNER C., « Attentions, intentions, and the structure of discourse », *Computational Linguistics*, vol. 12, n° 3, 1986, p. 175-204.

[GUZ 96] GUZZONI D., CHEYER A., JULIA L., KONOLIGE K., « Many Robots Make Short Work », *AAAI Robotics Contest*, Menlo Park, CA., 1996, SRI International, AAAI Press.

[HOC 02] HOCKEY B.-A., AIST G., HIERONYMOUS J., LEMON O., DOWDING J., « Targeted Help : Embedded training and methods for evaluation », *Proceedings of Intelligent Tutoring Systems (ITS)*, 2002, (to appear).

[HOU 86] HOUGHTON G., « The Production of Language in Dialogue : A Computational Model », PhD thesis, University of Sussex, 1986.

[KUP 00] VAN KUPPEVELT J., HEID U., KAMP H., « Best Practice in Spoken Language Dialogue System Engineering », *Natural Language Engineering*, vol. 6, 2000.

[LEM 98] LEMON O., « First Order Theory Change systems and their Dynamic Semantics », GINZBURG, KHASIDASHVILI, VOGEL, LEVY, VALLDUVI, Eds., *The Tbilisi Symposium on Language, Logic, and Computation : selected papers*, p. 85 - 100, SiLLI and CSLI Publications, Stanford, 1998.

[LEM 01a] LEMON O., BRACY A., GRUENSTEIN A., PETERS S., « Information States in a Multi-modal Dialogue System for Human-Robot Conversation », KÜHNLEIN P., REISER H., ZEEVAT H., Eds., *5th Workshop on Formal Semantics and Pragmatics of Dialogue (Bi-Dialog 2001)*, 2001, p. 57 - 67.

[LEM 01b] LEMON O., BRACY A., GRUENSTEIN A., PETERS S., « A Multi-Modal Dialogue System for Human-Robot Conversation », *Proceedings of North American Association for Computational Linguistics (NAACL 2001)*, 2001.

[LEM 02] LEMON O., « Transferable Multi-modal Dialogue Systems for Interactive Entertainment », *AAAI Spring Symposium on Artificial Intelligence in Interactive Entertainment*, Technical Report SS-02-01, Menlo Park, CA, 2002, AAAI Press, p. 57 - 61.

[LEV 77] LEVIN J. A., MOORE J. A., « Dialogue games : Metacommunication structures for natural language interaction », *Cognitive Science*, vol. 1, n° 4, 1977, p. 395-420.

[LEV 98] LEVIN L., THYME-GOBBÉL A., LAVIE A., RIES K., ZECHNER K., « A Discourse Coding Scheme for Conversational Spanish », *Proceedings of International Conference on Speech and Language Processing (ICSLP)*, 1998.

[LIT 00] LITMAN D., KEARNS M., SINGH S., WALKER M., « Automatic Optimization of Dialogue Management », *Proceedings of COLING 2000*, 2000.

[LUP 98] LUPERFOY S., LOEHR D., DUFF D., MILLER K., REEDER F., HARPER L., « An Architecture for Dialogue Management, Context Tracking, and Pragmatic Adaptation in Spoken Dialogue Systems », *COLING-ACL*, 1998, p. 794 - 801.

[MAN 88] MANN W. C., THOMPSON S. A., « Rhetorical structure theory : Towards a functional account of text organization », *Text*, vol. 8, 1988, p. 243–281.

[MAR 99] MARTIN D., CHEYER A., MORAN D., « The Open Agent Architecture : a framework for building distributed software systems », *Applied Artificial Intelligence : An International Journal*, vol. 13, n° 1-2, 1999.

[MCT 98] MCTEAR M., « Modelling Spoken Dialogues with State Transition Diagrams : Experiences with the CSLU Toolkit », *Proc 5th International Conference on Spoken Language Processing*, 1998.

[MOR 97] MORAN D., CHEYER A., JULIA L., MARTIN D., PARK S., « Multimodal User Interfaces in the Open Agent Architecture », *Proc IUI 97*, 1997, p. 61 - 68.

[PIT 96] PITTMAN J., SMITH I., COHEN P., OVIATT S., YANG T.-C., « QuickSet : a Multimodal Interface for Military Simulation », *Proceedings of the Sixth Conference on Computer Generated Forces and Behavioral Representation, Orlando*, p. 217-224, 1996.

[POW 79] POWER R., « The Organization of Purposeful Dialogues », *Linguistics*, vol. 17, 1979, p. 107-152.

[RAY 00] RAYNER M., HOCKEY B. A., JAMES F., « A compact architecture for dialogue management based on scripts and meta-outputs », *Proceedings of Applied Natural Language Processing (ANLP)*, 2000.

[RIC 01] RICH C., SIDNER C., LESH N., « Collagen : Applying collaborative discourse theory to Human-Computer Interaction », *AI Magazine*, vol. 22, n° 4, 2001, p. 15-25.

[ROS 95] ROSÉ C. P., EUGENIO B. D., LEVIN L. S., ESS-DYKEMA C. V., « Discourse Processing of Dialogues with Multiple Threads », *Proceedings of the Association for Computational Linguistics*, 1995.

[ROY 00] ROY N., PINEAU J., THRUN S., « Spoken Dialog Management for Robots », *Proceedings of ACL 2000*, 2000.

[SHI 90] SHIEBER S. M., VAN NOORD G., PEREIRA F. C. N., MOORE R. C., « Semantic-Head-Driven Generation », *Computational Linguistics*, vol. 16, n° 1, 1990, p. 30-42.

[STE 99a] STENT A., « Content planning and generation in continuous-speech spoken dialog systems », *Proceedings of KI'99 workshop "May I Speak Freely ?"*, 1999.

[STE 99b] STENT A., DOWDING J., GAWRON J. M., BRATT E. O., MOORE R., « The CommandTalk Spoken Dialogue System », *Proceedings of the Thirty-Seventh Annual Meeting of the ACL*, University of Maryland, College Park, MD, 1999, Association for Computational Linguistics, p. 183-190.

[TAY 98] TAYLOR P., BLACK A., CALEY R., « The architecture of the the Festival speech synthesis system », *Third International Workshop on Speech Synthesis, Sydney, Australia*, 1998.

[XU 00] XU W., RUDNICKY A., « Task-based dialog management using an agenda », *Proceedings of ANLP/NAACL 2000 Workshop on Conversational Systems*, 2000, p. 42-47.