MULTI-AGENT SUPPORT FOR MODELLING CO-OPERATIVE WORK

Mihhail Matskin
Department of Computer and Information Science, Norwegian University of Science and Technology, N-7491 Trondheim, Norway

Key words: Agents, Distributed AI, Group Communication Modelling

Abstract: We consider a development of multi-agent support [5,6] for modelling co-operative work in information systems. The development includes conceptual foundation of the method and its implementation issues. Basic elements of our approach are a flexible conceptual method for modelling co-operative work by identifying work participants and co-operative points and a concept of Agora as facilitator of co-operative work. An essential feature of our approach is its openness. The approach is demonstrated by an example of modelling co-operative work of a Program Committee of a conference.

1. INTRODUCTION

Development of computer networks dramatically changes a way of representation and usage of information. Instead of centralised archives and databases more and more decentralised information sources are available. A complete modelling of such distributed information sources is quite problematic.

The main problem is as follows. Most of former modelling approaches were designed for analysis of information in completely specified environments. These environments contain all sufficient information for operation at the environment. It does not mean that a completely specified environment should be only static. However changes in these environments are presented explicitly as well as actions which are also described

---

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35581-8_35

T. Yongchareon et al. (eds.), *Intelligence in Networks*

explicitly. In the other words our environment ("world") is closed in this case.

In the case of decentralised information sources in the network an environment in many cases cannot be specified completely. It is problematic (if possible) to describe completely all available information sources in the network as well as to specify precisely and completely all possible actions and changes. The distributed environment ("world") is not closed any more and, generally, there can be no restrictions for affecting the "world" and being affected by the "world".

A constructive approach for work in an open environment could be similar to human's way of behaviour in a real world which is also open. This approach is based on perceiving the world, having a rational model of behaviour and having intentions/motivations to be fulfilled by implementing corresponding goals. Of course, it is a schematic view to "human's way of behaviour", however, it emphasises autonomous, pro-active and distributed nature of exploring information sources and it is constructive in the case of open world. This approach becomes widely acceptable as an agent-based approach where agents are software entities/systems to whom users delegate some activity to be performed. For definition of agents we refer to [14], where agent is defined as an autonomous, pro-active, social and reactive system. We emphasis these features as essential ones for exploring information sources in distributed environment.

However not only information sources become distributed - control also becomes distributed and decentralised. In order to support decentralised control the role of co-operative mechanisms, such as co-ordination, negotiation and communication, increases very much. Exploring information in the network and perception of an open environment often requires co-operative efforts and global/local co-ordination, negotiation and communication activities of agents.

In this paper we consider an architectural support for modelling co-operative work between agents in a distributed information system. The paper is based on a generic architecture proposed in [5,6]. Compare to the previous papers ([5,6]), which are focused on justification of the approach, here we emphasise conceptual foundation of the approach and its implementation issues.

2. WORKING EXAMPLE

As a working example for our approach we use a simplified version of a Program Committee (PC) work modelling from [5].

In general we refer to intuition and experience of most of readers who participate in the work of a PC and we assume that PC includes a chairman (it may be co-chairman also) and PC members. PC is formed by the PC chairman by personal invitation of members after their agreement. Papers are submitted to PC chair(s) and then allocated to PC members for reviewing. Generally, PC members review themselves but they also can ask other experts to review a paper on their behalf. Critical co-operative points in the work of PC are:

- allocation of papers for reviewing,
- making agreement about paper evaluations between several reviewers,
- satisfying deadlines for reviewing and notifications of the acceptance or rejection of papers.

3. GENERAL APPROACH

The main question we would like to answer is as follows: what are possible/convenient ways of modelling co-operative work in an open and decentralised environment?

We choose agent-based approach as a general paradigm. The basic advantages of agents refer to their ability to autonomous, pro-active and social behaviour in an open environment (this is briefly considered in Section 1) as well as to ability to represent user-delegated goals and tasks. A detail practical justification of this choice is given in [5,6] and some experience remarks are presented in Section 5.

Briefly speaking we started with analysis of generic scenarios of co-operative work from different perspectives: functionality, data sources and co-operative work support. The functionality and data sources mostly relate to user's view of a system and the co-operative work support relates to a system organisation view. Of course functionality and data are important sources for designing system organisation, however, their influence is mostly implicit. But co-operative work perspective may provide us ideas for system organisation explicitly. As basic elements of such organisation we identify co-operative mechanisms and participants. In our case co-operative mechanisms include co-ordination, negotiation and communication and participants correspond to agents.

The main idea of our approach is as follows. We consider a co-operative work as a set of communications, co-ordinations and negotiations between participants (agents). In this case modelling of such work assumes identification of communication, co-ordination and negotiation points between participants in the co-operative work and providing support for these co-operative points. In the case of PC work, examples of such co-operative points are as follows:

- PC formation,
- allocation of papers to PC members,
- allocation of papers to reviewers (if different from PC members),
- making agreement about evaluation of a paper (if discrepancies exist),
- satisfying deadlines for reviews and notification of the acceptance or rejection.

Examples of participants in the case of PC work are PC members and other reviewers represented by agents in a computer environment.

In order to support work in the co-operative points we introduced a notion of Agora [5,6]. Agora is an area or meeting place where agents can advertise themselves and establish a common context. In order to be able to use Agora, an agent should be registered. Registration of the agent means pointing out it's name and public information about it's properties, which may include agent's goals, tasks it is interested in, tasks it can perform etc. In addition to advertising information about agents, Agora contains information about the problem or situation it should support, common knowledge, references to related Agoras, organisational structures of co-operatively solvable problem and knowledge about information sources. Hence, a common work context can be established by explicit representation of the contents of Agora and by registration of its agents.

More generally, Agora is a facilitator of co-operative work (co-operation, negotiation and communication) in a way similar to KQML facilitators which facilitate only communication [2]. In addition to registered participants agents each Agora has default (manager) agent and registered co-ordination or/and negotiation agents (which maintain a particular co-ordination or negotiation protocols).

In accordance to the above-mentioned the elements of co-operative system modelling are as follows (see Figure 1):

- Agents: registered software agents,
- Agoras: facilitators of co-operative work,
- Default agents (managers): system agents which maintain Agora's functioning.

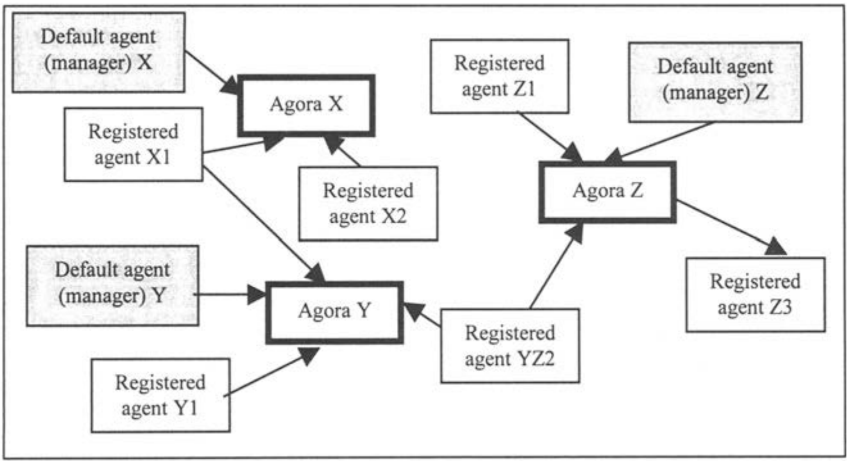

Figure 1. Agents and Agoras

4. AGORA CONTENT

Our next step in modelling co-operative work is implementation of Agora as a facilitating infrastructure for such work. In order to implement Agora decisions about Agora's content should be made. We consider the following three groups of elements in the Agora contents:

- Information about registered agents (individual agent's information),
- Information about Agora itself (group information),
- Information about other Agoras and common knowledge (surrounding information).

All Agora components have standard (default) implementation in a generic Agora shell. However the default implementations can be overridden by an Agora creator by attaching corresponding agents (in this case the components in Agora architecture become proxy components with pointers to actual components). The functional architecture of Agora is presented in Figure 2.

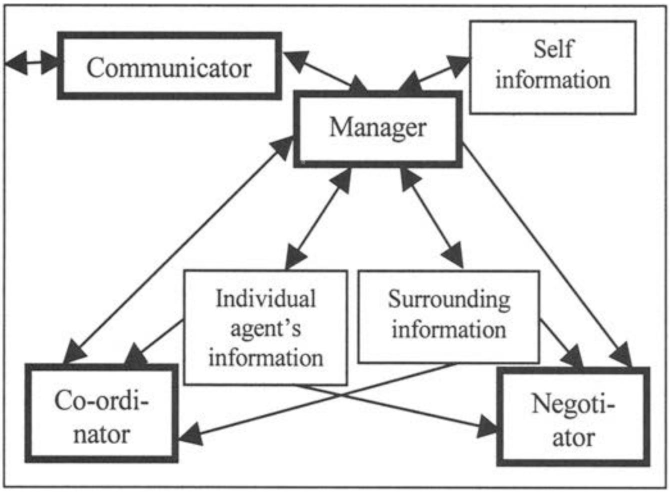

Figure 2. Agora functional structure

The communication component (Communicator) is similar to the usual agent communication block (it includes Agent Communication Language component for communication with agents and a signal-level communication component for communication with environment) and we assume a traditional solution based on KQML/FIPA [2,12] for it. We also assume that co-ordination and negotiation components (Co-ordinator and Negotiator) implement some co-ordination/negotiation protocols (e.g. [11]).

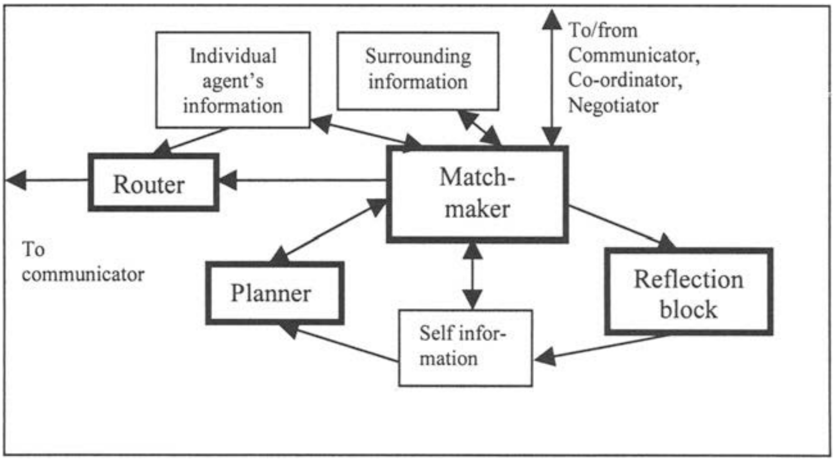

Figure 3. Agora manager

The manager is a central element in the Agora architecture and it contains the following blocks (see Figure 3):

- router - if message which is sent to the Agora should be processed by another agent then router redirects it to the corresponding agent,
- matchmaker - matches dependent/relevant information sources in Agora structure and decides what to do if matching succeeds/fails,
- planner - makes plans for action based on self information in Agora,
- reflection block - updates self information by observation of matchmaker work (more details about the reflection block can be found in [7]).

4.1 Agora information about registered agents

Agora needs to store different information about registered agents. This information includes:

- agent name and address,
- goals - they can be single (expressed by a word) , complex (expressed by an expression like *finish(reviewing(before(28.02.99)))* or multiple,
- agent's beliefs, desires and intentions - this refers to quite common for agents BDI-architecture [8],
- tasks - they are divided into two sets: tasks the agent can perform and tasks the agent is interested in,
- friend agents - other co-operating agents.

The information about agents can be used for message sending or for matchmaking. Matchmaking is an important operation on Agora content and it can be performed on tasks, goals, beliefs, desires and intentions. The purpose of matchmaking is to find another agent in Agora who can help in problem solving. Matchmaking can have different level of complexity from search for keywords to complex deductive procedures.

In the case of PC work modelling PC-Agora is created and each PC member is represented by an agent. The agents are registered in the PC-Agora where they present their goals (e.g. reviewing before deadline), tasks they can perform (e.g. papers/subjects they can review), tasks they are interested in (e.g. papers/subjects they are not experts in and where they ask some other experts to be reviewers) and friend agents (e.g. other agents who help them in reviewing). Papers to be reviewed can also be presented as agents which specify area of expertise of the papers (e.g. by keywords) and tasks they are interested in (e.g. to be reviewed by experts). However in our simplified example we don't consider papers as pro-active agents but rather as passive information sources.

4.2 Information about Agora itself

Agora information may include:

- organisational structure of agents participating in co-operative work including roles they may play in common work,
- Agora's goals and tasks – they may differ with goals and tasks of some registered agents,
- protocols for communication - in case of KQML/FIPA-like communication it may be a set of performatives which are accepted by the Agora,
- among others there should be default (manager) agent and at least one co-ordination or negotiation agent, which support corresponding protocols,
- information about current activities (own and other registered agents) including current tasks, goals etc.,
- a list of registered agents.

The above information can also be used for matchmaking and for advance interaction of agents with Agora. For example, if a role of an agent in Agora's organisational structure is restricted to reviewing only then agents having other roles will not be allowed to register themselves at the Agora. If we present papers as agents then matchmaking of areas of reviewer's expertise and paper's subjects can be done for allocation of papers to reviewers.

4.3 Information about other Agoras and common knowledge

Agora can contain information about related Agoras and about common information sources and knowledge.

In the case of PC work, related Agoras are those created by PC members for organising reviewing their papers and common information sources are, for example, papers submitted to reviewing.

Common knowledge is usually represented as ontology.

5. PUTTING PIECES TOGETHER OR LET US RUN OUR EXAMPLE

We briefly considered basic features of the approach and its support by Agoras. Now we consider running of our working example (PC work) in accordance to the approach and present experience remarks. Some elements of performing the example are distributed around different chapters of the paper, however, here we try to put them together.

The first step in modelling PC work is making decision about participants of the co-operative work or in the other words – about agents which will inhabit our system. There are the following two possibilities: PC members can be represented by agents and/or papers for reviewing can be represented by agents. Our first decision is to restrict a set of agents to PC members agents only (we do that for simplicity only but there are no restrictions for other decisions).

The next step in our approach is identification of co-operative points in the co-operative work. Some of them (we consider only the basic ones) are PC formation, allocation of papers to PC members and to other reviewers and discussion of paper evaluation. These co-operative points should be supported by different Agoras.

When PC work starts PC chairman creates a PC-Agora (see Figure 4), registers his/her agent at it and asks potential PC members to participate in the PC work (for simplicity we assume that this is done via email, phone, fax etc.). Those who agree to be in PC should register their agents at PC-Agora (during this registration they also specify areas of their expertise in the field).

PC chair also makes decision about a procedure of allocation of papers by registering corresponding co-ordination agent. We consider two basic way of doing allocation of papers: centralised and decentralised.

In the case of centralised allocation, a co-ordination agent is a matchmaking agent. It matches keywords in papers (or titles) and areas of expertise of PC members. Conflict resolution and general constraints can be embedded into the co-ordination agent.

In the case of decentralised co-ordination agents, more advance protocols can be used. We consider only one of possible protocols: Contract Net Protocol (CNP) [11]. In this case co-ordination agent is a manager who announces work to potential contractors (PC members) as a list of papers to be reviewed. PC members submit their bids for doing this work by selecting papers they wish to review from the list and attaching priority numbers (prices) to the selected papers. Co-ordination agent analyses bids and awards papers to the higher bidder. In case of collisions, a negotiation between corresponding PC members agents happens (this may be supported by creation of a temporal negotiation Agora for these agents as in Figure 4).

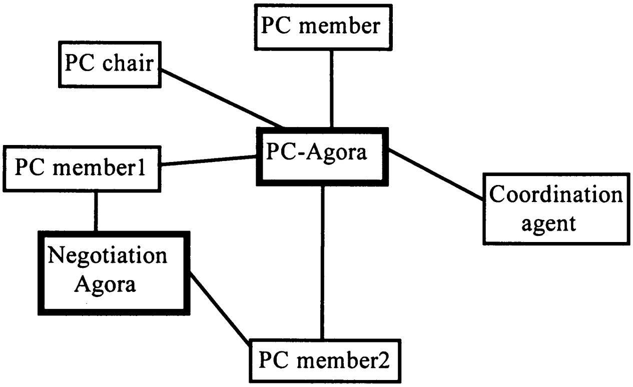

Figure 4. PC formation and allocation of papers

Next, PC members may create their own Agoras (see Figure 5) if they ask other experts to help them in reviewing - this process is similar to above-described procedure of PC formation.

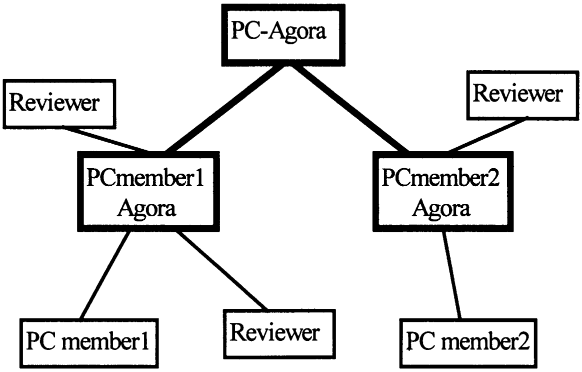

Figure 5. PC members Agoras

After that PC chair can register another co-ordination agent to the PC-Agora which provides awareness functions for notification of the deadline for reviews.

PC members submit their reviews to a co-ordination agent who analyses them and try to find discrepancies in evaluation of papers. If discrepancies are found then a negotiation between reviewers should start. In order to support such negotiation a temporal Agora is created (see Figure 6) and a negotiation agent for support of particular negotiation protocol (for example, Monotonic Concession Protocol (MCP) [9]) is registered at it.

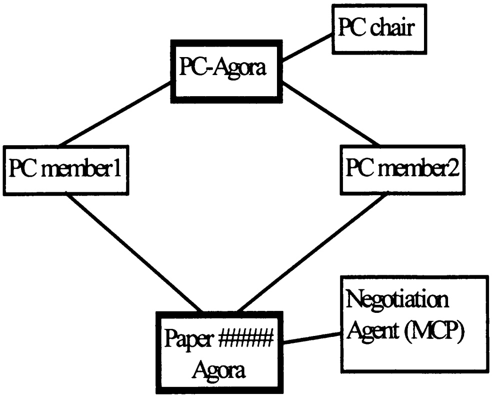

Figure 6. Discrepancy resolving

From the above example and our experience in using Agora approach we identify the following basic benefits of the multi-agent approach to co-operative work:

- Agents add speed and volume into process. This means that agents may react faster and, what is more important, in a more guaranteed way than humans they represent. For example, it is important for a fair allocation of papers to PC members to react promptly to many requests when collisions happen. However humans often do not reply to emails in time or ignore requests they think to be irrelevant – this may make the whole process of negotiation meaningless. By added volume we mean, for example, ability to more detailed negotiation via a large number of interactions with partners which are often ignored by humans.
- Automation of non-trivial time-consuming stages of the problem-solving process. In our case, conflict resolution (for example, in case of discrepancies in paper evaluation) is a critical and time-consuming stage. Multi-agent approach allows provide an automated support to such stages by means of utilisation of negotiation and co-ordination techniques.
- Personalising of software. Because of agents always represent their creator in a computational environment they should inherit user's goals, tasks and profiles and this enforces creation of personalised software.

- In the case of our working example the Agora approach may result in a higher fairness of paper allocation and in a higher quality of discrepancy resolving because of more detailed and personalised decisions.
- Agoras give a practical support for negotiation and co-ordination in the multi-agent system by providing an infrastructure and templates for co-operative points. For example, standard types of Agoras can be provided for PC work support, for group learning, for project managing etc. This means that Agoras provide not only support for a particular co-operative work but also accumulation and re-use of knowledge about such support.

6. RELATED WORKS AND CONCLUSIONS

Much research has been done, from different perspectives, in the areas of Distributed AI [4] and Computer Supported Co-operative Work (CSCW) [10] in order to support co-operative work. However most of the works consider negotiation, co-ordination and communication in co-operative work separately and do not pay enough attention to their integrated support. We try to overcome this in the case of Agora approach.

An interesting concept of communication facilitators was proposed in [2]. However our approach differs with it in the following moments:

- we allow dynamic creation and destroying of Agoras,
- we allow more flexible matchmaking by customisation of matchmaking components via possible overriding,
- functionally Agora is oriented to facilitation of different elements of co-operative work rather than only communication as in [2],
- we allow interrelated Agoras.

Another well-known approach – blackboard architecture [3] – is usually compared when Agora architecture is presented. However Agoras differ with blackboards in the following:

- we use registration of participants and do not allow global access,
- Agora allows flexible creation and it has more flexible control unit,
- Agora is oriented to co-operative work support and it always contains negotiation and/or co-ordination agents,
- Agora has richer content than blackboard and we consider our architecture to be higher level than traditional blackboards or hierarchies of blackboards [1].

Finally, we summarise the basic original features of the Agora approach as follows:

- we propose a flexible, conceptual method for modelling co-operative work by identifying work participants and co-operative points,

- we keep a proper level of abstraction in the multi-agent architecture (not too low and not too problem-oriented),
- we propose a concept of Agora as facilitator of co-operative work,
- openness is an essential feature of our approach (all basic elements of the considered co-operative work support - negotiation and co-ordination protocols, default services etc. - can be redefined by the user or agent).

A current implementation of the Agora architecture is done in Java and KQML (supported by the JATLite package [13]) is assumed to be a default communication protocol for agents.

ACKNOWLEDGEMENTS

This work is partially supported by the Norwegian Research Foundation in the framework of the Distributed Information Technology Systems (DITS) program and the ElComAg project. We would like to thank Monica Divitini and Sobah Abbas Petersen for fruitful discussions during the early stages of this work and Tom Arne Asbjoernsen and Reza Tahriri for their work on implementation of the Agora architecture.

REFERENCES

1. P. R. Cohen, A. Cheyer, M. Wang and S. C. Baeg. *An Open Agent Architecture*, in *Readings in Agent*. M.N. Huhns and M.P. Singh, Editors. 1998, Morgan Kaufmann Publishers: San Mateo, CA. p. 197-204
2. T. Finin, Y. Labrou, and J. Mayfield, *KQML as an Agent Communication Language*, in *Software Agents*. J.M. Bradshaw, Editor. 1997, AAAI Press/The MIT Press: Menlo Park, CA. p. 291-316.
3. B. Hayes-Roth, *A Blackboard Architecture for Control.* Artificial Intelligence, 1985. 26 (3): p. 251-321.
4. G.M.P. O'Hare and N. Jennings, eds. *Foundations of Distributed Artificial Intelligence.* 1996, John Wiley & Sons, Inc.
5. M. Matskin, M. Divitini and S. A. Petersen. *An Architecture for Multi-Agent Support in a Distributed Information Technology Application.* International Workshop on Intelligent Agents in Information and Process Management on the 22nd German Annual Conference on Artificial Intelligence in Bremen (KI-98), TZI-Bericht Nr. 9, Germany, September 15-17, 1998, pp.47-58.
6. M. Matskin, M. Divitini and S. A. Petersen, *Agora: a Multi-Agent Support for Distributed Information Technology Applications.* NIK'98, Tapir, November 23-25, 1998, pp. 281-292.
7. M. Matskin and E. Tyugu, *Shells for Multi-Agent Applications (An Architecture for Agents and Agoras).* Technical report, NTNU, IDI-nr. 4/99, ISSN-NO: 0802-6394, February 1999, 43 p.

8. A. S. Rao and M.P. Georgeff, *BDI-agents: from theory to Practice*. ICMAS'95 - first International Conference on Multi-Agent Systems. 1995. San Francisco, CA, USA, 12-14 June 1995: AAAI Press.
9. J. Rosenchein and G. Zlotkin, *Rules of Encounter: Designing Conventions for Automatic Negotiation Among Computers*. MIT Press, Cambridge, Massachusetts, 1994.
10. K. Schmidt and L. Bannon, *Taking CSCW Seriously: Supporting Articulation Work*. CSCW Journal, 1992. 1(1-2): p. 7-40.
11. R. Smith and R. Davis, *Frameworks for Co-operation in Distributed Problem Solving*. IEEE Transactions on Systems, Man and Cybernetics, 1981. SMC-11(1).
12. URL: http://fipa.com.cotec.jp/proxy/fipa/
13. URL: http://java.stanford.edu
14. M. Wooldridge and N. Jennings, *Intelligent Agents: Theory and Practice*. The Knowledge Engineering Review, 1995. 10(2): p. 115-152.