Supporting Self-Facilitation in Distributed Group Decisions

Constantin B. Zamfirescu
Katholieke Universiteit Leuven
Dept. of Mech. Engineering, Division P.M.A.
Celestijnenlaan 300-B (Leuven) - Belgium
zbc@acm.org

Florin G. Filip
National Institute for R&D in Informatics
8-10 Averescu Avenue
71316 Bucharest 1, Romania
ffilip@acad.ro

Abstract

The paper addresses the problem of supporting self-facilitation of distributed autonomous groups of decision-makers from an agent-oriented perspective. In contrast to the monolithic designs, based on functional decomposition, it is argued that an agent-oriented approach is the appropriate means to provide better support for group meetings. The focus is on distributed and asynchronous meetings, detailing the conceptual modelling aspects of an implemented agent-based GDSS which exhibit an emergent functionality able to support in an effective way the dynamic of the group decision process.

1. Introduction

Supporting in a flexible way a wide range of group decisions is still a challenge for Group Decision Support Systems (GDSS) research. GDSS are defined as interactive computer-based systems that support concerted team effort towards completion of joint tasks. As a combination of computers, communications and decision technologies working in tandem to provide support for problem identification, formulation and solution generation during group meeting [1], GDSS reveals its sociocentrical nature. Early GDSS consisted of a meeting room with special facilities, but rapid advances in communication, computer and decision technologies enabled a web-centric approach to group decision support.

Besides structuring and supporting decisional process, GDSS radically changes the dynamics of group interactions. Supporting a group of workers as a self-directed team does not automatically create self-management capability. A key component in GDSS settings is the meeting facilitator who helps the group to achieve its own outcome. Facilitation has emerged as one of the most important factors in the effective use of GDSS. However, despite the fact that a skilful facilitator is not easily available, for distributed and asynchronous GDSS such facilitation is even unfeasible.

Many studies have addressed the possibility of supporting an inexperienced facilitator [2] in setting up the decisional process. But virtual teams, by their very nature, can be considered self-managed groups. Elden and Chisholm [3] note that asynchronous and distributed meetings can be self-managed because of their long duration. In such settings, it is argued that appropriate facilitation is given by the group itself, and not by an external facilitator, allowing the group to adopt in its own way the technology under use.

Taking into account that self-facilitation represents the most appropriate way to address some of the problems encountered in distributed and asynchronous GDSS practice, the remainder of this paper is organized as follows. Section 2 will give an overview of promising research directed toward a flexible and a natural integration of software agents into human teams. In section 3 a general decisional process framework will be proposed and discussed. The system design of the co-ordination mechanisms used to achieve the desired flexibility in meeting planning will be outlined in section 4. Some concluding remarks and future work directions will be given in the final section.

2. The synergy between GDSS and MAS

Extensive employment of Simon's model becomes an obstacle for the evolution of decision support systems theory and practices [4]. Due to their incompleteness, the rigidity of decisional models employed in GDSS has been criticized on a number of grounds. Ad hoc arrangements that are in the best interest of the group may not be compatible with the discipline imposed by the system. In classical organizations, decisions are often related, creating a chain of temporal dependencies. Therefore, GDSS has to exhibit sufficient flexibility to support decisional process in very dynamic settings.

In the last decade, the approach used to solve complex problems has shifted from developing large and integrated software systems, to developing small and autonomous software components that can interact with human beings, with other software components, and

different services or data sources. Payne *et al.* [5] identify four main dimensions along which agents may support teamwork: team situation assessment, team-supporting behaviours, team leadership/initiative and communication among team members. These are in line with the adaptive structuration theory which address how groups use and adapt structures that direct group process [6]. The theory argues that the system used is more important than GDSS itself.

Due to the inadequate support of GDSS to adapt its internal structure on how the group is appropriating the system, the incompleteness and rigidity of decisional models used, and the uncertainty carried out in meeting planning it becomes inevitable that GDSS design is complicated enough to discourage wide use of the system. Fortunately, the multi-agent system (MAS) paradigm represents one of the most promising approaches to address such kinds of problems. As with any other technology, they have certain capabilities and for GDSS design, at least two such capabilities are of special interest: the complexity of large and highly distributed systems may be controlled more naturally and constructing scalable systems is easier since the addition of more agents becomes a non-disruptive task.

3. Modelling the group decision process

The group decision-making process involves a mixture of contingencies which may produce unexpected constraints during the decisional process. Such constraints are difficult to factor in and, consequently, to support. Following Ackoff [7] suggestions for interactive planning, such as flexible problem solving, a group decision-making process is necessarily participative, coordinated, integrated and continuous.

The decisional model described below is inspired by the shared-plans theory [8], which provides a well-defined framework for a group decision process. Moreover, it subsumes all the prospects given by Ackoff [7] for an effective decisional process. Its focus is on providing a general coordination pattern among participants, and not on the variety of tasks to be carried out. Shared-plans theory states that the participants need to have mutual beliefs about their goals and actions to be performed. It is assumed that participants have the required capabilities, intentions, and commitment to execute these actions.

A shared-plan (SP) is defined as a seven-tuple (P, G, a, T_p, T_a, R_a, C_a) denoting that the group *G* 's plan *P* at time T_p is to do action *a* at time T_a using recipe R_a in the context C_a. For a GDSS, the plan represents the group decision process structured in certain phases or actions with a defined deadline until these actions have to be executed by the group. The recipe is simply the tool used to support this action, i.e. brainstorming tool, voting tool, cardinal ranking tool. The context is refers to what Briggs et al. [9] denote as being the tool configuration, which has considerable impact on group outcomes.

Several aspects of SP theory are relevant from the GDSS viewpoint. Firstly, the decisional process may start with only a partial preparation. The whole decision process often requires groups to move between multiple intertwined processes, being considered to evolve over time as group members reactively decide the next step based on the in progress context. A continuous cycling between decision plan generation, alternative assessment, plan monitoring, commitment, plan elaboration, and plan execution is implicitly supported by the GDSS to mediate the decisional process. These steps should not be considered as imperative actions, but as opportunities provided by the system.

Secondly, the model can incorporate both a group decision and an individual decision. Not all the decision steps belong to a SP, some being relevant only for a certain decision-maker. This kind of service can be found also in some GDSS architectures, where decision-makers have exclusive access to particular types of decision support system (DSS), but it is not settled explicitly into the collective decision plan.

Thirdly, the model supports delegation and negotiation**.** Depending on the complexity of the problem, both the goal and the task may be decomposed. The model supports delegation at different levels of abstraction, from individual delegation to team delegation. Moreover, the context of each step could explicitly stipulate if the decision has to be made by a certain number of decision-makers from a designated group.

Fourthly, the model supports both synchronous and asynchronous collaboration. Discussion rules are defined within the context of each action. The meeting initiator must identify conventional limits according to the characteristics of the problem and the group. These limits could be constantly updated during the decisional process as they are an important feature for distributed meetings.

Fifthly, the model constitutes the basis for an action research tool inside the group. The model is close related to the action research theory [10], which involves method and tools inspired by the social and behavioural sciences to study the in situ use of the GDSS technology. From this perspective the participants become active researchers having the opportunity to intervene directly in the decisional process, evaluating and learning the consequences of their actions, and improving the practice and knowledge of the group.

Situations where disagreements regarding the

commitment to follow the course of actions emerge will lead to communication within the team. An important purpose of communication between participants is to determine the appropriate course of actions to execute and who should do what. The model used to support negotiation among participants about the meeting plan is based on the rationale discussion IBIS model [11]. All the decisional plan parameters are mapped automatically as distinct Issues together with the Arguments of the meeting initiator to employ them. It is expected that the communication will decrease in time as the group appropriate the system and the experience in using it will contribute to the effectiveness of the meeting [12].

4. System design

In order to gain wide acceptance of the deployment of agent technology in industry, during the last decade many researchers paid special attention to agent-based software engineering methodologies. Our methodology follows closely the guidelines presented in [13], including three main phases: identification of agents and roles played by them, inter-agent communications and internal behaviour of the agents. The representations in the following subsections are depicted in AUML [14], an under development UML-extension standard to facilitate the formalization of the agent-based system development lifecycle.

4.1. Species of agents

In contrast to the functional decomposition of classical software engineering technique, in MAS the agents are assigned to the particular entities involved in the system. This concern preserves the system's flexibility to provide an emergent functionality as consequence of the interaction among agents. SSDG functionality is not dispersed among many functional units keeping the system inept for the required dynamic of an effective group decision support, but the group appropriates the SSDG. For a GDSS three main entities may be identified: the users, the tools used to support each decision phase and the problems.

Each user is assisted by a *personal assistant agent* (A_A) in order to maintain his profile for different roles, tasks, contexts and tools used during each meeting. The users' profile is used afterwards to suggest adequate advice during system use. Tool capabilities are stored as meta-information in the resources' profile, but these capabilities have to be tailored and integrated according with the problem context, as well as the skills and experience of the users. In this way, how the system interacts with the users is decoupled from the task at hand and the tools used to support it.

Resource agents (A_R) provide tools-related services for the rest of the system, offering at the same time intelligent access to a heterogeneous collection of possible decision tools and data. They make functional details transparent to users, supplying specialized or periodic information in performing some task or service based on the given information. For the current implementation, the resources are limited to four types of voting tools [15], a rational discussion module based on the IBIS model [11] and one web-based AHP decision tool [16].

Plan agents (A_P) stand for the agents that represent each active plan of execution. On the one hand, by matching their requirements with the resource capabilities, plan agents mediate the communication between the personal assistant and the resource agents to better support the group decision problem. On the other hand, they coordinate actions across multiple plans. In short, the plan agents deal with four main functions: communication and coordination; planning; scheduling; and execution.

4.2. Agents' roles

Any agent may participate in several scenarios playing distinct roles in each. The A_A carries out the following roles: 1) *adviser* – to advice the user how to elaborate a decisional plan, i.e. selecting the most appropriate composition of the group and choosing the right tools which will be used, encourage participation in brainstorming sessions; 2) *negotiator* – when meetings are scheduled and tools to support them are negotiated between the personal assistant of the meeting initiator and the personal assistants of the intended participants; 3) *observer* – to keep track of user's actions in order to refine his profile concerning his skills and preferences in using a specific tool or collect problem related data for a post-meeting analysis; and 4) *informer* – to inform the user about the modifications occurred on his interest aspects. For the A_P the next roles are the most relevant: 1) *mediator* – to match the problem demands and users' profiles with the services provided by the resource agents; 2) *coordinator* – to coordinate the decisional process activities across several problems and users. Concerning the A_R, it acts only as a *provider* – to manage its associated tool.

4.3. Inter-agent communication protocol

Accordingly with Parunak et al. [13] an agent interaction protocol describes a communication pattern as

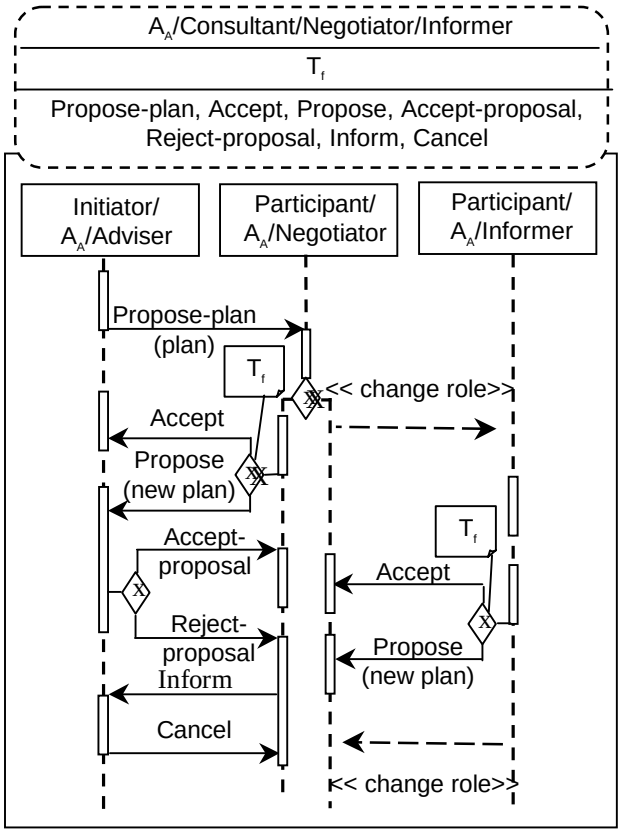

Figure 1. A sample from the communication protocol between personal assistants

an allowed sequence of messages between agents together with the constraints on the context of those messages. Figure 1 depicts merely a simplified example of the interaction protocol when a new plan is generated, where the A_A representing the initiator of the plan and the A_A of an intended participant are involved. When the initiator generates a new plan, his assistant sends the *propose-plan* message together with its explicit description (e.g. purpose of the meeting, deadline to accomplish the composite tasks, the tools used to support them, the group composition who will work on it) to the intended participant. The last, accordingly with its delegated responsibilities, could either take the entire decision on its own (e.g. when the scheduled date is unfeasible, the proposed tools are unacceptable) *accepting* the *plan* and/or *proposing* a new one, or could engage in a dialogue with its user. When this happens, the agent's role is changed from negotiator to informer; the participant's answer being forwarded to the initiator's agent. In its turn, the initiator could *accept* and/or *reject* the participant's *proposal*, in which case the participant's A_A will inform the initiator's A_A about his commitment to the action plan.

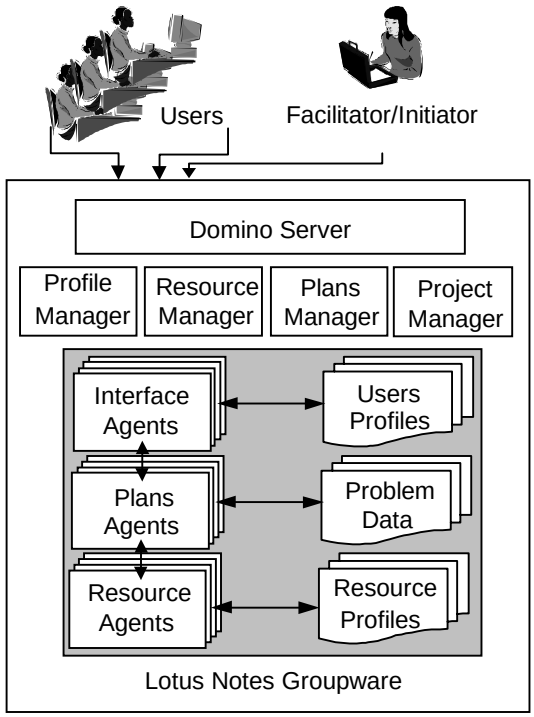

Figure 2. Agent-based GDSS architecture

4.4 Architecture and implementation issues

This work started from the compulsion to augment group decision capabilities with an existing project management tool implemented on Lotus Notes groupware. The system is used to coordinate among multiple teams situated in different countries software projects development. The Domino server is used to facilitate access over the Internet to related documents and to coordinate the development process. In any case, a significant amount of time is still being spent travelling around many places for very specific decisions in order to accommodate the programming environments, the software patterns employed or even to renegotiate the requirements at the performance level. The presented MAS is intended only to provide the level 3 of GDSS support [1], and to take on GDSS capabilities. This level of support deals with the group dynamic management. The other two levels, i.e. the individual support and the coordination level, are supported by a project management application implemented in Lotus Notes as well. The MAS is entirely programmed in Java and runs on Lotus Notes, which is now in trial phase. In Figure 2, the general architecture is outlined with three additional modules attached to the GDSS presented above. Profile Manager and Resource Manager are particularly intended for the exclusive use of system administrators, enabling them to introduce additional characteristics for

the users that have not been captured before and to plug in new tools into the system. A user profile is built initially with regard to the organizational structure, the skills and responsibilities that cover these positions, and the broad range of expertise in using technological issues. This initial profile is refined later during system use for each decision tool subsequently adopted. On the other hand, a coherent description of a resource requires special skills to define and classify resource capabilities, accessing and execution protocols, input and output parameters in order to facilitate necessary integration with other system components. Special attention is given to the context in which these tools will be executed. The execution context is provided in terms of pre-condition and post-condition rules that try to define tasks or group conditions that should be fulfilled before running the tool. The Plans Manager provides the forms to display, generate and elaborate the existing decisional plans offering at the same time an effective way to capture the inter-teams' memory of group.

5. Conclusions and further work

Given the fact that facilitation in distributed and asynchronous GDSS is rather unfeasible, it is argued that the group provides the most effective means of facilitation for any given problem, context and GDSS. The problem of self-facilitation has been addressed from a double perspective.

Firstly, to provide a generic and comprehensive framework in which the group decisions can take place, the process modelling is undertaken from the action research approach. The proposed model provides a generic setting not only for registering the order in which planning and execution occur, but also for deciding how to interweave them. It emphasises the itinerary to achieve the meeting goal rather than the goal itself. The decisional process emerges from the group interaction during the meeting and is not prescribed beforehand. It provides awareness-oriented collaboration, supports participants' auto-reflection during the meeting, captures the decisions and rationale behind each of them. The model has not been viewed as a substitute approach for the existing ones, but as a complementary advance able to expand the capabilities, range, and effectiveness of distributed and asynchronous GDSS meetings.

Secondly, from the software engineering perspective, the agent-oriented approach is arguably the most promising way to deal with incompleteness and uncertainty in which the participants appropriate the GDSS in distributed group meetings.

Although the approach described in this paper has already been implemented, its validation in real situations is still unconfirmed. Several variables are under investigation to measure the efficiency and effectiveness of the system along with group satisfaction in using it.

6. References

[1] G. DeSanctis, and B. Gallupe, "A Foundation for the study of Group Decision Support Systems", *Management Science*, 1987, pp. 589-609.

[2] P. Antunes, and T. Ho, "The Design of a GDSS Meeting Preparation Tool", *Group Decision and Negotiation*, 2001, pp. 5-25.

[3] M. Elden, and R.F. Chisholm, "Emerging varieties of action research", *Human Relation*s, 1993, pp. 121–141.

[4] A. Angehrn, and T. Jelassi, "DSS Research and Practice in Perspective", *Decision Support Systems*, 1994, pp. 267-276.

[5] T. Payne, T.L. Lenox, S. Hahn, M. Lewis, and K. Sycara, "Agent-Based Team Aiding in a Time Critical Task", *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, IEEE Press, Los Alamitos, 2000.

[6] G. DeSanctis, and M. Poole, "Capturing the complexity in Advanced Technology Use: Adaptive Structuration Theory", *Organisational Science*, 1994, pp. 121-147.

[7] Ackoff, R.L., *Redesigning the Future. A Systems Approach to Societal Problems*, John Wiley & Sons, New York, 1974.

[8] B. Grosz, and S. Kraus, "Collaborative plans for complex group action", *Artificial Intelligence***,** 1996, pp. 269-357.

[9] R.O. Briggs, G.J. Vreede, F. Nunamaker, Jr., and D. Tobey, "ThinkLets: Achieving Predictable, Repeatable Patterns of Group Interaction with Group Support Systems", *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, IEEE Press, 2001.

[10] P. Checkland, "From Framework through Experience to Learning: The Essential Nature of Action Research", H.E. Nissen, H.Hk. Klein and R. Hirschheim (Eds.) *Information Systems Research: Contemporary Approaches and Emergent Traditions*, Elsevier Publishers, 1991, pp.397-403.

[11] J. Conklin, "Capturing Organizational Memory", *Proceedings of Groupware*, 1992, pp.133-137.

[12] C.B. Zamfirescu, C. Candea, and S. Luca, "On Integrating Agents into GDSS", *Preprints of the 9th IFAC/IFORS /IMACS/IFIP Conference on Large Scale Systems*, Bucharest, pp. 231-236.

[13] H.D. Parunak, J. Sauter, and S.J.Clark, "Toward the Specification and Design of Industrial Synthetic Ecosystems", *Intelligent Agents IV: Agent Theories, Architectures, and Languages*, Springer, Springer, Berlin, 1998, pp. 45-59.

[14] http://www.auml.org

[15] Bui, T.X., *Co-oP: A Group Decision Support System for Cooperative Multiple Criteria Group Decision Making*, Lecture Notes in Computer Science, Springer Verlag, Berlin, 1987.

[16] J. Mustajoki, and R.P. Hämäläinen, "Web-HIPRE: Global decision support by value tree and AHP analysis", *Journal of Information Systems and Operational Research*, 2000, pp. 208-220.