MULTIMODAL INTEGRATION IN MMI^2: ANAPHORA RESOLUTION & MODE SELECTION

JL Binot, L. Debille, D. Sedlock and B. Vandecapelle

BIM

H. Chappel and M.D. Wilson

SERC Rutherford Appleton Laboratory

Abstract

The approach taken in the MMI^2 ("Multi Modal Interaction with Man Machine Interfaces for knowledge based systems") demonstrator to integrate multiple interaction modes between the user and the system for a computer network design task is described. Interaction is possible through English, French or Spanish natural languages, command language, by direct manipulation on a graphical display, through design gestures with a mouse or non-verbal audio.

The system has been designed in accordance with five guiding principles. The first two of these impose a common meaning representation language and a common dialogue context for all modes. These principles support reference resolution both within and between modes (including natural languages). Examples of the classes of reference which can be resolved are presented.

The third and fourth principles relate to the evaluation of user's input in the application domain and in the interface domain allowing side effects between these domains. These structure the multimodal integration of system output by allowing graphical changes on the screen to correspond to changes in the application KBS resulting from the formal evaluation of the logical meaning representation of the input. Examples of the resulting output using multiple modes are given.

The fifth principle permits the use of informal evaluation mechanisms which allows extra power to process pragmatic and plan based phenomena, although there are no formal constraints to control their implementation. It is shown how most cases of mode integration do not require this additional power, although some examples which do are given.

INTRODUCTION

The MMI^2 system (Binot et al, 1990) demonstrates an approach to multimodal co-operative dialogue between user and computer system. The user can input to the system through English, French or Spanish natural languages, command language, by direct manipulation on a graphical display or through design gestures with a mouse. The system can output through the natural languages, graphical display, or non-verbal audio. The interaction of these modes with the underlying demonstration application KBS for designing computer networks is governed by four principles:

The work reported in this paper was partly funded by the CEC through Esprit project 2474, MMI^2. The partners in the project are BIM (Belgium), Intelligent Software Solutions S.A. (Spain), ADR Centre de Recherche en Informatique applique aux Sciences Sociales (France), Ecole des Mines de Saint-Etienne (France), Insitut National de Recherche en Informatique et en Automatique (France), SERC Rutherford Appleton Laboratory (UK), University of Leeds (UK).

1) **there is a meaning representation formalism, common to all modes, which is used as a vehicle for internal communication of the semantic content of interactions inside the interface and also used as a support for semantic and pragmatic reasoning.**

2) **mode integration should mainly be achieved by an integrated management of a single generalised discourse context.**

3) **there are different model theories for the evaluation of symbols in the meaning representation formalism for the application domain and for the interface domain.**

4) **The effect of formally evaluating communication actions against a domain can cause side effects in the other domain.**

To produce co-operative dialogue it is still necessary to include a fifth principle:

5) **Informal processing of dialogue may be performed for user utterances (by the informal domain expert) and system output (by the communication planning expert).**

The architecture resulting from the application of the five principles is shown in figure 1 and an examples of a typical screen when using the demonstrator for computer network design is shown. in figure 2. In this, user interaction takes place through one of the presentation layer modes. These will produce, and take as input packets of information in the Common Meaning Representation (CMR) which describe the content of a communication action (by the user or the system) in a logical form, along with the mode used, the time the packet was created, and the force of the utterance (imperative, declarative or interrogative). All operations in the dialogue management layer take place using this representation language. The CMR is first passed to the dialogue controller which performs some presupposition checking and resolves references in conjunction with the context expert. It will be passed to the user model which will derive information about the user's preferences, knowledge and misconceptions from it. It will then be passed to the informal domain expert to determine if their are any pragmatic problems with the communication action. After this it will be formally evaluated against the formal domain expert or the interface expert to perform the command or find the answer to a question. The formal domain expert will interact with the underlying knowledge base which knows about network design. The answer to the question or response to the command resulting from formal evaluation will then be passed to the communication planner which will produce a reply to the user. Similarly, if the informal domain expert identifies any pragmatic problems these will also be passed to the communication planner to form the system's next move in the dialogue with the user. The system output will be created in the form of a CMR packet which will be passed to the dialogue controller and then out to the appropriate mode (named in the CMR packet) for presentation to the user.

One objective of the demonstrator is to test and illustrate an approach to co-operative dialogue. A second is to investigate multimodal integration. As with Bolt's famous "PUT-THAT-THERE" system (Bolt, 1984), much of the project has been absorbed by issues associated with natural language and co-operative dialogue at the expense of mode integration per se, however, this paper will attempt not to describe the mechanisms for co-operative dialogue beyond those involved in multimodal integration. Giv-

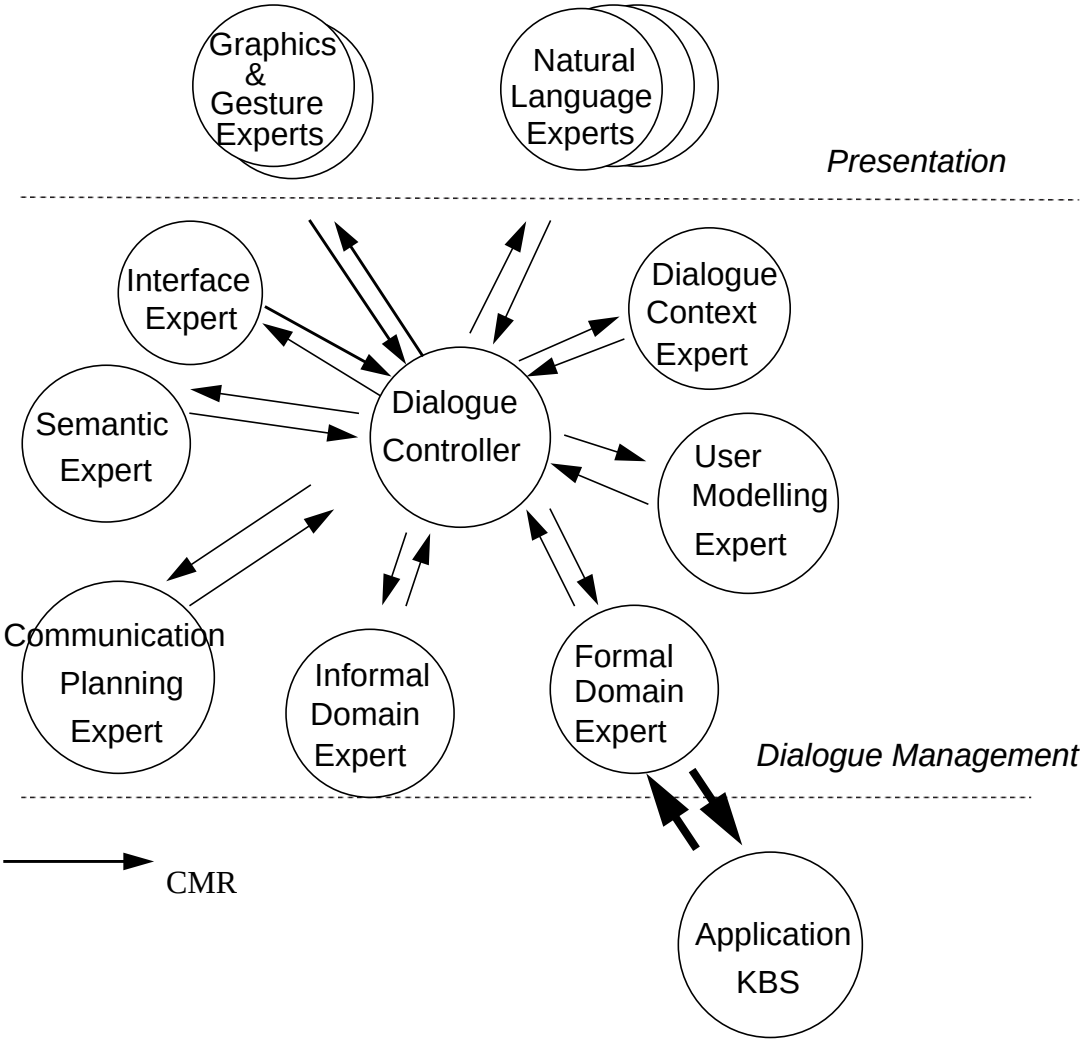

Figure 1: Architecture for the first MMI^2 demonstrator

en the design principles behind MMI^2 and the architecture resulting from them we will describe the two major classes of multimodal integration: the integration of user input through anaphora resolution, and the integration of modes for system output.

ANAPHORA AND DEIXIS RESOLUTION.

Dealing with anaphora in multimodal contexts is a powerful means of integrating modes. It allows users to select for each part of their commands or requests the appropriate mode without losing the possibility to reference across modes. The mechanism to support anaphora resolution in MMI^2 relies mainly on the first two design principles listed above.

An example situation is where a user wants to add a disk to each of a number of machines on the network. Without intermodal anaphora, the following options would be open to him:

1) a series of NL commands (or their equivalent in CL), repeating

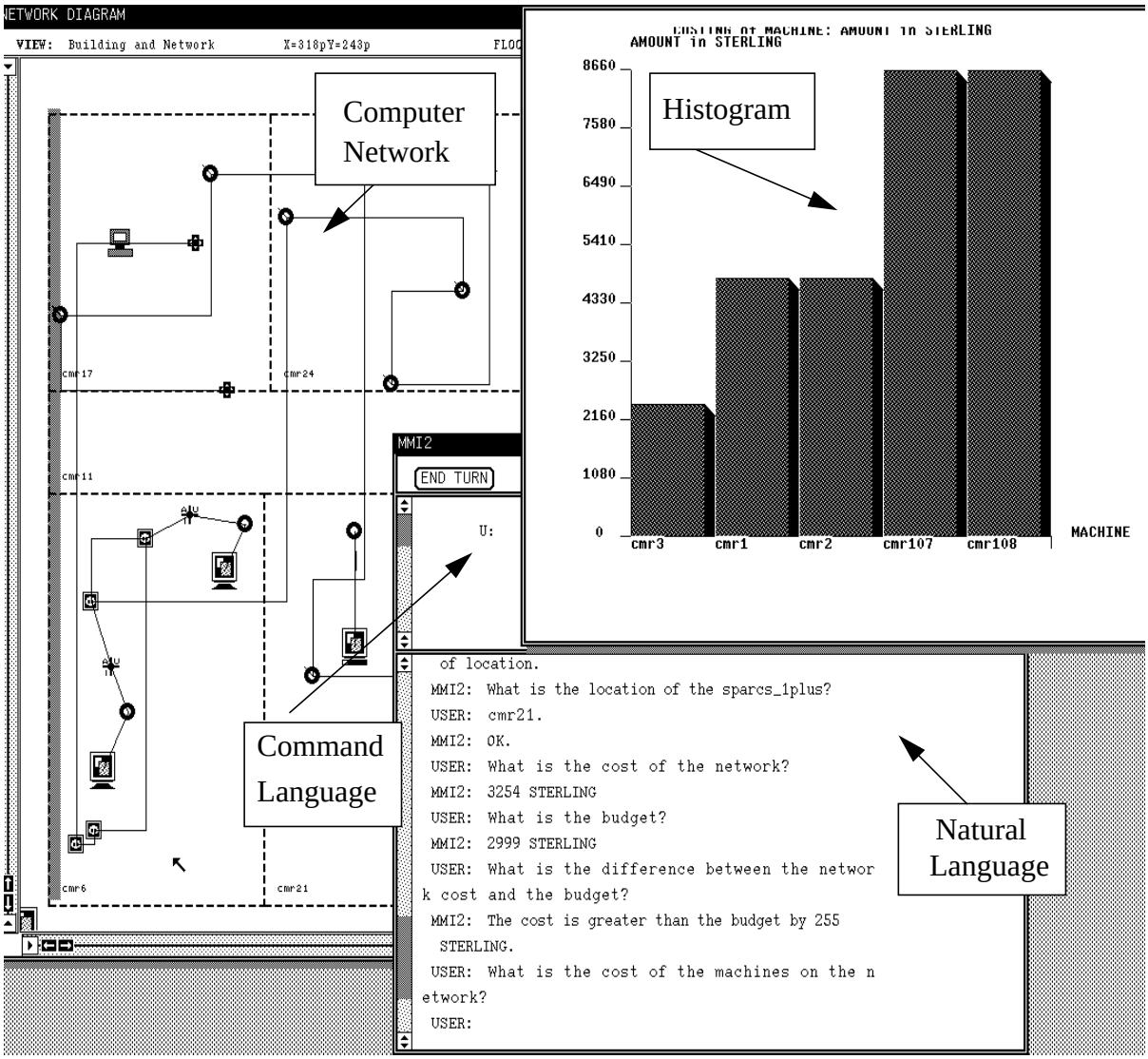

Figure 2: A typical screen display of MMI^{2}in use.

User: Add a disk to <machine-name>

for however many machines he wanted to add a disk to - notice that in casemachine names cannot be read off the graphical display, he will have to look them up first.

2) a series of graphical actions explicitly adding a disk icon to each of the machines and in each case establishing the appropriate connection.

With the availability of intermodal anaphora, the user can perform such actions much more efficiently by selecting the machines in the graphical display (either through multiple clicks, through grouping by mouse gesture, or by a zooming mechanism) and then issuing a command in NL:

User: Add a disk to these machines.

Anaphora resolution across modes adds an additional level of difficulty over and above the usual (i.e. intra-NL) complexity of identifying valid antecedents from the context and establishing anaphoric relations. The additional difficulty stems from the

fact that the system is forced to deal with non-textual modes; modes in which notions such as ‘establishing a context’ and ‘mentioning an object’ take on quite different meanings.

In NL, for a user to mention an object and thus make it part of the context and available for later reference, is to either name the object (MachineX567) or to describe it (the machine to the left of the server). This is done by using the words as part of typed user input. Except in rare metalinguistic statements and requests (“What does ‘workstation’ mean”) mentioning an object unambiguously indicates reference to the domain object corresponding to the meaning of the linguistic expression. Notice that when one does want to refer to the word itself rather than to its meaning, one usually makes this explicit by quoting the word and by omitting the article.

Graphical object selections are - for once - less predictable than their NL counterparts as to their intended interpretation. When a user clicks on the graphical display - clicking is the most common selection device -, his intention can be any of the below:

1 reference to a particular location on the screen (xy coordinates of pixel clicked on)

2 reference to the graphical icon displayed where the click occurred

3 reference to the domain object represented by the icon

In fact, finding out what the user’s selection refers to can often only be determined through making sense of the request or command that follows. The following NL input disambiguates a preceding click as a location reference (1), a graphical object reference (2) and a domain object reference (3):

1 Put one end of the cable here.

2 Add a PC to its left.

3 Which server is this machine connected to?

Mouse gesture, the other MMI2 non-textual input mode, generates the same kind of ambiguity with respect to intended referents: when a circle is drawn on the graphical display, the system cannot safely assume one single interpretation.

Another significant difference between textual and graphical modes is the role focus plays in the process of reference resolution. Textual focus (i.e. the topic or theme) is strictly tied to the actual time of interaction. Once the user changes the topic, the previous one goes out of focus until explicitly mentioned again. On the other hand, the graphical display - or any subdisplay the user has at some point zoomed into - remains present throughout the interaction. It would seem impossible to force the user to ignore this display as a possible visual context for NL reference at any time. This persistence aspect of the visual focus as opposed to the fleeting textual focus must be considered in anaphora resolution.

Both considerations above have contributed to a design decision for the structure of MMI2’s Context Expert, the module that is responsible for managing dialogue history on the one hand, and for delivering potential antecedents to the anaphora resolver on the other hand.

Most systems handling dialogue features and anaphora resolution to the degree that MMI^2 does, implement a specialised structure to represent dialogue history, discourse structure, focus shifting, antecedent stack etc. Every user and system input

and output is analysed with respect to its impact on the dialogue and all relevant information is retrieved and stored for later use. In these systems, most of the work is thus done at storage time, and retrieval of antecedents is made more or less trivial.

In MMI^2, we have adopted a different approach: every interaction is simply stored in the CE as it comes from or is sent to the different modes. The complexity is shifted from the extraction procedures that manipulate the internal representation and construct the discourse structure to a body of rules that retrieve context related information on demand by the anaphora resolver. This approach has the following advantages:

- It saves time as well as space. Since the retrieval rules are only activated when an anaphor is detected and needs resolving, all the time spent analysing input and output that never gets invoked in contextual interpretation is saved. As for space, it is highly likely that storing a discourse representation that makes explicit a lot of information that is implicit in user and system interactions will take up a lot more space than simply storing the user and system interactions themselves.

- It is more practical and more flexible from the point of view of software design. Extracting a particular discourse representation for the purpose of contextual interpretation is almost necessarily biased by the kind of dialogue phenomena one intends to handle. It is not obvious that a structure that was designed for easy retrieval of antecedents will serve as efficiently in ellipsis resolution. Even if one stays within one functionality, say antecedent retrieval, there is always a risk that extending the scope of anaphora resolution to new types of antecedents will require more information to be extracted from the dialogue interaction than was originally foreseen. Transferring the complexity to a dedicated antecedent retrieval rule component - next to an ellipsis resolution component for example - and storing the dialogue history as a simple transcription of the dialogue interactions themselves avoids the risk of being forced to change the basic context representation, which may have an impact on all context based interpretation modules, to accommodate further development in one of them.

In the context of multimodal systems, resolution often has to deal with a type of anaphoric relation that increases the importance of the previous advantages even more. It is a fairly frequent phenomenon that anaphoric elements are used in NL, yet no suitable antecedent is explicitly present in the context. The resolver then has to try and construct an antecedent itself, given what is explicitly present in the context, and given a number of rules on how to build valid antecedent-anaphor relations. This process of creative antecedent retrieval is called ‘accommodation’.

Accommodation occurs in purely textual dialogue as well. Plural antecedents are very often built from antecedents that occurred as singulars in the context. For example:

User: Connect 3 PCs to the server.

User: Move the machines to the upper floor.

In some cases, it is not the antecedent itself that needs construction, it is the relationship between the anaphor and the antecedent that requires more creativity than is common. In the example above, the user refers to a specific machine as ‘the server’. This description is intended to uniquely distinguish this machine from any of the other machines currently in context, thus illustrating the validity of the ‘role of’ relationship

between anaphor and antecedent. Another example of a conceptual relation that implicitly validates an anaphoric relation is ‘part of’ as in:

User: <clicks on a machine icon in the graphical display>

User: What does the disk cost?

The multiple interpretation possibilities of the graphical click also give rise to accommodation. In this example, it is the type of the anaphor and reasoning over the type of antecedent it requires that allows the system to make a decision as to the meaning of the user click.

Finally, the persistence of visual focus leads to a more or less permanent supply/ stock of antecedents which can be invoked at any time if called for.

With such a multitude of possibilities for antecedent construction, creative anaphoric relation definitions, ambiguous antecedents and different principles governing different contexts, it seems quite unsuitable to try and extract all relevant contextual information from every user and system interaction. It would lead to a large body of information which would be costly on processing time and storage space and potentially very difficult to manage.

The decision to store interactions directly in the CE rather than constructing a special discourse representation is partly made possible by the design principle of MMI^2 that the same representation language (CMR) is used by all modes, i.e. all 3 natural languages, as well as command language and the graphical modes. This uniformity of input and output representation has the following 2 considerable advantages:

1. The resolver deals with anaphora in all 3 natural languages uniformly. Since the resolver operates on CMR representations, which are input language independent, it need not know which language the CMR it is resolving originated from. The minor differences in the resolution systems of the languages dealt with by MMI2 were largely outweighed by the advantages and the interest of having a central anaphora resolver.

2. The rules in the CE that retrieve suitable candidate antecedents need not distinguish between looking for antecedents coming from graphical or textual modes. Since again, the CE stores the CMR representations themselves as making up the dialogue history, and CMR representations are input and output mode independent, the antecedent rules can uniformly apply to the complete history of interaction. Mode origin, however, is kept as a trace as part of each CMR representation, allowing the CE rules to access this information should the need arise.

INTEGRATION OF MODES IN SYSTEM OUTPUT.

Whereas the integration of user input from different modes was mainly brought about through the application of the first two design principles behind MMI2, the integration of modes for system output is mainly due to the third and fourth principles.

For example, the question:

a) User: What is the cost of the network?

is a communication action which would result in a CMR representation which would be evaluated against the application domain and produce a reply to the question which would also be a communication action:

b) System: 13000 Sterling.

In contrast, the question:

c) User: How many graphs?

would be evaluated against the interface domain which represents the visualisation of graphs and other interface aspects of the system to produce a reply:

d) System: 3

A second example which would appear to be evaluated against the application domain is in MMI2 evaluated against the interface domain also:

e) What machines are on the left of the bridge?

In the design KBS used for MMI2 the exact spatial locations of objects were not represented. Therefore it is necessary to interrogate the graphical display to provide the semantics for the spatial relation ‘on the left of’ and to determine which machines are on the left of a bridge (a box which connects two computer networks) whose exact identity will have been determined through anaphora resolution.

The first of these questions applied to the application domain, the second to the interface domain and the third to both; all three produced communication actions as answers which were returned to the user in natural language. A command such as:

f) Display a bar-graph of the cost of the computers.

requires data to be acquired from the application domain about the computers that exist and their costs, but the evaluation of the ‘displaying’ of this data takes place in the interface domain, with the result that a bar graph is displayed. The communication action in reply to this command is an acknowledgement that it has succeeded:

g) System: ok

in natural language. The display of the bar graph is neither a communication action in reply to the command, nor is it a side effect of some evaluation, but it is the consequence of successfully updating the ‘displaying’ in the interface domain. Further complexity arises for a command such as:

h) User: Add a Sparc SLC to Room23.

which would be evaluated against the application domain to make it true that a Sparc SLC existed in Room23 of the building displayed on the screen, but a side effect of updating the application domain would be to update the interface domain so that the icon of a Sparc SLC appeared in Room23 in the display on the screen. In this example, the reply to the user resulting from the command arises from the success of the update to the application domain and would be f) which would be a communication action by the system. There would be no communication action by the system corresponding to the graphical update of the icon on the screen since this was a side effect of the update to the application domain, and was not explicitly asked for or commanded by the user.

If instead of g) the user had typed:

i) User: Add a Cray XMP to Room23.

the update would have failed on the application domain since the design KBS does

not know about Cray computers. The answer to the user resulting from the failed update would then be:

j) System: No way.

which is an appropriate communication action as a reply to the failure in the application domain, and no side effects occur in the interface domain. Through the use of side effects in one domain from operations in another, system replies remain consistent with the force of the user's input whilst also causing changes in other modes than that used to express the communication action of the reply.

The use of side effects of operations in one domain to produce operations in another domain may appear trivial in this example, however the side effects can be very complex. For example, if a user issues the command:

k) User: Design a network.

when the prerequisites of a network design have been completed (e.g. the building is known and displayed, the budget and machine locations are known) then the application domain will design a network by placing cables, connectors and junction boxes in the building. The answer derived from the application domain successfully updating the design operation will be that a network will exist in the application knowledge base. However, a side effect of this update is that the network will be displayed in the graphical display of the building - the network may contain several hundred objects which have to be added to the display, located on it and connected together.

Despite the power of multimodal integration offered by the use of side effects and the use of evaluation over separate domains, some examples require a further informal communication planning provided by the fifth principle. For example, the question:

l) User: Where is Machine4?

would result, after formal evaluation in the application domain, in the answer:

m) System: Room3.

which makes no use of the multimodal facilities available in the interface. In some task circumstances what the user would like to see would be the room where the machine is located highlighted on the graphical display. In order to achieve this the communication planning expert must add to the formally produced answer which is already a communication action to the user for the natural language mode, an additional communication action to the graphics mode which will highlight the room.

Even when the form of the answer is presented in a single mode, the choice of this mode can be made by the communication planner. For example, the answer to the query:

n) User: Which computers are in which rooms?

will produce a list of pairs of computers and rooms after formal evaluation in the application domain. Both natural language and graphics are capable of presenting this type of information, and the communication planning expert chooses the mode which will more effectively present it to the user. In this case the mode would be graphics and the graphics mode itself would then decide on the most effective form of graphical presentation (a choice between changing the network display diagram; a bar chart; a line graph; a pie chart; a hierarchy, or a table) and would select a table, which

would then be presented to the user. Here a single mode is used to present the information, but the choice of mode must be made by on the basis of knowledge about effective communication.

One last example appears at first consideration to introduce the informal domain expert into the choice of which mode to use for presentation:

o) User: Put the network in a building.

p) System: Describe the building specifications - Displays a graphical tool for the user to draw the building.

In o) the user has commanded that a network be put in a building. The informal domain expert stores plans for completing tasks and has identified that for this to be done the network must be designed and the building described (the preconditions which were met for example k). Having identified that the building need be described by the user it states the need for this as a system desire which the communication planner attempts to fulfil. It is the communication planning expert again which decides that the way to express this request to the user is to use both a natural language request and to provide a graphical tool in which the user should answer that request. therefore although it may appear that the informal domain expert is choosing modes as a result of planning information, it is in fact the communication planning expert again which is choosing the mode as a result of the system need which has been identified from the plans. A further mechanism for choosing modes of interaction is not required due to the careful division of roles between the experts in the dialogue manager.

CONCLUSION

The first four principles appear to be sufficient to govern multimodal interaction in most cases without having to resort to the freedom offered by the informal approach of the fifth principle. However, both for some principled occasions such as the selection of the most effective presentation mode, and some conventionalised ones (as in example p) above) informal communication planning beyond that provided by the first four principles is required. Future work should include basic cognitive science and ergonomics research to redescribe those aspects of the informal communication planning in a formal way.

The present mechanism for synchronising the presentation of different modes is crudely that CMR packets are presented when they are received by the modes. If a sequence is presented at once to the modes, they are presented simultaneously. Clearly, if the system is to synchronise text with segments of video or animation this is inadequate and a richer formalism such as the HyTime standard would have to be adopted. Otherwise the identified core set of multimodal interactions can be supported through the mechanisms described here.

REFERENCES.

Binot, J-L., Falzon, P., Perez, R., Peroche, B., Sheehy, N., Rouault, J. and Wilson, M.D. (1990). Architecture of a multimodal dialogue interface for knowledge-based systems. In Proceedings of Esprit ‘90 Conference, 412-433. Kluwer Academic Publishers: Dordrecht.

Bolt, R.A. (1984) The Human Interface: where people and computers meet. Lifetime Learning Publications: Belmont, Calif..