The Case for Distributed Decision Making Systems

R. C. Thomas
Department of Computer Studies, University of Leeds, Leeds LS2 9JT, UK
A. Burns
Postgraduate School of Computer Science, University of Bradford, Bradford, UK

Decision support systems are described, together with previous work on how they support organizational and group tasks. A case study illustrates the need for several linked decision support systems in a manufacturing company and the nature of co-operation and conflict in organizations is discussed. The components and characteristics of Distributed Decision Making systems are then stated and justified. The possible advantages of using production systems to improve the explanations of decisions is considered. Plans for the future development of supportive software are outlined.

INTRODUCTION

One of the growth areas in commercial computing in the last decade has been the development of systems which support specific decision making processes, the resulting tool being called a decision support system (DSS). This paper briefly describes what these systems are, and then goes on to discuss how they fit into organizations. It is clear that a mechanism for co-operating decision support systems in an organization does not yet exist, even though networking is now becoming common. The organizational needs from such a mechanism are introduced through a case study, and we then go on to describe what we call Distributed Decision Making (DDM) which is an extension of the DSS concept. A key point to emerge is that DDM must have the capacity to deal with the conflict which inevitably arises between users of several decision support systems.

DECISION SUPPORT SYSTEMS

Decision support systems have their roots in MIT's project MAC. They represent a loosely defined area of study which has gained considerable momentum in the last few years. For the purposes of this paper, decision support systems are interactive systems which aim to support, rather than replace, managers in the execution of semi-structured tasks. Semi-structured tasks entail work which is neither so well understood that all the rules are known and can be automated, nor so judgemental that there is no prospect of providing computer assistance. Thus, it may be quite feasible to get the computer to perform some function, such as plot data A versus data B, but it is then up to the manager to make a decision on the importance of this plot. An authoritative work on decision support systems is that of Ref. 1.

A difference between conventional software engineering and DSS development is that for a DSS it is often not clear when, if ever, the project is finished. The reason for this is that the designer and user continually stimulate each other to do more and more as they learn about the application. It can be seen, therefore, that the construction of a DSS is a highly creative activity which can be difficult to manage. The problem is compounded when it comes to evaluating a DSS because it is aimed at improved decision making rather than replication of the decision making process. The benefits are intangible but sometimes substantial.

An early but typical DSS is described by Scott Morton.[2] It concerns the operation of the laundry equipment division of one of the largest corporations in the USA. Every month a production and sales plan for the next 12 months was created, which involved thousands of calculations and subjective evaluations of sales trends and similar data. The plans had to fit marketing and production constraints and objectives. Marketing aimed at flexibility with ample stocks of product, whereas Production wanted to minimize stockholding and manufacturing costs. The people involved in this planning had the basis of conflict in these differing goals. A DSS was constructed to support this planning process. It performed many of the calculations and presented much of the data graphically. The result was that it became possible to explore many alternative plans in an atmosphere of co-operation rather than conflict. It also enabled the decision making cycle to be structured by the problem, rather than by the limits of the data-manipulation process. The computer supported rather than replaced the monthly decision making process.

Keen and Hackathorn[3] argue that there are really three components of a Support System: '(1) Personal Support (PS) focuses on a user or class of users in a discrete task or decision (e.g. setting a price, selecting a stock) that is relatively independent of other tasks. (2) Group Support (GS) focuses on a group of individuals, each of whom are engaged in separate, but highly interrelated, tasks (e.g. office activities). (3) Organizational Support (OS) focuses on an organizational task or activity involving a sequence of operations and actors (e.g. building a divisional marketing plan, capital budgeting).' They observe that '... several of the most successful applications of decision support involve GS and, more recently, OS'. They are speaking, though, of several people co-operating around one DSS rather than of interactions between several decision support systems each of which supports a single user or group.

In order to explore in more depth the use of a DSS to

CCC-0010-4620/82/0025-0148$02.50

provide GS and OS, we now present a case study and then go on to discuss how DDM might augment the decision making process.

CASE STUDY

Part of the organization chart of the major division of a multinational manufacturing company is shown in Fig. 1. The division has a number of factories in the UK, the largest employing several thousand people. There is a Managing Director who has Production, Engineering and Marketing Directors reporting to him. Being a market oriented company, new product opportunities are explored by Marketing which may eventually put a case to the Board of Directors to introduce a new product line. This will generally involve a pilot launch, in a small region of the country, selling products made on an experimental production line with a larger than usual labour force. This is an expensive operation. Once a product has been launched successfully, costs can be reduced by building a special-purpose production plant. The time taken to do this, a year or more, has a strong influence on the product's prospects.

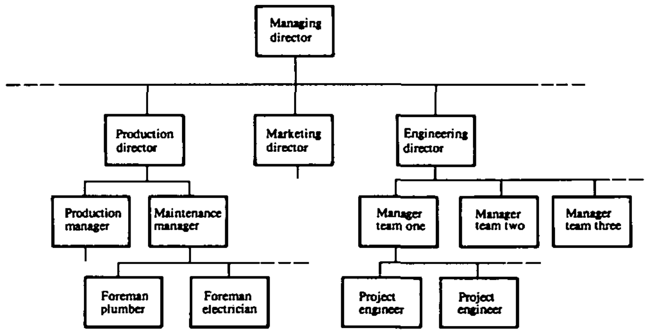

Figure 1. Divisional organization chart.

The design and project management of a new plant is the responsibility of the Engineering Director. The actual labour used (electricians, plumbers, etc.) is, however, drawn from the maintenance force which reports to the Production Director, whose prime task is to keep existing lines working and only then to introduce new plant as required. Thus there is the basis of OS for the decision making involvement of the Marketing, Engineering and Production departments, i.e. we have a sequential process. Marketing realizes the need for more plant; Engineers submit to the Board plans for the plant; Engineers and Maintenance install it; Production use it. There are also present the seeds of conflict because a new plant is usually required quickly, and yet the resources to implement the development could easily be engaged on maintenance or the installation of other plant. Even if the priorities of the twenty or so major engineering projects are accepted by all concerned, conflict will arise in the day to day deployment of the maintenance force because of unforeseen circumstances such as the plant breaking down. Regular daily GS is needed here in order to integrate people whose tasks conflict with, but in some sense depend upon, each other.

The company's data processing service was run as a part of head office. The larger of the two available mainframes provided a teleprocessing service to local and remote users, RJE links for the distribution division, various batch facilities, an APL service and a database system. Systems analysis and programming were separate departments. The analysts were divided into teams along functional lines, e.g. wages or management accounting. The APL facility was new and was mainly used by the operational research (OR) unit. There was a history of projects starting in this unit and being transferred to the analysts once the distinction between the OR and DP components became clear.

The company had systems to help marketing, purchasing etc., and effort was being expended to improve the range of services offered to the maintenance and engineering departments. It so happened that one of the engineers had seen and liked a bureau's project management system based upon the critical path analysis model. The OR department was called in, and it recommended that it would be cheaper to install an in-house system, which the engineers accepted.

Development time was crucial because the bureau could supply a package, via a terminal, within days. It was therefore decided to use APL, as it was believed to cut development time dramatically and there was a program available which provided a starting point. Without realizing it, the DSS bandwagon had been boarded. The user and programmer sparked each other off and the resulting system was very user friendly. It was apparent, however, that an APL system might be very good for a single project, but it was questionable whether it would handle 200 or even the 20 largest projects. As a precaution more conventional packages were examined, but it became clear that they were inferior from the user's point of view and that they would not easily fit into the company without restructuring the way projects were controlled.

The APL DSS was used quite heavily by two project planners. One created a very simple plan concentrating on the major phases of a project in a remote factory and he made sure that the overall structure was agreed by all concerned at the fortnightly project control meetings. The other went into great detail and used his personal contacts in the nearby maintenance department to draw up intricate plans for each week, a task which was eased by the addition of extra facilities in the DSS. Neither approach was clearly better than the other, and comparison was particularly difficult as the projects were so different, but both planners could claim substantial success. After a year, however, the system was hardly used at all. A slowdown in investment was partly to blame, but the overall inadequate OS and GS between the planner and the other parties on the project was a significant factor. There was conflict in both cases but it was resolved differently. For example, in the remote factory project, people could examine the plan for the fortnight and agree changes during the meeting, if targets were unrealistic due to conflicts with previous commitments. In the other example, maintenance foremen tended to resolve conflict amongst themselves, and the planner would actively update the plan to reflect what was expected to happen.

A factor which made the OS possible was the small size of the factory. The maintenance manager was in a position to know all the facts. The GS was, perhaps, the only practical approach in the other factory because the organization was so large. It can be argued that a DSS could have been constructed to overcome this size problem, but it would have needed information on the engineering projects, which would have come from the DSS we were discussing for the engineers, which needs data from the maintenance DSS . . . Thus the decisions of the project engineers are influenced by the decisions of the maintenance foremen and vice versa. It might be considered that this could be overcome by giving the Engineering Director his own labour force. Then there would have been interaction between different projects and between the decisions of the marketing department on project priorities. Similarly, there will always be links between the Project Engineer and Production when it comes to commissioning plants. It is clear that this co-operation is very often involved in sharing resources. Well managed organizations do not have much spare capacity, so there will be conflicting claims on the use of these resources. The case study has indicated that co-operation is essential, and that this involves the resolution of conflict. These points will be considered again later, but let us now turn to the problems of implementing computer systems in this organization.

Implementation of DSS and DP projects

Over a two or three year period various DP systems, such as engineering spares control or plant register, have been implemented to improve the efficiency of the areas under discussion. Decision support systems have been built for both Engineering and Maintenance, although the latter was developed almost completely independently of the former and is concerned with preventative maintenance rather than plant installation. Are there guidelines for creating such a range of systems?

In principle, the design and implementation of the DP systems is no particular problem. Top down methods can be employed since purchasing, inventory control, etc. are well understood. These systems can then feed the decision support systems which in turn can interact with each other. This makes the unlikely assumption that the users of the decision support systems are prepared to wait till the DP systems have been implemented. Alternatively, it is possible to implement the various decision support systems, which would then be enhanced when the DP systems became available. The problem here is that a DSS may need the data from the DP systems to be at all effective. What is not possible is to treat the DP and decision support systems as one unit to be designed and implemented together in classical top down fashion. The evolutionary nature of DSS development prohibits this. What is needed is to be able to install a 'co-operation grid' into which we can plug the various decision support systems as they evolve. The DP systems can plug into the decision support systems directly, as and when required, see Fig. 2.

There is another reason why it is necessary to adopt a flexible design and implementation approach. The problem of 'improving the way new plants are installed' is not well understood. It is very likely that analysis, partial reorganization and a better supply of information would all contribute towards the general goal of improved

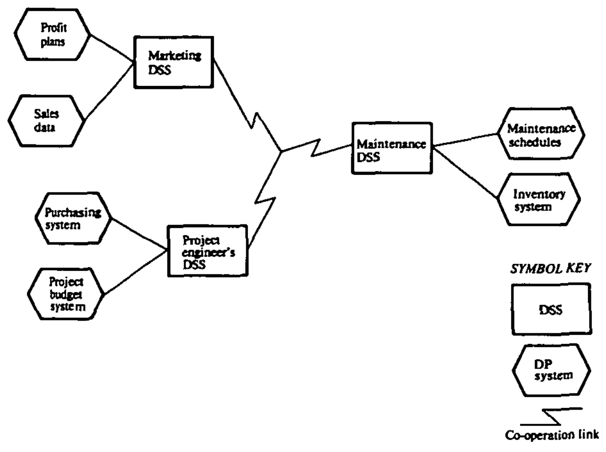

Figure 2. Co-ordinated DSS and DP systems.

effectiveness. The trouble is that it is not possible to say how much each tactic would help, or how they would interact. This makes it difficult, if not impossible, to design a better organizational decision making system, starting at the top and working down. In Ackoff's[4] words, we should 'design a desirable future and invent ways of bringing it about'. We hypothesize that a desirable future would include better co-operation and co-ordination between various parts of the organization, and would therefore propose to install a DDM system to bring this about. Actually, this was realized at the time conventional project planning packages were being examined. Unfortunately, it was evident that the company's system software, although modern, was not easily able to support it, nor was it practical to develop a DDM system during a project.

CO-OPERATION AND CONFLICT

An organization may be considered as consisting of a collection of parts between which there may be information flow. Here, we are not concerned with establishing a general information feedback model, rather, we accept that entities within an organization do *de facto* co-operate with each other, with more or less equal authority, at least for particular tasks. It is also accepted that the decisions of any one part will affect others, and that compromises have to be reached. This is in keeping with the work of Lawrence and Lorsch,[5] who have found that effective organizations show both high differentiation and high integration between business functions. Integration requires co-ordination between various levels of the hierarchy not just at the top.

In the case study, considerable differentiation existed between the project engineers and the tradesmen. They were not, however, well integrated. Most of the time they could work completely independently of each other, but efficiency dropped when each other's resources were needed. To some extent this was overcome when the critical path analysis system (CPA) was installed. In true DSS style, the critical path analysis system evolved from being a calculating aid to being one which provided genuine group and organizational support, within the engineering department, over the planning of individual projects. It was of limited value, however, when planning actual equipment installation, because the decisions of the trades foremen were important but were not fully incorporated into the CPA. Conversely, it would be hard to create a DSS for a foreman, without involving the engineers. Each party, therefore, needs to know about the other's decisions. When they are not acceptable to each other, there are two ways of resolving conflict. Either take it to a more senior person (which one, the MD?) or sit down and explain *why* the decisions are as they are and attempt to reach a compromise. The latter is often effective because co-operation almost always improves when reasons are given for a decision. Explanation, then, should be a feature of DDM. Keen[6] observes that decision support systems '... immensely ease the efforts required in explanation' and wonders whether '... joint problem solving may be facilitated by having at a key organizational node ... a vehicle that permits sharing of ideas, analysis, and explanation of logic'.

Having thus explored the basis of co-operation and conflict amongst decision makers with or without associated decision support systems, it is now possible to consider DDM in detail.

DISTRIBUTED DECISION MAKING SYSTEMS

Requirements

For the following, the term *node* is used to mean a point in an organization at which some decision making process occurs. From the above discussion of co-operation and conflict, it can be seen that a DDM system needs to meet the following requirements: (i) to support decision making by many people, in separate, but interrelated, nodes of an organization; (ii) to provide mechanisms for communicating relevant decisions amongst nodes; (iii) to enable decisions to be explained, and support the resolution of conflict between nodes; (iv) to allow for the evolutionary development of the system as a whole and of the decision support systems within it; (v) to recognize that the members of a system act as a peer group rather than as a hierarchy.

Components of DDM systems

The above requirements lead directly to the components that need to be available in a DDM system: (i) Support for individual decision makers needs a DSS style system, which could consist of operators (e.g. APL functions corresponding to LIST, PLOT, etc.), a database, and a software interface between them and the user. Evolution of the system consists mainly of adding new operators. (ii) Communication between nodes is provided at two levels. First, there is a need to establish a connection between two nodes and then negotiate for facilities. Second, it is important that data and operators are seen as being owned by individual nodes. For any external node only the function of the operator is visible, the coding and the data itself are not automatically available. Communication is therefore structured around the access of operators that are 'exported' from nodes. (iii) Very often conflict arises between decision makers simply because they have different perspectives of the same problem. In these cases, explanation can take the form of providing more detail on pertinent constraints than just the decision in question. For example, if a particular request for a plumber cannot be fulfilled because of the high level of sickness, then a response of the form 'plumber not available because three people are off sick this week' is likely to satisfy most reasonable requestors. This type of facility could probably be implemented in the form of special purpose operators. Sometimes explanations will need to be far more detailed: 'you cannot have a plumber this week because three of them are on essential maintenance tasks, four are on higher priority projects, and one is on a critical job on a low priority project. From the information you have given, your project will not be delayed if you do not have a plumber this week'. This type of explanation is more difficult to provide. It needs an understanding of the rules used by the foremen when allocating plumbers. (iv) Evolutionary development of DDM is possible because new nodes can be added without disrupting existing ones, and the same applies to operators within a node. Unilateral deletion of nodes and operators is possible, as the DDM system supports the activities of people who are entitled to stop using the computer if they wish. (v) The nodes would be loosely coupled together in a network. All possible pairs of connection would be needed, but not simultaneously. Sequential decision making (OS) would involve sending a decision from one node to others in a required order. Pooled decision making (GS) is not so easy as it is usually necessary for participants to obtain an overall picture of the constraints: it is possible, therefore, that Keen's[6] key organizational node concept might be employed for supporting *specific decisions*. These requirements therefore imply a flexible network topology which changes to meet required goals.

Relevance of production systems

The requirement to explain decisions hints at the use of production systems, which were developed as a branch of artificial intelligence (see Ref. 7 for an overview). Programs are expressed in the form of a series of rules of the type IF condition THEN action, which are 'executed' in the following way: all the rules are evaluated to see which conditions are true at that time; an algorithm selects a true rule and performs its action thereby altering the truth of other conditions; the cycle is repeated. A consequence of this is that the system can explain not only what it is doing but why! Production systems are employed in expert systems, of which the most famous is MYCIN,[8] a system for diagnosing certain diseases. Medical people can read its 'program' in a style which reflects their medical, rather than computer, background. Another, called RITA, can be programmed to have limited negotiating powers.[9] Suppose one wished to organize a weekday seminar, then one would instruct RITA to contact possible attendees and canvass their views. RITA would then report the recipients' responses to the meeting organizer who would then make a decision on the timing and inform the others through RITA. In the context of the case study this sort of system appears to have potential for tasks such as arranging to have tradesmen available to help in the installation of a section of plant.

CONCLUSIONS

The case for DDM has been stated. When making decisions there is a need to provide organizational and group support and to recognize and resolve conflict. These needs can be fulfilled by increasing the co-ordination and co-operation amongst decision makers using the DDM mechanism. The components and characteristics of DDM systems have also been stated. They represent a departure from mainsteam thinking on commercial computing in that: (i) evolutionary methods are recommended; (ii) the system development as a whole is not orientated towards specific functional goals; (iii) there are some unusual facilities such as explanation of decisions.

There are many possibilities to consider in the construction of the nodes. Smith[10] has suggested that APL could be used, but there is a case for using production systems with their power of explanation and potential for user 'programming'. The implementation vehicle needs to provide concurrent processing, internode communication via operators, and secure data and code. ADA and some other languages provide these. DDM systems could have interesting stability problems if decisions are made and then reversed. It would be necessary to determine which sort of rules begat stability, and what the requirements are of the channel to the actual decision makers who would have to resolve the stability problems.

There are potential problems associated with the complexity of a DDM system. How would one node know about the existence of another? In manual systems much of this type of knowledge is held informally by people, but in a computer system conscious decisions would need to be taken to relate together the informal and formal worlds without excessive complexity.

In order for practical DDM systems to be constructed, supportive software is required. The exact nature of this software is still open to theoretical conjecture, but we intend to produce a definitive description of a DDM system on which we will base future work. We will then be in a position to compare the provisions of current software with the demands of our model, and thence to make recommendations for the development of additional features to graft onto existing software. We would hope to construct practical systems for mainframes and networks in due course, and would welcome further suggestions and comments from people with potential applications.

REFERENCES

1. P. G. W. Keen and M. S. Scott Morton, *Decision Support Systems: An Organisational Perspective*. Addison-Wesley, Reading, Massachusetts (1978).
2. M. S. Scott Morton, *Management Decision Systems: Computer Based Support for Decision Makers*. Division of Research, Harvard (1971).
3. P. G. W. Keen and R. D. Hackathorn, *Decision Support Systems and Personal Computing*. Sloan School of Management Working Paper 1088–79, Cambridge, Massachusetts (1979).
4. R. L. Ackoff, The Future of Operational Research Is Past. *Journal of the Operational Research Society* **30**, 93–104 (1979).
5. R. R. Lawrence and J. W. Lorsch, *Organisation and Environment*, Division of Research, Harvard (1967).
6. P. G. W. Keen, *Decision Support Systems and the Marginal Economics of Effort*. Sloan School of Management Working Paper 1089–79, Cambridge, Massachusetts (1979).
7. R. Davis and J. King, *An Overview of Production Systems*, Artificial Intelligence Laboratory Report STAN-CS-75-524, Stanford University, California (1975).
8. E. H. Shortliffe, *Computer Based Medical Consultations: MYCIN*. Elsevier, New York (1976).
9. D. A. Waterman, A Rule-Based Approach to Knowledge Acquisition for Man-Machine Interface Programs. *International Journal of Man-Machine Studies* **10**, 693–711 (1978).
10. A. C. D. Smith, *APL—A Design Handbook for Commercial Systems*. Wiley, Chichester (to be published).

Received August 1981