Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 6 (1): 27-30, 2006
ISSN 1812-5654

A Distributed Data Management in Knowledge Based Group Decision Support Systems

[1]K. Raja and [2] S.K. Srivatsa
[1]SathyaBama Institute of Science and Technology, Chennai, 600 044, India
[2]Department of ECE, MIT Campus, Anna University, Chennai, 600 044, India

Abstract: Many critical decision-making situations happen in a dynamic, rapidly changing and often unpredictable distributed environment. In such situations, we need highly decentralized, up-to-date data sets from various resources. Unlike other conventional decision-making tools, Decision Support Systems (DSS) designed for such situations are challenged by the need to access this decentralized information overcoming all spatial and temporal constraints, while still exhibiting a high degree of collaborative activity. This study proposes a new approach to DSS in such contexts. Its main contribution is the integration of the co-operative nature of Group DSS (GDSS) with the decentralized advantages of distributed data management. This framework utilizes intelligent agent communication and a graphical user interface. The prototype, comprising a distributed database management system and the standard enabling architecture of a Group Decision Support System, will seek to evolve the existing DSS methodologies into one that lays emphasis on collaborative decision-making using data from diverse sources.

Key words: Distributed data management, DSS, GDSS, KBGDSS

INTRODUCTION

Most of the managerial decisions are taken by groups of responsible managers. This is because, nowadays, due to the information revolution, any business system in the global scenario is solving complex decision problems frequently. However, there is a need to take quick and efficient decisions in a rapidly changing world. Some of the problems associated with this are:

- Large number of mathematical and statistical calculations, selection procedures and methodologies.
- Difficulties in determining the most effective solution.
- Varied nature of decision-makers' domains.
- Decision-makers are generally not very comfortable using intelligent systems.

We consider the importance of communication between agents and the decision-makers involved in-group decision-making. Also, there is a possibility for managers from different domains to use the same DSS. As such, maintaining large data and model bases is very inefficient. The concept of distributed databases is factored into the system, to enable optimum use of database resources. This is achieved by viewing the GDSS as being essentially split into two parts: the container and the contents.

GROUP DECISION SUPPORT SYSTEMS

Many major decisions in organizations are made by groups. Getting a group together in one place and at one time can be difficult and expensive. Furthermore, traditional group meetings can take a long time, and the resulting decisions may be mediocre. A group support system is any combination of hardware and software that enhances groupware. GSS is a generic term that includes all forms of collaborative computing. GDSS evolved after information technology researchers recognized that technology could be developed to support the many activities normally occurring at face to face meeting like idea generation, consensus building, anonymous ranking, voting and so on.

There are several ways to classify DSS applications. The design process, as well as the operation and implementation of DSS, depends in many cases on the type of DSS involved. Here our studies concentrated database-oriented DSS in-group environment. In this type of DSS the database organization plays a major role in the DSS structure[1]. Early generations of databases-oriented DSS mainly used the relational database configuration. The information handled by relational databases tends to be voluminous, descriptive and rigidly structured. A database-oriented DSS features strong report generation and query capabilities.

Corresponding Author: K. Raja, Assistant Professor, Department of Information Technology, Sri Sairam Engg College, Chennai 600 044, India

DISTRIBUTED DATA MANAGEMENT

The complexity of most corporate databases and large-scale independent Management Support Systems (MSS) Data Bases sometimes makes standard computer operating systems inadequate for an effective and efficient interface between the user and the database. A Data Base Management System (DBMS) is designed to supplement standard operating systems by allowing for greater integration of data, complex file structure, quick retrieval and changes and better data security, to mention a few advantages.

Typically, in context of group decision support systems, distributed computer systems are characterized by the following:

- Computers are dispersed throughout a single organization or several organizations.
- Computers are connected by a data communications system.
- A common database is shared by all, but additional databases might exist.
- All computers are coordinated with an information resource management plan.
- Input and output operations are done within user departments.

The primary motivation for such a system can easily be determined from the following example. Consider a corporate DSS environment, where the managers for the Human Resources, Marketing and Finance departments etc., are brainstorming over some particularly dicey issue. If a large, common database is used, then its failure will affect all of them. However, the presence of a common corporate database and multiple shared departmental databases[2], with a model base for each will ensure that the session is only slightly affected.

CONVENTIONAL DISTRIBUTED G/DSS

Conventional distributed G/DSS involves multiple users connecting to the core DSS and then querying it after giving the necessary inputs. The rest of the data are called from the various distributed semantic units of the G/DSS.

The next step depends upon the age of the DSS. Older DSS used distributed computing to merely extend their capabilities, and perform only subsidiary tasks fulfilled on behalf of traditional, local decision support activities. More recently, the power of distributed computing has been harnessed to change the intrinsic behavior of the system. For example, an integrated electronic mail system can be used as a productivity-

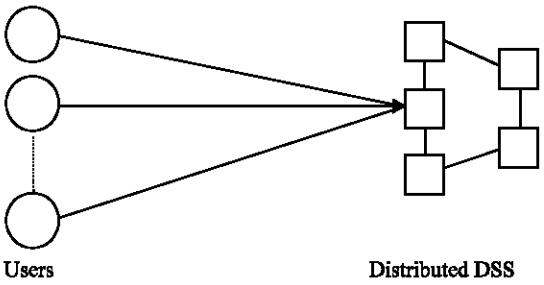

Fig. 1: Distributed DSS

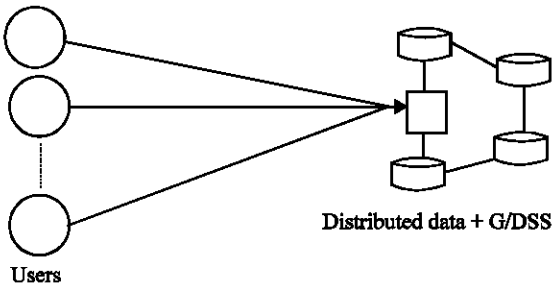

Fig. 2: Distributed data + GDSS

enhancement tool. Technological advances have now enabled multi-participant and Group DSS to avail of group productivity and communication tools, such as two-way interactive video, white boards, bulletin boards, chat systems, etc[3]. Such systems can be simply depicted in Fig. 1. The main disadvantage of such systems is that they are expensive to set up. Most organizations already have a G/DSS in place that meets their current needs. When they need to scale up to the advantages of decentralization, scrapping the entire existing framework is a colossal waste of money and resources.

DISTRIBUTED DATA + G/DSS

Instead, it would be much more convenient, especially to small and medium-size organizations, to adapt the existing system into a viable solution for their distributed computing needs. In this way, their previous investment can be protected, and they can still withstand the scalability requirements.

Such a solution can be effectively implemented by integrating a complete distributed database management system into the G/DSS. This results in the organization meeting their 'separate database for separate departments' goal, and achieving the same without the expense of setting up a full-fledged distributed G/DSS.

In Fig.2, the users connect to the G/DSS and enter their queries. The G/DSS then proceeds to perform the calculation locally, and retrieves the results using the distributed database management system.

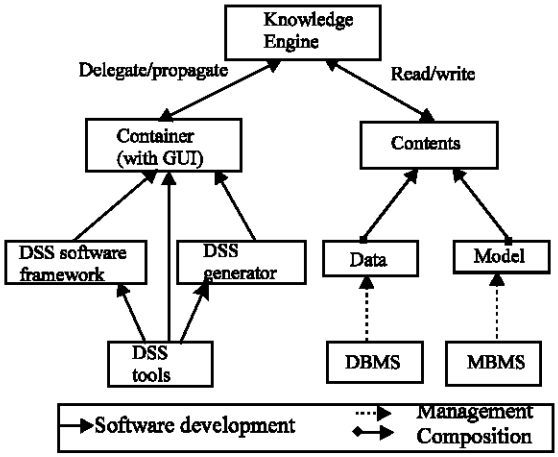

Fig. 3: KBGDSS

Such an approach is possible due to the bipartite nature of the approach to the G/DSS. The novelty of this bipartite approach lies in the clear and generic separation between the container of the DSS (responsible for the software engineering part) and the contents of the DSS (responsible for the knowledge engineering part) is shown in Fig. 3.

This bipartite approach decouples the development processes of:

- The purely technical components of the DSS
- The knowledge base of the DSS.

As the underlying technologies (for example, programming languages on the container side and modeling systems on the contents side) evolve in very different ways, this clear separation can greatly increase productivity when developing specific DSS.

The container: The container is responsible for the *system* part of the DSS. Its development follows the traditional life cycles described in software engineering. A typical development cycle for a container lasts a few months, up to a couple of years. Due to the dynamic and rapidly changing nature of programming languages and software engineering paradigms, containers are naturally subject to continuous, short-lived and deep changes in the underlying technology.

Separating the container from the contents raises many issues. To be successful, a container that should be reusable with different contents must definitely concentrate on generic decision support activities, such as specification of decision situations; definition of decision support tasks; generation of alternatives; analysis, documentation, and comparison of the alternatives; multi-criteria evaluation of the alternatives; support of the final choice using a preferences model, etc. The container should focus on these generic stages of a typical decision making process, rather than on specific details dealing with mathematical, relational, dimensional, document-, rule-or logic-based concepts.

The contents: The contents are responsible for the knowledge base of the DSS[4,5] and can be designed according to various paradigms (model-based, rule-based, document-based, data based, knowledge-based, etc.). Their development is evolutionary and follows the declarative methods of knowledge engineering. A typical evolution cycle for the contents can last a few years, up to a few decades. Well-defined contents should be extensible and reusable in different containers.

Contents represent critical assets for an organization. Knowledge bases often formalize many years of experience and expertise in the various domains of an organization. Contents are considered as reusable if they can be coupled with different containers. Unlike containers, the technology underlying contents development is less subject to rapid changes. Thus, as container and contents do not evolve at the same pace, they should definitely be loosely coupled. Otherwise, changes in the container technology could imply unnecessary changes in contents and vice versa.

Interface layer: At the highest level, the thin interface layer between the container and the contents is realized in the DSS component referred to as the knowledge engine. The knowledge engine is traditionally considered as the brains of the DSS. Of course, the knowledge engine then needs to provide handles to allow the container to communicate with it. This knowledge engine is able to retrieve the data and to compute model optimizations.

At a lower level, this interface layer between the container and the contents is realized in components called decision support objects. Decision Support Objects (DSO) can be informally described as reusable software envelopes placed by the container at the disposal of the contents. A DSO fits perfectly in an object-oriented software paradigm on the container side. It can easily be exchanged in a distributed environment. As a DSO remains closely related to the specifics of a decision making process. On the contents side, it encapsulates a business logic centered on decision support. The knowledge engine is the cornerstone of a bi-directional and loosely coupled communication between the container and the contents of the DSS, ensuring that the data, the models and the container remain dynamic components[4,5].

CONCLUSIONS

We have examined the issues of the various approaches-that exist today-to Group Decision Support Systems. Taking into consideration numerous aspects of these approaches, including the technical as well as the business side, we have arrived at an optimum solution to the implementation of such GDSS in small to medium-size organizations. This solution involves the partitioning of a Decision Support System into two constituents, and keeping the front-end traditional, while applying distributed concepts to the back-end. This will ultimately result in the organization exploiting the benefits of fully distributed DSS, while still maintaining their original configuration to a large extent. Not only is this simpler to implement, it is also a cost-saving measure that will crop organization expenditures.

This innovative approach still has a few limitations that should be dealt with in the future. First, it gives quite a bit of responsibility to the knowledge engine component, which has to be able to communicate with both the container and the contents of the DSS. Then, this approach relies on the traditional view of the knowledge base of a DSS, based on data and models. However, newer DSS paradigms, such as document-driven DSS, do not necessarily use data and models and it remains to be shown if our approach still holds for such paradigms.

REFERENCES

1. Turban, E. and J. E. Aronson, 2001. Decision Support Systems and Intelligent Systems. Addison Wesley Longman, Singapore, pp: 261-272.
2. Gachet, A. and P. Haettenschwiler, 2003. A decentralized approach to distributed decision support systems. J. Decision Sys., 12: 141-158.
3. Dube, L. and G. Pare, 2001. Global virtual teams. Communications of the ACM, 44: 71-73.
4. Shim, J.P., M. Warkenstein, J.P. Courtnry and C. Carlsson, 2002. Past, present and fututre of decision support technolgy. Decision Supp. Sys., 33: 111-126.
5. Gachet, A., 2001. A Framework for developing distributed cooperative secision support systems. Informing Science International Conference, Krakow, Poland, pp: 214-221.