*Proc. Second Int. Conf. on Cooperative Information Systems* (CoopIS-94), Toronto, Canada, 255-263.

# Cooperation in Distributed Medical Care

**Jun Huang*, N. R. Jennings+ and John Fox***

* Advanced Computation Laboratory, Imperial Cancer Research Fund, 61 Lincoln's Inn Fields, London WC2A 3PX, England (jh@acl.lif.icnet.uk; jf@acl.lif.icnet.uk )

+ Department of Electronic Engineering, Queen Mary and Westfield College, University of London, Mile End Road, London E1 4NS, England (N.R.Jennings@qmw.ac.uk)

**Abstract**

*Cooperative clinical information systems (CCIS) are destined to become an invaluable tool for managing care processes in which a number of interrelated care tasks are performed by a network of cooperating agents. After introducing this important new domain, the key mechanisms and structures needed to support CCIS are detailed - a set of communication primitives are developed, the concept of agent commitment is described, the structures needed for reasoning about allocating tasks to appropriate agents are presented, and the mechanisms for managing change in the cooperating community are outlined.*

## 1. Introduction

The care of a particular patient is often shared by several different clinicians who are located in a variety of physically distributed health care settings. For instance, a general practitioner (GP) may suspect that his patient has breast cancer. However, as he neither has the knowledge nor the resources to confirm this hypothesis, he must refer the patient to a hospital specialist who can make a firm diagnosis. Having confirmed the presence of breast cancer, the specialist must devise a treatment plan - this plan typically involves his hospital nurses, the patient's GP, and a home care organisation, jointly executing a series of interrelated tasks. This distributed care process requires a number of semi-autonomous problem solving agents - each with their own data, knowledge, resources and responsibilities - to cooperate with one another to attain a common goal (i.e. keep the patient healthy). In such scenarios there is a clear need to support coordinated action and to manage the flow of information between the various participants - "the forces of patient need, opposed by the pressure for cost containment, are likely to produce strong incentives to develop better coordination and integration across the boundary between hospitals and the community they serve" [Pritchard 1992].

The quality of care provided to a patient is very much affected by the degree of cooperation and coordination among the care participants. In particular, three key types of interaction have been identified. Firstly, there is the management of the flow of appropriate medical information to relevant parties. Inappropriate communication may result in delayed or even incorrect care acts. For example, if a doctor requests a pathology lab to administer a blood test for a crucial evaluation of cancer progression, the lab nurse who is actually responsible for the test may deliver the result to the patient, who, in turn, may not pass the information on to the doctor who requested it. The second type of social interaction involves the allocation of care tasks to appropriate participants. If a task is allocated to the wrong person then it is highly unlikely to be executed in a timely fashion. In serious cases, such delays may even result in death - early diagnosis and treatment are the keys to patient care! The final type of social interaction involves the timely dissemination of information about changes in the status of the cooperating group. Such interchange is needed because the participants are situated in dynamic and uncertain environments in which their commitments to tasks in the care program may alter over time. In such cases, the participants need general policies (*conventions* [Jennings 1993]) for monitoring their commitments and for interacting with others.

In the health care process, as it stands at present, these social interactions are controlled by a number of *implicit* and *ad hoc* rules of thumb. Unfortunately, in many real world cases this implicit representation has many serious consequences (as illustrated by the above examples). Therefore this work sought to identify *explicit* mechanisms for practical cooperation so as to provide the appropriate computer support for distributed care management.

Currently, clinical information systems are restricted to the storage and retrieval of individual patient records, using various commercial, largely relational, database management systems. However as more clinicians are exchanging information over the computer network and jointly taking part in inter-related care tasks, demand for integrated information management and decision support is increasing [Renaud-Salis 1992]. To meet this need, the next generation of computerised support - *cooperative clinical information systems* (CCIS) - must provide appropriate facilities for information management, communication, and cooperation for the three types of social interaction which have been identified as central to this application. As an illustration of this new technology consider the example of the shared care of terminally ill cancer patients. An oncologist from a specialist cancer hospital examines current patient data and prescribes the appropriate drugs. He may then allocate various tasks to the patient's GP, such as taking charge of regular administration of the drugs and scheduling follow-up blood tests. The GP may in turn request the most suitable home care organisation, say the nearest to the patient's home, to perform some further tasks (e.g. acquire and administer the prescribed drugs and take blood tests). The types of information that need to flow between the care participants in this scenario include: patient data, partial plans, requests, and responses. These interchanges require the CCIS to have appropriate message structures - including a set of well defined communication primitives and a communication protocol. The CCIS will enhance coordination and cooperation by providing explicit mechanisms for allocating tasks to the most appropriate agents (eg deciding upon the most suitable home care organisation), establishing agent commitment to tasks (eg ensuring the GP agrees to administer the drugs and schedule the follow up blood tests), and monitoring commitments so that the group of agents behaves in a coherent and efficient manner (eg ensuring the GP actually does what he has agreed to).

This paper is structured along the following lines: firstly, the vision of the distributed care process as a hierarchy of care tasks undertaken by a network of cooperating agents is presented, and the various task and agent types are described (section two). Secondly, the communication primitives and protocol necessary to underpin the cooperative care process are detailed (section three). The processes of allocating tasks to agents and formulating agent commitments are then discussed, and a convention for monitoring these commitments in the face of changing circumstances is described (section four). Finally, some conclusions are drawn and the prototyping work to develop a CCIS which is being undertaken at the Imperial Cancer Research Fund is outlined (section five).

## 2. Distributed care as hierarchical tasks and cooperating agents

A care process consists of a hierarchy of dynamically planned tasks (as in [Corkill 1979]) - see figure 1 for an example of the care process for treating breast cancer. This plan involves three high-level tasks: first-cycle EMV chemotherapy followed by either Gm-CSF chemotherapy or second-cycle EMV chemotherapy (depending on the patient's temperature). Each task in the plan has an associated state [Gordon 1993] - `established` (if the task is to be applied to a given patient problem); `cancelled` (if the task is no longer to be

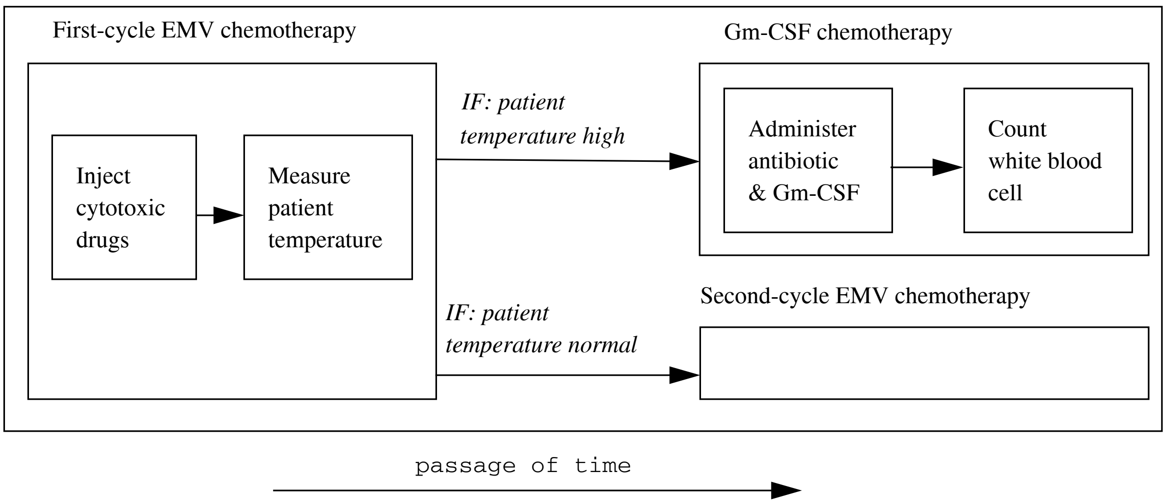


**Figure 1. Part of a Care Process for Treating Breast Cancer**

performed); `started` (if the task is being performed); or `performed` (if the execution of the task has been completed).

A given patient may be involved in multiple care processes - for instance, the concurrent management of breast cancer and maternity - and these processes may interact with one another in terms of their effect on the patient. Harmful interactions can be avoided by employing global patient records, such as patient-held smart cards, though the practical delivery of these records is limited by many social and human factors (such as confidentiality and privacy). Here we restrict ourselves to the management of a single distributed care process; however dynamic changes to the states of care tasks are allowed if previously unknown information becomes available from another care process. For example, the state of the task second-cycle chemotherapy may be changed from `established` to `cancelled` on receiving the data 'patient pregnant' from a maternity care process.

A task hierarchy for a particular care process is dynamically generated by the network of cooperating agents once a patient initiates a top-level task such as 'please do something about my breast cancer'. For a particular task to be executable, it may have to be decomposed into a number of subtasks. All tasks and subtasks are associated with two agents: one who manages the activity and one who actually executes it. When a manager delegates a task, the contractor becomes committed to its execution, i.e. the contractor endeavours to perform the task unless its circumstances change substantially. The contractor may decompose its activity into a further series of subtasks which it can subcontract to others in the community. Agents may be both a manager and a contractor for the same task if they allocate it to themselves. In order to facilitate the task allocation process each task is designated as being of a specific type (see section 4.1 for more details). For instance, first-cycle EMV chemotherapy is of type 'breast cancer therapy'. Tasks also have an associated local priority which varies from urgent to low.

In the context of this paper, the term 'agent' refers to an integrated entity involving an intelligent computer system and its user. Such agents interact with one another across a distributed care network using a variety of media. This interaction may be between the humans (using post, telephone or fax), or it may be between the computer part of the agents (via electronic mail). The intra-agent interactions between a computer and its user are not modelled here, but they may be informally described in the following manner. The computer is capable of performing some desired inference functionalities, e.g. decision making, task management, communication and cooperation. However, the computer must inform its user of the results of these inferences - i.e. decision options, task state transitions, proposed task allocation patterns, and so on - because it is the user who ultimately has to endorse and authorise the inference results before they can be sent to external agents. The user also has the unique capability to undertake physical actions (such as administering drugs and taking blood pressure). On the other hand, the user has the obligation to record the results of actions in the computer system so that it can use them during its inferences. Finally, the user is responsible for storing and updating knowledge and data, including meta-level knowledge about agents, knowledge of tasks and domains, and patient data.

As the task hierarchy is developed, an associated hierarchy of agents forms an abstract dynamic organisation in which data flows up and control flows down. To take an example, consider the task hierarchy of Figure 1. Suppose a senior oncologist in a cancer hospital accepts the task of 'treat breast cancer'. As the contractor for this task, he decomposes it into a first subtask of 'first-cycle EMV chemotherapy' and a second subtask of either 'second-cycle EMV chemotherapy' or 'Gm-CSF chemotherapy' depending upon the patient's reaction to the first-cycle EMV chemotherapy. He then allocates the first subtask to a junior oncologist, in the same hospital, but takes no further action about the second subtask. Therefore, the senior oncologist becomes the manager of 'first-cycle EMV chemotherapy' and the junior oncologist becomes its contractor. The junior oncologist then further decomposes its activities into two successive subtasks of 'inject cytotoxic drugs' and 'measure patient temperature'. He allocates the two subtasks to an external home care nurse who agrees to execute them at the patient's home. In this agent organisation, data flows up the hierarchy in the following pattern: the home care nurse has to inform the junior oncologist about the outcome of 'inject cytotoxic drugs' and 'measure patient temperature' (e.g. 'drugs injected' and 'patient temperature high'); and the junior oncologist has to inform the senior oncologist of the results of 'first-cycle EMV chemotherapy' (e.g. 'first-cycle EMV chemotherapy completed' with 'patient temperature high'). Control flows down the hierarchy in terms of commitments and expectations: the senior oncologist expects the junior oncologist to be committed to 'first-cycle EMV chemotherapy', and the junior oncologist expects the home care nurse to be committed to the 'inject cytotoxic drugs' and 'measure patient temperature' tasks. It can be seen that such organisations do not correspond to a single physical body, such as a hospital or clinic, rather they are a collection of agents related to one another through a web of commitments and expectations [Gerson 1976].

## 3. Communication in distributed care

After careful analysis of the three key social interactions involved in cooperative care organisations, a set of communication primitives were defined. These primitives are loosely based on speech act theory [Searle 1969] and are believed to be the minimum set of commands which can sustain the desired social interactions. Each primitive has a type and a content, as well as a certain effect on the receiver of the message. Having a well-defined set of primitives is important in this application because it means that the ambiguity in message interchange is substantially reduced - each primitive has a clear meaning and must be responded to in a particular manner. There has been similar work on communication primitives elsewhere (e.g. in the Contract Net Protocol [Smith 1980] and more recently in the 'cooperative primitives' of [Lux 1993]), but these primitives do not support all the interactions needed by this application.

### 3.1 Communication primitives

The primary purpose of a communication primitive is to identify the content of a message and its effect on the receiver. In distributed patient care, the messages are usually directed towards specific agents (individual clinicians or a representative body such as a chest clinic) whose responsibilities are known to the sender (the mechanism by which this is achieved is described in section 4.1). This directed message passing, as opposed to the alternative of unfocused broadcasts, significantly reduces the network's communication traffic and the amount of resource wasted by agents in needlessly processing irrelevant messages. Given this basic mode of operation, the following primitives are required to support cooperative care management: *request, accept, reject, alter, propose, inform, query, cancel* and *acknowledge* (Table 1) - note that the items in square brackets are optional. The primitives *request*, *accept*, *reject* and *alter* are used during the formulation of agent commitments. Using the example from the previous section, the junior oncologist may first allocate the task 'inject cytotoxic drugs' to the home care nurse by *requesting* her to perform it on the following day. The nurse may *accept* the task exactly as specified by the oncologist, or she may *reject* it because she is too busy during the next few days (insufficient resources to honour the commitment). Alternatively, the nurse may indicate that the task cannot be completed by the following day, but it could be finished two days later (*alter*). The junior oncologist may accept the proposed change or find the new date clinically unacceptable, and *request* some other nurse to inject the drugs as originally planned.

A *propose* act may be the result of a *query* - for instance, proposing a treatment protocol after being asked how to treat stage-2 breast cancer. *Inform* usually follows an *accepted* request to perform a certain task. For example, having accepted to measure patient temperature for the junior oncologist, the home care nurse will inform him of the actual body temperature of the patient on the specified date. The fact that *cancel* is included as a primitive type is due to the dynamic nature of the distributed patient care process. In certain circumstances, agents may modify their commitments - this issue is discussed more extensively in section 4.3. Finally, all messages must be *acknowledged*.

### 3.2 Communication protocol

Having introduced the primitives, the high-level communication protocol that defines the syntax of inter-agent messages can now be given. Note that the '*' superscript denotes repeated entries and that PRIMITIVE_CONTENT is as defined in the previous subsection.

```
<message> :==  <sender> <receiver> <date> <time>
               <patient> <transaction_primitive>*
<sender> :== <sender_name> <contact_address>
<sender_name> :== <first_name> <surname>
<first_name> :== NAME
<surname> :== NAME
<contact_address> :== <email_address> | <fax_number> |
               <postal_address>| <telephone_number>
<email_address> :== EMAIL_ADDRESS
<postal_address> :== POSTAL_ADDRESS
<telephone_number> :== NUMBER
<fax_number> :== NUMBER
<receiver> :== <receiver_name> <contact_address>
<receiver_name> :== <first_name> <surname>
<date> :== <day> <month> <year>
<day> :== NUMBER
<month> :== NUMBER
<year> :== NUMBER
<time> :== <hour> <minute>
<minute> :== NUMBER
<hour> :== NUMBER
<patient> :== <patient_name> <date_of_birth>
<patient_name> :== <first_name> <surname>
<date_of_birth> :== <year> <month> <day>
```

| Type | Content | Effect on receiver |
|---|---|---|
| *request* | task;<br>[provisional schedule];<br>priority: urgent or not;<br>response_by date | ReceiveAgent evaluates whether to accept the request, and informs SendAgent of decision. If recipient decides to accept the request, he becomes committed to the task. |
| *accept* | task;<br>[accepted schedule] | ReceiveAgent knows SendAgent is committed to the request and that SendAgent will inform him of the outcome of executing the task. SendAgent becomes the contractor for the task, and ReceiveAgent the manager. |
| *reject* | task;<br>[provisional schedule] | ReceiveAgent has to request someone else to perform the task on the provisional schedule |
| *alter* | task;<br>provisional schedule;<br>acceptable schedule | ReceiveAgent to evaluate the acceptable schedule and decide whether to replace the provisional schedule with the acceptable schedule. If so, he sends SendAgent a new request. Otherwise, ReceiveAgent has to send the original request to someone else |
| *propose* | task;<br>[proposed schedule] | ReceiveAgent may or may not adopt the proposal |
| *inform* | any information:<br>data, domain knowledge<br>or partial plans | ReceiveAgent may use the information for local problem solving |
| *query* | a question: what, how, whether, and so on | ReceiveAgent must answer the query, possibly involving extensive local problem solving (e.g. diagnosis and investigation). A reply may be of type 'propose'. It may also be 'inform', possibly giving the answer 'unknown' to the query |
| *cancel* | any message of the above types: from request to query | ReceiveAgent should ignore the earlier message |
| *acknowledge* | any message of the above types: from request to cancel.<br>All messages need to be acknowledged except acknowledgement messages themselves | ReceiveAgent is aware of the successful transmission of the message |

**Table 1: Communication primitives**

```
<transaction_primitive>:== <primitive_type>
                           <primitive_content>
<primitive_type> :== REQUEST | ACCEPT | REJECT |
              ALTER | PROPOSE | INFORM | QUERY |
              CANCEL | ACKNOWLEDGE
<primitive_content> :== PRIMITIVE_CONTENT
```

Using the above specification, the following is an illustration of a syntactically correct message from a Prolog environment:

```
message(from('Tony Burg', 'tb@acl.icrf.ac.uk'),
        to('Jean-Louis Penn', 'jlp@fb.y-net.fr'),
        date('1993 06 01'), time('12 00'),
        patient('Mary Taylor', '1925 10 30'),
        request(task('treat breast cancer'),
                priority('urgent'),
                response_by('10 06 1993')),
        inform(date('1993 05 30'),
               finding('tumour size', '10mm x 5mm')),
        inform(date('1993 05 30'),
               finding('tumour location', 'left breast')))
```

which could be first acknowledged and then responded to in the following manner:

```
message(from('Jean-Louis Penn', 'jlp@fb.y-net.fr'),
        to('Tony Burg', 'tb@acl.icrf.ac.uk'),
        date('1993 06 02'), time('10 00'),
        patient('Mary Taylor', '1925 10 30'),
        accept(task('treat breast cancer')) )
```

The message from Tony Burg to Jean-Louis Penn includes an urgent request to treat Mary Taylor's breast cancer, as well as some patient data that is thought to be relevant (i.e. the most recently-measured tumour size and location). The response message simply says that Jean-Louis Penn accepts to undertake the requested task.

## 4. Cooperation in distributed care

When specifying the mechanisms which support cooperative patient care, there are two important system properties which must be born in mind. Firstly, a CCIS should be able to tolerate uncertainty. Uncertainty typically arises from a lack of complete information relevant to decision making - as patient data is rarely globally available to all the relevant professionals. Without proper mechanisms for uncertainty management incorrect decisions may be made, sometimes causing irreversible damage to the patient. Secondly, a CCIS should be capable of delivering a quality solution, i.e. the coherent development and implementation of a systematic care programme. The symbolic decision procedure proposed in [Fox 1992] and later formally described in [Huang 1993a] has been used to support medical decision making with uncertain data and knowledge. Therefore we concentrate on the latter aspect of ensuring that the cooperating network delivers a solution which is of a high quality. Quality is attained by ensuring that there is a timely flow of medical information, that tasks are allocated to the most appropriate agents, and that the cooperating group responds efficiently to unplanned events or important changes in the status of its participants.

In this work, the underlying mechanisms on which social interactions are based are *commitments* (pledges which agents make to undertake a particular task) and *conventions* (means of monitoring commitments in changing circumstances) [Jennings 1993]. The former means that if an agent agrees to undertake a task then it will endeavour to actually execute it at the appropriate time - this means the agent must be capable of performing the task and also that it must have the necessary resources. Conventions are needed because commitments are not irrevocable. As agents' circumstances may change between the making and the execution of their commitments, agreed actions may turn out to be undesirable or even impossible to perform - meaning commitments may be reneged upon. For example, the home care nurse who agrees to administer cytotoxic drugs to a patient in two days time might be unable to honour this commitment because she cannot obtain the prescribed drugs at this point in time (an unexpected lack of resources). Conventions define the conditions under which commitments can be dropped (e.g. when a task is no longer necessary) and specify how to behave with respect to other participants in the cooperating group (for instance, if the contractor drops his commitment to a task, he must inform the manager of the task so that the manager can re-allocate the task without too long a delay).

Given that robust cooperation is founded on commitments and conventions, a number of key issues need to be addressed for the distributed patient care application: (i) what are the key mechanisms by which agents decide to whom they should allocate particular tasks? (section 4.1); (ii) what is involved in establishing a commitment between a manager and a contractor? (section 4.2); and (iii) what type of convention is appropriate for monitoring commitments in the given care organisational structure? (section 4.3).

### 4.1 Allocating tasks

The key structure used by a manager when making decisions about task allocation is that of *accountability.* Accountability represents a static relationship between two agents which is generally established before they jointly care for patients. The subset of the network's members which appear in an individual agent's accountability relations are termed its acquaintances. Accountability defines both for what and to whom an agent is responsible and it can be expressed in the following manner: accountable(Agent1, Agent2, TaskType). This means that Agent1 is accountable to Agent2 for performing tasks of type TaskType. For example, a general practice nurse may be accountable to her GP for immunisation tasks, while a hospital nurse may be accountable to one or more doctors for monitoring patient data such as body temperature and blood pressure.

A task manager uses these accountability relations to pick the most appropriate contractor for a given activity. If there is only one potential contractor, then the manager has no option but to ask the appropriately qualified acquaintance. However if more than one acquaintance has the potential to undertake a given task, then accountability needs to be augmented by a preference relation. For example, patients tend to prefer to consult a specific doctor for a specific type of task; this knowledge can be expressed in the CCIS by the following generic rule:

```
IF   Task is necessary, AND
     Task is of type TaskType, AND
     Acquaintance is accountable to Agent to perform
         tasks of TaskType, AND
     Agent prefers to interact with Acquaintance
         concerning TaskType
THEN request (Agent, Acquaintance, perform (Task))
```

If the chosen acquaintance subsequently rejects the request (eg because of insufficient resources) the manager will request the next most preferred acquaintance to perform the task. Task allocation is not complete until the contractor becomes committed to the task (as described below).

### 4.2 Establishing commitments

Accountability alone does not guarantee commitment; to commit to a specified task, an agent must also have the necessary resources which are required to perform that task. These necessary resources can be related to both time and material. For example, a hospital specialist may be accountable to patients for breast cancer surgery, but he will not become committed to surgery on a specific patient until he has the time (temporal resource) and the right equipment (material resource) to perform the operation. Although agents know what resources are available to themselves, generally they do not have information about the resources of their acquaintances, therefore a task may have to be iteratively delegated to a number of acquaintances until a specific agent becomes committed.

When an agent accepts a request he becomes committed to performing it (i.e. he commits to undertake the role of contractor to the originator for that task). The first action of a contractor is to inform the manager that the task has been accepted. These two facets of agent behaviour can be expressed in the CCIS by the following generic rule:

```
IF Acquaintance is requested by Agent to perform
      Task, AND
   Acquaintance is accountable to Agent to perform
      tasks of TaskType, AND
   Task is of type TaskType, AND
   Task requires Resources, AND
   Resources are available to Acquaintance
THEN Acquaintance becomes committed to Task,
      inform (Acquaintance, Agent,
                accept(Acquaintance, Task))
```

Commitment to the role of contractor also entails an additional responsibility - when the task has been completed the contractor is obliged to inform the manager that the task has finished and also of any results which have been generated. This behaviour ensures that there is the appropriate flow of medical information within the organisation; it can be encoded in the CCIS by the following generic rule:

```
IF Task is completed and it produces Results, AND
   Acquaintance is committed to Agent for Task
THEN inform (Acquaintance, Agent,
                performed(Task),
                results-produced(Task, Results))
```

### 4.3 Adaptive management of commitment changes

In most cases, when an agent commits itself to perform a task then that task will actually be executed. However in a number of well-defined circumstances it may be appropriate for an agent to renege on its commitment to a given activity. For instance, doctors may have to postpone or even cancel tasks to which they were previously committed because of an unforeseen lack of resources - eg extremely urgent tasks such as emergency operations may arrive, or the agent contracting a task may unexpectedly become unavailable.

Another class of reasons for dropping commitments are that task execution may no longer be necessary. Firstly, the need for the task may cease to exist - for instance, because of the sudden disappearance of the patient's disease, because the patient is unwilling to have the task performed, or because of the unexpected death of the patient. Secondly, it may no longer be feasible to execute a given task. This situation would typically arise as a consequence of the receipt of new patient data - for example, a planned chemotherapy may have to be withdrawn because the patient has a high temperature resulting from the toxic effect of the drug.

Having detailed the conditions under which commitments can be reneged upon, the social aspect of conventions needs to be addressed (figure 2). If a contractor drops its commitment to a task, then it is imperative that it informs the manager. Similarly, if the manager realises that a task which it has contracted out is no longer valid then it should inform the relevant acquaintance so that no unnecessary effort is expended. Empirical evaluation has shown that conventions such as these are an essential ingredient of robust coherent cooperation in environments in which agents possess neither complete nor correct beliefs about their world or other agents, have changeable goals and fallible actions, and which may be subject to interruption from external events [Jennings 1992]. Embodying this convention in the CCIS ensures that the cooperating care agents behave coherently in the face of dynamic and unpredictable changes in the network.

## 5. Conclusions

This paper describes the need for, and feasibility of, cooperative information systems in the real world

REASONS FOR RE-ASSESSING COMMITMENTS TO A TASK:

- Task is no longer necessary
- Resources for Task become unavailable
- Commitment to the super-task of Task is dropped

ACTIONS:

R1: IF Manager of Task believes that Task is no longer necessary
THEN *request* (Manager, Contractor, drop-commitment(Contractor, Task))

R2: IF Contractor for Task believes that Task is no longer necessary
THEN *inform* (Contractor, Manager, not-necessary(Task, Reason))

R3: IF Contractor for Task drops his commitment to Task, AND
Task has a SubTask
THEN *request* (Contractor(Task), Contractor(SubTask), drop-commitment(Contractor(SubTask), SubTask))

R4: IF Resources allocated to Task become unavailable
THEN Contractor for Task drops his commitment to Task, AND
*inform* (Contractor, Manager(Task), drop-commitment(Contractor, Task, Reason))

R5: IF Manager of Task is *informed*
that Contractor for Task is no longer committed to Task, AND
Manager believes that Task is still necessary, AND
Manager has another Acquaintance accountable to him for Task
THEN *request* (Manager, Acquaintance, perform (Task))

**Figure 2. Convention for Adapting Commitments**

domain of distributed patient care. The type of organisation required by the cooperating care agents has been characterised and the key types of social interaction have been identified. Using this framework, the minimum set of communication primitives required to sustain cooperation in this application have been specified. The mechanisms for supporting robust cooperative behaviour have also been described: accountability helps an agent decide to which acquaintance a given task should be awarded, commitments provide the basis of trust which is essential for distributed action, and conventions provide the means of monitoring commitments so that the cooperating group performs in a coherent and efficient manner.

As a consequence of the findings of this paper, a prototype CCIS is being developed at the Imperial Cancer Research Fund. This system implements the proposed cooperation rules together with the readily available symbolic decision procedure and a task manager for handling task state transitions [Huang 1993b]. Emulated mail facilities have also been built for message passing among simulated care agents, based on the proposed communication primitives. The prototype is currently being evaluated by cancer specialists and clinical managers and it is anticipated that this technology will become widely available in the next ten or twenty years.

## Acknowledgement

The authors are grateful to Jeremy Wyatt, Colin Gordon and Andrew Jackson-Smale for their help with the preparation of the paper. They also wish to thank the EC Advanced Informatics in Medicine (AIM) programme for funding the DILEMMA project.

## References

[Corkill 1979] Corkill D D, "Hierarchical planning in a distributed environment", in: *Proc. of the Sixth Int. Joint Conf. on Artificial Intelligence* (IJCAI-79), Tokyo, Japan, pp.168-175

[Fox 1992] Fox J and Krause P, “Qualitative frameworks for decision support: lessons from medicine”, *Knowledge Engineering Review* 7, pp.19-33

[Gerson 1976] Gerson E, “On quality of life” American Sociological Review 41, pp793-806.

[Gordon 1993] Gordon C, Herbert S I, Jackson-Smale A and Renaud-Salis J L, “Care protocols and healthcare informatics”, in: *Proc. of Artificial Intelligence in Medicine Europe 93* (AIME-93), Munich, Germany, pp.289-309

[Huang 1993a] Huang J, Fox J, Gordon C and Jackson-Smale A, “Symbolic decision support in medical care”, *Artificial Intelligence in Medicine* 5, pp.415-430

[Huang 1993b] Huang J, “A protocol task manager based on the DILEMMA Protocol Model”, Technical Report, the DILEMMA project, Advanced Computation Lab, Imperial Cancer Research Fund, London, England

[Jennings 1992] Jennings N R, and Mamdani E H, “Using joint responsibility to coordinate collaborative problem solving in dynamic environments”, *in: Proc of 10th National Conf. on Artificial Intelligence* (AAAI-92), San Jose, USA, pp.269-275.

[Jennings 1993] Jennings N R, “Commitments and conventions: the foundation of coordination in multi-agent systems”, *Knowledge Engineering Review* 8, pp.223-250

[Lux 1993] Lux A, de Greef P, Bomarius F and Steiner D, “A generic framework for human computer cooperation”, in: *Proc. of Int. Conf. on Intelligent and Cooperative Information Systems* (ICICIS-93), Rotterdam, the Netherlands, pp. 89-97

[Pritchard 1992] Pritchard P, “The role of computers in referral”, in: Hopkins A & Wallace P (eds.), *Referrals to Medical Outpatients*, Royal Colleges of Physicians and General Practitioners, London, England, pp. 79-89

[Renaud-Salis 1992] Renaud-Salis J L, Lagouarde P, Gordon C and Thomson R, *“Requirements for decision support in cancer shared care*”, Technical Report, the DILEMMA project, Fondation Bergonie, Bordeaux, France

[Searle 1969] Searle J R, “Speech acts: an essay in the philosophy of language”, Cambridge University Press.

[Smith 1980] Smith R G, “The contract net protocol: high-level communication and control in a distributed problem solver”, *IEEE Transaction on Computers*, C-29(12), pp.1104-1113