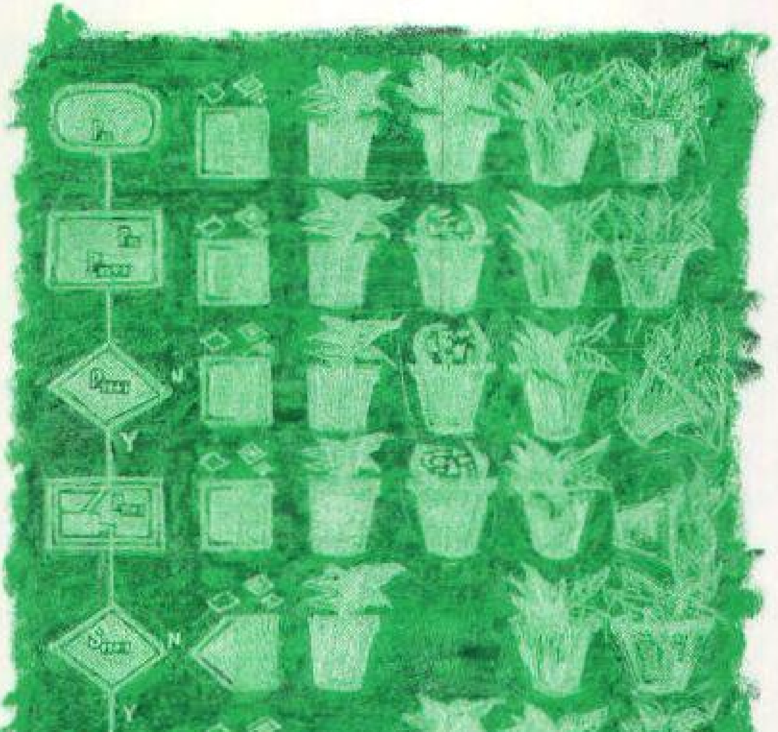
Modesty in Modelling

on the Applicability of Interactive Planning Systems

with a case study in Pot Plant Cultivation



Gert Jan Hofstede

Propositions

with the Doctoral Dissertation

Modesty in Modelling

Gert Jan Hofstede

20 May 1992

1 This study's findings about decision support systems for tactical planning can readily be generalized to non-agricultural domains.

This dissertation.

2 Designers of planning systems serve the users best if they take on the role of intermediaries between problem owners, modelling techniques and information technology.

This dissertation.

3 If one adopts a pragmatic point of view on Operations Research techniques, heuristic algorithms are often not less efficient but, on the contrary, more efficient than optimizing ones. This holds in particular for heuristics which allow the user to intervene at any time during each iteration.

Müller-Merbach, H. (1981), "Heuristics and their design: a survey." *European Journal of Operational Research* 8, pp. 1-23.
Chapter two of this dissertation.

4 In the design of planning systems, the chances of producing a system that is valued by users are highest if the first step is the development of a user-system interface that is understood and accepted by the user.

Chapter three and four of this dissertation.

5 Mitroff's 'Systems View of Problem Solving' is of limited practical use as a conceptual tool during the development of automated support for tactical planning situations.

Mitroff, I.I., F. Betz, L.R. Pondy and F. Sagasti (1974), "On managing science in the systems age: two schemas for the study of science as a whole systems phenomenon." *Interfaces* 4:3, pp. 46-58.
Chapter four of this dissertation.

6 Because of the close association of theoretical, technical, biological and social disciplines, combined with a social involvement and a practical attitude, Wageningen Agricultural University is preeminently suited to help cope with contemporary problems related to man, society and environment.

7 Educational institutions offer pupils exercises in solving formal rather than open problems. This practice is harmful to future DSS developers.

> Ackoff, R.L. (1978), *The Art of Problem Solving*. New York: John Wiley.
> This dissertation.

8 Working groups of peers are very beneficial to the progress of researchers.

> TIPSY (1990) (ed.), *Promovendidagen Informatiesystemen 1990*. Delft: Technical University, department of Information Systems.
> TIPSY (1990), *De Contouren van een Onderzoeksnetwerk, Verslag Derde Promovendidagen Informatiesystemen*. Den Haag: Werkgemeenschap Informatiesystemen SION.
> TIPSY "Problemen met Planningsproblemen." Forthcoming in *Informatie*.

9 The properties that qualify women for leading positions in present-day society are the very ones that keep them away from these positions.

10 Ofschoon zij juist is wekt stelling VI uit het proefschrift van Pieter Bots ten onrechte de suggestie dat de voornaamste beperkingen van ontwerpers van informatiesystemen van cognitieve aard zouden zijn.

> Bots, P.W.G. (1989), *An environment to support problem solving*. Doctoral Dissertation, Delft: Technological University, department of Information Systems.

11 Ten onrechte behoren noch de vakgroep Informatica, noch de werkgroep en afstudeerrichting Bestuurlijke Informatiekunde van de Landbouwuniversiteit, tot de *enacted environment* van de nederlandstalige Informatica-wereld.

> Weick, Karl E. (1969), *The Social Psychology of Organizing*. Reading, Massachusetts: Addison-Wesley.
> *Informatie* 30 (1988), themanummer 'Onderwijs in de informatica en informatiekunde in Nederland en Vlaanderen'.
> *de Volkskrant* (1991), 31 oktober, advertentie Universiteit van Amsterdam.

12 Het besluitvormingsproces bij benoemingen van hoogleraren aan de Landbouwuniversiteit zou gediend zijn met deelname van de betrokken vakgroep in de benoemingsadviescommissie.

13 Als het broeikaseffect doorzet verliest de nederlandse glastuinbouw elke grond.

Modesty in Modelling

on the Applicability of Interactive Planning Systems

with a Case Study in Pot Plant Cultivation

Promotoren: dr ir J.L. Simons
hoogleraar in de Bestuurlijke Informatiekunde
aan de Rijksuniversiteit Groningen

ir M.S. Elzas
hoogleraar in de Informatica
aan de Landbouwuniversiteit Wageningen

Gert Jan Hofstede

Modesty in Modelling

on the Applicability of Interactive Planning Systems

with a Case Study in Pot Plant Cultivation

Proefschrift

ter verkrijging van de graad van doctor
in de landbouw- en milieuwetenschappen
op gezag van de rector magnificus,
dr. H.C. van der Plas,
in het openbaar te verdedigen
op woensdag 20 mei 1992
des namiddags te vier uur in de Aula
van de Landbouwuniversiteit te Wageningen

Hofstede, G.J. (1992). *Modesty in Modelling: on the Applicability of Interactive Planning Systems, with a Case Study in Pot Plant Cultivation.* Doctoral Dissertation. Wageningen: Agricultural University, Department of Computer Science.

keywords: decision support systems, tactical planning, cooperative development, interactive heuristics, horticulture

Printed by Thesis Publishers, Amsterdam.
Cover illustration by Mascha Wasserman.

**THESIS PUBLISHERS
AMSTERDAM 1992**

Wageningen Agricultural University
Department of Computer Science
Dreijenplein 2, 6703 HB Wageningen, the Netherlands
e-mail: hofstede@rcl.wau.nl
telephone: (+31)8370.84630

Levin had that winter begun writing a book on agriculture,
the idea of which was that the temperament of the agricultural labourer
was to be treated as a definite factor, like climate and soil,
and that therefore the conclusions of agronomic science
should be deduced not from data supplied by soil and climate only,
but from data of soil, climate, and the immutable character of the labourer.

Leonid Tolstoy (1878 / 1978, p. 168)

Contents

Preface

> There is nothing here on the subject of friendship itself. But to discuss in public what has hitherto been the most important thing in my life is, I have found, quite impossible. I beg my friends' pardon, if not my readers'.
>
> Bernard Levin (1979, p. xix)

An anthropologist travels among a number of peoples unknown to him, makes a broad study of each, and writes down his observations. A member of one of these peoples reads the account. What does he see? The first thing that strikes this reader is that some details about his own people are inaccurate and others downright erroneous. Reading on, he gets amused, thinking "I've never thought about it that way". The traveller's account puts familiar notions into a wider perspective by holding them against practices in other cultures.

This book can be seen as a metaphorical travel through the (inter-)disciplines of Information Systems, Operations Research, Systems Theory, Psychology and Organization Theory. The 'anthropologist' is really a population biologist who underwent a brief period of brain-washing in automation. The book is intended for any reader who has a background in one of the disciplines mentioned. It is my hope that the book has upon the reader the kind of effect the anthropologist's account had on its reader.

Putting the metaphor aside, I believe the book can act as an eye-opener and provides useful entries into the disciplines covered to students, researchers, developers and even users of information systems. Likewise, the book contains some fairly practical ideas for those who deal with tactical planning situations, in particular for planning system designers.

A great many people have contributed in some way or another to this book. My thanks go to all. I wish to mention a number of them by name. During 1986 and 1987 I developed some inarticulate ideas about 'DSS that are adapted to their users'. At a doctoral consortium in Delft in December 1987 Henk Sol told me that my ideas were good enough and that all I needed to do was to find a sounding board. Some of those that helped me focus my views during these early months were Aart Bosman, Hans Crombag, Henk Sol and Charles Vlek. In March 1988, John Simons and Maurice Elzas agreed to act as promotors and the study formally began.

Preliminary work which had led to the PROPLAN project had begun as early as 1985, when Peder Jongejan experimented with the pentomino environment. Since then a fair number of students, some of whom made a substantial contribution, have participated in the PROPLAN team for three months or more.[1]

The PROPLAN project brought me into contact with a number of people in horticulture, most of whom have been very helpful. I wish to thank in particular Gérard den Hollander, Frits van Horssen, Jos van der Knaap, Ad van Nuenen, Eric Persoon and Wim van Rijn for the time, valuable in at least two ways, which they granted me.

I have cooperated with fellow researchers in the research programme 'DSS in agriculture and horticulture' from 1985 on and hope to carry on this work. I had numerous discussions with Eligius Hendrix and Bert Annevelink. Jan Bots allowed me access to unpublished results. It is a pleasure to thank Jan van Niejenhuis for allowing me to work with his students during three consecutive years.

I am especially indebted to Jan Ockeloen, who programmed PROPLAN HAND. Thanks go to Rob Hartog for many an *impromptu* discussion and for his helpful comments on the formal model of the placement problem. It has been a pleasure to work with Adrie Beulens on an article which I used in Chapter three of the book.

It was not always easy to remain motivated when I was still groping my way around, and while other obligations were making demands on me. But each time I attended a conference, a workshop or a doctoral consortium I returned full of enthusiasm. During 1991 I was strongly motivated by discussions and experiences at the ICIS 1990 Doctoral Consortium and Conference in Copenhagen. The closest and most rewarding research circle in which I have participated, and hope to continue participating in the future, is the one formed by Henk Akkermans, Pieter Bots, Richard de Jong, Dieta Mietus, Peter Verbeek and Alexander Verbraeck: in one word, TIPSY.

The actual writing of this book was a process of its own. I could never have done it alone. Josephie Brefeld and Rokus Hofstede gave me valuable suggestions for changing the structure of the book to make it readable. Checks on contents, syntax and style were performed by Adrie Beulens, Geert Hofstede and Sjoukje Osinga. Klaas-Jan Leutscher checked the 'introduction to pot plant cultivation'. Peter Verbeek commented on SCAPSIS. Time and again, Jan Ockeloen drew the pictures in this book. Masja Wasserman struggled with chaos and structure and pot plants to design the cover. The English on almost every single page of the book has benefited from Josephie Brefeld's red pen. She denied the English language many a new word, but then again her corrections and suggestions made the text much clearer.

[1]In what is roughly a chronological order the PROPLAN students during the years 1986 to 1990 were: Luit Rietema, Hans Vester, Bert Huijskes, Wim van Lemmen, Yunus Aliskan, Mario de Deugd, Henk Compagner, Joop Wijdeven, Cor van der Vleuten, Bart Lebbink, Karst Ziel, Aad Vollebregt, Caroline Schropp, Ewoud Endlich, Ronald de Ruig, Jan Hof, Lyda Dik, Henk Doeven, Jan Schrama, Rudolf Beekman, Astrid van Triet, Hans van Bohemen, Martien van Nieuwkoop, Arend-Jan Both, Wilbert Wouters, Adri van Doesum, Rita Karsten, Teunis Biemond, Reinoud Visser, Riena Tienkamp, Ron Wunderink, Ben Tubben, Jaco Wisse, Serge Loosveld, Martijn de Vries, Jan-Willem Reytenbagh, Ingrid de Groot, Flip ter Horst, Dick Brinkman, Willem van Breukelen, Serge van Dellen and Roel Veerkamp.

Two people were involved throughout in the creation of this book, and helped make it a rewarding activity. They are my promotors. In 1985, when I first entered the department of Computer Science of Wageningen Agricultural University, Maurice Elzas told me there was a need for generalists in Information Systems. Since then he has enabled me to work at becoming one by supporting my metaphorical travel. My closest companion during this journey, who light-footedly kept me on course, was John Simons. Many of the ideas expressed in this book owe at least as much to him as to me.

Hopefully, the contributions of all these people will not be wasted. This book asks to be read and discussed.

Wageningen, March 1992

1 Introduction

"Hallo!" said Piglet, "what are *you* doing?"
"Hunting," said Pooh.
"Hunting what?"
"Tracking something," said Winnie-the-Pooh very mysteriously.
"Tracking what?" said Piglet, coming closer.
"That's just what I ask myself. I ask myself, What?"
"What do you think you'll answer?"
"I shall have to wait until I catch up with it," said Winnie-the-Pooh.

A.A. Milne (1926 / 1978, pp. 31-32)

Like Pooh's hunt, this study started very much in an open-ended way. It was supposed to be a study on automated support for decision making in organizations, in particular for tactical planning situations. When the study started, my only certainty was that it would contain a case study which would have to result in an automated system for assisting pot plant growers in cultivation planning.

Before discussing automated support, decision making in organizations, or tactical planning situations, and also before elaborating on the research questions and goals of the study, I would therefore like to introduce the reader to pot plant cultivation.

1.1 An introduction to pot plant cultivation

Pot plant cultivation in the Netherlands

Pot plants are plants cultivated in plastic or earthen pots. They often end up on windowsills, where they are put for decorative purposes. Either the leaves, the flowers, or both, make out the decorative value of a pot plant. The Netherlands are among the world's first pot plant growing nations. There are about 800 pot plant nurseries, producing 1,500 million Guilders'[2] worth of plants per year in 1990, a large share of which is

[2]The Dutch Guilder was worth about half a U.S. Dollar in 1990. It will be used as the currency unit throughout this study and is indicated by the symbol *f*. The symbol ',' is used as a separator in the notation of large numbers, whereas the symbol '.' is used to delimit the fractional part of a number. For instance, *f* 1,500 stands for one and a half thousand Guilders.

exported. In the Netherlands, pot plants are cultivated in greenhouses that can be quite large and high-tech. Most nurseries are still run by one person, who is often assisted by family members, but this situation is changing rapidly (Kruijk 1988). A new type of nursery, with division of labour, extensive process automation, and up to forty employees, is spreading. Size and profitability of pot plant nurseries are strongly correlated.

No two pot plant nurseries are the same. Consider for instance the fact that there are over a thousand different registered products, which is more than the number of nurseries. Almost all of these products demand a specialized production process. Bots (1991, p. 27) gives an impressive enumeration: the products differ in the number of cultivation phases, in density per phase, in phase length, in the number and nature of manipulations, in the climate conditions they require by day and at night, in light-, water-, nutrition requirements, in treatments against diseases, in growth regulation, in way of reproduction. Seemingly insignificant details can alter the aspect of a pot plant to such an extent that it is considered a different product. Furthermore, it should be noted that new products are introduced every week. No wonder that hard data about pot plant product cultivation, such as growth curves, are sparse.[3]

Rate of innovation

Pot plant culture is different from most agricultural sectors by virtue of the tremendous rate of innovation and the independency of the individual pot plant grower as an entrepreneur. Successful operation of a large pot plant nursery depends to a large extent upon capacity for innovation (Alleblas 1987). Initiatives by pot plant growers include setting up nurseries in Brazil or Spain, planning labour peak periods during school holidays, creating a brand name or a new product, setting up their own genetics research, following Paris fashion magazines to determine what colours to cultivate, etcetera. All this goes to show that pot plant growers are alert and active in many domains.

A look at a pot plant nursery

What does a pot plant nursery look like? From the outside, one sees a glasshouse of a size averaging one hectare. Figure 1.1 gives the floor plan of a hypothetical nursery. Usually a nursery is not as straightforward as the figure shows; there may be several greenhouses of differing age, size and technical cultivation system.

The plants can be grown either in the ground, on concrete, on fixed tables, on tables that can be rolled aside, or on automatically transportable tables ('containers'). In the last case the plants are transported to a central location for each manipulation. The greenhouse is highly automated: opening and closing of windows, regulation of CO_2 level, of light, temperature, and watering are usually all fully automated. Yet there is still a lot of work to do: manipulations like making cuttings, planting cuttings, and preparing the finished products for transport can hardly be automated yet for a vast majority of products.

---

[3]For a more extensive but still comprehensive overview of the sector the reader is referred to Bots (1991). Leutscher (1991) gives a more elaborate treatment.

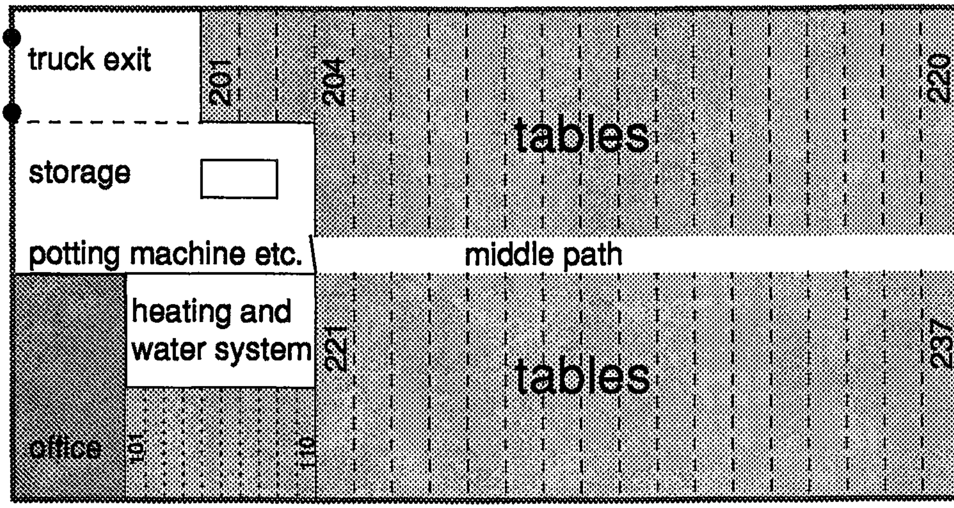


Figure 1.1 Floor plan of a very simple pot plant nursery (scale: 1 mm = 1 m)

Tactical planning

Tactical planning[4] in a pot plant nursery means cultivation planning. During cultivation planning the grower takes a number of decisions. First he must decide which products to cultivate during the next planning cycle, which usually means during the next year. This includes deciding on the cultivation process. Then he has to establish how much of each product to pot during each planning period. Most growers work with planning periods of one week; therefore the word 'week' rather than 'period' will be used henceforth.

A cultivation plan has some dependent plans. These may include a financial plan (specifying costs, sales and liquidity), a labour plan, a space plan (specifying the occupation of each greenhouse location during each week) and a materials plan. The tactical plan consists of both the cultivation plan and the dependent plans. The extent to which the different constituents of the tactical plan are formalized differs greatly from grower to grower. Usually the time spent on planning increases with the size of the nursery. Some growers are strongly opportunity-oriented, and do not use a cultivation plan. Others just have no interest in planning. They see themselves as growers, not managers, and trust their 'green fingers'. Yet others, more production-oriented, make a detailed cultivation plan and adhere to it strictly. For flowering plants the selling date tends to be more important than for green pot plants. For instance, in the Netherlands

---

[4]The notions of tactical, strategic and operational planning are taken from Anthony (1965). They will be explained in section 1.2.

'Mother's Day', the second Sunday in May, is a favourite occasion for offering flowering pot plants. Thus a grower of flowering pot plants has a strong interest in working towards a sales peak just before Mother's Day.

The tactical level has strong links with the strategic and operational levels. An example of the link between strategic and tactical planning is that a new greenhouse, clearly a strategic investment, is designed for accommodating the intended cultivation plan. On the operational side, the knowledge of the grower about the flexibility of constraints is quite important. Consider for instance space constraints. A grower may use the pathways between tables as temporary storage, an operational solution for a planning bottleneck. During cultivation planning he may anticipate on this possibility.

Spacing

A greenhouse as highly automated as the one just described is quite costly. Depreciation costs may be about *f* 10.-- a year per square metre. For this reason growers avoid having empty space in the greenhouse. This is not easily accomplished because almost all products grow quite a lot during their stay in the greenhouse. Potting them at final density from the start would be a waste of space. The practice is that they are first potted close together and that spacing is postponed as much as is possible without hindering growth. Often, spacing is accompanied by some other manipulation.

There are a number of transport media, often invented by the grower himself for his particular greenhouse and cultures. Except in the case of automatically transportable tables, however, the handling of the plants is rarely automated. Spacing to an adjacent table can be done in one manipulation. For transport over larger distances the plants have to be moved via an intermediate substrate, such as a conveyor belt or a chariot.

Spacing requires an empty destination table. Unless part of the plants remain where they were, it also leaves an empty source table. In a well-filled greenhouse it is not easy to find an adjacent table.

Space planning

For most growers space planning is an important aspect of tactical planning. Some only make a space requirement plan, stating how much of the available space is used each week. Others make an exact space plan. Some do not make a cultivation plan before making a space plan but integrate the two activities. The reason for this is that a space plan is a means to check the feasibility of a cultivation plan. In particular, transportation costs cannot be estimated from a cultivation plan without knowing the placement of the plants.

The usual representation for a space plan is the so-called *log plan*, with time horizontally and space units vertically displayed.[5] Figure 1.2 is a screen dump from the

[5]A log plan is in fact a special version of a Gantt chart. See Verbraeck (1991) for a tutorial on Gantt charts and other representations of plans.

system PROPLAN HAND[6] developed during the case study. The figure shows part of a log plan for the nursery depicted in figure 1.1. The plan is still almost empty. Two production lots are shown: Lot 1 from week 1 to week 26, lot 2 from week 1 to week 17. One sees that lot 2 is spaced twice, so that its space occupation rises from one table during weeks 1 to 5 (table 211) to 6 tables at final width, when the plants are fully grown. Knowing that the numbers at the right-hand side of the grid represent the surface in m^2 per table, one sees that lot 1 is spaced three times. In week 9 lot 1 is transported from table 101 in department 1 (measuring 20 m^2) to table 204 in department 2 (100 m^2).

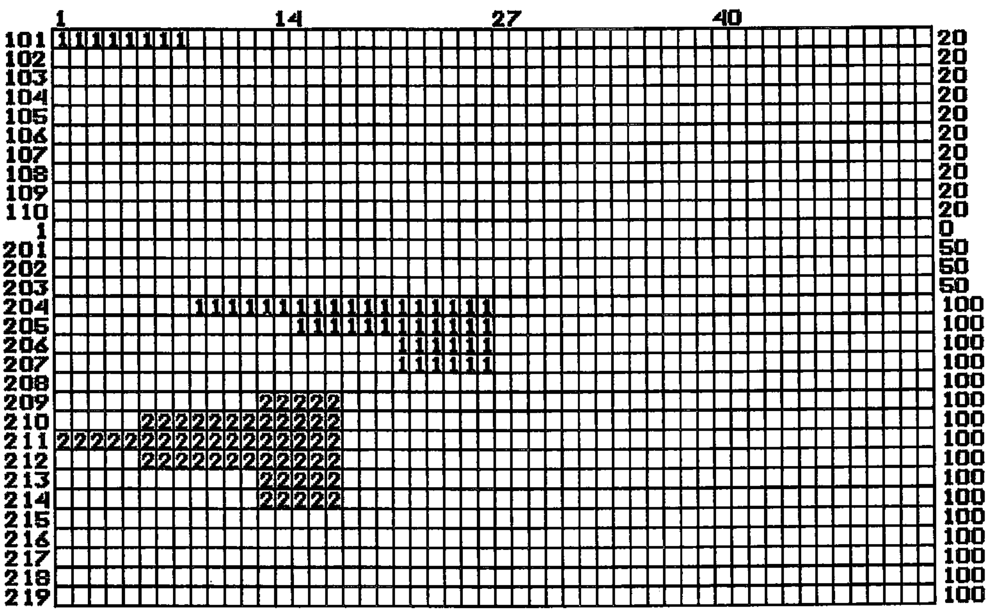


Figure 1.2: log plan as displayed by PROPLAN HAND
(N.B. For a full-colour version of the same screen, see p. 168).

Space planning involves balancing the cost of empty space against that of transport labour. Growers usually prefer minimizing empty space even if this requires some extra handling. There are also other considerations, which may even be more important, such as robustness of a plan. Robustness is defined as the extent to which the plan remains feasible if unexpected events happen during execution. Seeing a log plan, a grower is in a better position to estimate its robustness under adverse weather conditions or other

[6]'PROPLAN' stands for 'PROtotype PLANning aid for pot plant growers'. HAND is the latest version. The project PROPLAN is the subject of chapter five.

eventualities than on the basis of a cultivation plan alone.

The environment of a pot plant grower is very dynamic. A flexible approach to planning and control allows a grower to react to opportunities or adverse conditions. The two most important factors are the weather and the price formation. Both are poorly predictable. Average price fluctuations during the year can exceed 30% for most products. This means that space plans cannot be overly specific but must leave some room for operational adjustments. Sometimes adjustments are not sufficient so that plans have to be revised. Growers anticipate on this by spending progressively less time on later parts of the planning horizon. This is not always true, however; a Poinsettia[7] grower, for instance, must aim at selling at Christmas time. This is the main objective of the cultivation plan which is made a full year in advance, and other products that occupy the greenhouse before the next batch of Poinsettia arrives are used as fill-ups.

Automation in pot plant nurseries

In the Netherlands the primary processes in the greenhouse are almost totally and universally automated. For management purposes, however, only a modest minority of some 5% of the pot plant growers use computers (SITU 1991). This percentage is rising at a moderate pace. It should be borne in mind that these 5% certainly account for much more than 5% of the country's annual production of pot plants.

A survey among 15 innovative Dutch pot plant growers (Werkgroep MBP 1990) indicated that only five of them possessed a commercial software package for registration or planning purposes, whereas 14 of them used spreadsheet or database packages to build their own decision support applications. The survey does not indicate whether the growers indeed used the commercial packages. Typical reasons for developing one's own system were "At least then I understand what it's doing" and "By developing my own system I get exactly the figures I need".

The problem with automation in pot plant nurseries is that the sector is very diverse and the number of growers is limited. Most growers have their own particular wishes (see e.g. Annevelink 1990[a]). Yet administrative software should enable a grower to compare his management with that of his colleagues (Vegter 1990). For commercial software vendors the sector is therefore only moderately interesting. To reconcile software vendors and growers software is needed that contains as many generic components as possible, yet can be adapted to the wishes of a particular grower. The survey mentioned above is part of a number of initiatives to improve the situation with regard to automated support for the management of pot plant nurseries. The case study in this book has a place in a few of these initiatives.

---

[7]Poinsettia, Euphorbia pulcherrima, is popularly known as 'Christmas flower'.

1.2 Scope of the study

History of the case study

The idea for the present study originated from a government-initiated investigation called 'Information Model for Pot Plants'. This study, based on James Martin's Information Engineering method, revealed the needs and possibilities for automation in Dutch pot plant horticulture. A number of institutions, among which Wageningen Agricultural University, the Station of Floriculture at Aalsmeer (PBN), and the Dutch Study Association of pot plant growers (NTS), participated in this investigation. One of the conclusions of the investigation was that the tactical level of management at pot plant nurseries, i.e. cultivation planning, was in need of more support, and that the combinatorial complexity of the planning problems at this level called for automated support systems (Beers et al. 1985, INSP-LO Werkgroep 'Plantaardige Produktie' 1986).[8] This need for research was picked up by the department of Computer Science of Wageningen University and incorporated into a research proposal 'Decision Support Systems in Agriculture and Horticulture'. Research started in 1986. Pioneering work of the Institute of Agricultural Engineering (IMAG) in Wageningen together with the department of Operations Research of the University made a quick start of the research possible (see e.g. Saedt, Hendriks and Smits 1986, Barendse 1986).

Tactical planning and DSS

The term 'tactical planning' was introduced by Anthony (1965). Anthony distinguished three levels in management. The strategic level is concerned with the long-term objectives and activities of an organization. The operational level, on the opposite side of the spectrum, is concerned with the daily functioning of the organizational processes. The tactical level is intermediate: it concerns the making and monitoring of plans within the boundaries set by the strategic level, during the next production cycle.

According to Anthony, two related activities have to be carried out at all three levels. The first is planning, the second is control. Control is the monitoring of the actual processes against the plan, so that appropriate corrective actions can be taken if necessary. Anthony's first book was quite controversial, but since then his ideas have gained almost universal acceptance (see e.g. Anthony 1988).

This study is about *tactical*, possibly *operational planning*. The demarcation between the two is not always sharp, as the case of pot plant cultivation shows: although this is clearly a tactical planning situation, operational considerations certainly play a role in the mind of the grower while he is planning. It is not accidental that Anthony explicitly excludes small organizations, that is, organizations in which a single person is in charge of different management levels, from the set of organizations which he discusses.

To be clear: strategic planning is not taken into account in this study. Strategic

[8] More conclusions from this investigation are given in Elzas and Simons (1987).

planning processes are of a non-repeated, barely delineated, open, often political nature and problems such as group consensus are important. All these aspects make them very different from tactical and operational planning, where the cognitive aspect of the situation usually dominates over the political, and where the situation is comparatively well-delineated.

The common elements in the tactical and operational planning situations considered in this study are that they occur repeatedly (e.g. once or twice a year), that they consist of matching the organization's 'means of production' to each other during a certain period in the future and that the planner (or the team of planners) has some freedom to manipulate the planning situation. In the case study, the means of production that are matched are e.g. production lots of plants, locations, labour, and materials. The grower, who is the general manager or at least a member of the management team, is clearly free to alter the cultivation planning problem, being as it were the owner of the problem.

This study is also about *information systems*. An information system is usually defined (e.g. Bots et al. 1990) as a quintuple consisting of (1) machines, including computer hardware, (2) computer programs, (3) people, (4) organizational procedures, (5) data sets.[9] Together, these five elements store data sets in order to provide information: hardware operates according to software, people work according to procedures. An information system intended for supporting managers, planners or executives in an organization is often called a *decision support system*, abbreviated DSS. DSS are automated systems intended to support, rather than replace, decision makers. A DSS might for instance aggregate data into graphs, perform optimization on the basis of input data, or make drawing and erasing easier than it is on a planning board or with pencil and paper. In this study a DSS for planning situations is called a *planning system*.

Planning systems: the problem solving approach

From a management point of view, planning is an instance of management decision making. The literature on management decision making has been dominated for some time by Herbert Simon's views on problem solving (e.g. Simon 1947, 1960), which hardly need introduction to those who study information systems. Simon wrote his first books in a time in which an ideal of economic rationality, i.e. striving for the optimal solution in an economic sense, existed for decision making in organizations. He argued that this ideal is too remote from actual decision making in organizations, and introduced the concept of *bounded rationality*. The essence of this concept is that since people who are faced with a problem usually have imperfect information and limited time, and especially since they have limited intellectual capacities, they cannot obtain solutions that are optimal in an absolute sense. Within the bounds of their capabilities, they do as best

[9]Often the term 'management information system' (MIS) is used. This sometimes means the same as 'information system', but it may also have a more specific meaning. MIS is then used to refer to computer programs that enable a manager to aggregate data generated by an organization's primary processes.

they can. Simon regards problem solving as a process the structure of which can roughly be described by the phases intelligence - design - choice. He describes these phases as follows (Simon 1965, p. 54) :

> The first phase of the decision-making process - searching the environment for conditions calling for decision - I shall call *intelligence* activity (borrowing the military meaning of intelligence). The second phase (inventing, developing, and analyzing possible courses of action - I shall call *design* activity. The third phase - selecting a particular course of action from those available - I shall call *choice* activity.

Many authors who refer to Simon attach a rather mathematical meaning to these phases, in which the design phase includes formal modelling of the problem.

Another influential view on problem solving is the one by Mitroff et al. (1974), which distinguishes the phases conceptualization, modelling, model solving and implementation. It is necessary to explain the difference between conceptualization and modelling. A conceptualization of a situation is an abstraction from reality that is made more or less explicitly and may exist only in the mind of its maker. A model is a conceptualization according to explicit rules and described in communicable form, usually in a formal language such as mathematical notation or a programming language. In Mitroff's view, formal conceptualization and modelling are explicitly included.[10]

For those who adopt the problem solving approach as adapted from Simon or taken from Mitroff et al., the research approach for learning how to build effective decision support systems seems clear. One has to study the problem in question and find out what goals, variables and constraints play a part. Then comes the research challenge: one has to devise a suitable conceptualization, on the basis of which one can then build models and automated tools, in our case planning systems. Each planning system which is actually used in the organization for which it was made is a living proof that the conceptualizations inside the system are suitable for the planning problem in question. With time, the experience gathered through a body of cases will allow the creation of development methods. After all, methods are said to be nothing but 'canned experience'.

This problem solving approach to management decision making has been widely followed since the Operations Research activities in World War II. It has been applied in a great variety of situations, including tactical planning. In the early seventies the concept of Decision Support Systems became popular (e.g. Scott Morton 1971). Since the advent of DSS, there are definition problems: is DSS a new sub-field of Operations Research, or is Operations Research one of the disciplines that contribute to DSS? Whatever the answer, the Operations Research approach to developing DSS is abundantly present in the literature.

[10]A discussion of this approach can be found in chapter 4; see e.g. figure 4.4 (p. 127) for a pictorial representation.

Problems with the problem solving approach

For some reason or another, this problem solving approach does not always work as well as might be hoped. For instance, huge amounts of time and money have been spent on designing support systems for tactical planning problems such as scheduling and timetabling. Sophisticated models that can generate plans unassisted by humans result from these endeavours. Many planning systems are being designed and built, especially in the academic world, but far fewer are being used. A sizeable body of systems never gets beyond a research phase, despite ambitions to get them into actual use. In fact a fair part of the investments in planning systems does not result in DSS which are actually used, in improved planning practice, or in better DSS development practice. Those planning systems that do actually function in organizations are often disappointingly simple from the problem solving point of view described above. Typically, DSS for planning situations that are in actual use contain but little, if any, model-based support. As a consequence Mitroff's four-phase model cannot be recognized in the systems since they do not contain the type of formal models used by these authors. The problem solving activities are left to the planners, whereas the systems are 'stupid'. Their benefits reside mainly in improved administration and reduced time consumption of the planning process. In quite a number of cases this is so despite the circumstance that models are available that can generate plans of higher normative quality than the planners can. Why is this so? From a problem solving perspective *à la* Mitroff one gets the feeling that opportunities to do better are lost.

1.3 Research questions

When this study began the research question was the straightforward engineering question: "*What should a DSS for tactical planning look like, if it is to be useful in practice?*". Since I defined planning as an instance of decision making, the question was how to support this decision making. This means that my scope was wider than just the normative quality of the plans generated by a system. I wanted to find out what elements of a planning system contributed to its usefulness in practice. In particular I suspected that the type of *models* used in a planning system had a strong influence on the usability of the system, and that part of the cause for the problems mentioned above with the problem solving approach lay with the models applied.

After some experience with the design of a planning system for the pot plant case, as well as some literature study, I was able to articulate the research question in more detail.

A planner working on a plan is engaged in problem solving. My hypothesis became that the models used in a DSS should be adapted to the planner's problem solving activities, for if they are not, the planner will not want to use them. Under this hypothesis the question can be asked: "*Which functions make a DSS adaptable to a user's problem solving activities?*". Since interaction between user and DSS is the only way in which the user can make the model do what he wishes to have done, this interaction is especially important. Realizing this I added two questions which specifically address user-DSS interaction. The first is: *"In which DSS functions is interaction desirable, and what type of interaction should be possible?"*. For instance, how important is it to have interaction during execution of a model, and what should the interaction consist of: should it allow the user to exit from the current session, to modify model parameters, or even to override a partial plan generated by the model? The second question about the interaction between user and DSS is: *"What steps in problem solving can be allocated to the user, and what steps to the system?"*. For instance, the designer can choose to include a particular constraint in a model (and implement the model in the DSS), or he may not model it and leave it to the user to guard the constraint. More generally, one can ask to which extent a problem situation should be predefined in a planning system.

I also became aware that there are enormous differences between planning situations. The literature showed this, and so did my own acquaintance with various growers. It seemed very plausible that different types of planning systems would be needed to accommodate different types of planning situations. Perhaps there was not just one ideal approach. Perhaps the problem solving approach was as good as any, but only suited for a certain category of planning situations. All in all I now had a broader perspective on planning systems, which could be called a contingency or context perspective: "It depends on the situation what a planning system should look like". This meant that my research question became embedded in a second question: *"For which planning situations is it appropriate to develop a DSS?"*. It turned out that, although implementation problems were occasionally briefly mentioned, the organizational context was by and large missing from the literature on planning systems.

I also realized that if a DSS is a dedicated system built for a specific person (and this is usually so), then this person has an important role to play. During the case study, I had made early assumptions about the nature of the planning situation. Although these assumptions were solidly based on information analyses, they turned out not to be specific enough. A turning point occurred at the moment when my contacts with growers began to give me a feeling for how growers really planned. This made me realize that the nature of the development process of a planning system is important, in particular the cooperation between developer and future user. In other words *what* and *how* are two interdependent matters in the development of planning systems; one cannot just drop a planning system on a user's desk. I could now formulate another research question, complementary to the first research question. It was: *"How should a planning system be developed?"*.

Thus, the central research questions of this study can be summarized as *the what and how of interactive planning systems*. The question 'for which situations' is needed as a third question to put the two central questions into perspective.

1.4 Outline of the research

Research activities

In order to answer the research questions, I carried out three activities. These are literature study, generation of applied theory, and the development of the planning system for the growers. These three research activities were closely intertwined; for the sake of clarity, a somewhat sharper distinction is suggested here.

Three bodies of literature were studied. To investigate the 'what' and 'how', I studied literature about DSS for tactical planning situations. A diversity of research schools was considered, because the aim was to compare different approaches. I expected that a comparative description of the various approaches would enable me to discover the strengths and weaknesses of each.

The second body of literature studied concerned quality assessment or evaluation of DSS. Again the objective was to elucidate the 'what' and 'how'. The assumption was that because this literature adopted a 'how to do it' perspective, it would complement the views obtained from the literature on developing planning systems, which was mainly of the 'here is what we did' type.

The third literature study was meant to answer the question 'for which situations'. This question explicitly addressed the organizational context of planning systems, which as mentioned above is hardly dealt with in the literature on planning system design. For this reason it was necessary to turn to organization science, psychology and systems theory. I expected to find a relation between characteristics of a planning situation and desirable properties of a planning system built for it.

Besides providing answers to the research questions, the study of these three bodies of literature was also meant to generate some applied theory intended to improve the situation with regard to development and evaluation of planning systems. My investigation of the 'for which situations' question led to some ideas about the systematic analysis of a planning situation prior to starting a DSS development project. The answers to the 'what' and 'how' questions, taken together, could be used to develop means to evaluate DSS and DSS development processes respectively.

My third research activity, the case study of developing a planning system for cultivation planning at pot plant nurseries, has been a continuous source of insights used for inductive generalization. In the case study, a PROtotype PLANning system for cultivation planning at pot plant nurseries was developed: PROPLAN. From 1986 on, several pot plant growers participated in the case study, which has in fact made it a collection of case studies. Because at each point in time there has only been one prototype system under development, however, the various cases shall be referred to as 'the case study'. The case study was conducted by prototyping[11], combined with small-

[11]In information systems development, 'prototyping' means iterative design with short cycles, so that system versions are obtained rapidly to allow effective communication between developers and users.

scale laboratory testing of current versions on students. The approach taken for the case study is depicted in figure 1.3.

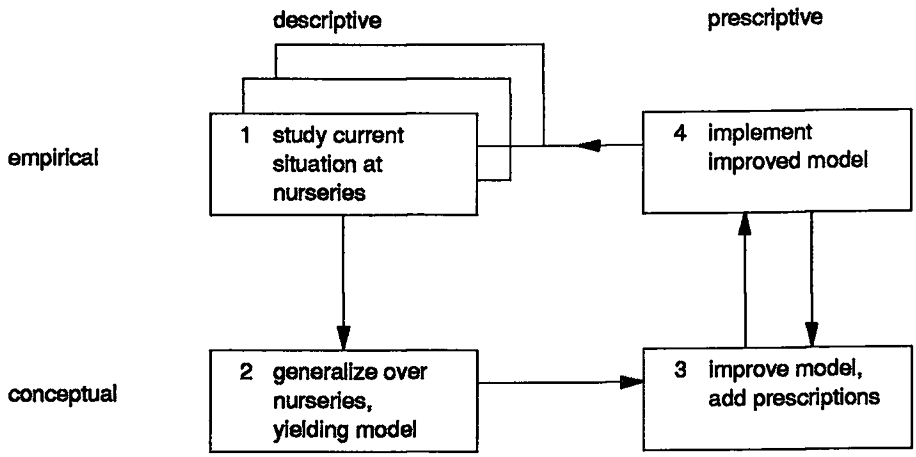


Figure 1.3: Design approach for the case study.
Adapted from Van der Ven (1989), p. 5.

The PROPLAN project set off with an information analysis (1, top left) which led to some modifications to the descriptive conceptual model which was already given in Beers et al. (1985) (see Hofstede and Simons 1987). The design phase (3, bottom right) included research into heuristics. This led to a succession of PROPLAN prototypes (4, top right). There were seven cycles between 3 and 4, in which prototype versions were designed and built. Exposure of the one-but-last of these prototypes (PROPLAN 0.2) to practice (1 again) led to renewed development activity, in close cooperation with growers. This resulted in PROPLAN HAND. PROPLAN HAND was developed as a planning aid for two of the growers.

Working goals

The working goals for this study are:

1 To find concepts that help a developer make planning systems which are easy to master and use and which agree with his way of working. These concepts concern both the process of development of the planning system and the end product, or in other words both the 'how' and the 'what'.

2 To develop a means of assessing *a priori* what type of planning system, if any, is suitable for a given tactical planning situation, depending on characteristics of the situation and of the organization in which the situation occurs.

3 To carry out a project in which an interactive planning system is made which helps pot plant growers with cultivation planning. The resulting prototype system should be usable enough to allow real on-site use but need not be marketable.

Related work

This work is closely related to that of the other members of TIPSY, with whom there has been fruitful informal exchange of ideas. TIPSY is the 'Team Interactive Planning SYstems', a group of Dutch researchers working on doctoral dissertations in the area of planning systems. TIPSY was formed in 1989. Presently, the dissertations of Bots (1989), Verbeek (1991), Eiben (1991) and Verbraeck (1991) have already appeared whereas those of Akkermans, de Jong and Mietus are soon due.[12] Other Ph.D. or doctoral dissertations in the field of information systems also resemble this study to some extent. Many of these works share the method of inductive generalization from case studies given in figure 1.3. They also share a perspective on decision making based on Herbert Simon's intelligence-design-choice division and on his paradigm of bounded rationality. In fact it is not easy to find a recent Dutch doctoral study in this field which does not state that it adopts Simon's views. For instance Van Schaik (1988), Boersma (1989), Bots (1989), Vellekoop (1989), Van der Ven (1989), Frowein (1990), Verbeek (1991) all use intelligence-design-choice in some way. Only Eiben (1991) does not. A study of research abstracts reveals that Simon is the most widely quoted author among 55 Dutch doctoral students present at the Dutch National Doctoral Consortium on Information Systems in 1990 (TIPSY 1990).

The present study does not take Simon's three-phase distinction for granted. Instead it investigates descriptive organization theory literature and looks into factors that surround and possibly modify actual decision making in organizations. The result is that the intelligence-design-choice division loses its central place as a guideline for the design of DSS.

Perhaps mirroring the development of the field, several current European Ph.D. or doctoral studies in information systems give attention to a diversity of non-quantitative aspects. These are inspired by disciplines ranging from philosophy to the social sciences. Often, attention is given to the human side of processes of change that occur when information systems are introduced (See for instance Angehrn 1991, Holwell 1990, Keane 1990, Näslund 1992, Saarinen 1990). The present study fits into this tradition.

---

[12]De Jong's doctoral dissertation will probably have appeared after typesetting but before publication of the present manuscript.

1.5 Structure of the book

The overall structure of this book is: an introduction, three theoretical chapters, a chapter about the case study, and a concluding chapter. The book is intended to be read from cover to cover, and should be comprehensible to relative outsiders to the research fields of DSS and planning systems. Readers with a background in a specific area could focus on specific chapters without having to read the rest. The pot plant case study is occasionally used as an example in all chapters.

The order of the theoretical chapters reflects the order in which my ideas developed during the study. The literature in chapter two, 'Planning and automated support', was needed fairly soon to get ideas for the case study and to find out whether my intuitions about the importance of interactivity were shared by other designers of planning systems. When I became gradually aware that the literature on planning systems hardly dealt with the setting in which the systems were to be used, I engaged in the literature study reported on in chapter three, 'Decision making in organizations'. With the accumulated experience of the case study and the study of the two bodies of literature, I was in a position to think about quality aspects for DSS. In chapter four, 'Quality of DSS', I first review the literature on quality assessment of information systems, in particular DSS, and then present my own views on the matter. In fact all three theoretical chapters contain not only literature studies, but also discussions, as well as the theory which I developed on the subject.

Chapter five, 'The PROPLAN project', is an elaborate description of the case study, including a discussion of both the development process and the systems that resulted.

Finally, chapter six summarizes the book, presents an overview of the perspective on planning systems and their development which emerges from the study and includes a few general discussion topics.

1.6 Some definitions

Many of the definitions given below are treated in more detail in other chapters. The overview here cannot be omitted because words like 'decision making', 'unstructured decision', 'problem' and 'planning' are far from uniformly used by authors in the fields which this book covers. As a matter of fact, there is much confusion about terminology in the field of Information Systems research. In order to avoid adding to the mess, it is necessary to specify in which sense a number of crucial words will be used here. The purpose of the present section is to discuss especially the differences and relations between terms.

Decision making in organizations

Theories of decision making in organizations can either be *pre*scriptive (also called normative) or *de*scriptive. Although most such theories contain both prescriptive and descriptive elements, there are clear differences in emphasis. The prescriptive theories stress that partitioning activities and predefining a goal structure should be the basis for decision making, and such theories even define procedures, models or algorithms to be applied for decision making (e.g. Anthony 1965 and 1988, Kampfraath and Marcelis 1981, Lindley 1985).

Descriptive approaches stress the mess that is present in organizations: decisions interact, work is fragmented, goals cannot be quantified or made explicit, or they are even invented after the fact to justify decisions. In other words, the conditions under which formal methods such as mathematical optimization are of value may not be met, at least when considering decision making in organizations as a whole (e.g. Cyert and March 1963, Lindblom 1959, March and Olsen 1976, Mintzberg 1973).

In accordance with descriptive approaches to decision making, *decision making in organizations* is in this study seen as a *complex of interrelated activities in which decisions do not stand by themselves*. A *decision situation* is one of these activities considered apart for some purpose, including the context of the situation. A *decision problem* is a *decision situation which is formally described*, i.e. a conceptual entity abstracted from the organizational processes. A decision problem has goals and variables and operates on an 'object system': the part of reality which is taken into consideration for the decision, and therefore formally described as a system.

Five dimensions of decision making

When approaching decision making in organizations from a descriptive point of view one has the problem of choosing a terminology to describe what one sees. In this study, an attempt is made to be as open-minded as possible, so as not to miss relevant aspects. Five dimensions are distinguished along which decision making can be observed.

The first dimension is the *actual* nature of the problem to solve, or of the decision to make: is it an operational disturbance which must be dealt with quickly, or a problem with far-reaching implications? The second dimension is the *formal* management level in Anthony's sense at which the situation occurs. These two dimensions are independent of each other, as the following examples illustrate. A group of managers engaged in meeting on strategic planning (in terms of dimension two) may suddenly find that they need some data. This leads to the operational problem (in terms of dimension one) of getting the data. The reverse can also occur: during operational work a serious defect in a crucial machine may be discovered by chance which necessitates far-reaching decisions about how to replace the machine.

The third dimension is the nature of the problem solving *process*: is there a formal procedure, what steps are taken, etc. The fourth dimension concerns the *organizational context* in which the situation occurs. This includes such things as management style, corporate culture. The fifth is the *personal* dimension. It concerns the psychology of the decision maker, including elements such as personal style, personal goals and personal circumstances at the time of the situation. The third, fourth and fifth dimensions include an awareness of processes of change and learning at the level of an individual or of an organization.

These five dimensions will be used continually, often implicitly, in this study. Of course there exist other dimensions of decision making. However, aspects of decision making that transcend the decision maker and the organization, such as political, social, or ethical dimensions, fall outside the scope of this study, except where they play a role in one of the five dimensions described above.

DSS and structure of problems

Many authors of DSS papers agree on a few key characteristics of DSS which were first formulated by Keen and Scott Morton (1978). DSS are supposed to support decision makers, rather than to replace them. DSS offer support to people who have to solve semi- and unstructured problems. However, lack of structure of a problem is a problematic notion (see e.g. Verbeek 1991, p. 22). It is understood quite differently by authors with different backgrounds. It can for instance concern the demarcation of the object system from its relevant environment. Or, if the object system is known, data about the initial state of the object system, including available resources, can be unavailable; the goals to be achieved can be unclear; constraints can be soft or unknown; the consequences of decision alternatives can be poorly predictable. Lack of structure can also be of a practical nature, and for instance concern procedures and methods to obtain necessary information, to specify goal attributes and to generate and evaluate decision alternatives using available information systems and DSS. Finally, models of the object system may themselves be unstructured in the sense that the output fluctuates unpredictably given small input fluctuations (Simons and Wieringa 1986, p. 11).

This enumeration shows that lack of structure can reside in various aspects of a decision problem. In fact, since a decision problem is a conceptualization, by definition *lack of structure of a problem resides in the mind of the one who conceptualizes the problem.* If a problem owner and a DSS designer both contemplate a problem situation they may arrive at different conceptualizations. As a problem owner learns about his problem the problem becomes more structured. This is entirely a conceptual learning process which does not affect the problem situation as it would be perceived by a third person.

In this study it is assumed that a planning problem may possess all of the attributes of lack of structure mentioned.

Situations, open problems, formal problems, and puzzles

The meaning of 'problem' can vary from 'something which is not right' to a chess problem, that is, from something very vague to something exactly specified. More than one word is needed to convey the various meanings of 'problem' within one text.

At one end of the spectrum of definitions, Checkland (1981, p. 155) defines the word 'problem' as follows.

> A problem relating to real-world manifestations of human activity is a condition characterised by a sense of mismatch, which eludes precise definition, between what is perceived to be actuality and what is perceived might become actuality.

Checkland adds that a number of systems analysis projects in which he participated were completed successfully in the sense that they were judged so by both client and systems analyst despite the fact that 'the problem' had never been defined. He also remarks on the experience that problems usually keep changing over time.

For a more restricted definition, consider the definition by Ackoff (1981, p. 20) also adopted by Bots (1989, p. 13):

> By a *problem* we mean a situation that satisfies three conditions: First, a decision-making individual or group has alternative courses of action available; second, the choice made can have a significant effect; and third, the decision maker has some doubt as to which alternative should be selected.

This definition presupposes much more clearness than Checkland's. It centres on the fact that someone has to make choices in a particular situation. There is no mention of modelling the situation, except perhaps in the mind of the problem owner (i.e., conceptualizing). 'Problem' and 'problem situation' are synonyms.

In a more elaborate definition, Ackoff (1978) sees a problem as something with five types of components:

1. The decision maker(s) faced with the problem
2. The controllable aspects or variables
3. The uncontrollable aspects or variables
4. Constraints imposed from within or without on the values of the controllable and uncontrollable variables
5. The possible outcomes produced by the decision and the uncontrolled variables.

Ackoff stresses that variables and constraints can be any combination of quantitative and qualitative elements. The decision maker(s) can either solve, resolve or dissolve a problem, according to Ackoff. Solving means finding the optimal solution, resolving is finding any solution, and dissolving is evading the problem by redefining it.

Compare this definition with the definition by Eiben, of which some elements are quoted here just to give an impression (Eiben 1991, p. 24, or see Eiben 1989, p. 175):

> A *planning universe* is an ordered triple: $(S,A,(T,<))$ of non empty sets, where
> - S is a set of *world states* with a special element $\square \in S$ called *nil*;
> - A is a set of *actions*;
> - $(T,<)$ is a linearly ordered set of *time instances*.

followed (p. 26) by

> A (static) *planning situation* is a 5-tuple
>
> (S,A,T,α,e),
>
> where the triple (S,A,T) forms a planning universe, α is an allowability relation and e is an effect function on (S,A,T).

> A planning situation captures the most relevant factors of the world under consideration. To define a planning problem, however, we also need to know what a plan is and what the problem is, i.e. what kind of plan is wanted as a solution.

Eiben's definitions are very formal. The entire world under consideration is formally modelled. As Eiben expressly states himself (1991, p. 12), the role of the human planner is unimportant at this level. A 'planning problem' is an instantiation of a 'planning situation'.

De Leeuw (1982) draws the following distinction between situations and problems: in a given situation there can be as many problems as there are people involved. These people are called *problem owners.*

In this book I shall follow De Leeuw's distinction. Thus the word *situation* means 'something that occurs in an organization' much in the way Checkland defines a problem. The word *problem* denotes a predicament in which one decision maker finds himself, much like Bots' definition. If the *problem owner has some freedom to decide about the problem definition*, as is the case for the planning problems in this study, the term *open problem* may be used. A *formal problem* is an *abstraction of a situation or problem according to some formalism*, and is analogous to Ackoff's five-element definition without the decision maker. Often, an open problem is solved by extracting from it a formal problem, solving this, and converting the solution to a solution for the open problem. Finally a *puzzle* is a formal problem without uncontrolled variables.

Thus a planning situation can be described in general organizational terms, often in ordinary language, while a formal planning problem must be unambiguously formulated in a mathematical or other formal language. Some mathematical definitions of 'problem' do not even contain uncontrolled variables, so that they describe what I call a puzzle (e.g. Johnson and Papadimitriou 1985). In Operations Research and DSS literature a 'problem' generally is a formal problem. The notion of a 'problem space' belongs to the concept of a formal problem. Consider for instance the division between troubles, problems, and puzzles drawn by Gorry and Krumland (1983, p. 206):

> There are troubles which we do not quite know how to handle; there are puzzles whose clear conditions and unique solutions are marvelously elegant; and then there are problems which we invent by finding an appropriate puzzle form to impose upon a trouble. In management science, models are the puzzles to which attention is directed, and much of the work in DSS constitutes the imposition of those models on various aspects of managerial decision making.

Gorry and Krumland disregard the role of a problem owner so that the concepts of situation and problem are both captured by their 'trouble'. Their 'problem' is a formal problem.

Incidentally, tactical planning is usually not initiated because there is a trouble. Therefore 'planning situation' is used in this book rather than 'planning trouble'. Yet for a planner / problem owner the planning task may be troublesome. This is especially so because to the planner, the problem is not a formal problem but an open problem, so that the definition of the problem is problematic.

The reason for elaborating on the distinction between situation, open problem, formal problem and puzzle is that researchers in information systems have become so much used to abstractions such as 'problem' and 'problem space' that they tend to use these concepts even in situations where they are not adequate. Boland (1987) points to the dangers of specifying a situation too precisely. For instance, in this book it is important not to mistake an open problem for a formal problem.

Decision making versus problem solving

Another distinction concerns decision making and problem solving, yet two more related and diversely used concepts. The terms problem solving and decision making are related but carry different historical connotations. A lucid account of this is given by Smith (1988). Two quotations (p. 1490):

> The decision making paradigm dominates management science research on action-related thought, and is reinforced by a sizeable body of psychological work. The preponderance of this research, exemplified by expected utility theory, decision analysis and behavioral decision theory (...) has adopted an expectancy-value conceptualization of human decision making. (...)
>
> Quite different, and largely ignored by management science, is the perspective adopted by cognitive science researchers concerned with human problem solving. The problem solving paradigm regards humans as goal-directed beings who apply factual and procedural knowledge in determining how best to achieve their objectives.

Thus there is a historical difference in the scientific use of the two words. Outsiders, however, use the words interchangeably, although 'problem solving' is sometimes used to stress the creative or the search aspect and 'decision making' to stress the choice aspect.

I should like to introduce a distinction based on the number of solutions of a decision or problem situation. If the solution space, i.e. the set of possible solutions, is known this means that the situation at hand is a formal problem. If there are relatively few options - tens -, the situation is in fact a *choice problem*, often called a *decision problem*. It is usually the weighting of alternatives which is problematic and decision science comes into focus. If there are many solutions - hundreds to billions - the situation is properly called a *combinatorial problem*. Usually the search in the vast solution space is limiting and Operations Research, Artificial Intelligence, or Simulation techniques are appropriate. There are also formal problems for which it is not known whether the number of solutions is finite; these fall within the same category. If, finally, the solution space is unknown to the decision maker, the situation is really an *open problem* rather than a formal problem. Further problem definition is needed to turn it into one of the above types, or else -and this is the rule- the situation will have to be dealt with in a less formal manner.

Planning and related concepts

Takkenberg (1983, pp. 28-29) considers planning as 'decision making made explicit'. According to him, four phases must occur before decision making can be called planning:

1. Construct a conceptual, i.e. data void model of the planning situation specifying the planning problem in its object system with the planning goals.
2. Supply parameter values to the conceptual model. This yields an empirical model.
3. Generate alternatives.
4. Choose one alternative: the plan.

This definition looks much like Simon's division of decision making in general, in which the intelligence phase is represented by the first two steps, so that it requires setting up and supplying parameter values for a model. Again, this is the problem solving approach to decision making, this time applied to planning.

The definition of planning used in this study is not so restrictive. ***Planning is decision making directed towards making a plan, a plan being a sequence of proposed decisions or actions to be carried out in the future.*** The planner certainly needs working knowledge of the object system, of constraints, and of planning goals, but these need not be formalized. The kind of planning considered in this study has already been defined in the first pages, using Anthony's division. It is tactical to operational planning in which the planner is to some extent free to redefine the planning problem.

A word is needed on terms related to 'tactical planning' and 'operational planning', such as 'production planning', 'scheduling', 'rostering' and 'timetabling'. Here again, there is confusion in the literature. For instance, the term 'planning' as it is used in this book may be misleading to some readers who feel that it should have been called 'scheduling' instead. This would have misled others who would then have expected something about combinatorial algorithms for formal job-shop scheduling problems. Unfortunately there simply does not exist a common language among everyone. In this respect not much has changed since Conway et al. (1967) published the first 'Theory of Scheduling' in which they remark that work on scheduling has been fragmented and carried out in separate disciplines without much contact.

Some authors in Artificial Intelligence attach a special meaning to the word planning which sets it aside from scheduling-like problems. Others, working on strategic business planning, could also be misled. Therefore the term 'tactical planning' is used in this study, although it is often simply abbreviated to 'planning'.

The difference between tactical planning, scheduling, etc. is a matter of focus. Many a real-life planning situation can either be seen as a planning problem, a scheduling problem, and so on. All these problems contain a combinatorial aspect of *generating an assignment of n-tuples of resources to time spans*. In other words there is an assignment or matching subproblem and a sequencing subproblem (Badie et al. 1990) to a planning situation. The choice of a name implies that some aspects of the situation are emphasized.

The term *scheduling* has been in use for mathematical treatment of job-shop situations (e.g. Rinnooy Kan 1976). Conway et al. (1967) use it as "essentially a synonym" for sequencing. In a job-shop operations that are grouped in jobs have to be assigned to machines at a point in time. 'Machines' could be any type of resource, and there could be several types of resources simultaneously needed for an operation. A comprehensive overview of scheduling from a management point of view is given by Verbraeck (1991), who mentions some synonyms (p. 3): production planning, short term production planning, process planning. Verbraeck stresses that this type of problem has been neglected in the literature. He devotes a chapter to the presentation of a general descriptive model of scheduling problems, including a discussion of different representations of scheduling problems.

'Rostering' is scheduling of people. The word is sometimes used in the context of establishing shifts in factories and hospitals. Workers are assigned to tasks and to time spans, often in groups (see for instance Bots 1989).

Timetabling is assigning people to resources. The word is used in the context of scheduling at schools. Classes of pupils are assigned to teachers and classrooms at a point in time (see for instance Verbraeck 1990[b]).

How are these concepts used in the book?

The reader may at this point feel hopelessly immersed in a flow of words.[13] In fact, the aim of introducing all these notions is exactly the opposite: it is to provide material enabling the reader to safely sail the seas of literature of the next chapters. Let me try to help build a raft by reiterating a few core points.

The subsections above have introduced a number of notions found in almost all literature this study deals with. Whenever a publication is quoted, it should be borne in mind that the words in the quoted text are used according to the school of its author. In my own prose, I use the concepts in the sense of the definitions that are in italics in the preceding subsections. It will become apparent in the next chapters how my usage of the terms can be justified.

Planning, as it is used in this study, means any activity carried out by one or more individuals in an organization which is directed towards making a plan. A *plan* is a tactical or operational plan, i.e. a set of proposed decisions or actions to be carried out during the next production cycle. Those making a plan are referred to in singular as 'the planner'.

It is important to remember that I refer to a planning *situation* as something that occurs in an organization, whereas a planning *problem* is a planning situation seen from the point of view of one individual (the problem owner). Another distinction which is central to the book is the one between an *open* problem and a *formal* problem. An open problem can be altered by the problem owner, a formal problem cannot. Conversely, a formal problem may be solvable by a computer program without user intervention, whereas an open problem cannot be solved without the problem owner's help.

---

[13] ...Won sages drown in words, e.g. as now...

2 Planning and automated support

Competence about the user problem
is something that the manufacturers think users do not have
and that users of the system later discover that the manufacturers did not have.

Anonymous (in Bjørn-Andersen 1988, p. 390)

The present chapter deals with the 'what' of planning systems. It starts with an extensive review of literature on automated support for planning situations. Literature on planning systems is hard to classify. The emphasis lies sometimes on the planning situation, more frequently on the details of the models used. Research communities tend to have their own recognizable types of problem formulation. For this reason the review of the literature is structured according to research community.

After the literature review, the research schools are compared and discussed. The 'what' question is formulated more precisely: What do the systems described in the literature look like, and what should a planning system look like if it is to be usable in practice? Some literature which pays specific attention to the practical usability of planning systems, often related to a particular case, is brought to bear upon this discussion. How interactive a planning system should be, and what 'interactive' should mean, is discussed. The chapter concludes with some suggestions for components of usable planning systems, in particular about the application of interactive heuristics and recursive problem decomposition techniques.

2.1 Literature: Operations Research

History

The field of Operations Research, OR for short, has also become known as Management Science. It originated in the Second World War in a military context, then rapidly spread to industry, business, and non-profit organizations. In its initial years, OR was relatively unspecific, e.g. Churchman et al. (1957, p. 11):

> Ten years ago any interested person with a creative mind and a good training in science or engineering could easily become an operations researcher. This easy movement into OR is going by the wayside because as OR develops it requires more and more time to catch up with what has gone on and to learn what methods, techniques, and tools are available.

Churchman et al. stress the interdisciplinary team approach characteristic of the early years, when OR specialists by education did not exist.

During the forties and fifties the discipline has evolved. The 'Introduction to OR' by Churchman et al. is organized in a real-world-problem oriented way, with chapters like 'analysis of the organization'. The 'Introduction to OR' by Hillier and Liebermann (1967), written ten years later, is technique oriented, with chapters on 'Probability theory'. The same holds for the 'Principles of Operations Research' by Wagner (1969). Although many of the subjects treated are the same, e.g. linear programming, the organizational context is much less prominent in the latter two books.

Hillier and Liebermann (p. 5) define OR as follows:

> OR is applied to problems that concern how to conduct and coordinate the operations or activities within an organization. The approach of OR is that of the scientific method. In particular, the process begins by carefully observing and formulating the problem, and then constructing a scientific (typically mathematical) model that attempts to abstract the essence of the real problem. It is then hypothesized that this model is a sufficiently precise representation of the essential features of the situation, so that the conclusions (solutions) obtained from the model are also valid for the real problem. This hypothesis is then modified and verified by suitable experimentation. (...) to be successful it must also provide positive, understandable conclusions to the decision-maker(s) when they are needed. (...) An additional characteristic (...) is that OR attempts to find the best or optimal solution to the problem under consideration.

They state that the impact of OR is rapidly growing and "no slowdown is in sight". The scientific contribution of OR lies among others in

> (...) the structuring of the real life situation into a mathematical model, abstracting the essential elements so that a solution relevant to the decision-maker's objectives can be sought. This involves looking at the problem in the context of the entire system.

Gap between theory and practice

The specialization which went on during several decades has caused a part of the OR community to overspecialize. Such a tendency has been signalled at the onset of the eighties by experienced OR practitioners, e.g. McArthur (1980, p. 111):

> I can't recall a problem in industry that we couldn't solve for lack of mathematical know-how; heuristic computer programs can be written or simulation models can be used where analytical solutions aren't available. We focus too much attention on the tools and too little on how to use them.

McArthur illustrates with examples that it can be fatal for an OR study if commonsense considerations are overlooked, such as political motives of managers or time constraints on a decision process. He finds that the discipline has in part lost its orientation towards organizational problems and overly focuses on refining mathematical tools. This probably is an almost inevitable consequence of the growth of specialized OR knowledge. Ackoff (1979, 1987) has also repeatedly commented upon the evolution of OR. According to him the nature of organizational problems has changed. Once the post-war productivity problems had been overcome, problems like marketing, work conditions, and organizational goals became more important than the production scheduling, allocation and routing problems to which OR techniques were best applicable. OR did not possess tools to deal with human motivation and judgement, and chose to ignore problems in which those factors played an important role.

Formal problems

Ackoff's comments imply that problems like scheduling, allocation and routing problems do not require much knowledge about human motivation and judgement. Indeed a great body of literature, both older and recent, only considers formal problems, without organizational context or problem owner (e.g. Conway et al. 1967, Lawler et al. 1989). The reason for performing this abstraction is that is permits greater generality and thorough mathematical analysis.

Van Hee (1985) describes a planning situation in mathematical terms, even in its organizational context. According to him a model in a DSS should be a mathematical abstraction of a control situation in an organization. Such a 'control model' should possess

- a state space
- an action space
- a process mechanism
- evaluation figures.

The state space is assumed to adequately model the real system. The action space represents all possible actions in different states of the system. The state changes as a result of the actions. The process mechanism determines which actions are allowed in each state of the state space, and it possibly also determines in which order the actions may take place. The evaluation figures allow the decision makers to evaluate the actions which they have chosen. Van Hee adds that establishing these constructs is often difficult and may require considerable abstraction from the real system. Eiben (1989, 1991) has elaborated on Van Hee's ideas.

Such a type of formalization has but a limited capacity for including practical aspects and constraints on the real-world planning situation. This is indicated by some of the authors (e.g. Conway et al. 1967 p. 3, Van Hee 1985 p. 982).

OR models and user judgement

The realization that OR models should in many cases be supplemented by human judgement has led recently to the creation of a branch of OR which can be termed 'OR-based Decision Support' (Anthonisse, Lenstra and Savelsbergh 1988). Vellekoop (1989, p.7) describes this approach as one that strives

> to improve the quality of the decisions to be made by integrating OR techniques, human ability and computer technology in terms of effectiveness and efficiency.

The assumption is that by making good use of human judgement OR models can be used in Decision Support Systems. The models supply precision and computing power, the user supplies intuition, creativity and expertise. Obviously the integration of these two components is a design problem. That this problem can be successfully resolved is e.g. proven by Savelsbergh (1988). However, Vellekoop admits that he "only considers the models and algorithms of the model base" in his thesis (p. 9). A similar attitude, where an awareness of the importance of the user is expressed but not further articulated, can be found in more recent OR publications. Hendriks (1990) signals a growing practical orientation of Operations Researchers during the eighties. According to Hendriks the advent of personal computers has made this development possible because it enables Operations Researchers to cooperate directly with the users of their systems. Regarding the user interface Hendriks remarks (p. 10) that

> De Modelbase (...) is het eigenlijke hart van het systeem. (...) Het User-interface is het gedeelte waar de uiteindelijke gebruiker/planner mee te maken krijgt. De gebruikersvriendelijkheid van deze component bepaalt, uiteraard naast de voorgestelde beslissingsvarianten en -adviezen, of het systeem zal worden toegepast.[14]

For a case of job-shop scheduling, Serafini and Ukovich (1989) try very hard to make a usable system on the basis of a monolithic algorithm. They repeatedly stress the gap between the model and the real problem, and the importance of translating an algorithm-computed solution into a real-world plan, but only offer directions of a very vague nature for making this translation. Their last sentence is encouraging (p. 209):

> Needless to say, the ultimate evaluation of the project will be based on the way the system will be perceived by the decision makers and on the impact it will have on the production system. Obviously the actual decision makers are already engaged in the DSS project and most design characteristics are based on their suggestions.

2.2 Literature: Artificial Intelligence

History

The field of Artificial Intelligence (AI for short) has long carried an aura of false expectations, including that AI would be able to replace human judgement in general. There are a number of different AI streams. Two paradigms which have both existed for

[14]The model base (...) is the very heart of the system. (...) The user interface is the part the eventual user will have to deal with. The user-friendliness of this component determines, of course together with the proposed decision scenarios and -advices, whether the system will be used.

several decennia are connectionist learning and symbolic reasoning. The former seems at first glance to be the most promising direction for planning problems because of its human-like capabilities for learning, for pattern recognition and for handling combinatorial problems. Nevertheless it has known little progress during the last decennia (Minsky and Papert 1988). Although recently there has been much interest in connectionist networks, neural networks, and related concepts such as genetic algorithms, simulated annealing and the like, I shall no further discuss them because I believe that an actual impact of these ideas upon planning problems is still far away. Glover and Greenberg (1989, p. 122):

> The neural network approach has had most success in pattern recognition problems that arise in vision. Although its roots are in seminal works of 1943 and 1960, it is only recently that it has been considered as an approach to heuristic search. The experiments so far just show that it can be done, and it does look promising on small problems.

The present attitude towards AI is tempered by a long history of failures and partial successes. Waterman (1986, p. 129) flatly puts that Expert System development is not possible for a task which requires common sense. He argues that in many cases Expert Systems can be appropriate for well-understood cognitive subtasks of a problem. This is a far cry from the optimistic introduction to AI by Winston (1977) who sees a promising future for AI: "In farming, computers should control pests, prune trees, and enable selective harvesting of mixed crops" and "In offices, computers should schedule people and groups". The content of AI textbooks did not evolve all that much during the last decade but the level of ambition certainly went down.

AI planning

The domain of planning problems has seen developments similar to those in other domains in which AI techniques have been used. Since the early nineteen-sixties, AI techniques have been applied to planning systems with limited success in terms of practically useful planning systems (Levitt 1990). What makes practical planning so very hard is the 'frame problem' as defined by Wilkins (1988, p. 7):

> This central problem is determining how a complex world is affected by an event that occurs in the world, so that a system can reason about the world both as it was before the event and as it will be after the event.

Wilkins remarks that many AI planning systems do not address the frame problem:

> Many scheduling problems require constraints to be satisfied so that schedules can be correctly met, but do not require that the system reason about how the world changes as scheduled events occur.

Including the frame problem in the representation of a scheduling problem would make it combinatorially much more complex than it is already.

Combining generality with power remains the 'holy grail' (Levitt 1990, oral presentation) of AI planning systems. Usable systems have typically been applicable to a very specific problem only. Levitt gives an overview of the history of AI planners, starting with the General Problem Solver of Newell, including general purpose 'classical planners' found in many AI textbooks such as STRIPS, NOAH and NONLIN, continuing with a second class of domain-specific expert planning systems and ending with ideas about a third generation of planning systems, using model-based reasoning, that he expects will lead to the development of applicable planning tools in engineering design, particularly construction. According to Levitt (1990, p. 181):

> The fundamental limitation of classical AI planners is inadequate representation. They use only literals - either in propositional logic or first order predicate logic syntax - to define all of the knowledge about the problem.

Levitt presents some special purpose planning systems briefly, e.g. MOLGEN, LIFT, ISIS. I shall discuss ISIS in some detail because this system has had a considerable impact in the research world and because it was made for a job-shop scheduling problem.

ISIS

Smith, Fox and Ow (1986) give a summary of their ideas on Knowledge-Based factory scheduling systems, based on the development of the ISIS system at Carnegie-Mellon University (Fox and Smith, 1984, Fox 1987). Smith et al. motivate the need for a scheduling system in their case study by saying that (p. 48):

> The schedulers were simply overburdened by the complexity of the task.

As far as prior research is concerned, they state that (p. 48):

> Scheduling research to date has had relatively little impact on the real-world factory scheduling problem.

They note that OR methods have insufficient capacity for modelling the multiple constraints involved in job-shop scheduling. As for AI planning research they remark (p. 49):

> Overall plan duration has been the most common consideration in AI scheduling systems. (...) In relating these efforts to the factory scheduling problem, the chief point of divergence is the absence of conflicting constraints.

They employ a variety of concepts to try and capture the complexity of the real-world job-shop scheduling problem in a large factory. It is central to their approach that they base their system on a factory model with a frame-based structure and, most importantly, with constraints attached to the objects on which they act. The representation of constraints is very rich. This enables heuristic reduction of the search space, which is far more efficient than the rule-based reasoning found in most AI-based systems.

ISIS, the resulting system, certainly went a long way towards what was needed. The user was able to edit a constraint base. He could specify parts of a desired schedule in advance, which were then treated as additional constraints by ISIS. He could also make changes to the schedules generated by ISIS. When the user added jobs to the partial schedule, ISIS gave advice as to the effect of the user's actions on constraint satisfaction. Yet ISIS was not implementable in the target organization. The authors mention two reasons for this: ISIS was unwieldy in size and it was not integrated with the organization's other information systems.[15]

Model-based reasoning

To come back to Levitt (1990), his third generation planning systems use model-based reasoning which he defines as follows (p. 183):

> in the case of planning, this translates into reasoning causally about both the inclusion of activities in a plan, and the ordering of those activities.

He describes some example systems which use concepts like 'object abstraction' and 'action abstraction', and 'reasoning about activity dependency'. A core concept of this reasoning is an activity, defined as a three-tuple: activity (action, object, resources). A recent system is named after this 3-tuple: OARPLAN (see also Kartam and Levitt 1990). It promises to realize Levitt's claim of combining generality with power, albeit within the limited domain of construction project planning involving assembly of standard components. This is a domain with many explicit well-defined precedence constraints at the object level, which makes it suited to a model-based approach such as the one proposed by Levitt.

AI and user judgement

In many domains it is specifically the soft, diverse and unpredictable nature of constraints that causes problems. These are the constraints that are hardest to formalize. For instance, Hartog and Kroon (1991) implemented a model of the pot plant cultivation problem in the expert system shell KEE (Knowledge Engineering Environment). KEE contains data abstraction mechanisms and logic programming. Hartog and Kroon found that despite these facilities, disappointingly much procedural code was needed to accommodate constraints, and they are pessimistic about the perspectives of getting a usable planning system built in KEE.

---

[15]A more detailed but still comprehensive discussion of ISIS, as well as of various other recent AI planners, can be found in Verbraeck (1991).

Van Dissel, Borgman and Beulens (1990) examined whether user domain knowledge could be replaced by AI techniques for a complex task of making predictions on the basis of time series of data. They found that an active 'context-bound' role of the user in modelling is important for usability of the resulting DSS, even though this DSS may not be ideal from a scientific point of view.

Some authors are willing to complement the current shortcomings of AI methods by using the planner's knowledge, but only reluctantly, and only as long as it is unavoidable, e.g. Fiksel and Hayes-Roth (1989, p. 18):

> To accelerate the near-term availability of intelligent planning aids, we propose a departure from the traditional orientation of planning research. (...) This will allow the development of interactive planning systems in which humans play the role of problem solver and machines assist by analyzing, monitoring, and refining plans initiated by humans. Eventually, as research on problem solving methods matures, it should be possible (...) to achieve true system autonomy.

2.3 Literature: Heuristics

Introduction

Both Operations Research and Artificial Intelligence possess a set of well-developed techniques for solving problems. Both also have limitations. OR algorithms tend not to be very adaptive or flexible, whereas AI techniques are ill-suited for coping with combinatorially complex problems (e.g. Badie et al. 1990, Dhar and Ranganathan 1990). These limitations seem to be complementary, which has led many an author to make pleas for cooperation between OR and AI techniques (e.g. Fordyce et al. 1987, Simon 1987, Simons 1988, Glover and Greenberg 1989, Grünwald and Fortuin 1989). To a certain degree the two fields have met, especially in the area of planning systems (Institution of Electrical Engineers 1990). Typically there is a combinatorial aspect to a planning problem which can be tackled by OR algorithms, and a domain knowledge part which can be dealt with by AI techniques. An example of how AI and OR can cooperate is given in Badie et al. (1990). These authors describe research versions of two general-purpose scheduling packages, presented in more detail in Bensana (1987) and Badie (1988). Another example can be found in Mietus et al. (1991).

A question which remains is whether OR and AI together are sufficiently powerful to deal with real-world planning situations. I shall address this question on various occasions during this chapter.

OR, AI and Heuristics

The meeting point of OR and AI can be summarized in the word *heuristics* (see e.g. Glover and Greenberg 1989). Heuristics, or rules of thumb, are the subject of the remainder of this section. The word heuristic comes from the Greek verb εὑρισκειν: 'to discover'.

Human beings are accomplished creators and users of heuristics for everyday use. In this study the word is used to refer to problem solving processes in which computer systems play an important part. A usable definition of heuristics in fairly general wording is the one given by Beer (1981, p. 402):

> Heuristic (contraction of 'heuristic method'); a set of instructions for searching out an unknown goal by exploration, which continuously or repeatedly evaluates progress according to some known criterion.

Important ingredients of a heuristic which are present in this definition are that it is an exploratory search process, and that there are known criteria for progress even though the precise goal is unknown. Implied but not stated in so many words is that the search is guided by the evaluations of progress.

Most authors from OR use a more specific definition of heuristics. One such definition originally given by Nicholson and adapted by Silver et al. (1980, p. 153) is:

> A heuristic method is a procedure for solving well-defined mathematical problems by an intuitive approach in which the structure of the problem can be interpreted and exploited intelligently to obtain a reasonable solution.

The difference with Beer's definition is crucial: Silver et al. limit themselves to problems for which a well-defined mathematical model formulation exists, i.e. they exclude open problems.

Some OR authors give even more restricted mathematical definitions, for instance Müller-Merbach (1981, p. 1):

> In OR, the term 'heuristic' is usually understood in the sense of an *iterative algorithm which does not converge towards the* ('optimal' or even 'only a feasible') *solution of a problem.*

What this definition conveys most, apart from the iterative nature of a heuristic, is a certain sense of 'weakness' associated with heuristics in a mathematical context.

Pearl (1984) has devoted an entire book to heuristics. He defines heuristics as (p. vii):

> strategies using readily accessible though loosely applicable information to control problem solving processes in human beings and machines.

This definition stresses the nature of the information used to guide the search process. This information can be anything, as long as it is both accessible and useful. In practice it is usually very problem-specific.

In her classic textbook, Rich (1983, p. 35, or see Rich and Knight 1991) has a short AI-oriented definition:

> A heuristic is a technique that improves the efficiency of a search process, possibly by sacrificing claims of completeness.

This definition focuses on the trade-off between speed of search and quality of solution which using a heuristic involves.

Simons (in Huizing and Simons 1980) draws attention to what he terms the 'double principle' of a good heuristic. This refers to the fact that a good heuristic not only searches for a solution, but also seeks to recognize unsolvable problem instances as early as possible.

The use of the term 'heuristic' in this study is consistent with most elements from the definitions given here. I feel most at ease with the definitions by Beer and Pearl, mainly because they do not require or assume the existence of a formal model of the problem under consideration. To sum up, the main ingredients of heuristics are the following. Heuristics do not require the existence of an optimality criterium but do require that solution quality indicators exist. Heuristics are problem-specific. They use any readily available information on the problem domain. They allow to balance the quality of a solution with the time spent looking for it. Depending on the definition, they do or do not require that a formal model of the problem (including constraints) exists.

The difference in approach to heuristics between OR and AI is well worded by Pearl (p. 19). An operations researcher talks of a problem as a state space of preexisting solutions, from which solutions have to be *eliminated* by e.g. branch-and-bound methods. An AI researcher prefers to see the same problem as one of *creating* a solution in a world that is not specified in advance. The OR approach lends itself well to proofs of completeness and optimality, while the AI approach is more like the way a human being views a problem solving process.

OR and heuristics

OR techniques can broadly be classified into optimizing and non-optimizing methods. The former category contains the mainstream of OR research and practice. Utterances about the limitations of mathematical optimization techniques for solving practical problems have been made by OR practitioners right from the start. The preponderance of refined mathematical methods in the literature undoubtedly has to do with the suitability of these methods as objects of research. Status of the resulting publications also plays a role, as admitted frankly by Stainton and Papoulias (1984).

The second category, non-optimizing methods, is often called 'heuristic methods'. Heuristic methods seem to be a source of renewed interest in OR. In an editorial note about heuristics in OR, Fisher and Rinnooy Kan (1988, p. 263) say:

> The 1950s were a flourishing period for Management Science that saw many practical problems successfully attacked through the application of inelegant but effective heuristics. In the 1960s, attention turned to optimization, leading to the development of algorithms that employed more sophisticated mathematical constructs. While these algorithms were a significant research achievement, they failed to provide for reliable

> solutions for many problems. The 1970s seem to have been a period of soul searching in which combinatorial complexity results were discovered, providing evidence that those who failed to develop effective optimization algorithms should not be discouraged, since the problems were probably intractable anyway. As a result, some of the intellectual energy that had been devoted to optimization began to be directed to the study of heuristics, but from an enriched perspective that emphasized theoretical performance analysis, both worst case and probabilistic.

They also mention a third method of analysis of heuristics: empirical testing. In this context they mention that the effects of the inherent uncertainty that is present in many applications are still poorly understood. A particularly interesting remark is that (p. 265):

> the complexity of a problem may be caused by excessive assumptions (inherent in the way we choose to formulate combinatorial optimization problems) on the availability of precise information.

The authors probably mean that it may be hard for the computer to *solve* a problem which is formally specified as a combinatorial optimization problem. This remark is important because the reader can take it to imply that it may not even be worthwhile to *formulate* a problem as a formal combinatorial problem in the first place. Assuming less knowledge about a problem may both be more realistic and make a problem easier to solve. This is a supposition rarely heard among Operations Researchers. Usually one encounters the inverse argument, e.g. Silver et al. (1980, p. 153):

> (...) realistic formulations are likely to lead to mathematical problems which are very difficult, if not impossible, to solve exactly.

Unfortunately this last argument still seems to lead researchers away from 'realistic formulations', rather than making them wonder whether a mathematical optimization formulation is a 'realistic formulation' at all, and whether a computer should 'solve' the problem. In fact, the question is whether to define a problem as a formal problem or as an open problem.

AI and heuristics

Rich (1983) gives a good introduction to heuristics from the AI point of view. She places heuristics at the centre of AI (p. 37):

> (...) it should be clear that Artificial Intelligence can more precisely be described as a process of heuristic search. (...) Artificial Intelligence is the study of techniques for solving exponentially hard problems in polynomial time by exploiting knowledge about the problem domain.

Both Pearl and Rich define heuristics as problem solving *processes* as opposed to the *result*-based definition of optimizing methods. Heuristic methods are generally weaker than optimizing methods, i.e. the solution found by a heuristic is not guaranteed to be the best possible solution, but is merely guaranteed to satisfy certain constraints ('constraint satisfaction problems') or to meet certain minimum quality standards ('satisficing', Simon 1960). If an optimality criterion does exist but does not necessarily have to be reached as long as it is approximated sufficiently closely, the term 'semi-optimization' applies. Heuristics typically allow a better use of problem-specific aspects than optimizing methods. Often heuristic methods are highly customized to a specific problem instance. The other side of the coin is that this customization makes heuristics hard to apply to other problems.

Why use heuristics?

In the previous sections a number of arguments in favour of heuristics have already been given. Under which circumstances can heuristics be helpful, and why are they helpful? To begin with there are limitations to the power of formal models. Three possible limitations, ranked in order of increasing severity, are:

1. It is impossible to reach an analytic or iterative solution to a formally modelled problem. Existing techniques do not work or there simply is no existing solution technique.
2. A problem cannot be formally *solved*, although a formal model exists. Collecting and formalizing the necessary data either is not possible or would be too time-consuming.
3. It is impossible to formally *model* a problem. Usually this means that there is no problem owner able or willing to help formulate the problem formally, so that the modeller is left to guess about many aspects of the real-world problem.

The first two situations can be found in the literature, e.g. Silver et al. (1980, p. 154). As far as the third limitation is concerned, it is often simply neglected in the literature. Stainton and Papoulias (1984) address it when they plead for a 'relational approach' to designing heuristics. Taught by experience these authors state that starting an OR project by formulating a mathematical model of the problem at hand can be not only misleading but positively harmful. Instead, the problem owner and problem solver should work together closely and start with the decision rules that the problem owner currently uses.

The reasons for using heuristics that have so far been given are negative in that they stem from the non-applicability of mathematically stronger methods. There are also some purely positive arguments:

4. Even if stronger methods do exist for a given situation, heuristics can be so designed that they are much easier to understand by a problem owner than stronger methods (Silver et al. 1980). This can be crucial if a system is intended for use in practice rather than for research purposes.
5. Heuristics can be used as a learning aid, in much the same way as simulation models. They can help a problem owner to develop his intuition about a problem by allowing him or her to play with models or variables.

These two last arguments imply that there could be some evolutionary aspects to heuristics which are intended to support a problem owner. As the problem owner becomes familiar with the heuristic his wishes may change. In fact the likely evolution is from simple to more sophisticated heuristics.

It follows from the last two arguments that there could be special merit in interactive heuristics, because they would be especially easy to understand and to manipulate. The literature on this subject is discouragingly sparse, e.g. Ball and Magazine (1981, p. 216):

> The idea of using man-machine interaction within an algorithm came up on numerous occasions. It was generally accepted that little was known about this class of algorithms and that other criteria should be developed to evaluate interactive algorithms.

Most heuristics work incrementally, i.e. a plan is built step by step through actions which resemble the actions carried out during manual planning. If a heuristic works incrementally then it can easily be made interactive, at least in principle. Suggestions for the design of interactive heuristics are given at the end of this chapter.

The arguments in favour of heuristics that have been mentioned so far are rather broad. There are also some arguments which relate specifically to the search process. Generally heuristic search involves selective searching of a tree of possible solutions. Some advantages of heuristics over other tree searches are

6. Heuristics apply domain knowledge to prune the search tree before starting the search. This may save a lot of time, particularly so if the pruned tree reveals that the problem is not solvable in its present formulation.
7. Heuristics apply domain knowledge to guide the search into promising directions.
8. Heuristics enable the user to match the effort put into the decision making to the value attached to the result. In the words of Pearl (p. 15):

> Most practical problems are of a semi-optimization type, requiring some reasonable balance between the quality of the solution found and the cost of searching for such a solution. Moreover, because the search effort required for many combinatorial optimization problems can easily reach astronomical figures, relaxing optimality is an economic necessity. The basic problem in handling a semi-optimization task is to devise algorithms that guarantee bounds on both the search effort and the extent to which the optimization objective is compromised. A more ambitious task would be to equip such an algorithm with a set of adjustable parameters so that the user can meet changes in emphasis between cost and performance by controlling the trade-off between the quality of the solution and the amount of search effort.

These three arguments again imply that interactivity would be especially advantageous. If a heuristic were interactive, last-minute domain knowledge could be taken into account, which would reinforce arguments 6 and 7. Furthermore, providing an algorithm with parameters in such a way that the trade-off between solution quality and time consumption could be regulated by the user during the very solution process also seems very attractive.

Elements of heuristics

Pearl identifies three requirements for heuristic computer-based problem solving:

- *representation of the problem* as a database with objects. Objects include final solutions to the problem, elements of solutions, and preferably also partial solutions.
- *operators* transforming objects into other objects.
- *a control strategy*, determining in what order the operations must take place. This can include an evaluation function to determine the quality of an object.

Let us consider these three elements in some more detail.

Problem representation

The choice of a representation can be a matter of preference. Often a designer tends to use representations that are familiar to him or her. Sometimes a problem is best suited to a particular representation. Pearl (p. 26):

> Strategy-seeking problems are better represented by AND/OR graphs, where the AND links represent changes in the problem situation caused by external, uncontrolled conditions and the OR links represent alternative ways of reacting to such changes. (...) If the solution is a prescription for responding to observations, we have a strategy-seeking problem on our hands, and an AND/OR graph will be most suitable. If, on the other hand, the solution can be expressed as an unconditional sequence of actions or as a single object with a given set of characteristics, we have a path-finding or constraint-satisfaction problem and a state-space representation may be equally or more appropriate.

This implies that planning problems will be suited to a state-space representation if the plans to be generated are not likely to be modified during execution. If on the other hand the plans are subject to so many external influences during execution that continuous updating will have to take place an AND/OR graph (also known as problem reduction representation) may be appropriate. A problem with AND/OR graphs is that they are not so well suited to combinatorial complexity.

Another implication concerns the representation of the control strategy. If this control strategy is considered as a problem in its own right and automated support is designed for it then an AND/OR graph seems to be the indicated representation for the control strategy.

Operators

The choice of operators depends on the representation chosen. Generally operators manipulate 'elements' of solutions, transforming partial solutions into more refined or more complete partial ones until a solution is reached. But it is conceivable to use operators that can also make jumps, manipulating sets of objects in one operation.

Control strategy

The choice of a control strategy or search procedure is a subject of much research. The usual representation is to see the solution process as the traversal of a tree, starting at the root and ending in one of the leaves that represent admissible solutions. Branching points in the tree are called nodes, and exploring the alternatives in a node is called node expansion. The number of branches per node is called the breadth of the solution tree, and the number of nodes to be traversed from root to leaves is called the depth of the tree. Some generic search strategies are:

- *Generate-and-test.* In its simplest form this just implies trying one path after another until a solution is found. In humans this would be called trial and error.
- *Single-pass strategies.* Commonly these are so-called greedy algorithms, for instance *hill-climbing.* Hill-climbing consists of applying the operation which is locally best, until a solution is found. Hill-climbing is a fast strategy that works well if there exists a good selection criterion that keeps the search away from local maxima. However, it is irrevocable: there is no way to undo an operation.
- *Depth-first algorithms*, especially *backtracking algorithms*. These develop a single path from the root upwards but retrace their steps when a dead end is signalled. They proceed Last-In, First-Out, i.e. the elements that were added to the partial solution last are removed first when backtracking occurs.[16] A backtracking algorithm is powerful if a good criterion exists to prioritize nodes to expand, so that little time is spent in non-promising alleys. If the tree is of irregular depth a depth-first strategy risks spending much time in long and unpromising paths.
- *Breadth-first algorithms.* These develop all open paths simultaneously according to a First-In, First-Out mechanism. As long as a path is not found to be a dead end it is kept open. Therefore a breadth-first algorithm is guaranteed to find the shallowest solution possible. In a tree with a high branching degree and paths of equal depth breadth-first can be very inefficient. Specializations are cheapest-first or best-first strategies. Best-first is a version in which there exists a global evaluation function which is used to choose the most promising from all the nodes explored so far. In fact it is a mixture of depth-first and breadth-first search. A variant which has much been used in OR is branch-and-bound (Lawler and Wood, 1966) in which the search starts breadth-first but as soon as two paths lead to the same partial solution only the best of the two is pursued.

In practice a selective combination of search strategies is often advisable. For example, a selective breadth-first search may be chosen, in which not all nodes are expanded; or hill-climbing may be supplemented with local backtracking (Verbraeck, 1990[b]). Recent research in AI includes the use of techniques to make the search less rigid. Freuder

[16] A readable tutorial on backtracking programming techniques can be found in Bitner and Reingold (1975).

(1989) discusses the concept of partial constraint satisfaction for cases in which it may not be possible to satisfy all the constraints.

The choice of a strategy depends on characteristics of the search space, and in the case of an interactive algorithm, of the user. Humans are good at abstracting detailed information into high-level constructs such as 'promising' or 'too risky'. To automate this type of search strategy-directing notions is quite hard. High-level constructs for describing the characteristics of a search space are e.g. introduced by Fox et al. (1989) who, elaborating on previous work, present the notions of problem topology, problem texture, and problem objective.

A less formal concept is 'opportunistic reasoning' as discussed by Ow and Smith (1986). They define opportunistic reasoning as

> (...) a problem solving process whereby system activity is consistently directed towards those actions that appear most promising in terms of the current problem solving state. It is motivated by problem domains where it is necessary to balance the combinatorics of a large search space with limited processing resources.

According to this definition opportunistic reasoning looks much like human problem solving. The core is that a problem solving action to be carried out is chosen at a fairly high level of abstraction. Indeed, opportunistic reasoning has been proposed as a general model of planning in humans and computers by Hayes-Roth et al. (1979) and Hayes-Roth and Hayes-Roth (1979).

A complex search problem may require the use of a mixture of search techniques. This introduces a control problem: when should which techniques be applied? Opportunistic reasoning can be seen as a way of dealing with this control problem. It can be very powerful if a reliable measure exists to identify promising search directions.

2.4 Literature: other schools

Decision Support Systems

In many respects the field of Decision Support Systems (DSS) ten years ago resembled that of Operations Research in its first decade. Here again a group of result-oriented scientists is striving to improve the functioning of organizations through the application of a set of concepts and techniques. Many of these techniques are drawn from OR (Gorry and Krumland 1983, p. 206):

> Much of the work in DSS constitutes the imposition of those [OR] models on various aspects of managerial decision making.

The DSS community has worked on both strategic and tactical to operational problems. A substantial part of DSS theory and experiences is valid for both categories.

Early seminal DSS texts are Gerrity (1971), Gorry and Scott Morton (1971).

Reference works are Keen and Scott Morton (1978), Alter (1980), Bennett (1983), McLean and Sol (1986), Sprague and Watson (1989). 'Decision Support System' has been a 'buzzword' from the late seventies on, which has led many a researcher to re-label existing research directions and systems as DSS. The history and goals of the DSS movement were e.g. described by Bots (1989, p. 3). During the last years an increasing number of related definitions have seen the light of day, such as expert systems, executive information systems, expert support systems, interactive planning systems, and the like. Bots points out that it is futile to attempt a rigid definition of DSS and related terms. He does, however, give a definition, the one proposed by Sol (published in Sol 1991) who generically defines all these types of systems as 'Information Systems to support Decision Processes' (ISDP). This definition is appealing because it disregards implementation and focuses on the goal instead, which is to support the decision making process of an individual, group or organization. The definition indicates that all research and practice on both information systems and decision processes is in principle valuable for the design of this class of systems. Unfortunately, disagreement about the terms information system and decision process is to be expected, and the NAG[17] prevalent in the field is likely to continue to lead to new definitions.

One of the particular contributions of the DSS community is the explicit attention to the decision making or problem solving activities of the user of a DSS which has been advocated by many authors, especially in the early eighties. This perspective has led to substantial results in the form of guidelines for DSS development. In chapter four I shall come back to DSS research in the context of evaluation of DSS. At this point it is appropriate to mention the contribution by Verbraeck (1991), who shows how the concept of ISDP can be applied to production planning situations.

Simulation

Simulation is the activity of modelling some real system with a view to experiment with the model in order to learn about the modelled system.[18] If one takes a broad view this definition classifies many planning systems as simulation models. As soon as the system contains a model of the real-life planning situation (which is always so if 'model' is not taken too narrowly) and the user can input data to get 'what-if' type of answers (which is often the case) the system can be called a simulation model of the planning situation.

The scientific simulation community is large and diverse. A model is a simplified representation, and several types of representation exist depending on the characteristics of the modelled system and on the objectives of the modeller. A major division is between discrete event models (Mitrani 1982) and continuous models. Within the scientific simulation community a simulation model is implicitly taken to be dynamic, meaning that it represents the behaviour of the system over time. This means that a static

[17]'New Acronym Greediness'

[18]Reference works on Simulation are e.g. Neelamkavil (1987), Kettenis (1990).

representation, e.g. a spreadsheet model, would not be called a simulation model.

The applicability of simulation in practice is limited by lack of knowledge about the modelled system, particularly causal relationships between entities. Also, tools for modelling are not yet ideal; they seldom allow trial and error modelling. The most important limitation is that it is quite difficult to establish the validity, i.e. the correspondence between the model and reality, of a simulation model of a system involving human actors.

Systems perspective in simulation

From a systems science point of view, a simulation study contains three elements: input, system (i.e., the model), output. Two of the three are known, one is unknown. Three situations can arise (Elzas 1980). In order of increasing complexity these are

a. Prediction: knowing the input and system, predict the output. In DSS terms this would be called what-if analysis.
b. Identification: knowing the input and output, establish the nature of the system. In DSS development, the system is given, though perhaps not unambiguously: it is the decision situation.
c. Management: knowing the system and the desired output, determine the needed input.

According to Elzas, the situation in a management context where the input is the unknown element is the hardest to solve. Be that as it may, simulation can be very relevant to automated systems for supporting planning problems (e.g. Sol 1982, Verbraeck 1990[a], 1991). Simulation can be used in several ways. The simplest way is to have a simulation model answer what-if questions. In fact, this is almost a restatement of a definition of DSS by Simons quoted in Hofstede (1990[c], p. 139):

> Decision Support Systemen worden erdoor gekenmerkt dat ze de gevolgen van beslissingen interaktief en momentaan doorrekenen, waardoor deze gevolgen in hoge mate bekend zijn alvorens de beslissing daadwerkelijk genomen wordt.[19]

Obviously the value of such decision support is dependent upon the validity of the simulation model, that is, the extent to which it correctly models the behaviour of the aspects of the real system that it is intended to describe. The concept of validity has not received as much attention from outside the simulation community as from within, with some exceptions (e.g. Verbraeck, 1990[a]).

A more sophisticated what-if like application of simulation to planning problems is to feed the plans generated by or with a planning system into a simulation model of the object system. This can be useful in situations with high unpredictability and complex interaction between variables. Repeated simulation runs can generate an insight into the

[19]It is characteristic of Decision Support Systems that they compute the consequences of decisions interactively and instantaneously, so that these consequences are known to a large extent before a decision is actually taken.

robustness of a plan.

However, there are limits to the applicability of simulation models. There are optimists, e.g. Larsen and Alting (1990) who are of the opinion that simulating real-time dynamics of scheduling environments is promising. Yet simulation cannot diminish unpredictability or complexity inherent in a planning situation. In an unpredictable real-world situation a simulation model can at the utmost be used for tentative purposes. Fortunately, even if it does not yield reliable predictions, a simulation model can be useful. Elzas (1980, p. 14:)

> In the management of change, the witness and the steersman is more interested in the structural aspects [of a model] than in accurate predictions whose background he cannot fathom.

Although this quotation was made in the context of problems of international scope I feel that it is also relevant to tactical planning problems. It indicates that the contribution of a planning system to the planner's understanding of the situation may be valuable even if the plans are imperfect.

2.5 What do the research schools have to offer?

After this treatment of Operations Research, Artificial Intelligence, Heuristics, and - briefly - DSS and Simulation, the question can be asked what support for planning each of these schools has to offer.

Each of the four disciplines deals with what Sol would call 'information systems to support decision processes'. From the beginning the DSS community has emphasized that interactivity is an important feature of a DSS, and that as a consequence the interface is an important design issue. Other research communities have paid much less attention to this. Interaction between user and system has been positioned either before or after running a model, which has typically been 'monolithic': large and running without user interruption. Models have been the central design issue. Decision situations, c.q. planning situations, have been treated as formal rather than open problems.

OR and AI have done quite some work on formal planning problems. This has yielded a large body of theoretical knowledge. The importance of simulation in decision support is that a simulation model with a proper interface can be a good tool for 'what-if' analysis.

Types of models

Planning situations can be formalized according to different paradigms. For some planning situations the choice of a problem formulation is obvious, for others much less so. The following types of problem formulation are frequently used:

- Equations model. The situation is formulated as a set of equations in which different types of variables play a role: external (or uncontrolled), internal (or controlled) and dependent. Often an optimality criterion is defined in terms of dependent variables. This approach has much been used in Operations Research.

- Search graph. The solution process of the situation is modelled as a graph, often a tree, which has to be sequentially searched for solutions. This is an approach much applied in Artificial Intelligence.
- Constraint satisfaction model. The situation is formulated as an n-dimensional space. Constraints on the values of dimensions or combinations of dimensions limit the number of possible solutions. A solution is any element of the remaining subspace. This type of formalization borrows from OR and AI.
- Simulation model. The system in which the planning situation occurs is modelled. The model can be an equations model or it can be of a different type. This approach has a tradition in the field of simulation and modelling.
- Spreadsheet model. The situation is formulated rather loosely in quantitative, usually financial terms. This type of model has much been used in decision support systems.

Except for the search graph, these types of models are declarative rather than procedural. The question whether a declarative model, usually in the form of an equations model, is appropriate for DSS situations, is controversial. As far as equations models are concerned, their proven success in OR might make this question seem pointless. It may well be, however, that in some domains less formal modelling paradigms are more suitable. Such a proposal is e.g. put forth by Bosman (1986). He argues that in a situation of imperfect knowledge, as is almost invariably the case in organizational settings, process models are more appropriate than equations models. This is among other things because process models are more robust under imperfect knowledge of data or problem structure (see also Hofstede 1990[a]).

Frowein (1990, pp. 17 ff.) elaborates on the distinction between process models and equations models. Process models, he defines, are descriptive rather than normative, they can employ a variety of representation techniques and they must contain a time dimension in some way. Frowein does not specify whether this time dimension refers to the time in the modelled system or to the sequential nature of the process of problem solving carried out with help of the model. The former is probably intended. If so, a constraint satisfaction problem which includes time as one of its dimensions is a process model. If one took the latter definition of time, a process model would become a model of the solution process, or in other words a strategy model. A search graph is an instance of this.

Regardless of how models are defined and what types of models are employed, many authors agree that a model base is an essential constituent of a DSS. Gorry and Krumland (1983, p. 206):

> To a large degree, the success of a DSS depends on the recognition of a structure suitable for the intellectual task in question or on the creation of such a structure.

In practice the school to which a DSS designer belongs limits the choice of 'structures' at his or her disposition. This can lead to poor solutions according to Moscarola, who argued at an IFORS[20] Conference (1981, p. 101):

> La prise en considération du contexte rend alors nécessaire le renversement de la proposition "la méthode ou le modèle permet de définir le problème", bonne caricature de comportements fréquents par la proposition "le contexte oriente le choix de la méthode ou du modèle". Cette dernière attitude suppose la maîtrise de méthodes multiples et l'absence de dogmatisme méthodologique.[21]

The choice for a modelling paradigm should be made consciously, for fear of the error of the third kind (Mitroff and Featheringham 1974): the error of solving the wrong problem. I agree with the authors mentioned in the above article (p. 383) who make the obvious but sometimes neglected remark that:

> one of the most important determinants of a problem's solution is how that problem has been formulated in the first place.

Model ingredients

With the exception of DSS, the research schools reviewed above have in common that they take a given planning situation, formalize it, and build computer models of the formal representation. They differ as to the problem representations they use for modelling, the assumptions they make about the planning situation and the ambition level of the models.

Two similar lists of model ingredients have already been mentioned in this chapter: the requirements for a model by Van Hee and those for a heuristic by Pearl. Combining and rephrasing these, one can say that any model of a planning situation must include the following elements:

1. A 'world model' or a description of relevant dimensions, be they variables (including time) or objects.
2. A description of the limitations or constraints on the values of the dimensions, on actions or on combinations of values.
3. A description of actions.[22] These actions change an unplanned world of variables or objects into a plan.

---

[20]IFORS is the International Federation of Operational Research Societies.

[21]When taking into account the context it becomes necessary to replace the proposition "the method or the model allows to define the problem", which is a good caricature of frequent behaviour, by the inverse proposition "the context directs the choice of a method or of a model". This latter attitude presupposes proficiency in multiple methods and absence of methodological dogmatism.

[22]These are often called 'assignments' in the sense in which this word is used in programming languages: allotting a value to a variable.

4. A description of a strategy for using the actions to solve the problem, that is, attain the goals.
5. A description of goals, or quality criteria, in terms of the dimensions.

Van Hee's and Pearl's lists of model ingredients and the above list contain equivalent notions, but with differences in terminology and emphasis. Figure 2.1 below displays side by side the equivalent notions in the three lists.

Van Hee	Pearl	Hofstede
state space	database	world model and constraints
action space	operators	actions
process mechanism	control structure	strategy
evaluation figures	-	goals

Figure 2.1: related concepts in modelling for decision support.

Figure 2.1 contains one empty box: in Pearl's list, goals are implicit in the control structure. Apart from this, the only difference between the three lists is the world view behind the terms. Van Hee uses an OR-like paradigm. He assumes that essential characteristics of the situation can be abstracted into a mathematical model, including a variable-oriented world model, description of constraints, and formulation of a goal function. The nature of the actions is not important. AI-oriented scientists, among whom I count Pearl, adopt an object-oriented state-based world model. Constraints are either incorporated into the world model or into rules. Actions are explicitly performed as operations. Goals are used by the control structure to determine the priority of actions. My own list is purposely kept non-formal, reflecting the view that the model ingredients need not be fully automated.

Typically, OR oriented models have trouble dealing with uncertainty and multiple constraints while AI oriented models tend to become complex and time-consuming when used on real problems. Both types of systems are therefore hard to use in practical situations. A combination of AI and OR methods improves the situation.

DSS researchers tend to impose few constraints on what they call a model, as long as it is helpful. They have stressed the importance of the user's decision making process as a guideline for modelling. As a consequence, they argue in favour of active involvement of the user (e.g. Simons 1987).

The uniqueness of simulation is that in this field the emphasis is on modelling the system in which the planning situation occurs rather than the planning situation itself. This means that what Wilkins (1988) calls the frame problem in AI is included, i.e. changes in the system caused by planned actions are included in the model. Thus, making a simulation model of the system is an ambitious venture but potentially a very rewarding one, because if a valid dynamic model could be derived one would be able to use it as a stand-in for the real situation. This would greatly add to the predictive and adaptive capabilities of a planning system.

Strictness and looseness

Combining methods from these four disciplines can clearly be of value for building planning systems for practical applications. The techniques used in OR, AI and simulation can complement each other. There remains a problem of 'strictness' of the modelling techniques used, which endangers the usability of the resulting systems. This is a notion which I have not come across in the literature. It might be called 'formalness'. If a model is *strict* then the user cannot circumvent the model. The degree of strictness draws a distinction between the usual way of problem solving by humans and by computer programs. Formal representations of uncertainty, probability and fuzziness do exist but even these are strict in the sense just mentioned. Not surprisingly, the opposite of strict is called *loose*. For example, if a DSS developer wants the user to follow certain steps, advice in a user manual is a loose strategy for obtaining this result whereas program code is a strict strategy.

The distinction 'strict' versus 'loose' also reflects the difference between the attitude towards problems taken by academics and non-academics, and that could be why it is not found in the literature: no academic author sees any point in loose, sloppy models. There are very good and obvious reasons for this attitude, both from an academic point of view and from a user's point of view. After all, the user of a model wants to have something to hold on. The fact remains that decision making is generally a loose activity, and that the communication between a strict model and a loose user is problematic.[23]

A development approach oriented towards the user rather than the model, as proposed especially in the DSS world, can lead to systems that are better adapted to the user but it cannot alter the techniques used. As a result there might be a misfit between the user-adapted and the model-based components of the system. If this is the case and if the misfit cannot be resolved, user-oriented system components will probably prevail and the model-based components might be altogether abandoned.

There seems to be a parallel evolution of the research schools from loose beginnings towards a strict end. At least in OR and DSS there is a tendency to move from an initial goal-directed motivation towards more and more technique-oriented specialization. Such a development is detrimental to the applicability of the results. Perhaps it only reflects the division of the field into 'pure researchers' and 'application builders', of which the

---

[23]Consider the perceptive remark made by Beer (1981, p. 53): "The strange thing is that we live our lives by heuristics, and try and control them by algorithms."

latter group does not publish. Such a gap was already signalled and denounced decades ago in OR (e.g. by Ackoff) and ten years ago in DSS (e.g. by Keen), but apparently the process is hard to reverse.

To my knowledge, there are no fully automated tactical planning situations. It follows that the properties of human planners are relevant to such planning situations. A planner has a certain style and education. Furthermore, he may be ill or have a Monday hangover. Also, the planner may have his own private objectives which interfere with a strict formulation of the problem.

Constraint manipulation

Model-based components do not accommodate the flexible, context-dependent constraint handling of a human planner. For instance, a pot plant grower working on his cultivation plan might include a certain production lot because 'it has potential'. This is how he thinks of it, and he does not formalize it any further. When prompted for an explanation he gives a non-formal description including any number of variables and expectations of various natures. A model judging the attractiveness of the same production lot would yield a more explicit, but less rich judgement.

Constraint manipulation is not restricted to the on-line interface. The interaction between user and planning system is only one of the many planning activities which the planner carries out. Other possible planning activities include creating or dropping constraints by asking for additional information, and changing constraints by manipulating the organization (e.g. altering holiday dates).

The above considerations imply that in the design of a planning system it is important to specify which constraints will be modelled into the system, so that the system will be able to enforce them, and which constraints will not be modelled, so that the user will be in charge of monitoring them.

Besides deciding on which constraints will be system-guarded, one has to decide how to guard the constraints. By analogy with the database world (e.g. Hofstede 1990[c], p. 106) one can think of three levels of strictness at which to guard constraints:

- immediate.
  The constraint can never be violated. If an operation were to result in a violation, this operation will not be carried out.
- deferred.
  The constraint is checked after a series of operations which together form a logical unit, e.g. one week, one department. If a violation is found, the system will engage in some corrective action, or give a warning message to the user.
- afterward.
  The constraint is not checked until the entire planning session is over. If a violation is found, this will usually result in a warning message rather than in re-planning.

Hard and soft constraints

The order immediate, deferred, afterward is an order of decreasing system involvement and increasing user involvement in monitoring a constraint. Thus the harder a constraint is in terms of the opposition 'hard' versus 'soft', the more it seems appropriate to have the system guard it. However, as soon as immediate checking of a constraint is designed into a system it becomes impossible for the user to 'mess around' with that constraint. No 'loose ends' can be left hanging for a while and this can be an impediment to his creativity or his problem solving process. The moral is that, in an interactive planning system, even constraints that must hold in the final plan had better not be designed into the system as hard constraints during the creation of plans.

The usual way to enforce soft constraints is to incorporate them in such a form that the user is free to choose the level of enforcement and the relative importance of different constraints. Such a checking of constraints by the system limits the search space of the system. This is normally considered a good property, but it can also have drawbacks, such as limiting the freedom of the user and thereby his creativity and his learning about the planning situation. To get the best of both worlds, one could leave it to the user which constraints the system should check. One can expect that with experience a planner will specify more and more soft constraints to the system as his confidence grows.

2.6 Application-oriented literature

Research and practice

There is a considerable gap between academic research on planning problems and what happens in organizations. It is not easy to get evidence for this because planning systems in organizations are seldom documented, let alone published about in scientific media. Developers of planning systems do not only have the complexity of the planning problems *per se* to deal with. Besides, they face any number of complicating circumstances. These include such things as resistance of users against changes, lack of time of users because of other duties, hidden agendas, power struggles in which the developer is made to participate. Furthermore a developer is likely to view system sophistication quite differently from a user. Consider the quotation from the conclusion of Van der Ven's (1989) doctoral dissertation on a group DSS:

> (...) our design approach was explicitly to keep the system 'simple and sufficient'. In an internal Europhar publication, the system is described by Europhar as 'advanced and "sophisticated" '.

In a discussion about the impact of information technology on managerial decision making at farms, Schiefer (1991) draws a clear distinction between research and practice. He observes the same phenomena throughout Europe: information technology is in a pioneering phase, and fails to fulfil its promises because developers do not realize that a research product is not yet a usable system. Specifically, a usable application of information technology must, according to Schiefer (p. 11) possess the following characteristics:

- it has to fit neatly into the decision and management behaviour of farmers;
- it needs to have fitting links with communicating data sources and databases with their corresponding technical devices;
- it has to communicate with the farmer through an interface which fits his problem perception, his intellectual capacity, his informational and educational background and his preferences for input/output devices;
- it has to be embedded in a data collection and processing system which fits the data availability and the processing needs in terms of time, quantity and data format;
- if the application reaches beyond a farm border, it has to fit an existing organizational infrastructure and responsibility system or requires appropriate developments.

The key notion in this list is *fit*. Schiefer pleads for extensive usability testing in the field before putting an information technology application on the market. One might say that for developers of information systems in agriculture, Schiefer's law is 'survival of the fitters'.

Referring to the Dutch situation, Geuze (1991, p. 135) remarks that to achieve successful diffusion of information technology, the users, in his case the primary agricultural entrepreneurs, must be actively involved. It is they who must direct and coordinate developments.

Verbeek (1991) is an exception to the rule that researchers do not mention troubles between users and developers that occur during the development of decision support systems. Verbeek was involved in the development of DSS for different planning tasks at KLM airlines. The problems that occurred mainly had to do with communication (or, rather, lack of communication) between the parties concerned. Expectancies were unrealistic, which caused disappointments. Time and again functional requirements specified in advance and correctly implemented did not result in a satisfactory system. The troubles were eventually overcome and it became clear what had been wrong: the open-ended nature of the project as a DSS project had not been recognized.

Published and unpublished cases

The conference on 'Expert Planning Systems' held in Brighton in 1990 yielded a number of descriptions of organizationally implemented systems (Institution of Electrical Engineers 1990). The systems presented were very different in architecture, in techniques used and in application domain. There were also common traits. Typically the applications were simpler than the designers originally intended, did not use advanced concepts but used creative findings suited to the problem context, worked in a way similar to a human planner, emphasized reactivity rather than fixed planning, were highly interactive, were integrated with other information systems and were written in conventional languages, sometimes after prototyping in AI software. The users were typically satisfied as soon as the system could take over tedious, time-consuming aspects of the planning task and could guarantee that hard constraints were not violated in the plan.

This conference was no exception to the rule that the scientific justification gets into the written paper much more easily than the commonsense experiences. This is why references are harder to give in the present section than in the previous ones.

Apart from the general trends some specific points are worth mentioning. Cook (Cook et al. 1990) presented a case-based system for a two-dimensional spatial planning task. She noted that the experience accumulated in the case-base was especially helpful to novice planners, whereas experienced planners just used the system for drawing. De Waele (De Waele 1990) presented an ambitious system for job-shop scheduling, called ReDS, with a modular architecture in which some good design ideas are incorporated, such as hierarchical decomposition of the overall problem, and information abstraction. These two features enable the user to choose a desired level of detail (p. 12):

> Detailed assignments are made only for a limited horizon. This horizon is a function of the uncertainty prevalent in the environment and can be determined by the user.

De Waele is one of the few to refer in written form to implementation problems. The last lines of his article are (p. 15):

> Finally I would also like to thank the users of the prototypes in Siemens Munich who showed remarkable good-will and patience during the first implementation stages.

Ready (Ready et al. 1990) emphasized that the benefit of a knowledge base is not automating or modelling a planner but enabling the planner to play with the knowledge, thus enabling him to improve his understanding of the problem.

Summarizing the findings from this conference in Brighton, one can say that a usable Expert Planning System is first and foremost an expert's planning system.

Incremental problem solving

Angehrn and Lüthi (1989) emphasize that playing with knowledge about a planning situation is important to promote learning about that situation. While describing a DSS generator for geographical problems they stress the importance of incremental problem solving (p. 7):

> The support of modelling as an *incremental process* is the third basic characteristic (...). Allowing the user to incrementally build, complete and test his models not only makes the man-machine interaction much easier and concrete, but as the modelling process mainly stimulates *learning* about the unstructured situation under investigation, an important benefit also results from directly involving the decision-maker in this phase and thus supporting a gradually evolving learning process.

Applying the argument given by Angehrn and Lüthi would in many cases probably lead to giving the planner a spreadsheet package, perhaps with some added modelling capabilities suited to the problem domain.

Iterative problem solving

A notion close to the one of incremental problem solving is that of iterative problem solving as discussed by Geraedts (1986) and by Simons (1988). Simons describes VLUCHT, a DSS for timetabling problems. In this DSS the solution method itself is not interactive. However, explicit attention is given to the user's problem solving process. The user evaluates a solution or failure of the system and reformulates the problem on the basis of this evaluation by refining (adding constraints) or relaxing (removing constraints) the problem formulation. In tests among users it appeared that expert users of VLUCHT were experts especially because they chose the right reformulation of the problem, resulting in a quick iteration towards a good solution. Simons calls the knowledge required for this expert reformulation 'domain knowledge'. The domain in this definition is the combination of the timetabling problem and the DSS, and (Simons 1988, p. 17):

> Solution strategies for the formulated problem may differ considerably from solution strategies for the original real life problem. The user caters for this discrepancy by choosing alternative search directions when controlling the heuristics. This process of choice leaves room for implicit domain knowledge.

Between the lines of Simons' article it is suggested that a DSS creates a novel problem solving environment in which a user has to go through a learning process in order to become an expert. Simons, by the way, explicitly discusses the feasibility of automating an expert's problem solving knowledge. He concludes that there are three categories of problem solving knowledge:

1. Rules that always work. These are candidates for hard-wiring into the problem solving mechanism of the DSS.
2. Rules that usually work. Cases that fall within this intermediate category are candidates for inclusion into a knowledge base. In the case of VLUCHT, few such rules were found, but those that were found were important enough to make inclusion into a rule base possible and meaningful. The purpose would be to instruct non-expert users.
3. Rules that incidentally work. These can only remain non-automated.

Incremental algorithms

An incremental problem solving process can also be modelled within an algorithm. A DSS for timetabling based on such an incremental algorithm is presented by Verbraeck (1990[b]). The algorithm was designed by trying to model the human timetable maker.

Verbraeck does not forget to mention (p. 208):

> Manual lesson-planning has a number of advantages over automated planning. The planner has a clear understanding of the planning situation. Difficulties can be tackled in parallel. The planner knows where the initial information can be altered without creating trouble. The planner is able to identify partial problems, blocking lessons and

> small puzzles that can be solved apart. The time it takes to make a schedule can be controlled by the planner. The criteria the schedule has to meet can be changed when the process threatens to take longer than expected.

Important remarks here are that the planner is good at problem decomposition given a particular problem instance and that he knows which aspects of the problem are 'open'. But obviously, manual planning also has numerous drawbacks. The main ones are inconsistencies and errors, problems with accommodating changes in plans and excessive time consumption.

Verbraeck uses the computational problem characteristics to deduce that a conventional depth-first search will not yield acceptable run times and the 'organizational' problem characteristics to state some requirements for the algorithm in his DSS (p. 209):

1. The search time has to be both limited and under control of the planner.
2. In the allotted time, a schedule must always be presented, whether all lessons are included or not.
3. With growing search time, the quality of the schedule has to rise and the number of unplanned lessons has to fall.
4. The algorithm must be 'inconsistency proof': inconsistencies in the data should neither take much time to find, nor stop the algorithm.

The algorithm tries to model the human planner as closely as possible. It quickly creates a first schedule on the basis of a 'heuristic order function' which puts lessons in order of decreasing difficulty and then proceeds to make small alterations in this first schedule, guided by a 'heuristic planning function' to evaluate the quality of the current solution. It leaves great flexibility for the human planner to take control, e.g. by pre-scheduling certain difficult lessons or by fixing partial solutions. Both heuristic functions are simple linear equations of which the parameters can be easily modified by the planner. The resulting system works according to the requirements and is being commercially exploited. Planners typically use it in three steps. First they generate a 'quick-and-dirty' schedule, which they use to get an idea of where the bottlenecks are. They try and solve these bottlenecks by either manually solving parts of the problem or making phone calls to change the problem. Then they make a few longer trials. The best schedule obtained so far is left to be refined by the algorithm over the weekend in a third and final step.

Role of the planner and practical results

McKay et al. (1989) do not mince their words in an article which shows much awareness of the significance of interaction between planner and system in job-shop planning. For instance (p. 172):

> In general, the world of scheduling remains as it was in the fifties and sixties. Neither Operations Research nor Artificial Intelligence has made a significant impact on how scheduling is done in the real world.

and some lines below

> The first generation AI scheduling approaches have been in the Expert System category or they have used (...) constraint directed searches. (...) The major goal of the systems has been to create a 'good' feasible schedule. This has been done using a problem definition that is virtually unchanged from traditional Operation Research.

This last remark is essential. McKay et al. propose to use another problem definition. An investigation among over 300 human job-shop schedulers showed that not one of the planning situations possessed any of the following 7 properties:

1. well-known and stable manufacturing process.
2. simple goals not affected by hidden agendas.
3. predictable and reliable set-up and processing times.
4. relatively short cycle times to allow most work to start and be completed without interruptions.
5. accurate and complete drawings, routings, and bills of materials.
6. reliable, stable, and accurate forecasts of product demand.
7. known quality, quantity, and arrival times of the materials.

This means that in almost all actual scheduling situations the hard data or knowledge which fully automated planning systems require simply do not exist. Both objectives and constraints change continually. The data and knowledge that do exist are all uncertain or vague to some extent. This is precisely the reason why in such scheduling situations the planner is typically a highly experienced person who has an intuition about his scheduling problem gained from experience.

McKay et al. propose (p. 171):

> a preliminary model for an integrated scheduling system (...). The human scheduler is considered to be the major component of the system and is integrated with a domain manager that maintains a knowledge base of measures, rules, objectives, constraints, and results of past scheduling activities. A schedule comparator uses the measures, degrees of relaxation, etc. in the knowledge base to define a scheduling problem suitable for mathematical analysis. The schedule comparator invokes a schedule generator until the schedule is 'good' enough. The human scheduler is tracked as the generated schedule is modified. The expert system attempts to capture the reasons for the changes and thus enhance the rule base for measuring the quality of the schedule.

The 'schedule generator' which performs the actual scheduling is not the focus of McKay et al. They assume that an indirect interface of the planner with this schedule generator by means of a knowledge base and a 'quality-rule base' will allow the planner to be 'the major component of the system'. Their approach resembles Simons' iterative problem solving approach which is discussed above.

McKay et al. list the following requirements for a job-shop scheduling system:

1. The planners have confidence in accuracy and reasonableness of the generated plans.

2. The system uses industry-specific terminology.
3. Sufficient constraint relaxation is provided by the system to allow for dynamics and uncertainty in plan realization.
4. The human scheduler is in control and is able to violate and relax any constraint.
5. The system will accommodate the changes made to objectives and constraints over time.
6. The system is as flexible as the manual methods.
7. The system will be used as a tool for prediction and feasible loading.

These are severe demands, especially the fourth and sixth.

The authors speculate that the benefits of a system such as they describe will mainly be twofold. The system will on the one hand contribute much to rapid learning by novice planners, and on the other hand offer a better base-line schedule for experienced schedulers. The authors warn that blind confidence in the schedules generated by the system is never justifiable.

2.7 Whose planning problem is it?

The previous sections have made clear that there is a tension between academic research and practice. Although academic research certainly produces usable planning systems this is rather an exception than the rule. A few circumstances contributing to this predicament have already been mentioned, notably the strictness of the models used by academic developers and the lack of possibilities for the planner to introduce domain knowledge into the planning process. Both these causes point at a misfit between developer and user. I shall now discuss this issue at more length.

Modesty in modelling

A feature which is present in much of the research literature reviewed in this chapter is that planning situations are abstracted from their organizational context early in the development process. Developers of systems adapt them to a set of techniques, and then to the requirements of programming tools, before the formally modelled problems are fed back into the organization. In this development process there is but a limited role for the eventual users of the system. There is a high risk that the resulting system will be alien to the users and to the user organization.

It is not so easy to change this state of affairs. A developer has to get well acquainted with a problem if he is to make a planning system for it. A developer is motivated to make an elaborated, sophisticated system, using criteria from his own practice.

A prototyping approach in close cooperation between developer and user would seem to be a solution, but problems remain. User and developer do not speak the same language, especially if the models used are specialised and complex. Besides, user and developer do not perceive the same aspects of a planning situation. A developer is trained to recognize a formal declarative problem formulation with structure and rules. A planner is trained to see not a *declarative* formulation but a planning *procedure* with rules of thumb and with informal assessment of risks and exceptions.

If the developer is a researcher a further complication is that he or she is supposed to publish, preferably about a sophisticated system. It is generally easier to publish in a respected scientific journal than to get a planning system accepted by a user. An important step is for developers to realize at all times that the problem is not their problem but the user's problem. A developer should not take it for granted that a planning situation must be modelled exhaustively. Indeed, pragmatic developers are modest and leave parts of the problem unmodelled, allowing the user to fill in these loose ends explicitly or implicitly during planning. These parts of the problem mainly concern the constraints and the goals. This is because real-world constraints and goals are especially hard to formalize due to their low strictness and their context dependency. Such a design will result in a highly interactive planning system.

Open and closed models

If a DSS requires that a problem be specified in full detail before the system can be used this may make the system poorly usable. A model in such a DSS can be called a *closed model*. The opposite, an *open model*, is a model which the user can easily adapt, while working with the DSS, to possibly unexpected elements of new problem instances.

It is easy to see that non-automated techniques for dealing with planning situations, such as pencil and paper, are open because they are loose. But automated tools use models which are strict. Can they be open? The case of spreadsheet packages shows that the answer is affirmative. But how about algorithms for formal planning problems, can they also be open? This would mean that the planner was able to supplement the model with unanticipated information during planning. In other words, interaction between planner and system is the vehicle for openness.

Closing the gap

There is a gap between theoretical advances and practical results. The cause for this lies at the heart of the matter: the models that researchers take as a basis for their work poorly fit the actual planning situations. As a result, planning systems developed by researchers have had less practical impact than they might have had. Theoretically there are two ways to overcome this gap. One is to change the definition of the planning situations, the second to change the models.

Successful practical planning systems are generally very simple from a scientific point of view but well-adapted to the organization. Most of these systems give low-level support such as memory aid, drawing facilities and the like. This represents the first option for closing the gap: changing the definition of the situation to make it simple enough to be manageable. One could, by the way, go further and change not just the definition but the actual planning situation; if an organization cannot cope with its planning situations then organizational adaptations may be just as easy to realize and more profitable than elaborate planning systems.

However, some complex planning situations happen to exist and will have to be put up with. For these, there is a need for systems that offer more sophisticated support as well, but only if it is given in an optional, interactive, planner-controlled manner. This means choosing the second option for closing the gap: change the type of model. This is the option which is central to the planning system concept put forward in this book. A number of research directions are possible. Existing algorithms and other modelling techniques can be evaluated for their applicability in interactive, user-controlled systems and adapted accordingly. New, open, incremental methods can be developed. Synergy between research schools can be enhanced, in the result-oriented spirit characteristic of the early years of Operations Research. If researchers adopt this user-centred perspective, I trust that many advances can be made in combining usability with model-based support.

Yet I suspect that these things will be very hard to put into practice, though not for intrinsic reasons. Rather, impediments are the organization of the research world with its emphasis on theory, publication, teachability and formal problems, and the misfit which seems to exist by definition between the problem of a planner and the problem of a planning system developer.

2.8 Suggestions for usable planning systems

So far, the findings of this chapter are that the energy of planning system researchers is overly directed at developing elegant techniques and models, and insufficiently at developing usable planning systems. However, there are exceptions to this rule, some of which were also mentioned above. Some researchers have been aware of the fact that usability is the bottleneck and have acted accordingly. These researchers have come up with ideas about the desirability of possibilities to work interactively with a planning system. In this section I intend to discuss two ideas for making usable planning systems. These are *interactive heuristics* and *problem decomposition*. For now I only present the ideas in general wording. Both ideas were used in the PROPLAN case study and the result is presented in chapter five. Readers to whom the next sections do not appeal might feel more satisfied after reading that chapter.

Interactive heuristics

It has been pointed out in the section on heuristics that they are a potentially user-friendly type of computational support, and that interactivity of a heuristic can have a number of advantages. Unfortunately, interactivity is an ambiguous notion in the literature. Some authors use the word in the comparatively weak sense that the heuristic should allow manipulation before, after or between runs. See for instance the OR-oriented compilation by Lewandowski and Stanchev (1989) which has the term 'Interactive Decision Support' in its title. Likewise, Narula and Weistroffer (1988, p. 435) mention in an overview paper

> The basic steps of many currently available interactive algorithms to solve Multi-objective Nonlinear Programming problems can be succinctly stated as follows:
> 1. Find an initial feasible solution
> 2. Interact with the decision maker
> 3. Obtain a new solution. If this or some previous solution is acceptable to the decisionmaker, stop; otherwise, go to step 2.

Others use the word in a stronger sense (e.g. Ball and Magazine 1981, p. 216). By *interactive heuristic* they mean a heuristic that *permits interaction even during its operation*, not merely one that can be run repeatedly with different parameters. This stronger sense is the one used here. If a system contains a model the system should only be called interactive if the model itself is interactive or plays a modest role in an otherwise interactive system. Such systems can be built, e.g. Fisher (1985, p. 541):

> Optimization algorithms or heuristics in which the user interacts significantly either during the solution process or as part of a post-optimality analysis are becoming increasingly popular. An important underlying premise of such man-machine systems is that there are some steps in solving a problem in which the computer has an advantage and other steps in which a human has an advantage.

Anthonisse et al. (1988) equate 'DSS from an OR point of view' with interactive planning systems and bring up numerous reasons why a planning system should be interactive, such as (pp. 415-416):

> The human planner is superior in guiding the overall solution process, in recognizing global patterns, and in observing all kinds of *ad hoc* constraints. Secondly, these better solutions are obtained faster, because interactivity allows for flexibility in manipulating data and in selecting alternatives. Finally, an interactive system is more readily accepted. The human planner is not replaced but gets a versatile tool.

Given these and similar observations, it is somewhat amazing that Anthonisse et al. do not address the topic of interactive heuristics.

According to the weaker sense of the word interactive, almost any heuristic could be used interactively. In fact a better term for this weaker sense would be *'parameterizable'*. Obviously some of the advantages of interactivity also hold to a certain extent for 'parameterizability'.

Interactivity in the strong sense places some special demands on the problem representation, the operations, and the control strategy. The main reason for these demands is that the user of the heuristic can only fruitfully interact with the heuristic during its operation if two conditions are met: the heuristic uses concepts which are meaningful to the user and the system interface allows the user to monitor what the heuristic is doing.

The problem representation

- must consist of objects that are meaningful in the real world. Often these are aggregate objects.
- must make it possible to develop a comprehensive interface that keeps the user informed of the activities of the heuristic. This usually means a strongly visually oriented interface.
- must not be overly specific. In fact the problem representation may be no more than a very partial model of the real-world situation. This leaves flexibility to the user. Especially constraints can be left non-modelled, since the user can monitor their

transgression.

The operators

- must correspond to meaningful actions in the real world. This is especially important for modifications of a plan.

The control mechanism

- must enable the user to alter model properties during execution of the model. These options must be simple enough to be usable without distracting the user from the planning problem.
- must allow the user to make a trade-off between the quality of a solution and the time spent on it. A very desirable property for an interactive planning system based on the type of heuristics mentioned here would be to have the user choose and change at will the level of support he wishes. For example, a user may either have the system perform actions and just watch a plan grow, or he may perform actions himself, having the system calculate associated quality parameters, or he may even disregard system advice altogether.

Obviously any interactive planning system will have to be developed by, or in close cooperation with, someone who has a thorough understanding of the real-world planning situation as nearly all the requirements for an interactive heuristic mentioned here have to do with properties of the real-world planning situation.

It is equally obvious that the interface of the system is of utmost importance. The objects which make up the problem representation, as well as the actions performed by the operators, must be both visualized and manipulatable. For instance, a user must be able to play with constraints, changing them repeatedly, without having to quit a planning board in order to edit a constraint database, which would interrupt his thinking. If the user cannot see the relation between his actions and the (partial) plan then there is no way in which an interactive planning system can be successful.

Perhaps less obvious is the fact that an interactive heuristic will need to be incremental in operation. There need to be many intermediate steps in the solution process. At all these points the user must be able to intervene. This implies that non-incremental algorithms are ill-suited for use in an interactive planning system.

Problem decomposition

In a complex problem situation a planner may need more than one level of detail to manipulate. This is a very important observation for the development of interactive systems. For instance Anthonisse et al. (1988, p. 417), in their presentation of the concept of interactive planning systems (IPS), remark:

> The user of an IPS usually follows some sort of divide and conquer scheme in arriving at an acceptable plan. This scheme often involves a certain decomposition of the problem.

Many authors are aware of the importance of problem decomposition[24] and describe design features such as hierarchical problem decomposition, data abstraction or abstraction of reasoning. Such an abstraction or decomposition introduces a problem of coordination between the levels, and it prompts the question at which level or levels the user must participate. In general the lower, more detailed levels will be more highly automated and the higher levels more understandable. This is what Baumewerd-Ahlmann (Baumewerd-Ahlmann and Kalinski 1990) has in mind when she remarks (p. 260):

> Knowledge modelling on an appropriate conceptual level significantly enhances a planning system's transparency and maintainability by reducing the gap between the planner's world model and the system model. Our approach aims at integration of structuring techniques concerning
> - the object level,
> - the contextual perspective and
> - abstraction levels of inference processes.

Evidently the automation of all these structuring techniques is a very ambitious enterprise. In most cases a first and perhaps sufficient step is to make explicit the structuring techniques present in the system and to make them easy to manipulate. After all, goal-directed interaction during the solution process is only possible if the user has an insight into the structure of the model.

Decomposition of a planning problem can be sequential, hierarchical, or a combination of both, depending on the planning situation. Some form of problem decomposition may well have to be designed into the system. This will take a more or less imperative form, mainly to enforce precedence constraints between subtasks.

Advantages of decomposition are the fit with the human planner and the reduction of complexity. However, these advantages can turn into drawbacks. The decomposition can be a strait-jacket making the system inflexible. The decomposition can also lead to too much isolation between subproblems and prevent the use of overall solution techniques.

Intuitively, OR techniques seem to be more suited for solving well-defined subproblems than for controlling the decomposition of the larger problem. AI techniques seem more appropriate for controlling problem decomposition. Indeed, Mietus, Jorna and Simons (1991) incorporate into a hospital rostering system an adaptable problem decomposition using AI, whereas OR techniques are used within some of the decomposition elements.

If a planner who is planning manually uses some form of problem decomposition then this decomposition need not be strict. Constraints or goals from one subproblem can

---

[24] Indeed, problem decomposition is a special case of the omnipresent problem of rubricating (Simons 1989). In its generic shape this problem consists of having to divide a set of elements into disjoint classes, where the set is 'open', i.e. it contains an unknown number of incompletely known elements.

play a role while the planner is working on another subproblem, or some constraint from one subproblem can be willingly transgressed while working on another subproblem. As a result the subproblems are not entirely independent nor entirely dependent. By contrast, problem decomposition designed into a system is inflexible with regard to constraint propagation. Either a constraint is propagated to another subproblem, or it is not. A problem decomposition in a planning system should, if possible, address this drawback. Preferably it should be easy for the user to track back and forth through subproblems, as well as to control the propagation of constraints between subproblems. In the next section a decomposition model is presented which especially addresses switching between subproblems.

A model for problem decomposition in planning systems

As we have seen, decomposing a planning problem in a planning system should be done with care since any decomposition introduces some rigidity into the planning process. Of the two modes of decomposition, sequential and hierarchical, the latter is the more problematic. This is because a subproblem hierarchy, unlike a subproblem sequence, has to be passed through at least twice: first top-down, while formulating or solving subproblems in more and more detail, and then bottom-up, while translating solutions of subproblems into higher-level solutions (see figure 2.2). Therefore, some attention shall be given to hierarchical problem decomposition.

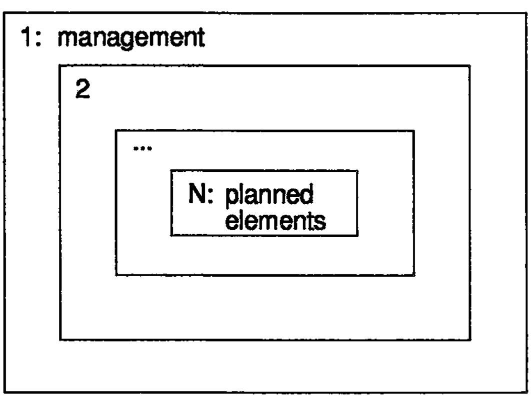


Figure 2.2: subproblem hierarchy in a planning problem

Figure 2.3 shows the problem decomposition model. In fact the figure shows one level in a recursive hierarchy such as shown in figure 2.2 and visualized differently in figure 2.4.

The model does not specify which activities are carried out manually and which are automated. It pictures the planning problem as a recursive series of problems at different context levels, from overall to detailed. Criteria for decomposition are of a practical nature. They can be found in the structure of the activities or processes being planned, or in the habitual routine of the planner.

At each context level a problem solving process consisting of the same elements is carried out. The model is recursive in that figure 2.3 repeats itself in the small box 4. See figure 2.4 for a pictorial impression of this recursion. But for the recursion level index box 4 is identical to the entire figure 2.3. The index runs from 1 to N. At each level there are three actions (2,4, and 7) as well as three points of evaluation (3,5, and 10). Boxes 6 and 9 are merely tests for success or failure at the top level.

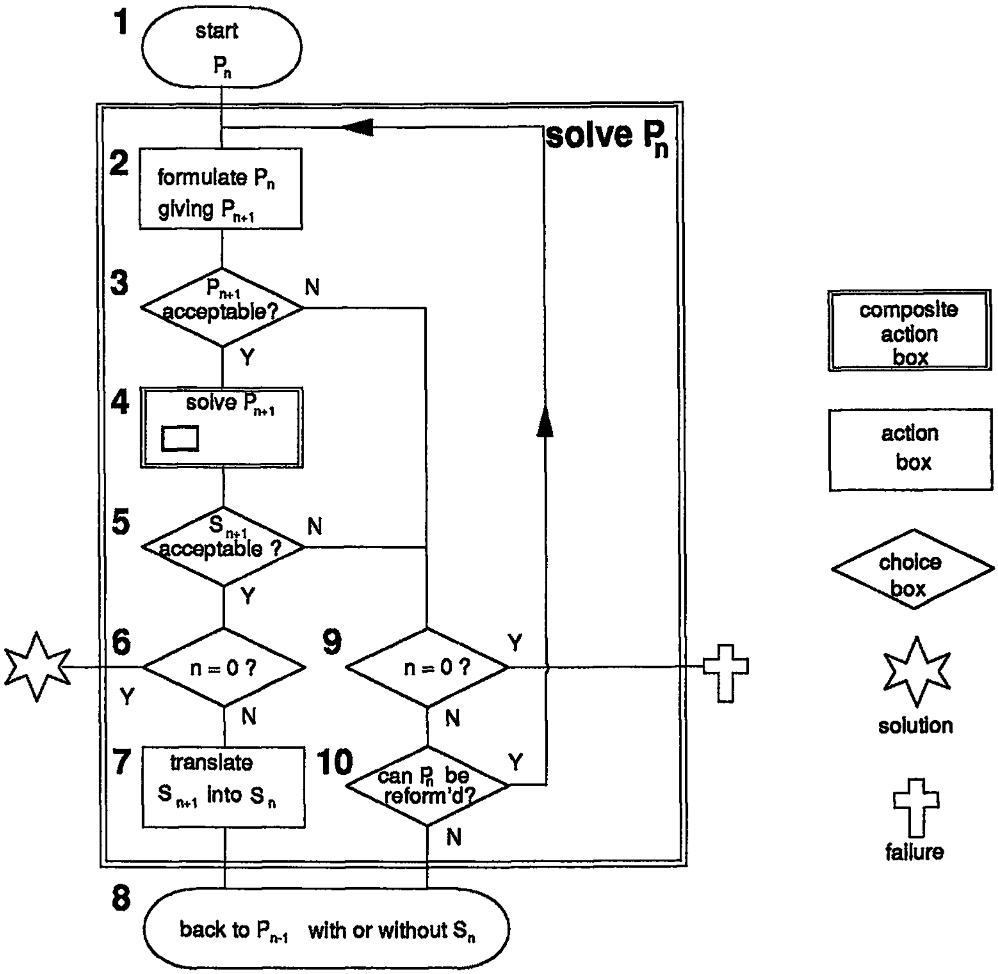


Figure 2.3: An approach for hierarchical decomposition of planning problems.

Each level in figure 2.3 has its own variables relevant to problem formulation and evaluation. During planning the planner and/or the system first formulate the problem at the uppermost context level (n=0) and then descend the recursion, formulating in more detail. During this descent, only boxes 1 to 4 are gone through. Incidentally, Simons' 'double principle' of searching for a solution as well as for unsolvable problem instances is realized in boxes 2 and 3. At the most detailed level, the problem is solved - which, by the way, means no more than that the yet more detailed problems are formulated that will have to be solved during execution of the plan. Then the solutions for subproblems

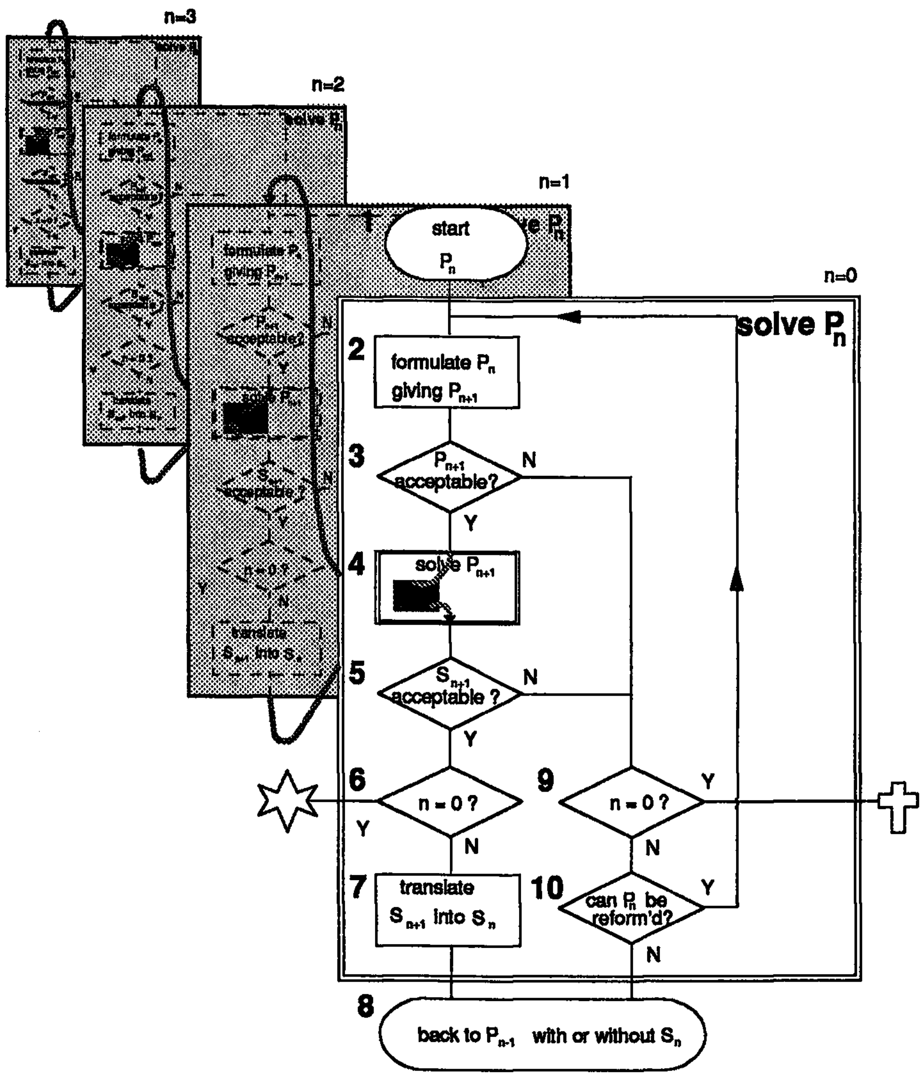


Figure 2.4: A visual impression of the recursion in figure 2.3

have to be translated to higher levels, ascending the context recursion again. If solutions of subproblems are found acceptable, then boxes 5 to 8 are traversed. If not, the path 5 - 9 - 10 is followed and then another attempt at formulating the problem at the current level n may be made before backtracking to a higher context level with an unsolved subproblem.

The translation of subproblem solutions to higher levels involves a translation into a different set of variables. This will in most cases remain largely the planner's work, especially at higher levels. The overall problem solving strategy, including when and why to switch between context levels, will also usually be determined by the planner.

When discussing heuristics we saw that search problems can be represented either as a state space or as a problem reduction representation. The context tree of figures 2.2 to 2.4 clearly is no state space. At a particular level, however, a subproblem can often be formulated as a state space, which allows the use of automated solution mechanisms such as Operations Research or Artificial Intelligence techniques.

The context tree is not exactly an OR graph[25] either, but it can be instantiated as one: in general a problem at a higher context level is solved by generating several solutions at the next more detailed level and choosing one of these. The branching degree of the tree is arbitrary. A difference between the context tree and a conventional OR graph is that in the context tree solutions of a subproblem have to be translated into a higher level and may not be acceptable at that level.

How to apply the model

Looking at figure 2.3 a crucial question is which actions and evaluations to automate and which ones to leave to the planner. Where it is decided to leave an action to the planner the system must present the relevant information. In many cases the system could leave it to the planner to decide whether he wants to take an action unassisted by the system, or whether he wants to be assisted or even wants the system to carry out the action on its own.

The model can be used to map a designed or existing system onto. During the PROPLAN project, it has been especially useful in signalling design shortcomings. Whether the model can be used from the start in a planning system project remains to be seen. It originated during the PROPLAN project when there was already a prototype system. I expect, however, that the model can be a valuable help in modelling a planning situation. This means that it can be helpful as soon as a suitable and working interface has been found.

[25]'OR graph' is intended in the sense given by Pearl (1986), where OR stands for 'choice', not for 'Operations Research'. An OR graph is a problem reduction representation: the problem is split in independent subproblems.

To sum up the presentation and discussion of the problem decomposition model, this model has the following attractive properties:

- it explicitly distinguishes actions from evaluations.
- it allows to make explicit design choices, e.g. choices about task allocation between planner and system.
- it allows problem decomposition into subproblems that are amenable to a state space representation and its mathematical advantages.
- it takes into account the crucial role of the planner in translating solutions of a subproblem into variables of a higher context level.
- it can act as a tool to check the completeness of planning system designs.

Despite these attractive properties it seems wise not to use this or a similar model too normatively in early stages of a planning system project. Such an attitude might hinder creativity. As soon as there is a prototype with a working interface, however, the model can be used to help design further supporting functionality.

3 Decision making in organizations

In the crowds of Grand Central Station,
Herzog in spite of all his efforts to do what was best
could not remain rational.

Saul Bellow (1964, p. 39)

This chapter deals with the question "In which situations are planning systems appropriate?". Chapter two has shown that the user is often neglected by developers of planning systems. Decision making and decision support will now be considered from the practical perspective of a user, to see whether such a perspective can shed a new light on automated support for planning situations. Under which circumstances and for which planning situations could a planner really benefit from using a planning system, and what would be required of such a system?

The chapter starts with some examples of decision making by pot plant growers. The literature on management decision making by pot plant growers is reviewed. The chapter proceeds with a lengthy review of a diverse body of literature on decision making in organizations. This review is given from an organizational, a psychological and a systems perspective on decision making. Then I develop some views on management decision making and decision support based on the literature and use these views in the analysis of planning situations.

3.1 Some cases of pot plant growers

In the next section the reader is introduced to three growers. The initials P., Q. and R. are fictitious, but the cases are real. The data were collected through interviews. Each nursery is briefly described in general terms, after which something is said about how this particular grower goes about cultivation planning.

Grower P.

Grower P. is a moderately large Ficus grower in the Westland, between the Hague and Rotterdam. Ficus are the number one green pot plant, with a yearly auction turnover of about *f* 120 million in the Netherlands. P. has recently doubled his nursery's size. It now covers a surface of 32,000 square metres of greenhouse which is good for a yearly production of *f* 2.5 million's worth. P. cultivates an assortment of some thirty Ficus varieties, dropping a few and adding a few each year. These range from small to huge plants. The larger ones take more than a year of cultivation before they are sold. P. usually cultivates from his own cuttings. He is oriented towards efficient production rather than market opportunities, with an accent on quality rather than bulk. Sales are via the auction.

Two sons of the founder run the nursery, together with a few more members of the family. In total there are 11 employees, who all meet formally each Monday and informally during three breaks a day. The two brothers also have weekly meetings together.

P. makes a strict cultivation plan, largely on the basis of last year's plan but with variations depending on profitability prospects (in money per week and per square metre). The cost side of production is known fairly exactly. Average empty space has been 9% over the last few years, 10-11% in 1991 due to storm damage. The ideal is 8%. The fixed costs of a square metre of space per week, that is, all costs which cannot be attributed to a particular culture, are currently *f* 1. To give an idea, reducing empty space by 1% would yield a benefit of *f* 1 x (32,000 : 100) x 52 = *f* 16,640 a year.

The profit side, in particular where price forecasting is concerned, remains guesswork. An important consideration is the total volume produced. In Ficus, production per variety fluctuates much compared to demand. Consequently, P. is keen on news about next years' production volumes, because these volumes give an indication of the market price that can be expected. Such news is mostly gathered through informal contacts at the auction and at Ficus growers' meetings. Also, P. avoids producing too much of a particular variety for fear of spoiling his own market.

The cultivation plan is cyclic, with almost identical three-week cycles repeated consistently throughout the year. There are three departments in the greenhouse with about twenty sections each. Each week one section in each department is sold out and refilled with spaced plants. The transport system, chariots, is simple, cheap in fixed costs, and flexible, and transport costs are almost distance independent. Transport is often combined with other activities, so that the cost of transport *per se* is not known. The cyclic nature of the cultivation plan and the little importance of transport considerations, added to the frequent informal contacts between managers and personnel, make a space plan unnecessary.

A less cyclic plan would probably be more profitable; however having a strictly cyclic plan has some advantages:

- cultivation planning is easy and not too time-consuming (2 weeks a year net time, spread over 4 to 6 weeks)
- carrying out a cyclic plan is easy for personnel; they can independently carry out tasks with minimal instruction
- providing constant supply at the auction is a service to customers.

P. strives towards a future situation with seasonal fluctuations in the plan depending on profitability of particular varieties in particular periods. First he wants to attain a better division of labour, because the managers still spend too much time on operational matters.

Since november 1990 P. has an automated planning tool at his disposal, coupled with the automated data recording system. On the basis of potting data provided by the user, the tool displays total space need per week, labour and financial consequences. The user can interactively play with potting numbers and dates and see what the consequences of proposed changes would be. P. was able to immediately adopt the system because he already used the data recording package intensively. The main benefit he expects from it is to be able to generate alternative plans and have the consequences computed. This in turn will allow him to experiment with less simple but more profitable cultivation plans.

Grower Q.

Grower Q. has a fairly large flowering pot plant nursery in Vleuten. He is in charge of strategic and tactical management himself. There are four major and some seven less important cultures. The main product is Poinsettia, which is sold at Christmas time. Other important cultures are Browallia, Marguerite and Fuchsia. The remainder of the cultures are used to make the nursery more attractive for clients. Q. sells mostly through the auction but has some export himself and has contacts with a number of important clients and exporters. These contacts are his 'market thermometer'.

The greenhouse surface is 14,000 square metres, not counting a neighbouring nursery which Q. runs jointly with a colleague. This means that Q. is among the 20% largest growers of flowering plants. Yearly turnover amounts to *f* 2.5 million. There are twenty employees, four of whom are involved in management, mainly operational management. Most of them have been with Q. for a number of years. There have not been any recent major changes, although the sister nursery has known enormous expansion during the last years. Plans to reallocate the Vleuten area for urban development have kept Q. busy during the last years. During the early eighties, Q. has fared well by a combination of innovative assortment and sound management. Recently the emphasis has been on better cultivation planning and more control of the growth process rather than on renewing the assortment.

Q. attaches great importance to the cultivation plan. He is unable to indicate how much time he spends on it yearly but thinks that the cultivation plan has his attention nearly half of the time. He stresses that cultivation planning is closely intertwined with strategic and operational concerns. On the strategic side, feeling for the market is a critical success factor. On the operational side, much can be gained by improving predictability and/or controllability. For instance, this year a sales loss of *f* 60,000 is compensated for by a cost reduction of *f* 100,000. All in all Q. has been able to predict his yearly turnover figures within a few percents for several years now. This is largely due to the spreading of risk caused by the large assortment, and further enhanced because the products are stable, relations with clients are stable and quality is constant.

Since a number of years Q. has done his space planning manually, resulting in a log plan which plays a central role in the nursery. An example of such a log plan is shown in figures 1.2 (p. 5) and 5.12 (p. 168). The log plan displays exactly which production lot is cultivated during which week and on which table. The main planning activities are carried out around October, when next year's plan takes shape. Although the process is ongoing and there is no formal partitioning, some steps can be distinguished:

1. Take last year's plan, including the evaluations which have already been made about it on a quarterly basis as well as per production lot. Decide on changes in assortment and on the direction in which changes in quantities will be carried out.
2. Use last year's log plan. Modify and refine the plan to yield next year's plan. Changes amount to some 10% of the plan. Taking last year's plan as a basis guarantees that a multitude of constraints are satisfied at the outset of the planning process.
3. Generate a financial plan derived from the cultivation plan. Q. has written a LOTUS 1-2-3 application for this, which he has been using for four years now. Q. does not backtrack from this activity to the previous.
4. During execution of the plan, monitor it. A quarterly financial evaluation is made, as well as an evaluation per production lot.

Grower R.

R. stands for a large, innovative pot plant nursery in the Westland. Most of R.'s products are green pot plants, for example Spathiphyllum. They have two sites, as well as a joint venture with a plantation in Brazil which furnishes cuttings and big tropical trees. R. also cultivates from seed or cutting himself, and gets some potting material from abroad.

The nursery is run by three sons of the founder. During the last years non-family members have entered management, e.g. a well-schooled young head of internal management, who among other tasks does the cultivation planning. Personnel amounts to 50 persons. The two sites cover 70,000 square metres. Nursery R. is well-known by researchers and software vendors for their willingness to try new ideas.

The planner has spent his first years getting to know all the intricacies of the firm and creating a working planning and control cycle with both automated and manual elements. He relies partly on a commercial data recording system, partly on self-written spreadsheet applications for planning. Currently, in 1991, he aims at a "completely automated" planning cycle, including use of optimization techniques as one of the first activities in cultivation planning. At the same time he is aware that the planning procedure cannot do without his own judgement. He remarks that one has to "incorporate context in the data that the software use". For the optimization module this means for example defining available space as 102% of really available space. The plans obtained which contain the nonexistant extra 2% space are translated by the planner into realizable plans. The surplus is resolved by temporary storage, alterations to cultivation schemes and similar measures. In other words, by fooling the optimization module the planner is able to use his own specific knowledge of the nursery to improve the plan. Of course this means that formal optimality of the plan is lost.

The introduction of new automated planning components is simultaneous with organizational changes. As he put it himself, the planner worked in a quick-and-dirty way during his first years. Almost any plan would do, resulting in a multitude of products cultivated under changing circumstances. He wants to change this into a situation where plans are solidly based on accurate recorded data and evaluation procedures. There should be fewer products so that the cultivation schemes can be stabilized, resulting in more efficiency. The cultivation plan should not fluctuate more than 10% each year.

Technically, R. strives for higher automation. Transport of plants currently takes place by conveyor belt, but robots are desired for the near future. Socially, R. tries to keep motivation and quality of labour at a high level. As one of the very few the nursery has issued a formalized organization chart which prescribes not only functional areas but also information flows. Recently a handbook describing jobs, functions and information flows was completed. It is given to the personnel members.

Finally, R. is actively involved in a number of external activities, such as a seat in the board of SITU, Institution for Information management in Horticulture. SITU is a semi-government institution concerned with automation, both technical and management-oriented, which acts as an intermediate between growers, commercial software vendors, extension and research. In 1990 and 1991 SITU participated in a project which aims at developing integrated data recording, planning and evaluation tools for pot plant growers.[26]

How far can growers automate cultivation planning?

The reader has now become acquainted with three growers with very different nurseries and cultivation planning practices. All three are successful, which shows that many roads lead to Rome in pot plant cultivation. Knowing something about growers P., Q. and R. we can speculate whether their wishes for automation of cultivation planning are realistic and sensible. R. wants to formalize both the organization and the automated components. Q. merely wants to automate his current activities without modifying them. P. is intermediate. Is R. too ambitious, or Q. too conservative?

It may be informative at this point to give a brief history of cultivation planning in the Dutch pot plant world. Starting in the early eighties there has been a current towards recording of labour, financial data, and cultivation data. The justification was that "*gissen is missen en meten is weten*" ("guessing is missing and measuring is knowing", see Van Horssen 1989). In the mid-eighties it was more and more felt that recording alone was not enough, and that in order to be beneficial recorded data had to be used in planning to evaluate past plans and to provide normative data for new plans (see e.g. Alleblas 1987). Bleijenberg (1983), for instance, provided growers with a guideline for an integrated data recording, planning and control cycle. His tools were manual: data recording sheets, log plan sheets etc. which the grower could fill in.

Simultaneously, commercial software vendors were developing automated data

---

[26]In 1992, SITU will merge with a number of related Institutions to yield ATC, Agro-Technological Centre. ATC will serve the interests of agricultural entrepreneurs at a national level for all agricultural sectors. Automation at primary agricultural firms, networking, and standardization of interfaces are ATC's main areas of concern.

recording tools. By the late eighties many innovative growers possessed such a data recording package. There were also a few planning packages, but these did not interface with the data recording packages. The call for automated integration of data recording with planning and evaluation was getting louder (Werkgroep MBP 1990). Growers used manual techniques or self-made spreadsheet applications.

Recently vendors have begun to produce planning aids integrated with their data recording packages. The planning support is at the level of space requirement planning with financial and labour corollaries. Table-level location planning or cultivation planning optimization are not yet integrated in commercially available form. More advanced features, such as the use of plant growth models, are far off yet.

The majority of pot plant growers are not interested in all the commotion around cultivation planning. They are conservative with regard to automation and do not use a computer other than for technical purposes such as climate control. Each grower has his own way of cultivation planning. P., Q. and R., being forerunners and prominent customers, have intensive contacts with their software vendors. But the software vendors, some four or five companies, are of moderate size and are busy protecting their market share so they cannot afford too much activity that does not result in direct pay-off. One software vendor cancelled an ambitious project for this reason. Research institutes, on the other hand, are supplying advanced planning tools, but these are as yet not integrated with the growers' data recording software and spreadsheet applications.

Experiences with automated cultivation planning

During the early eighties one package was developed for optimization of cultivation planning: IPP, IMAG Production Planning Aid, based on Linear Programming (Saedt 1982). This approach yielded plans of high normative quality. IPP is used by growers in practice, albeit on a much smaller scale than the developers originally expected. There are a number of reasons for this, not all of which apply in every single case. Hofstede and Simons (1987) found four likely reasons why IPP had but limited success with growers. Firstly, it might well be that growers used other criteria in planning than optimization of expected profit, such as robustness, continuity of labour requirement or continuity of output. Secondly, because of a combination of unavailability and poor predictability of data, the assumptions about labour and about price levels under which the Linear Programming matrix was built were not met in reality, so that optimality was an illusion. Thirdly, a lot of important factors could not be expressed in the Linear Programming model. A grower might change the cultivation process under time constraints, use the pathways between the tables to cope with space constraints, sell half-ready products if the opportunity arose, etcetera. Finally, growers had an adaptive and *ad hoc* way of dealing with their cultivation plan, and were not ready for optimization.

3.2 Research about decision making in horticulture

This chapter would not be complete without a treatment of the scarce literature on decision making in horticulture and pot plant growing. Although my views on organizational decision making are not limited to any single sector but are meant to be of general significance, the horticultural sector has some particular traits. Moreover, the literature on pot plant management decision making is of practical relevance for the PROPLAN project.

Alleblas

The agricultural economist Alleblas (1987), in a doctoral dissertation related to earlier management research, systematically studied what he calls the *management level* of horticultural enterprises. Decision making is central to his definition of management level. On the basis of extensive survey data he developed a tool to measure the management level and used the tool to advise 'fitting levels' of management, adjusted to enterprise and entrepreneur characteristics. Thus an ideal management level of 100% is not necessarily desirable for all growers. Depending on the conditions and goals of the grower a 'lower' management level may be better.

Alleblas took considerable trouble reviewing existing models for management quality assessment and adapting them for the horticultural enterprise. He divided management decisions in three levels, following Anthony. The tactical level is constituted by cultivation planning.

The quality of each decision is characterized by rating a number of aspects:

- a balanced amount of attention to the decision in relation to other decisions
- time of preparation, which must be proportionate to life span of the decision
- use of oral sources of information
- use of written sources of information
- use of models or calculations
- monitoring of actual execution and evaluating a plan against its realization to assess the quality of the plan and establish the causes of aberrations
- consultation of personnel about the plan.

Summarizing these aspects, Alleblas' quality characteristics stress three points: balancing the decision in its context, using appropriate information sources, and using feedback from realization for subsequent planning cycles.

Alleblas' model was applied to a sample of 63 growers, including a few pot plant growers. Unfortunately the pot plant growers were too few to be treated as a separate category. The average management level in the entire sample was 25 for cultivation planning, on a scale from 0 to 100. This level corresponds to decision making on an *ad hoc* basis, where a cultivation plan does exist but is not closely followed, nor monitored against recorded data and where no calculations are made.

The level of profit of the nurseries in the sample correlated positively and linearly with the level of management as determined by Alleblas' tool. 1% rise in management level corresponded with 1% profit rise. However, only 60% of profit rise could be accounted for by the quality of management. Relevant management aspects were mainly the modernity of the greenhouse and equipment (accounting for 28% of the variation), and the age, ambition and education of the grower, and less the quality of decision making in a narrower sense, as defined above. Of course modernity is a result of past decision making 'in a narrower sense'.

Alleblas checked the objective data against the growers' satisfaction with aspects of their jobs. The external influences, especially government regulations, turned out to be the main sources of stress for growers. The low level of tactical management was not perceived as stressful. Alleblas concludes that the growers were insufficiently aware of how profitable tactical planning and control could be for them.

In a new research phase, based on the same survey, Alleblas established 'fitting management levels' for various decision areas dependent on goals and circumstances of the grower. Not surprisingly, he found that a high management level for cultivation planning is most important for large, capital-intensive nurseries with hired labour. According to Allebas, such nurseries should be very systematic about cultivation planning and they might use quantitative techniques such as optimization. They should intensively use diverse sources of information and they should closely monitor execution of the plans.

Bots

In his Ph.D. research, the organization scientist Bots (1991) tried to establish relationships between the environmental and internal complexity, the quality of decision making, and the profit level at pot plant nurseries. His research was an in-depth study of eight nurseries. The results are in line with Alleblas' findings. 1% profit rise corresponded to 0.4% rise in decision making quality. However, profit level had a more pronounced relationship with complexity than with decision making. Growing complexity was detrimental to profit. Bots puts forward that, to a large extent, a grower chooses his complexity level by strategic choices, so that similarly to Alleblas' modernity, Bots' complexity can be seen as the result of past decision making. Bots' data further suggest that the quality of cultivation plan making is more important than the quality of cultivation plan realization, or in other words: it pays to plan.

Bots did not include either modernity or nursery size or personal characteristics of the grower as explanatory variables, but he suggests that these factors might be taken into account in subsequent research. Such research is currently being carried out, and preliminary results are given in Ziggers (1991).

3.3 Tactical planning as decision making

Tactical planning situations usually have some properties not present in all decision making which make them rather well-demarcated. To sum up:

- there is a precedent, viz. the previous occurrence of the planning situation, from which data, variables, and procedures can be borrowed

- the time horizon is explicit
- there is a person who is specialized to some extent in doing the planning job, and who is responsible for the resulting plan: the planner. This may be a team of planners rather than one individual.

On the other hand, and here I refer to the discussion in the previous chapter, tactical planning situations share a number of properties with most other decision making:

- usually, the planner is under time stress and has to attend to other problems simultaneously
- not all relevant data are available and of sufficient quality
- not all processes are reliably predictable
- the objectives cannot be formally specified. In practice, getting the job done without blundering grossly is the main objective.
- if a problem cannot be solved, it is reformulated, i.e. it is an open problem.

The situation at grower R. exemplifies these properties. The planner has been under enormous time stress just to keep up with continuing organizational changes. For instance, the size of the nursery changes frequently. A huge new greenhouse was built last year, creating an unusual planning problem: it had to be filled entirely as soon as possible after completion, in order to get return on investment. This meant that the entire cultivation plan had to be built around filling the new greenhouse. At the same time, new products were grown, and cultivation problems had to be overcome. Another disruptive event, entirely unforeseen this time, was a severe storm in February 1990 which ravaged the greenhouses, causing a lot of unexpected repair work and production losses. Unique events like these two arrived frequently. Until 1990 they have prevented the planner from carrying out his plans to automate.

In conclusion, tactical planning situations are admittedly better demarcated than most decision making in organizations, but nevertheless the similarities with other decision making are far more important than the differences. In particular, tactical planning situations are *open problems* to a planner, ridden with uncertainties, and do not come as neat *formal problems*. Therefore it makes sense to investigate the literature about decision making in organizations in general.

3.4 Introduction to the literature

The amount of literature on management and decision making in organizations is vast and it offers a multitude of perspectives. Although the present literature review does not attempt general coverage, it is quite large in scope. It emphasizes writings which are directly relevant to the argument of the book but it also mentions authors whose work is not used in the rest of the book. As Morgan (1986) puts it: each perspective is a different interpretation of reality, and effective analysis of organizations takes account of rival theories or explanations, rather than being committed to a fixed and unshakeable point of view. New insights about a situation emerge as one considers it from new angles. Three perspectives are adopted here.

The first perspective from which I shall approach the literature on decision making is an organizational perspective. From Fayol on, management theory has developed into a number of schools and approaches, both descriptive and prescriptive in emphasis. Descriptive approaches have gradually evolved from Simon's bounded rationality towards even more mundane concepts such as Lindblom's 'muddling through', and I shall present them in historical order. Authors such as Henry Mintzberg adopt a pragmatic view on management, describing the actual work context and activities managers carry out. This perspective is relevant for my case because in small organizations like pot plant nurseries planning is usually carried out by the manager.

From a more prescriptive organizational point of view, one might say a control perspective, authors such as Anthony and Kampfraath define how management as a whole should be partitioned. They include characteristics for the quality of decision making. The organizational perspective stresses that no management problem can be solved in isolation. There is a link, both with other problems and with the control cycle in the organization. Another property of this perspective is the orientation on processes rather than structures.

The second perspective is a psychological one. Cognitive scientists such as Simon, Beach and Vlek stress the bounded cognitive capacities of human decision makers. This is relevant for two reasons. First, it means that 'rational' computer-given advice will not necessarily be taken. Second, it implies that the interface of an automated support system is important. If it does not agree with the way in which the user wants to or is able to handle the problem, the system may not be adopted.

The organizational and psychological perspectives meet in the field of social or organizational psychology. Authors such as Simon, for example, have written from both perspectives.

The third perspective taken is a systems perspective. Further abstracted from the organizational context than the previous perspective, and structure-oriented, this approach offers possibilities for novel ways of looking at planning problems. Systems thinking emphasizes generality of phenomena and the open nature of organizations. Authors include Beer, Kramer, De Leeuw, In 't Veld, Checkland. The systems perspective offers a useful language, both verbal and pictorial, to provide abstract descriptions of organizational phenomena. Systems theory has had an influence on the problem solving approach to DSS development as introduced by Mitroff.

3.5 Literature: organizational perspective

From Management Theory to Muddling Through

Managers plan, organize, coordinate, command, and control. This has been widely taught since Fayol (1916). Some separate aspects of managers' work have been much studied since then, e.g. leadership, power, human relations and decision making. Throughout this century the image of the manager as a rational decision maker maximizing expected profit, '*homo economicus*', has remained popular. A new set of ideas, including the idea of bounded rationality, was developed by Herbert Simon, whose first major contribution was "Administrative Behavior" (1947). This set of ideas has gradually gained momentum, as it has become clearer that people do not usually act in an economically rational way.

The writings of experienced managers such as Sloan and Vickers (both in Pugh, 1984) have enriched the bounded rationality paradigm. According to Vickers, judgement is the most basic managerial skill, and it is a fundamental, continuous process, integral with our thinking. He distinguishes three forms of judgement: reality, value, and action judgement. Reality judgement deals with how we perceive the world, value judgement refers to how we evaluate what we see. Action judgement, the basis for decision making, is subordinate to the other two.[27]

A provoking author is Charles Lindblom, who was an early critic of the *homo economicus* school. He called management the 'science of muddling through' (Lindblom, 1959). According to Lindblom the *homo economicus* idea is utterly unrealistic because it overlooks man's inability to cope with complex problems, the usual lack of information, the cost of analysis, the problems of timing, and the difficulties of stating realistic goals. On this last point he remarks:

> But there is no practicable way to state marginal objectives or values except in terms of particular policies.

According to Lindblom a manager copes with complexity, lack of information, time constraints, and the ambiguity of goals by a combination of four strategies. These are: avoiding unanticipated change, taking small steps and awaiting feedback, continually adjusting goals to means, and attending to problems serially. These steps, called 'successive limited comparison' by Lindblom, are continually repeated as feedback from previous steps becomes available. Thus the method economizes on the need for facts, and it directs attention to the relevant features only, both because only small changes from current practice are considered.

Although Lindblom made his observations in the context of policy making, a lot of them can be applied to planning. For instance Lindblom's ideas imply that the usual planning procedure would be to take last period's plan and make incremental changes to it. This is a time-saving approach and it is indeed what usually happens in pot plant nurseries unless the planner is forced to make radical alterations. Another implication is that *a priori* criteria for plan quality cannot be stated, and that instead the planner will say "yes" or "no" to a given plan on the basis of qualitative judgement.

Behavioral Theory of the Firm

A major contribution from the economic school, related to Lindblom's ideas, was made by Cyert and March (1963) with their "Behavioral Theory of the Firm" in which they present a coherent alternative to *homo economicus*. The theory uses concepts like

---

[27]Vickers' views are given more attention on pp. 88-90.

imperfect information and cognitive limitations. The heart of the theory is formed by four concepts:

1. Quasi resolution of conflicts.
   Considerable latent goal conflicts can exist without disturbing the functioning of the organization. This is possible because people resolve conflicts using local rationality, acceptable-level decision rules, and sequential attention to goals.
2. Uncertainty avoidance.
   Organizations avoid the requirement to correctly anticipate events in the distant future by short-run adaptations and by arranging a negotiated environment.
3. 'Problemistic' search.
   Search is stimulated by a problem, depressed by a solution - any solution will do. Search is as simple as possible, and searching persons - managers - are biased.
4. Organizational learning.
   Organizations can adapt their goals, attention rules, and search rules.

Cyert and March are less radical than Lindblom. The focus is the firm rather than the individual manager, except for the chapter by Williamson who presents "a model of rational managerial behavior". This model is built upon the reasonable assumption that a rational manager will first and foremost pursue self-interest rather than the interest of the stockholders of his firm. Williamson uses the notion of 'organizational slack' to indicate the room for play available to the manager for pursuing his own interests without endangering the firm.

Garbage Can

March developed more controversial ideas in a subsequent publication: "Ambiguity and choice in organizations" (March and Olsen 1976). In this publication the authors develop the so-called Garbage Can model of decision making. In a garbage can situation, a decision is an outcome of several relatively independent streams of garbage within the organization. Streams include problems, solutions, participants, choice opportunities. These four streams are mixed and churn around in the garbage can, and choices are made somewhat fortuitously, highly dependent on timing and context. The result is an organized anarchy. In the garbage can (p. 31):

> Problems come to seek connections to choice-opportunities that solve them, solutions come to seek problems they handle successfully. Decision-makers use their energy in areas in which they have success, or about which they have concern, or in which they find pleasure.

Thus the independent status of a problem, as something which is approached in an orderly way performing intelligence, design and choice or similar activities, as it appears in other schools of thought, is absent from the garbage can model. Also, March questions the generally held presumption that goals precede choices. March contends that goals are not static but fluid and ambiguous. He proposes that some flexibility with regard to goals and values be built into decision making in organizations. This leads him to a variety of suggestions, including the following one on evaluation:

> As nearly as I can determine, there is nothing in a formal theory of evaluation that requires that the criterion function for evaluation be specified in advance. In particular, the evaluation of social experiments need not be in terms of the degree to which they have fulfilled our *a priori* expectations. Rather, we can examine what they did in terms of what we now believe to be important. The prior specification of criteria and the prior specification of evaluation procedures that depend on such criteria are common presumptions in contemporary social policy making. They are presumptions that inhibit the serendipitous discovery of new criteria. Experience should be used explicitly as an occasion for evaluating our values as well as our actions.

One can use the same argument for the evaluation of tactical plans in organizations. After a plan has been carried out, the evaluation should not be based on *a priori* criteria only. The plan may have proved to possess qualities or defects that were unforeseeable at the time it was conceived. Thus the evaluation is partly a restatement of goals, to be used in a succeeding planning cycle. In fact such an evaluation is an occasion for learning about the planning situation.

How do managers manage?

Still the obvious question "What does the working day of a manager look like?" remained to be answered. Economical theories, even behaviourally enriched, failed to answer this question, and a more social sciences-oriented research approach came into focus. Mintzberg (full study: 1973, brief version: 1975), inspired by a diary study by Carlson (1951) and by anthropological studies by Sayles (1964), performed an empirical study in which he shadowed a number of managers to find out what they actually did. He found that managerial jobs are surprisingly alike. One aspect of their work which is of special relevance to this book is (1973, p. 5):

> The prime occupational hazard of the manager is superficiality. Because of the open-ended nature of his job and because of his responsibility for information processing and strategy-making, the manager is induced to take on a heavy load of work, and to do much of it superficially. Hence, his work pace is unrelenting and his work activities are characterized by brevity, variety, and fragmentation. The job of managing does not develop reflective planners; rather it breeds adaptive information manipulators who prefer a stimulus-response milieu.

For pot plant growers this statement certainly holds. It may be less true for planners who are not themselves managers. Nevertheless, adaptiveness rather than reflective behaviour characterizes planning in most organizations. In many cases 'rolling planning' is used. This is planning in which the time horizons of subsequent plans overlap. The nearest future is planned more precisely than the rest, since re-planning of the later part of a plan occurs anyway. Under turbulent, unpredictable conditions, rolling planning is a way of combining short-time reactivity with looking ahead.

Managers and computers

There is some evidence that effective managers to a certain extent succeed in reducing the fragmentation of their working days (Ponder 1957). However, the continuous need for attention to the environment remains, and a high level of fragmentation is unavoidable. In this light it is not surprising that recent studies indicate that computers have hardly altered managers' work (Moss-Jones 1987) yet. Moss-Jones conducted interviews with a hundred managers and found that although information technology has had an important impact on their work, the work has remained fragmented, weakly defined, oral and action-orientated.

A survey by Millman and Hartwick (1987) among 75 Montreal middle managers indicates that generally they are satisfied with automation. The respondents deemed that their work had become richer, more demanding and more important.

In the case of DSS one can speculate that a DSS will necessarily change the work of the user inasmuch as the DSS structures some portion of the managers' decision making. The intuitive, reactive approach will be modified in some way, either by specifying the data to be considered, or by prescribing the type of analysis to be conducted, or by formalizing choice rules (Ginzberg 1978[b], p. 41). It can be conjectured that there is a tension between the need for redesign of managerial problem solving which a DSS generates and the demands imposed on managers by their work. Such a tension could preclude adoption of DSS by managers, even if the DSS would have been helpful if adopted.

An interesting viewpoint about how DSS could be not only attuned to managers but positively useful in dealing with unexpected variety is given by Migliarese (1985). Migliarese introduces the concept of 'organizational noise'. Organizational noise, he says, is unexpected variety which an organization's members have to face and which is always present in an organization. He then proposes to develop DSS which are sensitive to the unexpected, using a metaphor from process control. The emphasis is on the watchdog function of such a DSS, and on the organizational learning it allows. This learning occurs because the DSS contains a descriptive model of its subject matter. When inconsistencies between model performance and actual performance are signalled, the users come into action. Either they react to the unexpected event, or they improve the model.

Macintosh (1985) remarks that managers and DSS developers are competitors for information. Managers like to have a monopoly on important information. This is a keen common-sense observation that could be a strong reason why managers do not like to use DSS.

Prescriptive orientation

Prescriptive management theory owes much to Robert Anthony (1965) from Harvard Business School, who introduced the hierarchically related concepts of strategic planning, management control, and operational control described in the introduction. At the time of publication Anthony had his doubts about the acceptance of his views. Since then the three notions have become known and used by almost everyone dealing with management. I shall repeat a *caveat* which is relevant for my case study of pot plant management: Anthony presupposes division of labour. He excludes small organizations, especially those so small that the management consists of one person rather than a group (p. 9). Very likely, in such a small organization the strategic, tactical and operational level have much more interplay than they have in larger organizations.

Some authors mentioned in other sections have also developed prescriptive models of management. A recent overview of such 'control concepts' is presented by Bots et al. (1990). A prescriptive approach which is relevant here is the so-called Wageningen Control Approach (*Wageningse Besturings Benadering*) developed by Kampfraath (Kampfraath and Marcelis 1981). This approach is rather different from Anthony's framework, but also aims at larger organizations.[28] A feature to be mentioned here is the characterization of the quality of the decision making process given by Kampfraath (Kampfraath and Marcelis 1981 pp. 37-40, Bots et al. 1990 pp. 547-548). A translated and interpreted version of the quality characteristics mentioned on those pages is:

1. Integration.
   Is the situation considered in a larger organizational context before the decision is made? This enhances quality.
2. Prediction horizon.
   What future time span is taken into account during decision making? Ideally, the decision making process should look ahead as far as the end of the period of time during which the decision has an impact.
3. Feedback.
   How much information from past cycles of the same decision is considered, and is the decision monitored against actual performance? These two forms of feedback improve quality.
4. Structure.
   Is the decision making process explicitly structured in procedures, time allocation and the like? This enhances quality.

It can be noted that these recommendations have much to do with the real-world context of the decision situation, and that the use of a DSS or of models is not mentioned.

3.6 Literature: psychological perspective

Some would rather call the psychological perspective a decision making perspective, together with what has been discussed above as the organizational perspective (e.g. Open Universiteit 1985). I feel, however, that there is a distinction: the psychological perspective is about individual decision making whereas the organizational perspective stresses the interaction between people in an institutional context.

---

[28] Though not exclusively so, since it was tentatively applied by Bots and van Heck (1989) to horticulture under glass in general, and by Bots (1991) to pot plant nurseries.

Program and play

Does the fragmentation and adaptive behaviour prevalent in managers' work mean that managers face poorly structured decisions? Or is there some structure to these decisions? As Mintzberg states, what appears to the observer to be disorganized and 'unprogrammed' may simply not be understood. Or perhaps managers do not even *want* to organize or program their decision making beyond a certain point. With hindsight it is interesting to read what Geert H. Hofstede wrote in 1970, when Management Information Systems were in their infancy:

> In as far as we can see now, however, what can we expect the permanent problems in the interaction between managers and computer to be? Personally, I can think of three kinds of conflicts, which partly overlap each other:
> 1. Rationality [in the '*homo economicus*'-sense] versus non-rationality
> 2. One goal versus a plurality of goals
> 3. Determination versus play.

Hofstede sr. gives original arguments for his suspicion. Besides mentioning the usual arguments about bounded rationality and personal motives he hypothesizes that perhaps managers just do not *want* to be rational, and that perhaps they are highly motivated by play:

> Play can be defined as activity carried out for its own sake, without any apparent goals but the fun of doing it. Now it appears that it is in playing situations that man finds some of his strongest motivations to perform.[29]

Clearly, in the context of DSS the three conflicts foreseen by Hofstede sr. are still with us. OR-based DSS usually contain models that are characterized by economic rationality, by one goal and by determination. There are also DSS that ask less rationality, explicitness of goal, and determination of their users, for instance spreadsheet-based DSS. Still, these latter DSS leave much less room for play than unaided judgement.

Cognitive biases

During the last two decades an impressive body of empirical psychological research about human decision making and problem solving was done. In part this research, often named cognitive psychological research, is relevant to managerial decision making. A central

[29]For an elaborate treatment of play and motivation in the context of budgeting, see Hofstede (1968). A recent and valuable treatment of play at work, especially play with computer applications, is given by Starbuck and Webster (1991). This article includes many entries into the literature.

notion is that of *cognitive biases.*[30] Humans are endowed with strong abilities for filtering information and for deciding quickly without being fully informed. Sometimes these mechanisms can lead to severe errors of judgement. In the context of planning, a dangerous bias is the so-called illusion of control. This bias consists of underestimating the unpredictable and especially the uncontrollable developments which may occur during execution of a plan. The planner who suffers of the illusion of control falsely believes that his plan is guaranteed to work.

A brief overview treatment of cognitive biases can be found in Hofstede (1988[b]). Elaborate treatments are given by Kahneman, Slovic and Tversky (1982), Hogarth (1987). Von Winterfeldt and Edwards (1986) approach the issue from a prescriptive viewpoint. They convincingly argue that cognitive biases can be reduced by good decision making procedures.

Decision making under stress

The organizational current in psychology merges with what I have treated above as organizational perspective. A major work in this tradition is Janis and Mann (1977). These authors present a theory on the premise that every decision is accompanied by stress. This stress determines perceptions, attitudes and actions of the decision maker. Basically there are three attitudes towards a decision: vigilance, under normal stress; 'hypervigilance', under excessive stress; defensive avoidance, also under excessive stress. Hypervigilance can be expected under time stress, as popular wisdom knows.[31] It is not uncommon for managers to show 'hypervigilant' behaviour, e.g. to take rash decisions in order to end a situation of uncertainty or of inactivity. Defensive avoidance occurs in people who resist change, and is popularly known as ostrich policy. A planner on whom a planning system is forced can be expected to show defensive avoidance.

Organizational context

There is, to my knowledge, not much psychological research on decision making and problem solving in organizations. Experimental psychological evidence usually comes from laboratory settings, whereas organizational data do not tell much about the actual cognitive processes of problem solving. It is my conviction that this gap is with us to stay. An observer who wants to experiment with human problem solving needs to clear-cut a problem artificially so that he can be on the spot when the actual problem solving process takes place. On the other hand, observations in an organization will always show that no single problem or decision can be isolated without omitting essential context

---

[30]Recently, the notion of cognitive biases has been criticized by authors who state that since normative models have proven invalid to describe judgement in the first place, it makes no sense to talk of biases (Anderson 1986).

[31]Consider proverbs such as 'The more haste, the less speed' or 'Haste makes waste'. Similar sayings exist in many languages.

elements. To take an example from quite a different sphere, even in the well-defined context of a game of chess a player who makes a bad move does not necessarily display bad problem solving. The context might be that the player expects his adversary to overlook his chances, or even that he wants to lose the game to attain some other goal, e.g. to change the opponent's mood. An observer can do no better than guess at the problem solver's motives or ask about them after the fact and obtain an answer coloured by hindsight bias.

What we do know about human problem solving is that it is quite flexible and adaptive. So, short of entirely comprehending a problem to be supported by automated tools, if one can design a support tool that allows the user to work in a flexible and adaptive way, one may succeed in helping the user. Any automated DSS involves some standardization, but this need not be undesirable. In a hectic environment some standardization may be beneficial. Consider for instance Simon (1965, p. 97):

> People find the most interest in situations that are neither completely strange nor entirely known - where there is novelty to be explored, but where similarities and programs remembered from past experience help guide the exploration.

The above statement can be interpreted as a restatement of the importance of play in decision making, where play stands for freedom within given rules.

Rationality

Usually the adjective 'rational' is taken to mean 'economically optimal from the point of view of an omniscient decision maker'. This is also the kind of rationality meant in the *homo economicus* school. However, organizational psychology has provided less abstract and more human views on rationality. Translating Vlek (1989, p. 209) we can define rationality as something which makes sense in a given context and under given constraints. Vlek distinguishes four kinds of rationality:

- Representational rationality: know what you are talking about.
- Goal-value rationality: know what you want.
- Methodical rationality: judge and assess correctly within the problem setting.
- Synthetic-analytic rationality: judge whether time should be spent on this problem or on other matters?

The first three types of rationality can be improved, maybe even optimized, within the context of a given problem. The fourth kind of rationality is different: it is only defined if the problem is seen in its organizational context. It includes an assessment of the costs and benefits of spending additional effort on the problem. In my opinion, this fourth type of rationality has been neglected somewhat by researchers on problem solving and decision making, whereas it is central to managerial decision making.

Image Theory

A recent theory on decision making which is both psychologically grounded and takes into account the organizational context is image theory (Beach 1990). Image theory departs radically from earlier, rationalistic theories on decision making such as for example presented in Von Winterfeldt and Edwards (1986). This is understandable since image theory is intended as a purely descriptive theory whereas Decision Analysis as treated by Von Winterfeldt and Edwards is prescriptively oriented.

Image theory stresses the fact that most people, during most decisions, are trying to satisfy much more basic and much more consequential considerations than the maximization of profit. These considerations lie in the realm of personal goals and plans, and of the values and beliefs on which these goals and plans are based. Decisions are adopted if they seem fitting or right in the light of the decision maker's values and beliefs. Thus image theory does not argue with the notion that decision making is self-interested but expands the definition of self-interest as compared to classical *homo economicus* decision making. Image theory describes one decision made by a single individual, rather than the social processes in which such a decision is usually embedded. This feature distinguishes it from theories such as March's Garbage Can.

A decision maker is viewed as having three 'images' which each possess some decision-related knowledge. The *value image* contains values, beliefs and the like. The *trajectory image* contains an agenda of goals and related time-lines for the decision maker or his or her organization. The *strategic image* contains plans for accomplishing the goals. Thus there is a nesting of strategic image within trajectory image within value image. Note the way in which the word strategic, while carrying approximately the meaning which is usual in a management context, is for once at the bottom of a hierarchy.

Framing is essential to set the context for a decision. The context defines which elements from the three images the decision maker believes to be relevant. This selection of elements determines what will happen. The decision maker will usually try to relate the situation to familiar previous decisions. If the status quo is acceptable it will usually not be altered. But a number of motives, ranging from clear necessity to imaginative opportunism, can cause new plans, new goals or even new values to be adopted. Progress monitoring involves trying to imagine what would happen if a to-be-adopted plan were to be implemented. If no acceptable plan can be found, backtracking to the goal level occurs, and the goal may have to be abandoned.

Image theory places some stress on imagining and deliberating, in a way that is analogous to simulation in a computer-based planning context. Consider this quotation from p. 6:

> Deliberative thinking helps in the identification of the decision's consequences - the ways in which principles, goals and plans might be affected by the ramifications of the decision. It allows the decisionmaker to imagine possible futures and to experiment with variations on how to attain desired ends.

3.7 Literature: systems perspective

In contemporary science the notion of a system is omnipresent. This is a fairly recent phenomenon, a reaction to the analytical, mechanistic world view prevalent in science since the renaissance (Ackoff 1979, p. 95). Boulding (1956) has been one of the early disseminators of systems theory, and the first economist to adopt this approach which had hitherto been reserved for biologists, notably for von Bertalanffy. In his classic article Boulding presents what he calls the skeleton of science, meaning that systems theory can be the unifying framework to which the flesh and blood of all scientific disciplines can be attached. Boulding presents a hierarchy of increasing complexity in systems:

1. Static structure, or framework.
2. Dynamic system with fixed, predetermined motions, or moving corpuscle.
3. Cybernetic system with negative feedback, or thermostat.
4. Open, self-maintaining system, or living cell.
5. Genetic-societal system, or plant.
6. Teleological, self-aware system, or animal.
7. Self-reflexive, symbol-processing system, or human.
8. Communication system, or society.
9. Transcendental system, or unknowable.

Although one could accuse Boulding of anthropocentrism, the point of this hierarchy is clear and seems valid to me: organizations are systems of the eighth complexity level, with subsystems of all the lower levels. Thus organizations contain all of the complexity of the eighth level superimposed on the complexity of all lower levels. This should be borne in mind when one uses lower-level system metaphors (such as cybernetics or OR models) to describe organizational processes.

An accessible introduction to the systems approach is given by Kramer and De Smit (1982). They distinguish three main orientations within the systems approach as it is applied to management: an axiomatic current, aiming to develop an internally consistent set of concepts (e.g. De Leeuw 1974, p. 27); an 'organismic' current, using analogies from biology and useful for development of theory (e.g. Beer 1981); and a 'methodistical' current, which views the systems approach as a useful paradigm for solving practical problems (e.g. Ackoff, In 't Veld). I prefer to call the methodistical current a pragmatic approach. Kramer and De Smit's main message is that although there are many different variants of systems thinking, they all aim at studying wholes that are more than the sum of their parts, and that are embedded in a larger whole, the environment. Other authors confirm that the notion of a system has many related meanings. Anthony (1965) cites from Webster's Unabridged:

> (A system is) a complex unit formed of many often diverse parts subject to a common plan or serving a common purpose.

I believe this is a useful definition, although I would add that the plan is not a property of the system but of the person defining the system, and that the system's purpose may differ from the definer's. This feature is apparent in the definition by In 't Veld (1981, p. 9):

> Een systeem is een, afhankelijk van het door de onderzoeker gestelde doel, binnen de totale werkelijkheid te onderscheiden verzameling elementen. Deze elementen hebben onderlinge relaties en (eventueel) relaties met elementen uit de totale werkelijkheid.[32]

Thus, the notion of a system is context-dependent. The objectives and perspective of the definer determine which elements are included as parts of the system, or as parts of the environment, and which elements are not taken into account. Ackoff (1978, p. 77) puts it into words as follows:

> *System* and *environment* are relative concepts: they are not 'given' to us but are imposed by us on the field of our perception. Therefore, we can enlarge or contract either one.

I shall not use formal, elaborate definitions of 'system', such as e.g. presented by De Leeuw (1982). My interest in systems is mainly their partitioning property: the dividing of the world into object system, environment and irrelevant remainder.

Organismic approach: Stafford Beer

A highly original writer and proponent of the organismic systems approach is Stafford Beer, who in his book Brain of the Firm (1981) discusses (p. 17)

> (...) the contribution which cybernetics, the science of control, can make to management, the profession of control.

Beer uses the analogy between organizations and living organisms in his ideas about management. This leads him to envisage an important place for computer heuristics in control, by analogy with natural selection. Algorithmic genetic material leads to heuristic mutations, working towards an unknown goal. The crux is that the evaluation function - natural selection in this case - operates on another set of variables than the algorithm. Transposed to DSS and computers this implies an argument in favour of heuristic control, where the evaluation function is not programmed into the algorithm but rests with the user or possibly with a higher-order program.

Though valuable and worth careful reading, Beers ideas are not easily implementable. It may be that they are just too audacious for present-day people, or that they do not sufficiently take 'the human factor' into account. Perhaps the organismic

---

[32] A system is a set of elements to be distinguished within total reality, dependent upon the aims of the researcher. These elements have mutual relationships and may have relationships with elements of total reality.

metaphor, of level five in Boulding's hierarchy, is just too simple to use as exclusive modeller of a level eight system such as an organization.

Pragmatic approach: In 't Veld

A pragmatic systems-oriented attitude to aiding managerial decision making, using elements from the various perspectives discussed above, is exemplified by In 't Veld (1981). Discussing the lack of adoption of scientific method by managers, he remarks (p. 1):

> In de eerste plaats gaan te veel wetenschapsbeoefenaren er van uit, dat de manager tracht de absoluut beste oplossing voor een probleem te vinden. In de praktijk ontbreekt echter nogal wat aan die interesse. De manager is in wezen vooral geïnteresseerd in het vermijden van slechte oplossingen.[33]

In 't Veld asserts that the manager's main motive is self-interest, which leads to risk aversion. He notices that within large boundaries, any solution for a problem which is not totally unfeasible will work. He also notices that the actual quality of a decision can never be assessed, for lack of a control situation. In 't Veld also mentions the severe time constraints on managers, leading them to disregard time-consuming schooling activities. At best a manager is willing to try out a novel prescription for problem solving, but as a rule he relies on experience.

As a third reason why managers fail to adopt scientific method In 't Veld suggests that management scientists fail to address the relevant problems. Following this criticism, which uses the organizational, and psychological perspectives, he proposes systems thinking as an aid in considering and solving organizational problems. In fact he develops a control perspective based on systems thinking.

Soft Systems Methodology: Checkland

One current in systems thinking which Kramer and De Smit do not mention is Soft Systems Methodology (SSM), introduced by Peter Checkland (Checkland 1981). Soft Systems Methodology was developed at Lancaster University through action research, i.e. research combined with participation in solving 'wicked' problem situations in organizations. During its development Soft Systems Methodology was enriched with the views on judgement of Sir Geoffrey Vickers (Checkland and Casar 1986). The methodology has considerably evolved during the last decade (Checkland and Scholes 1990).

This current is different from the ones discussed until now in two essential regards. To begin with, organizations are not seen as goal-seeking systems. The second important difference bears on how history is taken into account. As I hope to make clear these are

[33]In the first place too many academics presuppose that the manager attempts to find the best solution, in an absolute sense, to a problem. In practice this is not so. What the manager is really most interested in is to avoid poor solutions.

indeed essential differences with other currents in systems thinking.

If organizations are not goal-seeking, what else can they be? What about strategic, tactical and operational goals, and especially what about tactical planning situations? In their research during the seventies, Checkland and his colleagues experienced that it was often impossible to determine precisely when and by whom a decision was taken. Somehow the decision took shape in the minds of the people involved. This observation did not agree with the image of an organization as a goal-seeking, organism-like entity. The only way he could make sense of it was to define organizations as follows (Checkland 1991, p. 16):

> Experiences of this kind, together with the whole experience of developing SSM, convinced us that a precise ontological definition of 'an organisation' would be that they are social collectivities (a hermit not constituting an organisation!) *the existence of which as an active agent is assumed by many people, both members and non-members.*

In other words, Checkland defines an organization as a social construct.[34] It follows that, to the extent that its members have different personal goals, an organization cannot have unanimously shared goals. It also follows that problem situations are social contructs, all the more problematic because no two individuals have the same perception of them. Likewise, the notions of strategic, tactical and operational planning and control only exist inasmuch as they are accepted and shared by the organization's members.

Checkland (1991, p.17) contrasts his view with that of the 'hard' systems thinking of which Mitroff's model (see p. 9 and p. 127) is an instance, which he terms Systems Engineering (SE):

> The SE approach assumes you can unproblematically name the system of concern, define its objectives, and, in the light of these, engineer the system to achieve the objectives. We found this epistemology too poverty-stricken to match the richness of the real life of organisations, in which shifting alliances continually seek temporary accommodations in response to changing perceptions.
>
> Naming the system and defining its objectives, which are given at the start of an SE project, were always, in our situations, part of the problem.

What about a tactical planning situation? At first glance, the situation and its objectives might seem to be well-defined. But a tactical planning situation does not exist in isolation from other situations. A grower, for instance, has strategic concerns (e.g. the existing market is expected to become less attractive, so that new products could be tried), as well as operational considerations (e.g. there is unexpected growth of a product, which might affect its market value) during tactical planning. Moreover, apparently unrelated other situations that occur simultaneously can keep the grower from carrying out as thorough

[34]By the same token, a decision taken by an organization is also a social construct.

a planning as he would have wished (e.g. a new and strict governement regulation on groundwater pollution necessitates unexpected changes in the water-giving system which take all the grower's time, so that he decides to simply repeat last year's cultivation plan although he knows he could make a better one). In March's words one could say that tactical planning situations are not fished out of the Garbage Can before they are dealt with. It is only in organizations with a clear division of labour among Anthony's levels that a tactical planner can be expected to have a reasonably clear-cut problem with reasonably well-defined objectives.

Judgement and appreciation: Vickers

Until now the argument still leaves room for the view that an individual can have clear goals which serve as a guidance for a planning process. This is a hypothesis sir Vickers has something to say about. As he lucidly points out, there are three different forms of judgement. *Reality judgement* has bearing on how one perceives events in the world. *Value judgement* bears on what norms one uses to make sense of what one perceives. Reality and value judgement together constitute what Vickers calls *appreciation.* If perceived events differ from norms, a third form of judgment comes into play: *action judgement.* Action judgement usually requires both reality judgements ("What will happen if...?") and value judgements ("How desirable is that consequence?"). Now the heart of the matter is that (Vickers 1961, in Pugh 1984 p. 197):

> We can as yet give no satisfactory account of the process by which we resolve problems of conflicting value. We only beg the question when we talk of maximizing satisfactions, for the satisfactions we maximize are set by ourselves; and there is no evidence that we reduce those disparate imponderables to a common measure, so that they can be added and weighed. There is indeed much evidence that we do not.

This leaves the human decision maker with, ultimately, nothing but his own responsibility or, as Vickers puts it, his sense of rightness.

Vickers' distinction of three forms of judgement is quite powerful. It can explain the role of creativity in planning as being able to make reality and value judgements about prospective actions. More generally it can explain the difference between men of thought and men of action: the former are good at reality judgement, the latter at action judgement. Vickers remarks that (p. 199) "the simplicity which characterizes the thought processes of men of action has often seemed to me excessive". As a case in point, problems of communication between developers and users of planning systems can ensue because the former think about the planning situation in terms of reality judgement, whereas the latter think of them in terms of action judgement. A third illustration of the force of his division is beautifully given by Vickers himself (p. 199):

> Finally, clear judgement of value and reality only makes more frustrating the common human state of helplessness, when no effective action can be taken; and this is as common in business life as in life at large.

So much for clear and unambiguous goals. In fact Soft Systems Methodology redefines goal-seeking as the occasional special case of the concept of *relationship maintaining*. Both at the level of individuals and at the level of organizations, it assumes, courses for action are selected on the basis of their perceived relevance to the maintaining or creation of relevant relationships. The validity of this view hardly needs arguing. One need only reflect on the importance which relationship maintaining (or rather, as Checkland renames it (1991, p.18), relationship managing) has in commercial settings.[35] In non-Western cultures, relationship maintaining is recognized much more explicitly as an important aspect of organized life than in our individualistic Western ones (Geert H. Hofstede 1991).

Now for the second difference of Soft Systems Methodology with the other systems thinking I presented. This is what one could call its historical consciousness. Checkland (1991, p. 18) summarizes Vickers' thoughts:

> (...) we select between possible courses [of action] using standards or criteria *which are themselves the product of the previous history of this system itself.* (...) Thus does Vickers allow *history* its important role in human affairs, turning his back on classical management science with its focus only on the *logic of situations*.

That the antecedents of a situation determine how it is perceived by those involved in it is really a very commonplace observation. "You never cross the same river twice." But to apply this observation to a systems perspective and conclude that any definition of a system is only valid for a particular observer at a particular moment is something else. It implies that to-morrow, the definition may no longer be valid. This concept of an 'appreciative system' is at odds with conceptualizations of a situation as a formal problem: although its parameters may change, a formal problem has fixed boundaries.

Checkland (1991, p.20) graphically summarizes the ideas Vickers wrote down in prose. Figure 3.1 shows the resulting picture. The picture expresses the dynamics of an appreciative system, such as an organization or an individual. In figure 3.1 time runs from left to right. The twin spirals at the top of the figure depict the flux of events and ideas unfolding through time.[36] The remainder of the figure bears on the actors involved in the system. The picture shows that decision making is a succession of 'appreciative cycles'. Alternatively, appreciation and action take place. The action interferes with the flux of events, the appreciation influences the flux of ideas, and the changes caused are observed in the next appreciation. Besides leading to action, appreciation both guides and is guided by standards of fact and value. As a result, the standards change over time. I hope to have made the point that Checkland's and Vickers' views account for what

[35]Indeed, SSM enables one to understand the success of the 'Soft Sell Method': sell the relationship first, then sell the product.

[36]In fact this flux corresponds to what is usually drawn in graphical displays of decision making as a cloud-like shape and called the environment.

happens when a planner is at work in his office much better than more mechanistic systems thinking does. The reader may also have noted that they accommodate quite a few of the ideas expressed by authors in the organizational and psychological perspectives.

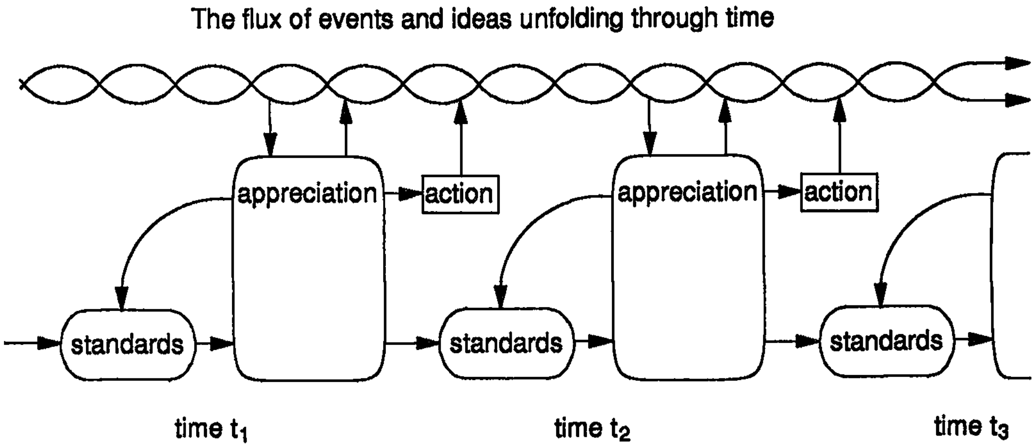


Figure 3.1: The dynamics of an appreciative system (from Checkland 1991, p. 20).

3.8 Discussion

In developing ideas about decision making in organizations and DSS I have made use of the organizational, psychological and systems perspectives in whichever way seemed helpful. I draw most heavily on the organizational perspective to stress the complexity, volatility and unpredictability inherent in planning situations, as well as the interrelatedness of planning situations with other organizational problems. The key notion in the organizational perspective is *context*; I believe that it rarely makes sense to try and describe, or, worse, to try and solve an organizational problem that is separated from its context. The examples from the growers P., Q. and R. have probably illustrated this point sufficiently. Briefly stated, planning situations are *open problems*.

The psychological perspective can help in designing problem solving possibilities of planning systems. It provided me with an awareness that 'what *you* see is not what *I* get'; in other words, the same interface may not mean the same thing to designer and user. Another conclusion from the psychological literature is that in order to get the most out of a user it is he or she who must have primacy in the interaction, because it takes involvement and deliberate thinking to bring out the best in the user.

The systems perspective has strongly coloured my thinking. Systems thinking as a whole is useful because it reminds one that the object of study, such as 'planning situation', is defined subjectively and has an environment. Boulding's hierarchy forces the awareness that organizational systems are inherently far more complex than the techniques available to model them. Soft Systems Methodology combines 'hard' systems thinking with insights from organization theory and psychology.

Describe or prescribe?

There is a discrepancy between prescriptive and descriptive theories of management decision making. Prescriptive models overemphasize the importance of goals and of problems as distinguishable entities. Personal motives and chance opportunities are disregarded. DSS based on such models are biased. They give too much weight to formalized data, which - even if up to date, complete and correct - capture only a fraction of reality and almost exclusively concern the past. These data lack context, which makes them not only inadequate but positively dangerous because they suggest more than they can provide.

Descriptive models show the opportunism and arbitrariness of actual decision making in organizations. They are not encouraging for developers of DSS. However, a developer who is aware of the defects of decision making can help an organization to reduce them. DSS can help make decision making less arbitrary and more explicit.

Fragmentation and adaptiveness

The assumptions about decision making in organizations which I make have important implications for the possible role of DSS. Fragmentation and adaptiveness seem to me to be key notions to describe a manager's work. This means that if a manager is to use a DSS it will have to be both learnable and operable in very little time, and it will have to be adaptive. A result must be obtainable in short time, usually a matter of minutes, and the same holds for changes to plans or what-if analyses.

A discussion along similar lines can be found in Beulens (1990). Beulens remarks that work fragmentation and interrelation of decision processes run contrary to the assumption in the Simon model of decision making that there are neat iterations of sequences intelligence-design-choice. In his view this has important consequences for many aspects of DSS. Beulens remarks (p. 98) that in so far as a DSS requires a certain procedure of decision making,

> Building and implementing DSS can heavily affect the organization of decision making processes and the tasks to be performed by users. It must be regarded as an organization innovation process that may have a great impact on the organization.

This is so because if a DSS is to be effective its use must be integrated in the decision making process. To mitigate disruptive effects of DSS introduction Beulens also remarks that a DSS must be flexible and adaptable to changes in the problem context, in the organization and in the decision making process. This is no small requirement, considering how evasive decision making processes have proved. Indeed, among the defects of current DSS Beulens mentions that they do not support qualitative aspects of decision making and do not provide for adequate task allocation between user and system. But there are other impediments.

Factors that affect DSS performance requirements

Beulens (1990) makes a number of pragmatic observations about organizational factors that affect performance requirements for DSS (pp. 105-106):

- It is difficult to give automated support for tasks which are unstructured with respect to the problem demarcation or the available data. DSS usually need high quality data. In the pot plant case, this means that automated data recording is very desirable in order to make automated support for cultivation planning practicable.
- Managers deal with tens of problems simultaneously. Thus they may have to use various DSS at the same time for related problems. This may be too demanding, resulting in failure to use the systems.
- Organizational roles and procedures determine which role a DSS user plays in decision making. A planner's perspective may not be beneficial to the organization as a whole.
- DSS are not always available when they are needed, e.g. while walking around the greenhouse or while attending a meeting.
- DSS usually change the work of their intended users. This means that implementing DSS is a process of organizational innovation, not just a new way of doing the same thing. The consequences cannot be foreseen, so one should be prepared to respond creatively to unexpected developments.
- Organizations change continually. DSS are usually not adaptable enough to cope with this.

It can be concluded that the actual decision problem, as an abstraction from the organizational whirlpool, plays no more than a modest role.

Ideal relationship manager-DSS

The relationship between a manager and his or her DSS can be similar to what Mintzberg (1975, in Pugh 1984, p. 436) proposes for the cooperation between a manager and the management scientists of his or her organization. Consider substituting 'DSS' for 'analyst' in the following quotation:

> Managers have the information and the authority, analysts have the time and the technology. A successful working relationship between the two will be effected when the manager learns to share his information and the analyst learns to adapt to the manager's needs. For the analyst, adaptation means worrying less about the elegance of the method and more about its speed and flexibility.

Sharing information with a DSS may seem a strange notion, but it makes sense if it is understood as the possibility for the user to introduce current, diverse information in the DSS to modify the DSS's functioning. The DSS must, like the analyst, worry more about speed than about elegance. However, the metaphor should not be carried too far since a DSS, after all, is not an analyst.

In fact the relation between users and developers of an information system is often ambiguous, since both have their private motives which need not totally overlap. Developers may use their power as experts to get the users where they want them (Willmott et al. 1990). Such a relation troubled by hidden agendas is not desirable, most definitely not in the case of DSS.

DSS: when?

Certain requirements have to be met if a model-based DSS in Mitroff's sense is to be usable. The real-world problem must be demarcated and then formalized. Then the resulting formal problem must be solved. Finally, the solution to the formal problem must be translated into a solution to the real-world problem situation.

These requirements will usually only very partially be met, as we have seen. The main reason for this is that organizational problems make little sense without their organizational context. They are not clearly demarcated and cannot be entirely formalized. On the other hand, there is agreement that efforts towards demarcation and formalization are to some extent desirable and can lead to better management. But it is crucial to keep in mind two things. When modelling the developers must keep to the perspective of the decision makers involved. Otherwise, the model will only reflect the developer's view of the problem. When solutions to the models are obtained an important step remains: the solution to the formal problem must be translated into a solution to the real-world problem. If this step is neglected, one is trying to solve a problem that is abstracted from its context, which is precisely what the organizational perspective warns against.

The user of a model-based DSS will have to model the problem and to translate the solution of the model into a solution to the real-world problem. These two activities are new and require changes in the user's way of dealing with the problem. As a consequence, introducing a DSS has much to do with organizational change. A DSS must fit within the organization, which sets limits to the use of models and may preclude the use of ambitious models. But a DSS can also be used as an agent of organizational change. In this case the requirement that the DSS fit would mean that either the level of support be gradually enhanced or else that one be prepared to make personnel or job changes.

DSS-assisted problem solving

It is not easy to find the right measure of prescriptiveness if one wishes to discuss the use of DSS in problem solving. Too prescriptive an orientation could lead to unrealistic requirements. In the pot plant case for example, even if a strictly circumscribed procedure for cultivation planning exists, chance events will invariably force the grower to improvise. In November 1990 grower Q.'s secretary fell ill and because a substitute who knew the nursery well was not available, Q. had to do her work during the very weeks reserved for planning next year's cultivation. Incidentally, he was all the happier to have PROPLAN HAND available.

On the other hand, DSS are intended to enable improved decision making, so that a prescriptive element in a model of DSS use is justifiable. In an organization which operates in a predictable, formal environment, DSS can be more prescriptive than in an organization which finds itself in a turbulent environment.

My way of looking at DSS-assisted organizational problem solving is depicted in figure 3.2 below (adapted from Hofstede and Simons, 1987). In terms of figure 3.1, figure 3.2 depicts a single appreciative cycle.

No single figure can capture every aspect of organizational problem solving. Figure 3.2 emphasizes the problem solving process for one isolated problem. Information enters at the left, provided by both environment and 'real problem' within the organization. The information is first filtered by the personal context, then by the management context. Only a selection of the information arrives at the problem context as supported by the DSS. The outcome of the DSS-assisted problem solving process is then considered in the light of the management context. The output may be modified here, because of considerations relevant to the management context but not modelled in the DSS. Then the modified output is considered in the light of the problem solver's personal context. Only an outcome which is satisfactory within this context is adopted. Each context has primacy over the contexts inclosed in it, so that the personal context has overall primacy.

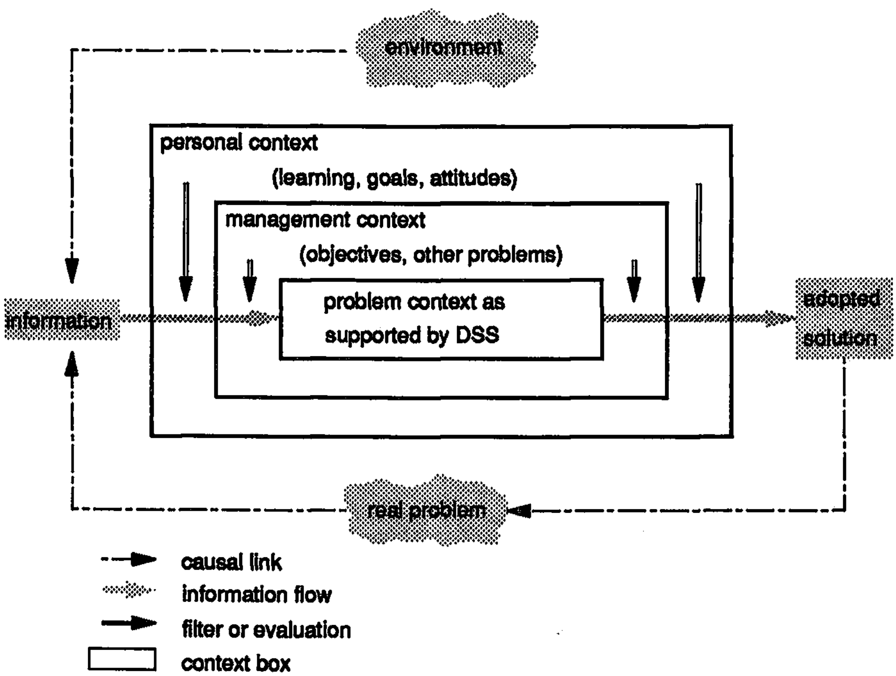


Figure 3.2: Problem solving with DSS

This phenomenon can for example take the shape of hidden agendas of the problem solver.[37]

Nested context levels

The nesting of the three context rectangles is crucial. It signifies that the management context determines both input and output of the problem context. The personal context, in turn, determines input and output of the management context. More generally, any context level overrules those nested within it (Hofstede and Simons, 1987).

At the input side a DSS can to some extent be shielded from the filtering effect of higher contexts, if data are automatically gathered. At the output side, the relative importance of personal versus management context varies greatly. If a person does not identify with his job, or does not feel for the organization in which he works, then the personal context strongly prevails. By contrast, in an 'excellent organization' in the sense of Peters and Waterman (1982) the individual problem solvers' interests coincide with those of the organization; as a consequence the decision maker is highly concerned with the 'real problem'. This is usually the case in a small enterprise such as a pot plant nursery. If, on the other hand, the decision maker is unconcerned with the real problem, as can be expected in the case of institutional planning, then the personal context serves interests outside the problem domain, such as minimizing effort. For example, the course planning department of a university will usually attend best to the needs of those who complain if unsatisfied. This attitude is consistent with their personal interest, but not necessarily with the management of the university.

Interrelated problem situations

Although figure 3.2 does not consider a problem as an isolated phenomenon, it does consider only one problem at a time. In reality a variety of problems occur simultaneously, as depicted in figure 3.3.

A number of decision situations, that is, problems or opportunities, occur simultaneously. These situations are dealt with in many problem solving processes which have multiple interconnections indicated by the lines departing from each problem solving process in the figure. Only a few are shown in the figure, but tens to hundreds coincide. Some of these problems, in fact a small minority, have been 'prefabricated' by designing and implementing a DSS for them. This mess of interrelated problems is abstracted by management, for the most part after the fact, as a number of separate decisions. There is no actual one-to-one relationship between problem and decision, although there is a formal correspondence.

---

[37]More fundamentally the personal context could be further divided into strategic image, trajectory image and value image as proposed by Beach (1990, see p. 82).

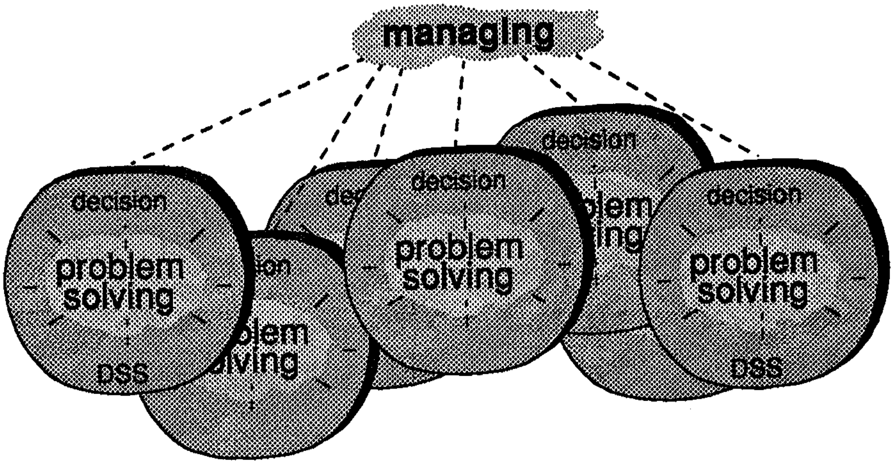


Figure 3.3: Management as a collection of interrelated problems

To give an example: suppose a conflict exists between workers at a pot plant nursery. A planner charged with rostering may try to solve this organizational problem simultaneously with a planning problem by an adroit coupling of workers into teams. Formally, the only problem solved is the rostering problem, but in reality the same problem solving process has also solved -or rather, resolved - the conflict.

Conclusions: modest models, active users

A synthesis of the various perspectives leads to the conclusion that the organizational context is the most decisive factor to guide the development of planning systems. This is particularly so because the development of a DSS is accompanied by *organizational change*. The capacity for change and the will to change on the part of the organization, *in casu* the planner, are limiting factors. If this is not kept in mind then the organization may fail to build commitment during development or it may reject the new DSS on delivery.

Likewise, the organizational context is a prime determinant during actual planning. *A priori* formalized knowledge is never as up-to-date, never as richly varied, never as organizationally embedded as the planner's knowledge. Therefore, the organizational context cannot satisfactorily be taken into account if the DSS behaves as a black box. Rather, the inclusion of context necessitates a central role for the planner during planning. The planner should be able to make use of any type of knowledge to influence the plans that are being made. Also, he must be able to treat the planning situation as an open problem, i.e. to change the problem while solving it.

Regarding an active DSS user, the psychological perspective has much to say. If a user is actively involved, the quality of his or her thinking will be high, and the quality of the resulting plans can also be expected to be high. Of course a user can only be actively involved if the DSS's interface allows it.

The systems perspective brings about the realization that the types of formalism used in DSS can by their nature provide no more than very partial models of a practical decision situation. This is another argument in favour of modest models and an active role of the user of a DSS. An important contribution from Soft Systems Methodology is that appreciative systems are continually shaped by their own history. One need only consider the team involved in building a planning system as an appreciative system to conclude that the process by which this planning system is developed matters much to the end result. Soft Systems Methodology is also useful for pointing to the close intertwining of system and environment. It reinforces the conclusions drawn from organizational theory.

3.9 SCAPSIS, A quick-scan aid for a planning situation

The previous sections have made it clear that a wide diversity of factors have to be taken into account when decision support for planning situations is considered. I have attempted to make the above perspective on decision making in organizations usable, and have created a tool which can be used by designers to perform a quick scan to assess chances for planning system development in a given situation. The acronym SCAPSIS stands for SCanning Aid for Planning SItuationS. The idea for the tool was born of the frustration of unsuccessfully trying to get pot plant growers to accept normative support. The tool investigates contingency factors that have to be taken into account when assessing what kind of planning system, if any, is suitable for a given planning situation. Given a decision situation in the area of tactical and operational planning, I propose to rate eight aspects of the problem context. Aspects that have bearing on the organization, the environment of the organization, and the direct decision making context are grouped in the category 'context characteristics'. Aspects that relate directly to the object system are grouped in the category 'object system characteristics'.

Context characteristics

1. How large is the organization?
   In a very small organization the tactical level is not separated from the strategic and operational levels, because the same person or group of persons attends to all levels. This means strong relations of the decision with many other decisions. Plans tend to be informal or even absent. If on the other hand the organization is large then the planning process becomes more detached from other organizational decisions and there may exist formal organizational planning procedures.
2. Is there a specialized planner?
   This question is correlated to the first one but deserves special mention. In the absence of a specialized planner, planning is often done by someone with many other obligations. It may be done in an *ad hoc* way, especially if the other

responsibilities of the person who does the planning are perceived as more important or more entertaining. If there is a person specifically responsible for the planning situation, which often occurs in larger organizations, then usually the planner is an expert, spends much time on planning, likes to do a good planning job, and has an interest in automated support (but perhaps also fears to be chased from his job by a computer). It might be that there is a team of planners instead of one planner; in this case the same holds for all the planners.[38] Each planner is often a problem owner, that is, a person who is faced with a problem and who can to some extent change it, e.g. by telephoning to try and change some constraints, or by wilfully neglecting aspects.

3. How stable is the environment?
   In a sector where the environment (in a broad sense) is dynamic and unpredictable, decision making has to be highly responsive to external signals. If the environment is comparatively stable, there is more to be gained by an orientation towards improving the efficiency of internal operations. Often an organization is to some extent free in defining its environment, as in the case of a pot plant grower choosing to sell through private channels or via the auction.
4. What is the frequency of the decision making process?
   Tactical and operational planning decisions are by definition recurrent. The present point is: how often does the decision making process occur before a decision situation changes so much that it must be considered to be different from the current decision problem? In most organizations changes are fairly frequent. For instance, if a planning decision is taken yearly and major organizational changes can be expected to occur once every three years, developing a DSS to support the generation of these plans may not be cost-effective. This is of course dependent upon the value of the decision in question.

Object system characteristics

5. How predictable is the object system?
   The better an object system can be predicted, the better an automated model of the object system can function as a means to generate decision alternatives and expected consequences thereof. An object system can be considered predictable when a planner accepts a model of the object system and its forecasts within a specified context. In fact when the object system is considered not to be predictable this can mean one of two things. Either it is known that the situation is inherently unpredictable (e.g the weather) or there is currently no predictive model but one might still be developed. Although in both cases a simulation model can be of great value, the potential predictive benefits of a model are larger in the latter situation.

---

[38]Be it remembered that in this study, the singular is used for convenience, i.e. 'the planner' often stands for 'the planner or team of planners'.

6. Are people unaffected?
   This point is no more than a special case of the previous one but so important that it deserves special mention. As soon as people's interests are affected by a plan then a variety of constraints come into play. Both during plan generation and during plan execution there will be soft constraints related to persons. Especially if planner and planned have a personal relationship, interpersonal considerations will play an important part.
7. How controllable is the object system?
   To some extent, adaptive control measures can counteract unpredictability, thus making planning feasible. An important assumption to make planning a sensible activity is that planners are able to generate plans that can actually be realized. For instance, in pot plant culture one can to a certain extent counteract unfavourable radiation circumstances by artificial lighting so that a desired growth rate can be met.
8. Are reliable data available?
   However predictable and controllable a process, if data needed by a model are not available, not reliable, outdated, or too expensive, the output of a DSS based on this model is of limited value.

Establishing the scales of the tool

SCAPSIS can be applied by scoring each of the eight characteristics on an ordinal scale from 1 to 5. Figure 3.4 shows what real-world properties may be associated with the extreme scale values. The height of the scores merely implies a reality judgement, not a value judgement!

For the time being SCAPSIS lacks empirical validation. However, it is my conviction that by using the tool critically it can evolve into a valuable and empirically sound instrument for assessing DSS chances. The following examples show how the tool can be applied, and how the scales relate to real-world problem aspects. The first is a planning situation which rates low on most aspects, whereas the second is one which rates quite high.[39]

---

[39]A more elaborate example with both high and low scores will be given in the last section of this chapter.

	feature	situation with score 1	situation with score 5
1.	organization size	1 to 5 people. At most one manager.	Over 100 people. Formally defined, clear function separation
2.	planner	Boss does the planning among many other tasks.	Specialized planner or team; does nothing else.
3.	stability of the environment	Not all relevant variables are known. There are major changes in external conditions on each repetition of the decision making process.	All variables are known, as well as their values. The conceptualization can cope with the changes of external conditions (markets, prices, laws, ...).
4.	frequency of the decision making process	Unique decision.	Decision process repeated unchanged many times for subsequent planning periods.
5.	predictability of the object system	No accepted model of the object system and/or essential variables are unpredictable within, say, 50%.	Accepted model of object system and all variables are assumed sufficiently predictable within allowed %.
6.	people	People are being scheduled. Their wishes are very important.	No interests of people are affected by the plan in any way.
7.	controllability of object system	Processes not at all controllable.	Processes very well controllable.
8.	data quality	Data are of unknown quality, often rumours, hunches, expectations.	Data are quantitative, accurate, complete, up-to-date, reliable.

Figure 3.4: SCAPSIS in detail

The case of grower R.

For the first example we return to grower R., the third grower to whom the reader was introduced earlier in this chapter. Grower R. wishes to go quite far in automated support of production planning; in fact the planner has been experimenting with Linear Programming-based DSS.

The scores on the SCAPSIS questions for grower R. are as follows:

1 Organization: 50 persons, in the process of specializing. Entire management meets weekly. Score 2, going towards 3.

2 Planning: Would-be specialized planner, much needed for other work. Score 3.

3 Environment: Highly dynamic. Markets, technology, products subject to yearly changes, so that conditions differ each planning cycle. Score 1.

4 Frequency: Each year has a different greenhouse assortment and different

objectives. Yet the principle is constant over time. Score 2, striving towards 3.

5 Predictability: Growth is unpredictable due to weather and to quality of basic material. Growth models are not available for most cultures. Fluctuations up to some 25% of cultivation time. Furthermore, unexpected events or opportunities may arise. Score 2.

6 People: No evident relationship, although planning has some consequences for labour, especially at peaks. Relation between management and labourers is valuable, so labour conditions are important. Score 4.

7 Controllability: Growth conditions can be controlled by extra lighting, heating, chemical treatment and other means, but costs are considerable. Also, new cultivars tend to show unpredictable reactions to treatments. R., for instance, unexpectedly had to cope with ugly flowers in some Spathiphyllum varieties in 1990. Practically speaking the process is hard to control. Score 2.

8 Data quality: Recording of production data is automated and sufficiently detailed, viz. to the individual production lot. The automated system does not, however, generate the desired reports. External market price data are not specific enough. For instance, differences between growers in prices for the same products can be significant but are not found in the statistics coming from the auction. Score 2.

There are many low scores of 1 or 2. If the innovations desired by R. are realized, some scores can rise a few points, but not drastically. On this basis I predict that the planner must be very cautious about automating cultivation planning. Optimization should be used with care, because its preconditions do not hold. There are too many factors which are unpredictable, or unknown, or non-quantified, or have to be decided *ad hoc*. A modest approach, aimed at generating feasible and robust plans, possibly at 'what-if' exploration of given plans, seems the highest level of support that will fit. A very important requirement for a planning system is that it enable re-thinking of a plan at short notice, preferably within minutes (for instance during a management meeting). Integration of a planning DSS with actual plan realization data is essential because it saves a lot of clerical work.

As of 1991 the only automated support system used in production planning by grower R. is a spreadsheet application built by the planner himself. By the way, most of the characteristics of grower R. are fairly generic to pot plant nurseries. An average nursery, being both smaller and less innovative than R., would have had even significantly lower scores.

The case of physical distribution in a large company

The second case is of a large manufacturing firm in the food sector which I shall nickname the BARS company.[40] The problem area is physical distribution. Several related problems occurred in this area, one of which is described here. This one, a

[40]This case is also described in Hofstede and Beulens (1991). It is taken from Beulens' personal experience as one of the managers involved.

problem of operational physical distribution planning, was to effectively allocate and dispatch finished BARS products to client-warehouses throughout Europe on a daily basis. The stocks at the factory warehouses had to be allocated to client-warehouses. Subsequently the allocated products had to be assigned to trucks under stacking, volume, and weight constraints. Data were available that those involved accepted to be controllable and high-quality: the allocation was based on up-to-date sales forecasts and stock data. An important objective was to attain equal expected service levels per product over all depots.

The logistics managers at BARS gradually developed a good insight in the way in which this problem could be dealt with. This does not mean they developed a perfect optimizing model. It does mean they developed a set of models and solution procedures that represented the main quantitative aspects of the problem well enough to be accepted by the planners. It also means that organizational procedures were developed that were accepted and adhered to by the parties involved (factories, country sales-units and warehouses) for the accurate and timely administration of sales forecasts, actual sales and stock data.

Over time a sequence of DSS have been developed and implemented for this problem. Versions of these DSS have evolved such that the task allocation between planner and system has dramatically changed. Likewise, the level of integration with other physical distribution systems has changed through the use of a shared database. Importantly, the roles of the planners have also changed. As the scope of functions provided by successive systems increased, tasks were performed in new ways using these sophisticated functions. The sophistication resides in the fact that models used for the allocation of finished goods to client warehouses and for determining the optimal loading and stacking of these goods into trucks can generate feasible truck-loads that are frequently accepted and dispatched without further planner intervention.

The last version of the DSS has been effectively used during a number of years to generate decision scenarios which the planners could further refine at a qualitative level. The system has provided the users with structured working knowledge of optimization models and their restrictions. They accept the models and their results as valid in the problem situation and claim that the effective execution of their tasks requires the use of the system.

If one looks at the context and object system characteristics of the problem situation just depicted it is easy to see that the score for all eight characteristics is high (4 to 5). Within tight organizational constraints the problem has been provided with structured solution procedures that are accepted by the planners. Thus there have been simultaneous processes of organizational change and of learning, thanks to which a sophisticated DSS offering model-based support with little planner intervention has been developed and implemented and is being used.

3.10 Discussion of SCAPSIS

Limitations

A *caveat* about this assessment tool is appropriate. SCAPSIS reveals something about the feasibility of a DSS for a given case, about the type of DSS that could be appropriate, but not about the actual opportunities for getting a DSS built and used! For example, it does not take into account preconditions for DSS success such as capacity for innovation of the organization, or personal acquaintance of the DSS designer within the organization, or the attitude of those who have to pay for the development, or a state of readiness to accept changes ('unfreezing') in the organization at the beginning of the development. These and similar aspects are not explicitly dealt with in this study. However, it stands to reason that the literature covered in this chapter is relevant to them. Building and maintaining a relationship between developers and user organization will be important.

Potential

The two examples show that the SCAPSIS scores bear a relation to the chances for DSS, in particular to the prominence of models in DSS. Given more empirical validation, SCAPSIS can grow into a valuable aid for *a priori* assessing chances of DSS development for a planning situation.

A different possible use of the tool is to scan a given decision situation not with a view to build a DSS but rather to gain insight into the situation itself. *In extremo* this insight could even lead to suggestions for altering the situation, e.g. by taking measures to improve the score on one of the aspects. Incidentally, it is good to note that improving the score on an aspect need not be desirable. This is for instance shown by Alleblas' 'fitting management levels' discussed in the second section of this chapter (p. 71-72).

Whatever the objective of using the tool, the insight to be gained by user and designer of a prospective DSS by considering the planning situation in the light of the tool is valuable. It may be a help in starting the learning process through which user and designer must go together.

I hope that the first effect of SCAPSIS is to generate a fruitful exchange of ideas among DSS researchers and practitioners. Perhaps the perspective on decision support to which the tool's name refers will prove to be even more important than the assessment tool itself.

3.11 An application of SCAPSIS

The case

In his doctoral dissertation, Verbeek (1991) describes two case studies on manpower planning in an airline company, KLM. He covers the entire development process, from the original problem situation to the working system. His description of both the development process and the resulting systems is unusually elaborate. In particular, he mentions those aspects of the development process that he sees as crucial, even if they are the sort of aspect not usually thought fit to write about. Communication problems between all the parties concerned with the development is such an aspect. In short, Verbeek's book lends itself very well to a mental exercise with SCAPSIS.

The two cases of Verbeek's book run along similar lines. The larger project concerns the system CAPTAINS for strategic manpower planning, and this is the case which shall be scanned here.

Verbeek includes a detailed description of the KLM organization in his third chapter. From this description, only very few points will be briefly reiterated here. Those who wish to acquaint themselves with the full complexity of the case are referred to Verbeek's book.

KLM is a large company with an explicit formal structure and stable procedures for planning. Short-term plans are made, which cover the next year, and long-term plans, which cover five years, starting more than one year after the conception of the plan. Long-term plans are made yearly, with a rolling horizon. KLM counts about one thousand pilots. There is a Crew Planning Department which has a number of responsibilities. In the area of strategic manpower planning their responsibilities are:

- To make sure that, given a future production plan, the right number of personnel of each category will be available at the right time. The notion of a category refers to the fact that pilots can only fly the types of airplane for which they hold a license.
- To report to higher management on expected bottlenecks. A combination of a function in the cockpit with a type of aircraft is called a *seat*. Training a pilot for a new seat is a time-consuming and expensive affair, and pilots are a scarce resource. Therefore bottlenecks in the availability of pilots for a seat are hard to remedy, and can endanger the flight plans. This in turn can do great damage to the public image of KLM as a reliable aircraft company, which is one of the company's main assets. Likewise, surpluses of pilots for a certain seat are costly. To make things worse, it is hard to predict which career steps pilots will wish to make and when. A bidding system is used to match demand for seats with supply of seats, but pilots may not bid as expected.
- To quantify the effects of proposals to change the Collective Agreements between the trade unions and the company. These Collective Agreements specify the labour contract rules which among other things determine how many pilots will be needed for a future flight schedule.

To deal with these responsibilities, the Crew Planning Department issues strategic plans, which contain as primary decisions (1) the number of transitions between seats and the moment at which they are scheduled, and (2) the number of new pilots to hire for each period.

The planning situation is highly constrained. The career path from the lowest to the highest seats is at most seven steps long. Thus, if a pilot in the highest seat retires, it may take seven changes in seats to fill the vacancy from below without affecting the number of pilots in each intermediate seat. Besides, there are many constraints on the type of transitions allowed and the time windows in which they should take place.

Current practice - before the start of Verbeek's work - was that the planners used a combination of huge wall planning boards, paper planning and several non-connected automated systems. On average it took seven days to produce a five-year plan. Because of the complexity of the task and the fragmented support, the first acceptable plan was usually the ultimate plan. The time that elapsed before questions from higher management about the plan could be answered was about three weeks, which was often too long to make changes possible. It was difficult to justify a plan to higher management.

Approaching the situation with SCAPSIS

In the strategic manpower planning case described above, a DSS project was started. This project will be discussed in the next chapter. But if SCAPSIS had been used at the beginning of the project, what would have been the result?

The ratings on context characteristics are:

1. Organization size: 5

   The organization is clearly very large. Functions are highly separated and formally defined.

2. Planner: 5

   There is a department specialized in the planning task under consideration, the Crew Planning Department.

3. Environment: 3

   The long-term plans which cover 6 years attempt to predict developments in the environment, such as expectations about the demand for air transport and expectations about political aspects, e.g. landing rights and fly-over rights. These plans are a useful basis for manpower planning, but they sometimes miss the mark.[41] However, relevant variables are known and the structure of the airline market is fairly stable.

4. Frequency: 5

   The planning process has taken place in similar form for several decennia. Since 1974, several automated systems have been used.

The ratings on object system characteristics are:

5. Predictability: 3

   The object system is well-defined. Aircraft and seat types are known, numerous hard constraints on transitions between seats exist, labour agreements are formally specified and adhered to. However, small perturbations in the system can have large consequences. On page 59, Verbeek gives an example of a situation in which one unexpected bid results in two nasty vacancies.

6. People: 1

   The plan is all about people, who moreover are highly valuable and scarce and whose wishes are therefore taken very seriously.

---

[41]The Gulf war of 1991 and its effect on airline companies are a case in point.

7. Controllability: 3
   Lower-echelon transitions are strictly constrained through a system of 'Regulated Flow'. The bid system for higher-echelon transitions is an attempt to regulate the mobility of senior pilots. Yet these pilots can for personal reasons make different decisions than expected.
8. Data quality: 2
   The data about the formal career constraints are hard. There are, however, also data of political nature about Collective Agreements which cannot be reliably estimated[42], and it is entirely impossible to get hard data on future personal career decisions of pilots.

It can be seen that, although the KLM strategic manpower planning problem looks quite well-defined at a first glance, SCAPSIS still gives low scores on a number of aspects. Interestingly, the context aspects score much higher than the object system aspects. This means that the context is well-defined; indeed KLM is a stable organization with many formally defined procedures. In such a context one expects planning problems to be attacked in a formal manner. However, the object system of strategic manpower planning is not as well-defined as its context. In particular, the behaviour of pilots is an uncontrollable and unpredictable component, of which small perturbations cause large perturbations in the adequacy of plans. Incidentally, KLM has long since recognized this, or they would not have made the elaborate attempts to control the flow of personnel that exist in the KLM organization. Studies on this subject have been carried out since 1961 and systems for regulating the mobility of pilots have been used since 1974.

The conclusion to be drawn is that probably the single best solution to this planning problem is to improve the predictability of the pilots' behaviour. Since shortage of pilots is so detrimental, it would be justifiable to make considerable costs for this, e.g. on a system of bonuses.

If changing the situation is not possible, then a planning system could comparatively easily be made which models the seat transition constraints and allows flexible sensitivity analysis of the bids and of the new positions which KLM should offer dependent on the pilots' choices.[43]

---

[42]Incidentally, the Collective Agreements are a case in which it may be more profitable to change the planning situation (by lobbying) than to take it for granted.

[43]In fact this has been done, see Verbeek (1991).

4 Quality of decision support systems

When I say, "Quality cannot be defined,"
I'm really saying formally, "I'm stupid about Quality."

Robert M. Pirsig (1974, p. 200)

This chapter addresses the question how to assess the quality of a decision support system. There are a number of aspects, concerning both the DSS as a *product* and the *development process* leading to that product, which are relevant for determining quality. Therefore both the 'what' and the 'how' of DSS, including planning systems, are considered in this chapter.

The chapter sets off with a review of the literature on quality or, with another word, evaluation of DSS. Quality of DSS has not received very much attention in research. Therefore the literature is both rather scarce and diverse, and some literature on management information systems or information systems in general is included. After the literature review there is a discussion integrating all literature.

Quality assessment of DSS is a reputedly hard issue. The literature review shows that after nearly two decades of research, no simple recipe has been found. Nor will this happen in the future. Existing evaluation approaches do not pay attention to all elements that are important. An alternative, which integrates elements from the literature with my own ideas, is proposed in the last sections.

4.1 A quality chain

To give structure to both literature review and discussion it is useful to distinguish the components that play a role in DSS-assisted decision making. Planning, it has been argued in the previous chapter, is an instance of decision making, and anything that holds for decision making in general also holds for planning.

The perspective on the role of a DSS in management decision making which is taken in this study is roughly schematized in figure 3.3 (p. 96). *Managing* involves taking numerous decisions. For each of these, a process of *problem solving* occurs. It is directed by the user's goal and by the available *DSS*. On the one hand, the problem solving activity results in a number of effects not shown in figure 3.3, such as e.g. learning by the user, or the recognition of another problem or opportunity. On the other hand, a *decision* results which in turn has an effect on management.[44] Although, as indicated in the previous chapter, figure 3.3 is a gross simplification of reality, it is useful because it shows one important chain of relationships, which runs between management quality, decision quality, problem solving quality and DSS quality. The relationship between the links of this chain is that the quality of the right-hand link is one of the determinants of the quality of the left-hand link. One can even add one more element to the chain, since one of the determinants of DSS quality is the DSS development process. Whereas figure 3.3 took the perspective of the manager, the DSS is central in the present context. Given this, 'management' becomes an ambiguous term, and may be misleading. Therefore, 'functioning of the organization' is used instead as the extreme left link. The '*quality chain*' then becomes the one shown in figure 4.1.

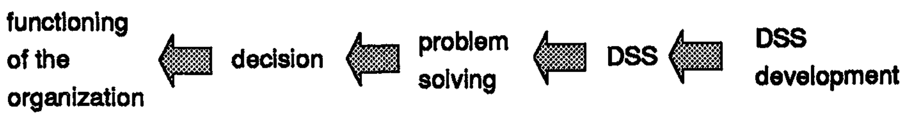


Figure 4.1: The quality chain of DSS

Whereas the ultimate justification of a DSS lies at the left side of this chain, in an improvement of the functioning of the organization, the role of the developers is limited to the right end: the DSS, or rather its development process. If quality aspects of DSS development or DSS are to be taken seriously the relationships through the chain must be known. If these relationships were not known the influence of DSS development quality or of DSS quality on the functioning of the organization would be impossible to assess and it would be impossible to justify DSS or DSS development guidelines by referring to the functioning to the organization.

Obviously, the actual situation is usually somewhere in between these extremes: the relationships through the quality chain are partly known. As put forward in the previous chapter, there is not only causality from right to left through the quality chain, but there is also ambiguity and chance. Likewise, there are causal shortcuts through the chain. After the literature review, I shall address these issues.

The quality chain is also, in a way, a process - product chain. Starting at the right-hand side, the *process* of DSS development results in the *product* 'DSS'. With this DSS, a decision maker can carry out a *process* of problem solving resulting in the *product* 'decision'. This decision then co-determines the *processes* that constitute the functioning of the organization.

[44]As remarked in chapter one, 'problem solving' and 'decision making' are treated as near-synonyms in this study. In the present chapter, 'problem solving' is used because it emphasizes the process rather than the resulting decision; the decision is given a separate treatment.

In this chapter, a division is drawn between the process of system development on the one hand and the rest of the quality chain, which represent the DSS in use, on the other hand. The development process precedes the other links not only in time, but also causally, and it therefore deserves special attention.

4.2 Literature: DSS quality in general

History

Main issues and problems of DSS quality assessment have been signalled as early as the DSS concept itself. In the nineteen-seventies, a community of researchers in the USA, especially at M.I.T. (Massachusetts Institute of Technology), expanded the concept of MIS to that of DSS. Some publications of researchers working during this period are e.g. Alter (1980), Gerrity (1971), Ginzberg (1978[a], 1978[b]), Gorry (1971, 1983), Keen (1975, 1978), Scott Morton (1971), Stabell (1983). They signalled the difficulties involved in evaluating DSS. However, the success of the DSS concept was overwhelming. In the eighties, the self-reflective DSS movement was overtaken by the 'DSS bandwagon' (McLean and Sol 1986) on which researchers and practitioners alike eagerly jumped, because there was glamour to be won and money to be made. Any computer program that could pass for a DSS was taken for granted. The quality issue was not given much attention and went largely out of focus, at least in the literature. At present, however, the bandwagon is under fire and DSS have to prove themselves just like any other information system.

Ginzberg (1978[b], p. 49) sums up nicely what makes assessing DSS quality so difficult:

> Conventional techniques also seem to be inadequate for the evaluation of DSS's. Systems designed to take over the performance of existing tasks can often be evaluated on the basis of the cost of task performance: a successful system should reduce this overall cost. DSS's, however, are designed to enable the performance of **new tasks**. This cost displacement model cannot be used to accomplish an evaluation of a system which aims to make the organisation more **effective**, as opposed to more efficient, by enabling it to do things it could not do previously.
>
> The DSS evaluation problem is particularly thorny. It is often difficult to place a value on the promised outcome of using the system, e.g. better decision making. We normally cannot even be sure that the DSS is the reason for the observed change; (...) The way out of this dilemma seems to be in recognizing that a prime objective of DSS development is learning, both individual and organizational. (...) Unfortunately, measurement techniques in this area are not well developed, and further research is required.

Given the thorny nature of DSS evaluation it is no wonder that managers evaluate the desirability of DSS in a highly intuitive way, as Hogue and Watson (1983) found in a survey.

The literature review will continue with a discussion of some multidimensional approaches to quality assessment, after which the literature on individual links of the quality chain will be treated.

Comprehensive approaches

Some comprehensive multidimensional approaches to measure the value of information systems have been suggested since the early years of DSS. Often, however, these focus on information systems in general and do not take into account precisely the qualitative, innovative aspect which is so important in DSS (e.g. Chandler, 1982). Three such comprehensive approaches will be discussed here.

Hamilton and Chervany

Hamilton and Chervany (1981) have proposed a useful set of concepts to evaluate the effectiveness of information systems. These authors' outlook fits rather well with the 'quality chain' so that elements of their framework for evaluation can be used in this study. Hamilton and Chervany distinguish a goal-centred and a system resource view on effectiveness. This distinction, they say, is analogous to the distinctions 'summative' versus 'formative', ends versus means, and outcome versus process. Obviously, in the absence of measurable objectives, a system resource view is the only feasible one. In their words (p. 56):

> The system resource model recognizes that systems fulfil other functions besides accomplishment of official objectives, and that these need to be considered in assessing system effectiveness.

Nevertheless, the authors proceed by attempting to make a goal-centred evaluation possible. To do this they define a "conceptual hierarchy of system objectives". Like the quality chain, this is a causal chain. In this hierarchy a division is made between efficiency- and effectiveness- oriented objectives. The efficiency-oriented objectives refer to the cost of system design and implementation. With regard to system effectiveness, three levels are proposed (figure 4.2): (1) the information provided by the system, (2) the use of the system and the effect on the organizational processes and performance of the users, (3) the effect of the system on organizational performance. A number of objectives and performance measures are distinguished for each of the three levels. Incidentally, Level 0 of the hierarchy is the information system itself. Level 3 is the level of relevance, but its quality is not directly measurable. Level 1 is well measurable if criteria for information quality are present, but level 2 determines whether better information does indeed have an organizational effect.

Hamilton and Chervany's levels can be recognized in the quality chain of figure 4.1. Hamilton and Chervany's level 1, the information provided by the DSS, is considered in figure 4.1 to be part of the DSS. Objectives at Level 2 (use of the system) are represented by 'problem solving'. Level 3 (organizational performance) corresponds to 'functioning of the organization'. The 'decision' from figure 4.1 is not directly

Level	Objective	Performance Measures
1. Information and support provided	Improved time of presentation	...
	Improved information content quality and quantity	...
	Improved presentation form	...
	Improved user support	...
2. Use process and user performance	Improved decision maker(s)	Understanding of problem Extent of common information Degree of cooperation and consensus Change in attitudes: toward job, toward MIS, toward MIS approach, toward confidence in decision
	Improved decision making process	Explicitness of goals/objectives Consideration of constraints, alternatives Comprehensiveness of alternatives Quantification of action consequences More informed use of MIS Length of time to make decisions
	Improved user organizational performance via:	
	- reduced information processing costs	Automate manual calculation/analysis Automate data handling / collection / correction Cost displacement (people, equipment)
	- improved asset utilization	Reduced inventory levels/turnaround Reduced number of 'backorders'
3. Organizational performance	Financial objectives	Sales revenue, Profit contribution, Return on investment
	Customer objectives	Customer satisfaction Regulatory agency compliance
	Organizational development objectives	Morale

Figure 4.2: Evaluation. From Hamilton and Chervany (1981, figure 3).

represented in figure 4.2, but as a rule, decision quality is represented in terms of the financial objectives that Hamilton and Chervany mention.

Hamilton and Chervany conclude that it is quite hard to make these performance measures operational at the important levels 2 and 3. That is where they leave the reader as far as the question "what is to be evaluated" goes. They make a valuable comment concerning who is to evaluate: a single evaluator viewpoint is often not adequate. Likewise, they ask the question: "What and whose purpose is the evaluation to serve?"

Finally Hamilton and Chervany make a comparison of existing evaluation approaches, using their conceptual hierarchy. They conclude that there are few effectiveness-oriented approaches, and of these only two address level 2 or 3: Post Installation Review (1 to 3) and Cost/Benefit Analysis (3). They also mention User Attitude Survey as an effectiveness-oriented evaluation method, but state that such a survey typically assesses only level 1 aspects.

Jagodzinski and Clarke

Another multidimensional approach to the measurement of DSS quality, in particular the interaction between user and system, can be found in Jagodzinski and Clarke (1988). They define their focus as human-computer performance, which they divide into four categories:

1. Technical performance. Here, the authors mean ergonomic properties of the system, such as response time.
2. Task efficiency. This category includes e.g. error rate and time efficiency.
3. Quality of users' conceptual model. This concept is an indicator of how well the users' cognitive characteristics, knowledge and goals are recognized in the system design.
4. Job satisfaction. This is a subjective assessment by the user.

These categories were designed with simple systems (like data entry) in mind. With some adaptations these four categories can be used to assess the quality of DSS. Category 1 corresponds to DSS quality, category 2 to decision quality, category 3 to quality of problem solving, category 4 to 'satisfaction'. Although job satisfaction is only one of the elements of the functioning of the organization (figure 4.1) or 'organizational performance' (figure 4.2), it is certainly an element that should be taken into account when assessing the quality of DSS. Perhaps the most important contribution of the article is that the authors stress the need for non-formal, multidimensional quality assessment in which the user has a voice.

Keen: Value Analysis

Aware of the fact that detailed quality measurements are impossible, Keen (in Sprague and Watson, 1989) suggests an informal, simple and explicit method to determine the desirability of a DSS. This method, Value Analysis, leads to a go / no go decision. It can and should be performed both before and during a project, being coupled to development by prototyping. What it does is compare expected benefits with the cost expected to achieve them. These benefits are usually of a qualitative nature, pertaining to organizational effectiveness in a broad sense. It is the task of the person who pays for the development to make the comparison. Keen stresses the importance of prototyping and of cooperation between developer and client; one must strike while the iron is hot.

In my opinion, Value Analysis provides a good framework for evaluation, which must be tuned to the particular cases in which it is used. For DSS development in a commercial setting the method proposed by Keen seems to be an excellent quality monitoring approach. In research projects, where costs play a different role and benefits are of a different nature, Value Analysis should be used with care.

4.3 Literature: Links in the quality chain

A certain amount of literature about the quality of individual links in the quality chain exists. This literature is not as readily applicable for DSS evaluation as the comprehensive evaluation literature discussed so far. Yet, it may yield useful insights.

Functioning of the organization

Finding adequate quality criteria for the functioning of the organization is even harder than finding them for DSS is because of the inherent complexity of the organization. It is hard enough to establish a relation between an increase in turnover or income and improved decision making. To make things even more complicated, a DSS can have other effects on an organization than just providing better decisions. It can lead to new ideas, it can make a change in time management possible, and so forth.

Since a real-life situation cannot be duplicated under controlled conditions, ambiguity about the contribution of a DSS, or a DSS development project, to the quality of the functioning of the organization cannot be avoided. By attempting to define quality one can locate and reduce this ambiguity. One should not fall into the trap of trying to eliminate ambiguity altogether, because this would imply relying on misleadingly precise quality parameters. As Pirsig would put it, one had perhaps better admit that one is stupid about quality than pretend that one is not.

Be this as it may, accepted criteria for the quality of management do exist. Some general criteria can for instance be found in Hamilton and Chervany's tool at level 3 and the third objective of level 2.

For planning systems specifically, the most visible benefits for the organization are improved *efficiency* or improved *effectiveness* of planning. In the case of improved efficiency the planner loses less time on administrative tasks and can reallocate this time. In the case of improved effectiveness the system allows the planner to make plans that satisfy more constraints or conform better to the goals. But planning is not just an activity at one point in time followed by unmodified execution of the plan. If a plan must meet strict constraints in a changing environment, as is often the case, it must be possible to modify the plan in the light of events that take place during its execution. To put it differently: *adaptive feedback* (De Jong and Sol 1991), is necessary. One could even say that the ability to adapt to outside information is a minimal requirement for real-world planning and control. The minimum planning level would consist of making no plan but just responding to each request individually in an *ad hoc* manner, which is what happens in many smaller manufacturing companies. By keeping track of execution data, a planning system can make better adaptive feedback possible. Thus, a planning system should improve rather than restrict organizational *flexibility*. If plan revisions are desired, they should be very easy to carry out at any moment during the realization of a plan. To the organization, the benefits of such flexibility are e.g. improved contact with suppliers and better service to clients or personnel.

Quality of working life
Quite a rich perspective on quality is brought in by those who see information systems in a wider context, including aspects of the quality of professional life. This view is predominant in the Scandinavian countries. In the words of Bjørn-Andersen (1988, p. 386):

> At the moment all we are doing is to adapt the technology to the known so-called 'human weaknesses' in order to reduce the resistance to using the technology, rather than providing a technology which will help to liberate the intellectual capabilities of human beings.

Bjørn-Andersen stresses that the level of quality information systems should aim at is, ultimately, the quality of life, rather than some financial measure. Based on his experience from cooperation with users and trade unions over a long period, he expects that in the future, users will demand systems that (1) contain no monitoring, (2) assume knowledge in the user, (3) permit high discretion and competence, (4) are modifiable, (5) are transparent, (6) support learning, (7) support feelings and intuition, (8) support social contact.

Such considerations fall within the 'functioning of the organization' link of the quality chain.

Decision

Basically there are two different approaches to measuring the quality of a decision. One can either assess the expected value of the decision as soon as it is made, on the basis of the decision variables, or one can carry out the decision and make an *a posteriori* evaluation. These two approaches do not measure the same things. In the former case, which I shall call immediate quality assessment, the assumptions on the basis of which the decision was made are taken for granted. In the latter case, the evaluation mainly serves to check whether the assumptions made during decision making were justified. Decision Science has developed methods for immediate quality assessment (see e.g. Von Winterfeldt and Edwards, 1986). Management studies have included *a posteriori* decision evaluation in theories about management. Such evaluations are problematic because of the ambiguity about the causal relation between the decision and the functioning of the organization. One never knows what to attribute to chance and what to the quality of the decision.

A quick alternative to *a posteriori* evaluation, allowing to take the effect of chance events into account, is to check the decision not against reality but against a simulation model. This requires a valid simulation model, and unfortunately it is quite hard to prove model validity. If, however, a valid model can be made, it will have the advantage of permitting repeated runs, thus allowing to obtain an estimate of robustness of the decision before it is actually effectuated (see e.g. Verbraeck 1990[a], 1991).

Alleblas' list of decision quality characteristics presented on p. 71 contains both immediate and *a posteriori* aspects, as well as aspects that I treat as parts of problem solving or of the functioning of the organization.

Problem solving and Decision making

There is general agreement among DSS researchers that the D in DSS has not received proper attention. In 1986, Keen mentioned as a priority for the next decade "focusing more specifically on the Decision component" (Keen 1986, pp. 234-235). As put forward in the introduction, decision making and problem solving are two sides of the same coin. Decision making is a result-oriented term, whereas problem solving is process-oriented, and the terms represent different bodies of research. Both are relevant to this literature review. Because of its management- and result-orientation, decision making research is more readily connected with the concept of quality than problem solving research is.

Decision making

The decision making literature includes Management Science, twin sister of Operations Research, as well as Decision Science, which field includes e.g. Multi-Attribute Utility Theory, MAUT (e.g. Lindley, 1985), and the related school of Decision Analysis (e.g. Von Winterfeldt and Edwards, 1986). Psychological literature which is oriented towards decision making is often termed Decision Theory. It uses concepts such as preferences, utility perception, cognitive biases (e.g. Kahneman, Slovic and Tversky 1982, Hogarth 1987, Hofstede 1988[b], Vlek 1989). Psychological literature which is oriented towards problem solving will be mentioned in the next subsection.

Much of the literature on decision making refers to well-defined decision situations. This often leads to the conception of formal models as representations of the decision situation. In OR and Decision Science literature, quality of decisions is measured in terms of a conceptual model in which there exists some goal criterion. Decision quality can be established by computing and rating the value of that criterion. This principle applies to criteria such as expected profit, expected utility and score relative to an optimum value.

There obviously exists a discrepancy between this quality concept and the organizational value of a decision. This has been signalled by many authors, for instance Ackoff (1987) and McArthur (1980). Both articles contain severe criticism of existing research practices. Ackoff directs his criticism towards OR practitioners, McArthur towards Decision Scientists. McArthur (p. 110):

> There's a gap between decision scientists and decision makers. The decision scientists don't understand decision making; and the decision makers don't understand decision scientists.

As this quotation implies, decision scientists are not so much concerned with decision making as a process as with immediate quality assessment of a decision as a product.

Behavioural decision theory is more psychologically oriented than decision science and decision analysis. A good overview of behavioural decision theory is presented in Vlek (1989). A quotation from the summary gives his conceptual framework (p. 207):

> The crucial question of "what constitutes a good decision?" is considered against four kinds of rationality: representational, goal-value, methodical and '(de)compositional' rationality, respectively. In practice, decision quality may be determined by looking at: (a) rationality of procedure, (b) robustness of final choice, (c) user satisfaction with the method followed. Often, increased cognitive control over the decision problem is a more important effect of decision support than the resulting 'best choice' advice.

The last sentence in this quotation clearly states the added value of behavioural decision theory for DSS research and practice. Behind this sentence lies a body of research on heuristics and biases that occur in unaided human decision making, due to cognitive limitations (Hogarth 1987, Kahneman, Slovic and Tversky 1982, Von Winterfeldt and Edwards 1986) and emotional factors (Janis and Mann 1977) .

Problem solving

Problem solving processes are hard to study in a scientific way. Todd and Benbasat (1987) suggest that this may be the main reason for the lack of progress in DSS research since Scott Morton (1971) discussed the impact of DSS on thought processes. In a comparative study Todd and Benbasat suggest verbal protocol analysis as a research approach for studying problem solving processes. They make a strong case in favour of protocol analysis as a means to assess how a DSS influences problem solving processes. They also suggest protocol analysis, applied less formally, as a means for prototype evaluation during development. The main obstacle for applying protocol analysis is the enormous amount of work which it generates.

Besides methodological problems there may be more fundamental difficulties in studying problem solving. This issue is addressed by Smith (1988) who stresses that there is no such thing as a general, problem-independent, problem solving system in humans. He draws on Simon (e.g. 1981). He concludes (pp. 1491-1492):

> Theories of problem solving have been supplanted by broad accounts of human cognition, e.g. information processing theory, and by narrower theories concerned with specific cognitive capacities, e.g. perception, memory, and inference. Theorists have learned that problem solving is a domain-specific activity, strongly characterized by the nature of the task being addressed (...).

Case studies in DSS development have provided experience in support of Smith's statements. For instance Stabell (in Bennett, 1983, p. 231) remarks:

> It is apparently quite difficult to say anything very useful or insightful about how decisions are made in a specific decision situation that is divorced from the what of decisions.

These two quotations go to show that there is only a limited set of general problem solving theories which can possibly be applied to DSS research, and that case studies will have to play an important role.

In view of the difficulties with descriptive research into problem solving and in view of the flexibility of humans in problem solving, an alternative often used by designers of DSS is to devise prescriptive strategies for problem solving. The rationale is that since humans adapt to situational factors so readily, presenting them with a normatively good problem solving procedure is a guarantee for good problem solving performance. A danger is that the procedure may require too strong an abstraction of the problem from its organizational context.

Quite a number of problem solving strategies or procedures have been proposed. Virtually all of these can be seen as variations on Simon's intelligence / design / choice model. An example is the procedure given by Weber and Konsynski (1987). These authors propose five steps: (1) problem finding, (2) problem representation, (3) information surveillance, (4) solution generation and (5) solution evaluation. They discuss the five steps and indicate which type of support a DSS can provide for each step. This results in general guidelines which can be adapted for use in particular DSS. It can be noted that in most DSS, the first two steps are fixedly designed into the DSS. In fact these steps constitute the requirements analysis and functional design of the DSS. This 'closed' nature of the models used reduces the flexibility of a DSS.

The brief overview of decision making and problem solving given here confirms that they are similar concepts, with a difference in emphasis. I agree with those authors who, like Smith (1988), advocate the integration of the research schools concerned, and therefore consider the two concepts as one. Problem solving is then seen as a process which ends with the actual making of a decision. I suggest that an informal, general conceptual model of a decision problem can consist of:

- process model: problem solving strategy, for instance the five steps of Weber and Konsynski mentioned above. Often, in the case of planning, the current planning procedure can be taken as a basis for a process model.
- design submodels: state spaces, variables, used for problem representation and solution generation.
- choice submodels: evaluation functions, sensitivity analyses, used for solution evaluation.

In open problem situations, the problem solving process is most important, and the role of design and choice submodels is limited; they may be absent altogether or they may be applied to subproblems. If a problem can be considered as a formal problem, design and choice submodels acquire more importance, since they allow the use of computer computational power.

In all situations it is an important consideration that human problem solving readily adapts to situational factors. It can therefore be expected that problem solving will change if a DSS becomes available which offers new ways for problem solving, provided that the problem solver perceives them as useful.

Obviously, *Human-computer interaction* is an aspect of problem solving which is of special importance when one uses a DSS to solve a problem. The interaction should be such that the user's train of thought is not interrupted by the system. For instance, the session should be *interruptible*. If the user is interrupted by, say, a phone call, it should be easy to interrupt the session as well as to pick up the thread afterwards. In psychological terms this means that the system should impose as small a load as is possible on the user's short-time memory. This can be achieved by features such as a 'pause' key, displayable parameters, self-explaining screens. Other interaction aspects are mentioned in the next paragraphs.

Learning

It is an important aspect of problem solving with DSS that the user *learns* about the problem situation by using the system. Vlek (1989) uses the phrase 'increased cognitive control of the problem'. Unfortunately, as was asserted by Ginzberg (1978[a]) some time ago but still holds true, literature on learning consists mainly of controlled experiments in which subjects had to perform relatively simple tasks like text processing and data entry, using different interfaces. This type of research has much been in vogue among cognitive psychologists.[45] It has resulted in guidelines for interface design to facilitate learning (e.g. Shneiderman 1987). Such guidelines include the number of items on a screen, the use of colours and of visual interactive presentation, and so forth, i.e. they are general rather than problem-specific guidelines of an ergonomic nature.

A composite and pragmatic quality criterion for an interactive system is whether it can be mastered by the user without help (Robert 1987). An important aspect of this is that errors must not be fatal (Rasmussen 1987). Experimental evidence on learning how to use text processors shows that users tend not to use help facilities or manuals if they can help it (Mack, Lewis and Carroll 1983). It seems that although interface quality is very important for initial learning, a user with experience in handling a task can cope with almost any interface if he must (e.g. Jørgensen 1987).

In the bulk of the literature about learning in the context of information systems, learning refers to the mastering of some *tool* to perform a rather simple task more efficiently. Another definition is learning how to perform a certain *task*. This latter definition is relevant in the context of DSS where the tasks are complex and may change through using the DSS. In this view, learning how to use the DSS is only a means to

---

[45]A typical example is Streitz et al. (1987), but there are many hundreds of similar experiments reported on (Bullinger and Shackel 1987).

learn to solve the problem better, i.e. more effectively. Indeed, better problem solving obtained through using a DSS may after some time make the DSS superfluous, although in the case of a system for tactical planning this is unlikely to occur.

The literature indicates that systems that are not easily mastered are not easily adopted either. Yet, sufficiently motivated people manage to cope even with very poorly learnable systems. Incidentally, this phenomenon occurs widely both inside and outside the domain of information systems. In deciding whether to take the trouble of learning a certain skill, a person is rather benefit-oriented than cost-oriented.

DSS

Input data

A DSS cannot do without input data. The user's need for information is dependent upon the type of situation for which the DSS is intended. If a DSS is meant to support problem definition, it is hardly possible to make *a priori* statements about the quality of information sources. For tactical planning problems, information quality can be assessed if one follows established standards. Criteria for information quality include completeness, correctness and timeliness (Bots et al. 1990, Van Hulzen and De Moel 1987). It is sometimes possible to assess the financial value of additional information (Davis and Olson 1985, Bemelmans 1987). Intrinsic unpredictability of data sets limits to the applicability of DSS (De Leeuw 1982), in particular to the use of predictive models.

Recent empirical evidence collected by Wassink and Van Rossum (1990) among government top managers suggests that the information needs of decision makers cannot be validly derived from interviews. In their case studies, functional decomposition performed by the information systems designers turned out to be a better basis.

It can be concluded that input data of adequate quality are a necessary but not sufficient condition for DSS. The availability and properties of data must be carefully considered before embarking on a DSS project.

Software quality

Boehm (1978) has devoted a book to the assessment of software quality in general. He has defined a hierarchical 'quality tree'. The adapted version by Bots et al. (1990), used to describe the quality of information systems, is shown in figure 4.3.[46]

The three main branches of Bots' quality tree represent different aspects of quality. Usability refers to the 'as-is' aspect, i.e. the quality of the information system in a stable environment. Maintainability refers to the 'as-should' aspect, i.e. the quality in a dynamic environment. Portability refers to the 'as-must' aspect, i.e. the ability to cope with a totally new situation. These three aspects of quality are often conflicting. One has to make choices as to which aspects prevail. *The* quality of an information system does not

---

[46] A number of similar quality trees exist in the literature (e.g. Bemelmans 1987, van Hulzen and de Moel 1987).

exist.

Besides the properties of the software mentioned above there are some down-to-earth quality attributes of an information system. For DSS, the following two aspects of usability are important.

To begin with, the DSS should be *available* when needed. If the system is to be used at various places this may mean that data must be easily transmittable between computers. If the system is to be used during meetings there are other requirements, and so on.

Also, the DSS should generate usable *outputs*, often other than on-screen. Designers tend to overestimate the importance of the on-line interface. Written reports can play a very important role, because no matter how well-designed a screen interface is, there are limitations in transportability and flexibility. The ease with which one can walk around an office with a paper report, or scribble notes on it, is hard to match.

For the quality of the DSS's output data, the same criteria hold as for input data, i.e. completeness, correctness and timeliness, plus ergonomic aspects, which are the subject of the next subsection.

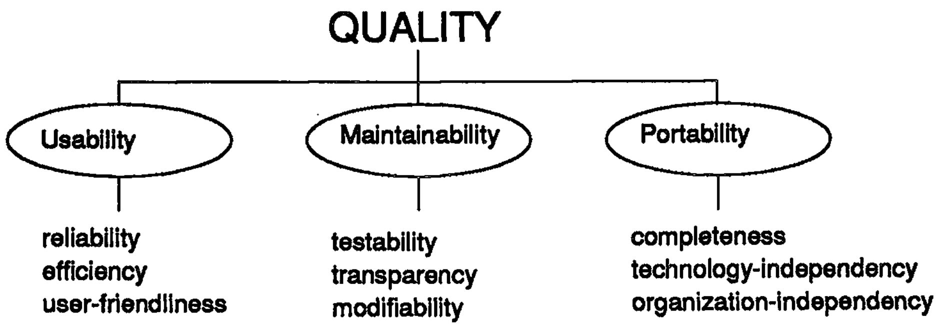


Figure 4.3: Quality of an information system, after Bots et al. (1990, p. 556).

Models

If the DSS contains models, it may be possible to assess their quality if there is a quality standard for the category of models to which those in the DSS belong. OR handbooks contain quality criteria for models; usually these are of a strongly mathematical flavour. Ören, Elzas and Cheng (1985) offer an elaborate framework for evaluating the quality of simulation models. In simulation, validating models against the real-world problem situation is usually the hardest issue.

As a rule, model quality criteria do not take into account the quality of the problem solving processes that the combined system of model and user performs; they aim at the quality of the model *per se*. Criteria that have to do with the problem solving activities a user performs with the help of a model are included in the problem solving link of the quality chain and they are therefore not mentioned here.

Human-computer interface

Human-computer interfaces, in particular screen monitor / keyboard interfaces, have been the object of a research boom during the eighties. Most of the research has been concerned with the ergonomics of relatively simple tasks. Research questions have related to either the human ("do people with different cognitive styles require different interfaces?") or to the interface ("which interface is best suited to a certain task?"). It would seem that individual properties of the user, such as cognitive style, are not a suitable basis for DSS design (Huber 1983), but that task properties determine what representation and what interface are most suitable (Gerstendörfer and Rohr 1987). Concerning the most suitable data representation for some specific task, however, there exists much contradictory evidence (Dickson, Senn and Chervany 1977, Davis and Olson 1985, Bullinger and Shackel 1987, Hofstede 1990[d]).

Despite this unfortunate predicament, some recommendations can be given. Usually a DSS is designed for an existing task and a user familiar with that task. The user has experience-based knowledge of the task. It would be counterproductive to force the user to work with a different interface than he or she is used to. So unless there are very good reasons to change the interface, the DSS should have the kind of interface the user is used to.

Tyszka (1986) provides an implicit argument to let a user work with a familiar interface, that is, an interface which allows the user to deal with the problem in the same way he used to do without the DSS. Tyszka has argued and empirically confirmed that people who are familiar with a decision are better able to use a satisficing approach than beginners. Beginners tend to rely on normative models, if available. Experienced decision makers are able to perform the more subtle and context-rich evaluation required to compare a proposed solution to some non-quantitative standard. They perceive more aspects of the situation. If a DSS user is confronted with an unfamiliar interface he can be expected to act like a beginner as long as he does not recognize his own conceptualization of the problem in the DSS.

Satisfaction

The concept of measuring user satisfaction as a *post hoc* evaluation method for information systems has received considerable attention in the literature. This is justifiable, since it is usually true (Ives, Olson and Baroudi 1983) that

> A 'good' information system perceived by its users as a 'poor' system is a poor system.

These authors give a review of methods for establishing user satisfaction and choose one of them for adaptation to their own purposes. The review confirms Hamilton and Chervany: level 1 is addressed almost exclusively. The instrument chosen and adapted by Ives et al. was developed by Pearson (Pearson 1977, Bailey and Pearson 1983). This instrument is a questionnaire consisting of 39 questions. Ives et al. administered Pearson's instrument to 280 production managers in manufacturing organizations in the USA. They performed a factor analysis on the results which divided the questions among four factors: (1) EDP staff and services, (2) information product, (3) vendor support, (4) knowledge or involvement. This last factor addresses level 2 from Hamilton and Chervany. It contains three questions, which deal with training provided to users, with users' understanding of systems, and with users' participation.

The information systems addressed by Ives et al. and by Pearson are large information systems with several users, not single-user DSS. When one measures satisfaction about a DSS, other things will have to be taken into account alongside the above-mentioned. One could, for instance, consider spreadsheet packages as single-user DSS.[47] I believe that the great success of spreadsheet packages for decision support lies in their malleability. They enable the user to carry out his very own problem finding and solving process. DSS designers can learn from this that flexibility and possibilities for control of what the system does are essential for user satisfaction.

Whether a DSS which satisfies the user is also satisfying from a normative point of view is not at all certain. The importance of user satisfaction depends on the goals of a DSS project. If user satisfaction is defined as a dominant aspect of quality, then a satisfied user coincides with a high-quality DSS. This makes sense in most cases, and it certainly does in the case of a DSS which is intended for voluntary use by a user who is competent about the supported decision situation.

4.4 Literature: DSS development process

As opposed to the quality of a DSS product, one can also consider the quality of the development process according to which this product is made. It could be argued that, since the quality of the four left links in the quality chain is not easily or not adequately measurable, it is more profitable to focus on the quality of the development process. I shall argue that there are even better reasons for paying much attention to the development process. For this purpose I shall discuss three types of problems that can occur with the design of DSS. Some recommendations to avoid these problems will then be made.

The first problem is that systems lack flexibility: they force the user to work in a way that might be unacceptable to him or her. Design involves a trade-off between offering guidance to the planner and reducing the flexibility of his way of planning. It has been shown in chapter two that this trade-off often emphasizes model-based support too much at the detriment of usability.

The second problem is that systems are sometimes incomprehensible to users. If for any of a number of reasons the users fail to understand a system, they will tend to mistrust the system's output and they will tend not to use it. By the way, this should not be taken too absolutely; a user does not always need to understand the DSS's functioning

[47]The term 'DSS generator' is sometimes used for packages with which a user can develop a specific DSS (see e.g. Sprague and Watson 1989).

at all levels of detail. If a user is prepared to trust a designer or system at the outset, for instance after previous good experiences, the user may accept that he does not understand the intricacies of a model (see e.g. Vellekoop 1989, p. 164).

The third problem is that in the design of a DSS, it is not at all certain that anyone knows in advance what the system will have to look like. More likely than not, the user has no clear idea of what an automated tool could do for him or her and the designers have a poor insight into the decision situation, especially the organizational contingencies. The result of this mutual lack of knowledge may well be a system that falls short of the possibilities which the situation offers.

I shall term these three problems *unacceptable design*, *incomprehensible design*, and *unknown specifications* respectively.

Unacceptable design

Many factors other than the properties of a DSS are important in determining its acceptance. For instance, a user who feels his job security is threatened or his competence challenged by a planning system may *a priori* refuse to work with the system regardless of the design. This phenomenon is known as 'counterimplementation'. In a situation of cooperative design one would not expect counterimplementation to occur because if a user is unwilling to accept a system no project will be started in the first place.

Even if the user is not *a priori* against a system, it may be unacceptable to him. This is especially to be expected if the user is a competent professional who 'has his own way of going about it' and the system prevents him from working in this way. If, for instance, a planning system is strongly model-based, it usually works on the basis of generalities. An experienced planner has internalized these generalities and focuses instead on exceptions. Each planning action by the planner involves a quick informal sensitivity analysis, based on previous occurrences that come to his mind in the current context.

A second cause of unacceptability is the treatment of constraints. An automated system deals with constraints in an all-or-nothing manner. The user, on the other hand, implicitly assigns relative importance to constraints. This importance can easily be altered.

Both the focus on generalities versus exceptions and the treatment of constraints are aspects in which models and human planners differ fundamentally. AI models tend to deal with exceptions more readily than OR models but still the treatment of constraints in AI techniques is rigid.

Incomprehensible design

If a design idea, however appropriate for a certain planning situation from a developer's point of view, is not understood by the user, the idea is of no use.[48] Näslund (1992) uses the suggestive term *shadowed design idea* for such an uncomprehended feature of an information system. In his own words (p. 1):

---

[48]An alternative might be to replace the user, but this is often neither feasible nor desirable, particularly if the user possesses unique domain knowledge.

> Each DSS is designed with at least one *support idea*, which reflects the designer's intended role for the DSS. In practical use, this idea is however seldom fully utilized. The reasons for this may vary, but involve often a misfit between the user, the system, the task and the organization. Such misfits can be the user's misunderstanding of parts of the system, or of the role the system is intended to play. Other common variants are the user's lack of confidence in the system, and insufficient training. This makes the support idea partly unusable. We call this a *shadowed support idea*, where the misfit is the shadow.

Näslund carried out an investigation of a large number of development projects and tried to find out whether early and continuous evaluation, in which the user participates, can reduce shadowing. His answer to this question is in the affirmative.

Unknown specifications

The problem that functional specifications of the system are not known at the outset of a project has been fairly generally accepted in the DSS research world for a number of years now. It is well worded by Alavi and Napier (1989, p.82):[49]

> The way of designing a DSS is different from that of a transaction processing system. A fundamental assumption in the traditional 'life cycle' approach is that the requirements can be determined prior to the start of the design and development process. However, Sprague stated that DSS designers literally 'cannot get to first base' because the decision maker or user cannot define the functional requirements of the DSS in advance. Also, as an inherent part of the DSS design and development process, the user and designer will 'learn' about the decision task and environment, thereby identifying new and unanticipated functional requirements.

In this quotation the last sentence is especially significant. There has to be some learning shared by designer and user to discover new functional requirements. The authors stress this open-ended aspect by talking of the situation for which the DSS is built as 'the problem or opportunity' rather than just 'the problem'.

Perspectives on design

The literature on designing DSS can be roughly divided into user-centred and model-centred. Some writings on user-centred design are discussed below. The user-centred perspective is compared with the problem solving perspective discussed in chapter one (p. 8-10) which is currently popular with academic DSS designers in the Netherlands and which is rather model-centred.

[49] The paper was originally published in 1984.

Alavi and Napier: adaptive design

Alavi and Napier (1989) suggest a strongly iterative prototyping approach which they call adaptive design. This design process should involve mutual interaction between user, builder and technical (hardware/software) system. It is important that the builder become familiar with the task for which the DSS is being designed. This requires intensive communication between user and builder. Close cooperation between the two will be beneficial to the quality of the design and will also build credibility with the user. The user for his part must work with the available DSS version to enhance his understanding and perception of the decision task and potential solutions.

Gould: early focus on user

Gould (Gould 1987, Gould et al. 1991), a successful designer of information systems, mentions a number of publications agreeing with his own view that the first design principle should be *early and continual focus on the users*. The other three principles that Gould puts forward are of the same nature. Principle two is *integrated design*, meaning that all aspects of usability need to evolve in parallel and under one focus. Principle three is *early and continual user testing*. Principle four is *iterative design*, which means being able and willing to spot deficiencies and make changes.

Vellekoop, an operations researcher engaged in a DSS project, relates that the sceptical attitude of the users changed as a result of working with the first prototype and frequent discussions with the author. He adds that close cooperation with the users was also crucial for his own understanding of what was wanted and what was not (Vellekoop 1989, pp. 164-165).

The recommendation to cooperate closely with the users is by no means simple to carry out, as any developer who has tried probably knows. There is often a considerable gap in perspective, language, and culture to overcome, not to mention suspicion or fear on the part of the user. If both parties are willing to make an effort, cognitive barriers can fairly easily be overcome. It is not so easy to take away distrust and fear. Moreover, these feelings are often justified in situations where the user is not the initiator of the development, and has no control over the development project (see e.g. Willmott et al. 1990).

Angehrn: User-centred development

The opposition between user-centred and model-centred decision support systems development is clearly stated by Angehrn and Lüthi (1989, see also Angehrn and Lüthi 1990 and Angehrn 1991). The following is adapted from their figure 2.

Traditional, method-centred development follows the sequence:

1. formal analysis of a decision *problem*
2. development of suitable 'solving techniques'
3. addition of so-called 'user-friendly' interfaces.

User-centred development as proposed by Angehrn and Lüthi proceeds thus:

1. analysis of the characteristics of a decision *context*
2. development of an interactive modelling environment

3. development and integration of supporting analytical models.

Angehrn and Lüthi mention as their first design principle: 'usability prior to functionality'. They justify this principle by stressing that if the functionality of a system is not transparent to the user, this functionality will not be effective because the user will either reject the system or use it incorrectly. Their second design principle is 'active cooperation'. They explain this second principle as follows (p. 3):

> While the human divides the complex planning job into partial tasks and assigns some problems to the machine to resolve, the system takes over the role of an 'advisor' or 'facilitator'.

Conclusion

In conclusion it can be asserted that the problems of unacceptable or incomprehensible design can be avoided, and that the problem of unknown specifications can be overcome. The remedy is straightforward: it is to focus on the intended users right from the beginning of a project, to work with the users, and to make and evaluate prototypes as soon as possible. Recommendations of this type have been made by practitioners and researchers alike from various fields. It can be noted that this advice implies viewing the decision situation as an open problem, not as a formal one. Incidentally, whether an acceptable and comprehensible system is also a good system is a different question.

Problem solving perspective on design

As stated in chapter one, there exists a perspective on the development of information systems inspired by hard systems thinking which centres on model-based problem solving and which is much documented. For example, Ackoff (1978), Mitroff et al. (1974), Bosman (1977, 1986) and Sol (1982) are authors with related views on problem solving that fit into this perspective. The common ground in their approaches is that one conceptualizes a given management problem situation with a view to designing something to solve it. The stress is on the designing and on the solving; the nature of the 'something' is secondary. In other words this is an inductive empirical, or engineering, approach. It does not adopt the *homo economicus* perspective on rational decision making; a description of current decision making practice is an important element, and normative decision quality parameters are not presupposed. Just like ideas on user-centred development, the application of the problem solving perspective to DSS design was first intended to reduce misfits between DSS and user.

Ackoff (1978) stresses creativity as the most important factor in problem solving. The usual reason why people fail to solve a problem, he says, is that they impose unnecessary constraints upon the solution. In fact Ackoff's views imply that a crucial aspect of problem solving is to question, and if appropriate to alter, the problem statement. Apart from solving a problem, one can also resolve or dissolve it (pp. 39-40). Resolving a problem means accepting the problem statement and adopting a compromise solution, which may not exactly solve the problem but is at least acceptable. Dissolving a problem means altering the problem, seeing it in a new light so that it no longer exists.

So, Ackoff puts particular emphasis on the importance of the first step in the succession conceptualize - design - solve. The other three authors, Mitroff, Bosman and Sol, pay relatively more attention to the other two steps. I shall discuss their design views in chronological order.

Mitroff's Systems View

A paper by Mitroff, Betz, Pondy and Sagasti (1974) entitled 'On Managing Science in the Systems Age' lies at the origin of the problem solving perspective. Clearly this perspective was inspired by systems thinking. Central to the article is the 'Systems View of Problem Solving' shown in figure 4.4.

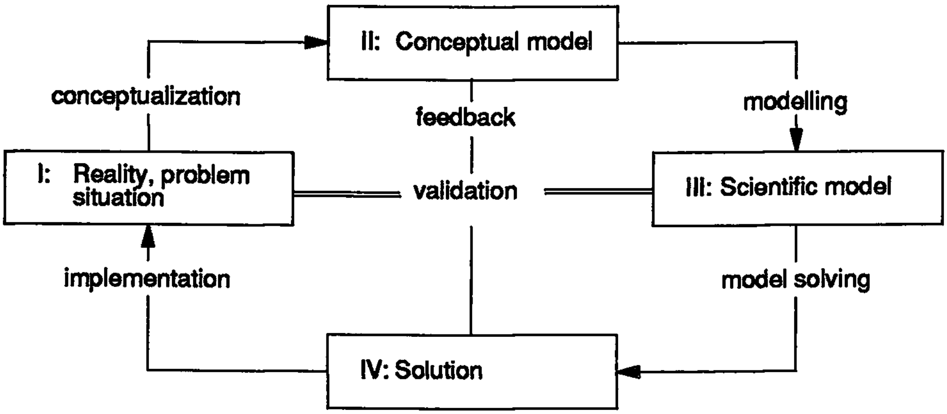


Figure 4.4: A Systems View of Problem Solving, from Mitroff et al. (1974).

Solving organizational problems is viewed as a network of activities in which a variety of paths can be followed. A clockwise cycle: Conceptualization, Modelling, Model solving, Implementation, is most usual. This resembles an OR approach to problem solving in that a problem, as a conceptual entity abstracted from an organization, is central to it, and that it is considered as a formal problem. The activities involved in the four phases are of a very different nature and each phase requires something else from the problem solver. Figure 4.4 is a useful basis for methodological discussion in information systems research and has been much used. Sol (1982), for example, has adapted it and integrated it with simulation.

Bosman (1977) integrates organizational and systems perspective with classical econometric methods and lays some of the foundations for the problem solving perspective, especially the approach further developed by Sol (1982) and currently followed in many publications from the department of Information Systems of the University of Delft. Bosman discusses Cyert and March and draws from them in his own 'metatheory of organizational behaviour'. Bosman addresses the issue of problem decomposition for computer-assisted problem solving. As early as 1977 he distinguishes a sequential and a simultaneous approach to problem solving, saying that a sequential approach is necessary if a problem is too complex to be solved in one step (Bosman 1977, p. 190). In a 1986 article he much further articulates these ideas.

Bosman: Procedural rationality and design

Bosman (1986) adopts the paradigm of *procedural rationality* as given by Simon: behaviour is procedurally rational if it is the outcome of appropriate deliberation. Simon puts forward procedural rationality as the 'bounded' alternative to *substantive rationality*, which coincides with economic rationality of an omniscient decision maker. Procedural rationality does not presuppose perfect knowledge or perfect reasoning. Bosman's argument is that in the design of information systems, if one adopts this paradigm of procedural rationality of designers and users, a descriptive model is needed of the current situation for which the information system is intended. Such a model will allow assessment of the rationality of a design.

This argument has resulted in a family of closely related design frameworks which are currently being advocated and used by researchers influenced by Bosman (e.g. Sol 1982, Takkenberg 1983) or by Sol (e.g. Bots 1989, Van der Ven 1989, Verbraeck 1991).

Figure 4.5 pictures the inductive - deductive research approach currently used to describe a lot of information systems research, including my own case study.[50] The research is based on a combination of empirical research - case studies - and theory. Unlike in figure 4.4, a *counter-clockwise* cycle is followed. Usually the research starts at the upper left quadrant by performing one or more studies of existing situations in which there are elements deemed to be open to improvement. This results in the first model: a descriptive empirical model of the case. The second phase is to abstract essential elements from one or more cases, giving a descriptive conceptual model (bottom left) of the type of situation of which the case studies are instances. Now design starts: the conceptual model is enriched with normative elements expected to improve the current situation. The result is a prescriptive conceptual model (bottom right). Then this model is instantiated in one or more actual information systems, giving prescriptive empirical models for the cases. This cycle can be repeated several times.

Taking into account the inverted rotation, the four boxes in figure 4.5 are analogous to those in figure 4.4. There is a less pronounced emphasis on abstraction but still both frameworks rely on conceptualization early in the design process.

Cooperation or problem solving?

One could ask whether the user-centred perspective and the problem solving perspective on information system design are compatible. In the user-centred perspective there is no reference to conceptualization or modelling. A user-centred developer would probably

[50]In fact figure 4.5 is identical to figure 1.3 but for the absence of references to the application domain.

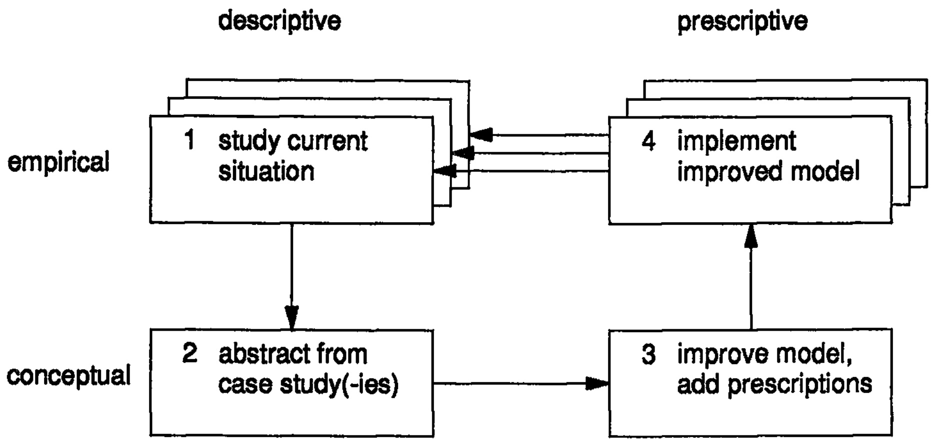


Figure 4.5: Design approach for information systems research.
Adapted from Van der Ven (1989), p. 5.

say "if you want to conceptualize, go ahead but only if the user is highly involved". In the problem solving perspective of figure 4.5 no reference is made to the user. The first phase, 'study current situation', leaves open the extent to which the users are involved. The second and third phases may transcend individual case studies, making it difficult to attain user involvement. In the fourth phase, 'implementation', users might be involved, but the cycle does not seem to agree with an open-ended type of system development: the functional specifications are to a large extent determined in phase three.

A developer who adopts the problem solving perspective would perhaps say "it depends on the case to what extent users are involved in the cycle". For example, if a number of cases showed much similarity in phase 1 then the designer could choose to work with only one or two users during phases 2 and 3, or even with none at all.

So, although the two perspectives need not exactly conflict, the priorities are clearly different. From the user-centred perspective, design is a *regulative cycle* consisting of the phases diagnosis - design - change - evaluation. The time aspect is explicit. In Vickers' words one could also speak of an appreciative cycle.

From the problem solving perspective, design is an *empirical cycle*, as figure 4.4 shows. The dimension of organizational change is missing in the problem solving perspective. This is a serious omission in the light of the conclusion drawn in chapter three that change is very important in the development of DSS. Any attempt to make changes induces reactions, either resistance to the change or further unanticipated changes. If only because of its attention to change, the user-centred perspective should be the most important in a DSS development project. Futhermore, experiences from case studies indicate the importance of cooperative development (Vellekoop 1989 p. 164, Verbraeck and Sol 1988 p. 11, Verbeek 1991 p. 177 ff.).

If the developer is predominantly model-oriented, and chooses a conceptualization that is incompatible with the user's planning approach, then the result might be an addition to the list of failures indicated by Takkenberg (1983, p. 19) in an enumeration of causes for disappointing OR impact:

> (...) het impliciet hanteren van het paradigma van het algoritmisch rationeel handelen, waardoor modellen en algoritmen worden gebruikt die weinig of geen aansluiting geven bij de wijze waarop beslissers gewend zijn hun problemen op te lossen. Hierdoor ontstaan moeilijkheden m.b.t. de acceptatie van nieuwe beslissingsprocedures. Er zijn diverse projecten bekend waar, na het verdwijnen van de OR-expert, het resultaat van vele jaren intellectuele arbeid geruisloos werd gearchiveerd.[51]

If, on the other hand, the developer focuses entirely on the user, he or she might miss opportunities for not only supporting but also improving a planning process.

To avoid either pitfall, the developer should start by working with and for the user for some time, focusing on building a manipulative interface, and consider only afterwards whether a model-based design cycle is appropriate. This is the succession of steps proposed by Angehrn and Lüthi. In this manner, models that are developed are guaranteed to work under an interface understood and accepted by the user.

In other words, a DSS development project is first and foremost a succession of regulative cycles. The design phases of these cycles may either or not include empirical cycles in the sense of figures 4.4 or 4.5.

Adherence to method

Prescriptions for developing information systems proliferate in the literature. Besides a great body of literature on information systems development in general (e.g. Olle, Sol and Verrijn Stuart 1986, Davis and Olson 1985, Bemelmans 1987, Bots et al. 1990), specific methods for DSS development have been proposed (e.g. Boersma 1989). Most methods contain quality monitoring activities. These usually deal with Hamilton and Chervany's level 1, i.e. a restricted view on the DSS link of the quality chain.

However, the emphasis on developing appropriate methods - often confusingly termed methodologies - for system development is countered by literature which dismisses methods as irrelevant or even harmful (e.g. Van Rees 1982). Basically, the critics say that methods are for stupid people; they hamper flexibility and creativity; an open-eyed and -eared cooperation between system developer and user is far more important than adherence to methods.

My view on this controversy is that the choice (Sol 1987 and 1989) and importance

[51](...) the implicit use of the paradigm of algorithmically rational behaviour, leading to the use of model and algorithms that show hardly any correspondence, or no correspondence at all, with the way in which decision makers are used to solve their problems. This generates difficulties with the acceptance of new decision procedures. Several projects are known where, after the disappearance of the OR-expert, the results of many years of intellectual labour were silently tucked away in archives.

(Kierulf et al. 1990) of methods strongly depend on the situation. Sol advocates a choice of paradigm, development process and project control dependent on problem type. For 'poorly structured decision support' he proposes object-oriented modelling, hypothetico-inductive, goal-directed, incremental development, and participatory project structure. This is in accordance with other authors (e.g. Keen 1986) and allows for user-centred system development.

Kierulf et al. (1990) reject methods altogether. Aiming at software development in general they write (p. 152):

> The vast literature on software engineering discusses, almost exclusively, the production and maintenance of software as an industrial process. (...) The source of all this knowledge is experience based on previous production of similar products.

Kierulf et al. then argue that innovative software projects do not meet the conditions required for *a priori* structuring because there are no 'similar products' on the experience of which to fall back. As examples of innovative software projects they take the projects that lead to the advent of spreadsheets and hypertext, but one can also think of any DSS project. Kierulf et al. give conditions under which innovative software development occurs. Among these is a strong relation between project and individual developers. They conclude (pp. 152-153):

> Even a casual acquaintance with several exploratory software projects suggests that they proceed according to rules different than those postulated in the software engineering literature. It is not just a matter of developing a prototype or two before starting on the product, and it is definitively not a matter of going through half a dozen phases of a life-cycle that includes requirements analysis, specification, coding, and testing.

After this they give some properties of exploratory software projects. These include a bottom-up approach centred around a single evolving system. The current version of this system is the focus for creativity and cooperation.

Kierulf et al. carry their point a little far. Methods obviously have great importance as a means for documentation and communication, for learning, and as checklists. On the other hand, the quality of an information system cannot be guaranteed just because the developers have adhered strictly to a particular method. A developer should never rely on a method to replace independent thinking. I agree with the opinion that open-minded and intensive contact between developer and user is more important than adherence to methods. This holds especially in the context of innovative systems such as DSS. As Verbeek (1991) suggests, a good method for developing DSS should include a fair amount of adequate communication between users and developers.

Project goal

Each DSS is developed with some formal goal as a justification for allocating time and money to the project. Keen (1975, 1986) has repeatedly stressed that quality of a DSS can only be assessed by comparing the system with predetermined goals. Therefore, even if the goals are mainly of a 'soft', qualitative nature, they should be explicitly stated before development and monitored throughout development. Otherwise, evaluation cannot deal with these soft goals without arousing the suspicion that the evaluators are only making an attempt at self-justification. Besides, ambiguity about the goals can lead to false expectations among users. In innovative projects, the goals may well have to be adapted during development, but they should remain as explicit as possible.

4.5 What makes up DSS quality?

Introduction

To sum up the literature review, quite some research has been spent on information system evaluation, but there exists a discrepancy between what is relevant and what is easily measurable. To be clear: what is relevant is not easily measurable, and what is easily measurable is often not very relevant. This is especially felt for innovative systems such as DSS. When it is insufficiently recognized that designing a DSS is a process of change and learning, problems occur.

The concept of quality implies a comparison of the evaluated entity with some standard. Therefore it is not surprising that emerging directions in research and development are not much concerned with quality, contrarily to more mature branches. An established discipline generally has accepted quality standards. In Information Systems, some subdisciplines are still in the innovative phase. Other branches are actively engaged in quality debates. A DSS is by definition innovative. At the same time, DSS evolve in a world of new technologies and methods, which makes comparison between systems and between concepts awkward. Therefore it will remain hard to find standards against which to evaluate DSS, and only a flexible, situation-adaptable approach for quality assessment is feasible.

What is currently evaluated?

As pointed out in the literature review, current evaluation methods usually deal with the *DSS* link in the quality chain. What is evaluated are such things as the quality of the input information, the quality of the software, or the satisfaction of the user. The quality of the *decision* is sometimes taken into account through immediate quality assessment. The quality of the *functioning of the organization* is at best taken into account as a rough assessment of improved effectiveness; Hamilton and Chervany's tool is the most explicit one that I found. It is understandable that there is not much to be found on realistic and detailed assessment of improvements in the quality of decisions and of the functioning of the organization. This is so partly because improvements in these fields are hard to measure, partly because they are taken for granted. Simulation of the organization can sometimes help, but introduces the problem of assessing the validity of the simulation model.

The quality of the *problem solving process* is another element which is not often taken into account. The problem solving process is often mentioned in the literature as something to which attention should be directed, but it remains elusive. It appears that, if they are motivated, human problem solvers are quite flexible and can learn to use a great variety of problem solving aids, in spite of their cognitive boundaries. The main problem in DSS design might well be how not to hinder the creativity and flexibility of the user, while at the same time providing the user with tools to help solve those aspects of their problems that can be anticipated during development.

Finally, the quality of the *DSS development process* is an altogether different matter, at least in the literature. It is dealt with by different authors in different publications than the other four links in the quality chain are. Authors who discuss the quality of DSS development tend to be very outspoken in their opinion that the nature of the development process is crucial for success or failure of the DSS. In particular, the various authors who advocate user-centred development stress that mutual understanding through a shared process of learning is a precondition for success of a DSS project.[52]

What should be evaluated?

Which links of the quality chain should DSS quality assessment aim at? Can the literature reviewed in this book help to substantiate or falsify the claim made by the proponents of user-centred DSS development that the development process is of paramount importance for success of the DSS?

Processes versus products

To answer these questions it is good to recall that the quality chain is also a process - product chain, as mentioned in the first paragraphs of this chapter. Now it appears from the literature review that the products (DSS and decision) are most often the object of quality assessment. The processes (DSS development, problem solving, and functioning of the organization) are much less prominent in the evaluation literature. The functioning of the organization is sometimes mentioned, but in rather 'product-like' terms such as 'organizational performance'.

At the same time, there is evidence in the descriptive literature on organizations that processes of change are very important. A few highlights shall be reiterated here. To begin with, any pressure towards change imposed on an organization or on a person induces a reaction, and so will a DSS development project. This reaction varies strongly with the attitude of those of which the change is required; it can vary from defensive avoidance, blocking a project, through sceptical indifference, to a surge of enthusiastic cooperation.

Secondly, it can be mentioned that managers usually have a very busy and highly fragmented working day, and will not invest time in a DSS development project unless highly interested and aware of potential benefits. In practice this often leads to a rather

[52]As Beers (1990) points out the same holds, *mutatis mutandis*, for information systems development in general.

indifferent attitude on the part of the intended users of a DSS, until they have a clear motive to invest time and mental effort. All in all there are strong arguments in support of the proposition that the development process of a DSS is very important.

A similar train of thought, though not quite as forceful, can be followed to argue that the process of problem solving is important. In chapter three, the psychological perspective uncovered the importance of the right degree of novelty and the right degree of stress for a problem-solver, so that he can create a context of play in which he can exert creativity. If possible, a DSS should allow the user to create such a context. This implies that DSS should be highly interactive and allow the user enough degrees of freedom. In this light one need only reflect on the enormous success of applications made by decision makers for their own purposes, usually in spreadsheet packages, database packages or other popular software packages, to realize that this success owes much to precisely the fact that these packages allow their users to 'play' with their problem in a creative, open-ended way. Implicitly, this is a forceful quality statement in favour of these packages as DSS tools.

Concluding remarks

It can be concluded that evaluation of the product links in the quality chain serves a different purpose than evaluation of the process links. A product can be critically examined at will, it can be certified, or it can be sold; its quality assessment has legal and commercial value. A process does not possess these properties, at least not so clearly; but processes determine the quality of their end products. The quality of problem solving with a DSS determines the quality of the resulting decisions to a large extent, and the quality of DSS development determines the quality of the resulting DSS. Essential elements of the quality of these processes are the sharing of expertise between developer and user during development, and the sharing of expertise between user and system during problem solving.

As to the relation between the quality of a decision and the functioning of the organization, there exists a clear correlation between the two in horticulture in any case, as shown by the findings of Alleblas and Bots discussed in chapter three. However, it is rather the process of decision making than the end product of this process that is correlated with parameters of the functioning of the organization by these authors. In terms of the quality chain, the 'decision' link is skipped.

The influence of the quality of the DSS on the quality of problem solving is not so clear. If insufficient attention is given to problem solving during DSS development, the resulting DSS may well remain without any effect on it. It may also happen, however, that a DSS leads to radically new ways to solve problems. For instance, shorter throughput times of the planning process may enable plan revisions that were avoided prior to the use of the DSS, thus making the organization much more adaptive.

It is also good to repeat that the quality chain is not just a chain. In particular, the literature gives evidence that the process of DSS development can have far-reaching effects because of resulting changes in attitude of those involved. In terms of the quality chain, this is a causal shortcut from DSS development directly to the functioning of the organization. It shows that the quality chain should not be regarded as more than a vehicle for discussion. The web of causalities and coincidences in a real DSS development process is similar in complexity to the functioning of organizations in general, and it is not unravelled in this study.

Because of the complementary nature of the product and process links, both types are relevant for quality assessment. The relative importance of the links of the chain depends on the goals of the evaluation. In the perspective of this study, the process links are more important than the product links, because high-quality processes are more of a precondition for high-quality results than *vice versa*.

So, the main conclusion from this chapter is that high-quality processes of DSS development and of problem solving are crucial if DSS are to be really used. We do not have to be stupid about quality: the quality of a DSS can be meaningfully assessed.

4.6 QUEST, A quality estimator for DSS

Introduction

On the basis of the discussion above, and with help of the literature, I have put together a list of quality characteristics for DSS. This list, together with some suggestions as to how it can be used and how data for its use could be collected, constitutes a QUality ESTimator for DSS: QUEST. QUEST does not pretend to yield the one and only valid quality statement on a DSS. One's quality criteria are dependent upon one's goals; therefore the relative weighting of the elements of QUEST will have to be established anew for each evaluation, and elements that are specific for the evaluated DSS in question will in most cases have to be added.

QUEST is structured in five main elements, which are the links of the quality chain. There are a number of possible quality measures for each of these. If one wishes to zoom in on particular quality elements, other tools can be used, of which a number are mentioned above. The value of QUEST is that its scope is very wide. In particular, QUEST pays attention to the processes of DSS development and use, rather than limit itself to the end products of these processes.

Functioning of the organization

Quality elements of the functioning of an organization can be roughly divided into efficiency-oriented and effectiveness-oriented elements. Hamilton and Chervany have provided a number of elements.

Efficiency-oriented elements are:

- automation of time-consuming activities
- cost displacement (people, equipment)
- reduced inventory level
- reduced number of backorders
- improved utilization of resources

Effectiveness-oriented elements are:

- improved contacts with suppliers, personnel, clients (i.e. more flexibility, better information)
- improved adaptiveness of organizational processes
- improved quality of work (in particular of the DSS's users, e.g. feelings of increased competence, but also of those affected by the DSS's output)
- more positive attitudes towards job, or towards DSS
- increased creativity in the tasks affected by the DSS

Decision

Here, the decisions reached with help of the DSS are seen as a product in their own right. Again, the distinction into efficiency-oriented and effectiveness-oriented elements is useful.

Efficiency-oriented elements:

- length of time to reach decision
- costs attributable to the decision (personnel, data collection)

Effectiveness-oriented elements:

- immediate quality assessment, i.e. comparison with normative criteria
- explicitness of criteria used in immediate quality assessment
- assessment of robustness (e.g. by simulation)
- *a posteriori* evaluation of premises and decision against reality

Problem solving

Problem solving is the link that connects the DSS to the decisions reached with its help.[53] The elements of problem solving quality listed here are taken from the previous chapters as well as from the present one. They are given in approximate order of increasing generality, starting with specific elements of the process, continuing with general properties, and ending with context-related elements.

Specific elements of the problem solving process:

- possibilities to manipulate constraints (which constraints? At which moment and in which way can they be manipulated? Who enforces the constraints, DSS or user?)
- possibilities to compare alternatives (entire plans, partial plans, individual actions)

General properties of the problem solving process:

- possibility to work incrementally (start 'quick-and-dirty', refine to the extent that time permits it)
- flexibility
- decomposition, modularity
- contribution to cognitive control over problem (e.g. explicit use of process model)
- ergonomic aspects of the interaction (possibility to interrupt, memory load, etc.)

---

[53]Indeed, one could say that problem solving is the DSS as a tool, or the decision as a process.

Context-related properties:
- correspondance with the practice that was current before the DSS came into use
- suitability of solution techniques and models inside the system for the intended users (do the users understand the notions they are supposed to use?)
- suitability of solution techniques and models inside the system for the intended situations (are the assumptions about predictability, availability of data etc. the DSS requires realistic?)

DSS

Here, the DSS is considered as a product, rather than as a tool. There is some arbitrariness in the distinction between DSS and problem solving, especially where interface aspects are considered.

Quite a number of quality elements of an information system as a product are given by other authors, often much more elaborately than they are given here. The literature review above gives entries into this literature. The present list pays limited attention to the software-technical dimensions of DSS quality. The legal dimensions are not taken into consideration at all. Elements of DSS quality considered here are:

Input data
- timeliness
- correctness
- completeness

Software quality
- usability (reliability, efficiency of code, user-friendliness)
- maintainability (testability, transparency, modifiability)
- portability (completeness, independency of technology and of organization)

Availability
- in time
- in space

Models
- performance versus running time
- quality of result relative to norm or optimum if available
- robustness

Interface
- familiar interface
- usable outputs (e.g. on paper)

Satisfaction
- explicitly stated
- implicitly shown (adoption, use)

DSS development

The development of a DSS usually takes years. The quality of a DSS development project is not constant during this time. It may deteriorate or improve considerably as a consequence of events. Apart from the fact that all people involved in a project have a continuous responsibility, explicit quality assessment should be carried out from time to time (for instance several times a year), so that unwanted developments in the project can be corrected.

Some of the elements listed below primarily deal with awareness, and can therefore be collected by directly asking the relevant individuals. Other elements primarily deal with actions. Here, explicit statements made in interviews or in questionnaires should at least be checked against observation of the actual development work. This distinction is of course not a black or white one, but merely refers to a difference in emphasis.

Awareness:

- recognition of the open-ended nature of a DSS project
- explicitness of the project goals to all parties, and agreement about these goals
- explicitness of the methods followed to all parties, and support for these methods

Actions:

- communication between user and developer (resulting in a shared conceptual model of the problem situation)
- active involvement of the user
- active involvement of management
- adaptiveness of the project to unexpected events (new goals or perspectives, obstacles, etc.)
- primacy of interface over solution techniques (this has to do with the attitude of the developer)

4.7 How to use QUEST?

QUEST can be used both during and after a DSS development project. First it can be used for monitoring the development process. Then, as soon as there are prototypes or working parts of a DSS, it can be used to assess their quality, particularly when it comes to their problem solving aspects. Finally it can be used at the conclusion of a project.

Most often an evaluation with QUEST will be used internally by the project members. The main input will consist of interviews, possibly supplemented by questionnaires. As a different kind of source, other documents can be used, such as minutes of meetings, progress reports, manuals.[54] Finally, the DSS itself is an obvious object of evaluation.

How the quality elements should be weighted depends on a number of things. The main determinants are the projects' goals and the goals of the evaluation. Another one is the status of a DSS. If it is not yet in actual use many measurements can only be done in simulation or laboratory tests, or by informal estimation.

[54]In the next section, QUEST is illustrated on the basis of a case description in a Ph.D. thesis (Verbeek 1991). Unfortunately, evaluative descriptions of DSS projects, such as the one Verbeek gives, are rare. Most literature on DSS projects is hardly suitable for an analysis with QUEST, because process aspects are not mentioned.

Some suggestions for collecting data for an evaluation with QUEST are given in the next paragraphs.

Functioning of the organization

Effects of a DSS on the organization can be assessed by evaluating trends in the organization's economic performance, by simulating DSS-generated decisions against non-DSS-generated ones, or by asking those affected: users, suppliers or clients. These three techniques all have their merits, and they also have serious drawbacks. The first technique is seldom feasible due to many disturbances, the second is not real-life, the third is easy but highly subjective and possibly misleading. If possible the methods should be combined.

Decision

One readily measurable quality criterion for the efficiency with which a decision is made is the time consumption with and without the DSS. Sometimes there exist straightforward financial criteria for the quality of a decision. This is usually the case in tactical planning. However, uncertainty about the values of the uncontrolled variables is also a general phenomenon, which makes these figures less useful. Other economic criteria for decision quality, e.g. robustness and labour quality, are harder to assess. Simulation can be used to assess robustness. Feedback from actual realization of a decision is not always a usable criterion because there is usually no comparison against an alternative decision so that hindsight biases can easily distort the evaluation. Goals outside the direct economic realm can only be assessed by observing or by asking those concerned.

Problem solving

In chapter three it has been pointed out that there exist criteria in organization science to evaluate the quality of a problem solving or decision making process. These, however, mainly concern the inputs to the process, and do not address the details, cognitive or other, of the actual problem solving activities. A difficulty inherent in describing problem solving processes is that they are largely subconscious. Intuition is a phenomenon which for a real-life decision is not easily decomposable.

Protocol analysis is a good, but very laborious, method to assess problem solving processes. Close observation is a 'quick-and-dirty' substitute which usually works quite well.[55] For instance, it can be highly instructive for a developer to see a novice user work with his software.

DSS

Software and interface quality criteria, which are alike for all information systems, are amply described in the literature. Both are measurable in various ways. Obviously, for

---

[55]An account of a close observation of grower Q during a session with PROPLAN HAND is given in the next chapter.

model-based DSS the quality, in particular the validity, of the models in the DSS is important.

DSS development

Documents of various nature that are generated during a project can be used to assess the quality of DSS development. To get an insight into the quality of the cooperation between developer and user, other sources of data are usually better. These are interviews, questionnaires, and, best of all, participatory observation.

4.8 An application of QUEST

In order to get an impression of what using QUEST means, it will be used to estimate the quality of two planning systems. PROPLAN will be dealt with in chapter six. Here, the CAPTAINS system introduced in chapter three will be evaluated. Verbeek (1991) is taken as the source text. This can be done because Verbeek includes extensive descriptions of both the DSS and the development process, as well as a discussion about the quality of both. Nevertheless this evaluation of CAPTAINS should be looked upon as nothing but an illustration of the use of QUEST. Having made this disclaimer, the CAPTAINS system can be evaluated as follows.

Functioning of the organization

- The *throughput time* which the Crew Planning Department needs to answer questions about a plan higher management came up with has been reduced from about two weeks to about two days. This has resulted in a much higher involvement of higher management and a decrease of time wasted by higher management in trying to recall what the problems and proposed solutions were that they had discussed during previous meetings.
- A *system manager* has been appointed who is in charge of CAPTAINS. This person has turned out to be a promoter of the use of PCs for various purposes. This is an unanticipated side benefit of the system. Presumably, although Verbeek makes no mention of this, the planners have also adopted a more positive attitude towards PCs due to CAPTAINS and to the system manager's activities.
- The planners have more *confidence* in their plans and advices. Although not explicitly stated this presumably improves their job satisfaction and, as a result, their performance.

Decision

- The *efficiency* of the planning process is greatly enhanced.
- The '*effectiveness* in a narrow sense'[56], as Verbeek defines it, is improved.

---

[56]This is what I call the immediate decision quality, i.e. the quality of a proposed plan according to certain standards.

- Longer *time horizons* can be studied.

Problem solving

- The basic *planning procedure* has not changed. It still consists of a two-level hierarchic decomposition. Firstly, 'starting points' are defined, i.e. assumptions about uncontrolled variables. Then a plan is made based on these starting points. If no acceptable plan can be reached, the planners backtrack and start again from different starting points. The system has made it possible to increase the number of backtracking steps from usually none at all to several. Thus, the planners get valuable insights about the influence of the choice of starting points which they did not get before they used CAPTAINS.
- When making a plan, the planner can switch freely between *manual* and *automated* planning. He can choose to make only parts of a plan by hand, or edit plans made automatically. Thus the system does not restrain his freedom to choose a strategy for making a plan.

DSS as a product

- The quality of the *input information* is improved, because consistency between the different planning boards is now guaranteed by the system.
- *Software* quality is not dealt with by Verbeek. However, the existence of an in-house system manager, as well as an application development department, make it likely that support, maintenance and updating of the system will not be uncommonly problematic.
- The *interface* is familiar to the planners, which is the main quality criterion for the DSS as a product. It is visual and easy to manipulate. This interface played an invaluable motivating role during the project.
- The users express *satisfaction* with regard to user-friendliness, usability, decision support, speed, and availability of the system. They are less satisfied with the support of routine administrative tasks and with the integration of the system with other procedures. These latter aspects did not have priority in the CAPTAINS project.

DSS development process

- Although management was committed right from the outset of the project to having 'something' built, the importance of a clear *project structure* and management was underestimated at first. Presumably this had to do with the fact that nobody knew what to expect of the project: it was not recognized as a DSS project with the associated open-ended nature and need for creative involvement. After a first working prototype became available in autumn 1989, there was a closer commitment from the people involved.
- Verbeek does not give much explicit attention to *learning*, but the very title of his book, 'Learning about Decision Support Systems', indicates that he regards it as

crucial, and his attention to the development process is an implicit acknowledgement of the importance of learning processes of both developers and users. He mentions that the system enabled the planners to obtain a much better insight into which constraints are limiting for a plan.

- The *communication* between users and developers grew into a severe bottleneck, after an enthusiastic start in which the developers attempted to follow the SDM-II[57] Global Design phase. It turned out that no valid functional requirements could be derived. Also, it turned out that the developers had a mistaken idea about the planning process. Verbeek characterizes the planners as behaving in a 'bounded rational' manner, whereas the developers implicitly assumed them to be 'rational' (pp. 178-179). For some time, the two parties did not think too well of each other. The confidence between the two parties started to grow from the moment when the first results became tangible. After the delivery of the first version, the users were so enthusiastic that the developers organized an all-day meeting outside the company to get an opportunity for informal and undisturbed exchange of thought. The developers feared that the users might otherwise expect too much from a second version. From then on, mutual understanding and respect prevailed.[58]

To sum up, it appears that the commitment of all parties concerned, the creativity of the developers in overcoming an impasse, and the close cooperation between developers and users, have been a *conditio sine qua non* for the CAPTAINS system. At the same time these factors have been a solid basis for quality aspects of CAPTAINS as a product.

The system has not drastically changed the nature of the problem solving processes or of the resulting decisions, but it has had important benefits due to reallocation of time, both the planners' time and higher management's. There are also side benefits due to a more positive attitude towards computers.

In the literature, pleas are often made for more attention to the decision making process, and the suggestion is sometimes made that this process is a mysterious entity requiring a lot of study. In this case study, attention to the decision making or problem solving process was indeed important but there was nothing esoteric about it. Attention to the actual decision problem and to the way the planners handled it were central elements. An atmosphere of open communication and mutual respect between the users and developers was important for success.

---

[57]SDM-II stands for the second version of the 'Systems Development Methodology', an approach for structuring information system development projects which is much used in the Netherlands.

[58]It is interesting to note that, in accordance with what Kierulf et al. propose, the project progressed well from the moment that there existed one system version to focus everybody's attention.

5 The PROPLAN project

Ja, der er planter, som slet ikke naar ned til Jorden.[59]

Niels Meyn (1921, p. 66)

This chapter describes the history and current status of the project PROPLAN, which has evolved around a succession of programs bearing the same name. PROPLAN is an interactive planning system for cultivation planning at pot plant nurseries. Most of the historical PROPLAN versions are now outdated. Nevertheless it seems fitting to treat them because readers may be interested in the research approach, especially the idea of using 'surrogate environments' as a playground for problem formulation and for solution techniques. This approach has allowed the PROPLAN team to gradually develop the kind of interactive, manipulative heuristics they had in mind.

After the historical overview the current PROPLAN versions are described. Experiences with the various PROPLAN versions, both in the laboratory and in the field, are given. Finally the PROPLAN project is evaluated with QUEST.

5.1 Project history

The project PROPLAN originated in 1985. It was initiated by John L. Simons, who was affiliated with the department of Computer Science of the Agricultural University at the time. The initiative was prompted by a need for more support of cultivation planning in pot plant culture. This had become apparent in studies by mixed teams in which researchers, extension consultants and growers participated (see Beers et al. 1985, INSP-LO Werkgroep 1986). There already was a system for cultivation planning (IPP), which was somewhat more strategic in nature, and which was based on linear programming (Saedt 1982). For this system an addition at the tactical/operational level was under investigation (Saedt et al. 1986, Barendse 1986). In fact the research was a continuation of these investigations. Simons had previous experience with DSS that used heuristics for combinatorial problems (Huizing and Simons 1980), and the problem of pot plant placement appeared to be amenable to a similar treatment.

Over the years, more than forty people have been involved in the project. Almost all of these were students who only participated for a few months. The group of people

[59] Yes, there are plants that never even reach the ground.

working on the project is referred to as 'the PROPLAN team'. Staff included in the team for a substantial amount of time were John L. Simons (1985-1988) and myself. References to progress reports of all participants till 1990 can be found in Hofstede (1990[f]).

After some preliminary work a first progress report was published in spring 1987 (Hofstede and Simons 1987). At that time the project bore the name GROMS for GROwer's Management Support system. Because this name had unfavourable connotations with the Dutch public[60], the project has since then taken the name of its first prototype system, PROPLAN.

Hofstede and Simons (1987) defined the goal of the project as follows (p. 10):

> The goal of the project GROMS is to develop a prototype DSS for the tactical planning process at pot plant nurseries in a commercial hardware environment.

More specifically, PROPLAN aimed at the placement problem. The first phase of cultivation planning, i.e. determining broadly what to cultivate in the context of available labour and finances, was already covered by IPP. During the first years of the project work was carried out along four parallel lines:

- Heuristics.
  Prototypes were built on a VAX mainframe computer to test the value of design ideas for heuristics. These heuristics were meant to offer interactive support for the placement aspect of cultivation planning.
- Functional design.
  Through interviews with growers and extension consultants, detailed functional requirements were derived for a prototype DSS. The result is described in Biemond and Hofstede (1988).
- Data modelling.
  The reference information model for pot plant nurseries given by Beers et al. (1985) was adapted to the functional design. A number of refinements were necessary. Conversely, the Beers et al. model turned out to be all too specific in some aspects.
- User interface.
  A literature study was made and experiments were carried out on a Symbolics LISP machine. Though promising these experiments were not continued for a number of pragmatic reasons. The machine turned out to be difficult to master, which hindered progress. Another disadvantage of the Symbolics machine was that the code was not portable to other machines, and that only one terminal was available.[61]

---

[60] In Dutch, 'grom' means 'growl'.

[61] Much later, a PROPLAN-like prototype DSS was developed on the Symbolics machine in KEE (Hartog and Kroon 1991).

The most successful line of research during 1986 and 1987 was that of heuristics. This research provided the project with the first working PROPLAN versions.

From the end of 1987 on the four research lines merged into one. To avoid duplicating efforts expected of commercial software vendors, the functional specifications were not fully implemented. The placement planning part was given priority.

5.2 Heuristics and surrogate environments

Research perspective in 1986

The aim during the first years was to develop heuristic methods for the placement problem that were interactive during operation but at the same time able to perform well without interruption by the user. This meant that the PROPLAN team had to formalize the placement problem. A middle-of-the-road realistic cultivation plan can include up to some 10 products, which can each be potted in various lot sizes, some 100 to 200 tables, and 52 time-slices. Given such quantities, the task of automatically generating a plan is of a tremendous combinatorial complexity (Hofstede 1990[b]). Because the researchers did not have the ambition to generate optimal solutions, but merely to reach any solution that satisfied certain constraints, backtracking (see chapter two) seemed the indicated general heuristic to cope with this complexity. It is a depth-first search technique, and it can reach a solution quickly if the search can be well guided. The researchers wanted to have the user guide the heuristic. Because backtracking is incremental, intervention by the user is in principle possible between any two iterations. Simons' earlier work with backtracking on a timetabling problem (Huizing and Simons 1980) had resulted in a DSS called VLUCHT. This DSS was used at the Dutch National Aircraft and Space Laboratory. The success of VLUCHT encouraged the team's attempt to use backtracking for the placement problem.

Constraints

Interviews with pot plant growers during the first years were usually followed by heated discussion among the researchers and students working at PROPLAN. The reason was that pot plant cultivation planning is not only complex but varies widely from one grower to the next. It was very hard for the PROPLAN team to find out how to distinguish between controlled and uncontrolled variables. Should temperature, for example, be treated as an independent variable which the grower could manipulate to influence the plan? It was also problematic to find out which constraints to treat as hard and which as soft ones. Were adjacency constraints important? What about location/product incompatibility constraints? Were cultivation schemes fixed or flexible? And so on.

Surrogate environments

The combinatorial complexity of the real-world placement problem, not to mention the doubts about the best problem formulation, were frustrating. However, Simons had in his earlier work used a clever work-around for a problem of similar complexity. This approach used so-called *surrogate environments*. To define these surrogate environments, the real-world problem situation was first analyzed and modelled in a mathematical way similar to Van Hee.[62] Then this model, the formal version of the 'real-world problem', was stripped of some of its elements, especially of constraints. This was done in an order of increasing importance. After each simplification, a surrogate environment one step less complex than the previous remained. In fact, intermediate surrogate environments were not mandatory at this stage. What mattered was that in the final and most simple surrogate environment only essential variables and constraints remained, as well as the most important properties of the real-world problem as it was first conceptualized.

Once defined the surrogate environments could be implemented in inverse order, that is, from simple to complex. This was a sort of prototyping cycle. Building the first surrogate environment was straightforward. It allowed experimenting with the environment to find out whether it was promising. If not, this would mean that the essential properties of the conceptualization of the real-world situation were ill-chosen and had to be changed. For the successive additions to the surrogate environment the same held: build it, experiment with it, backtrack if not suitable, else proceed to the next surrogate environment.

The PROPLAN team adopted this approach. The first surrogate environment was a rigorously simplified placement problem world: the 'pentomino environment'.[63] The idea was to work on progressively less simple, more realistic surrogate worlds, adding more and more elements of the real-world placement problem until PROPLAN would finally be working on the real-world placement problem.

Growers were found to use an informal backtracking approach to make their cultivation plan (cf. Jongejan 1986). They incrementally developed a plan until some space allocation conflict forced them to alter or remove production lots that were already planned. In parallel to the growers' approach, backtracking heuristics were to be developed for the successive surrogate environments, and the researchers hoped that in the end they would be able to derive a backtracking-based interactive system for the real-world problem.[64]

The project has known a succession of surrogate environments and prototype versions. They will be discussed in the next pages. Figure 5.1 gives an overview for reference.

---

[62]See chapter two, p.25.

[63]See the following pages and figures 5.2 and 5.3.

[64]Arguments in favour of backtracking which are of a more general nature can be found in chapter two.

environment	*year*	*version*	*key-word*
pentomino	1986	pentomino	Pentomino puzzle
	1986	time-space-figure	Time / space distinction
VAX-PROPLAN	1987	A	Log plan interface, transport cost function, interactive backtracking
	1987	B	Time locks
	1987	C	Space locks
	1988	D	Variable seed table
	1988	0.0 (on PC)	PC interface
PC-PROPLAN	1989	0.1	Variable lot size, feasibility check, dashboard screen
	1989	0.2	INGRES database, retrieve old plans
	1990	0.3	realistic transport cost function
PROPLAN HAND	1990	1.0	no backtracking, no constraint checking

Figure 5.1: Successive PROPLAN surrogate environments and versions

The pentomino environment

The first surrogate environment was the two-dimensional pentomino problem. This environment included placement of composite elements into a two-dimensional space, resembling production lots in a log plan (see figure 1.2) which had to be transported during cultivation, but excluded all complicating factors such as growers' financial or labour goals, or the multitude of constraints mentioned in chapters one and three.

Pentominos are n-angles in a plane of unit squares which consist of five conjunct units. Twelve different pentomino shapes exist. Examples are given in figure 5.3, and figure 5.2 shows all existing pentominos. In fact figure 5.2 depicts a solution to the common version of the 'pentomino problem', which consists of trying to fill a space of six by ten unit squares with pentominos, one of each of the twelve shapes.

The pentomino problem was chosen because it was thought to represent the essential characteristics of the pot plant placement problem. An additional advantage of this problem was that it is well-known as a formal problem and as a game (see e.g. Bitner and Reingold 1975).

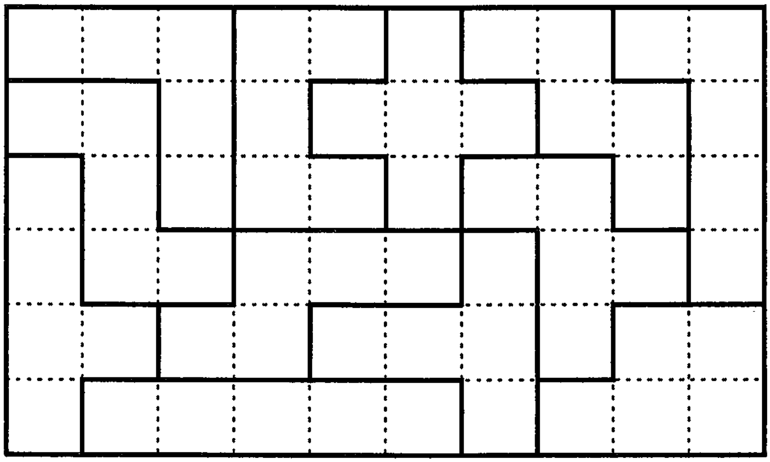


Figure 5.2: A solution to the pentomino problem.

In designing a backtracking heuristic for the pentomino problem the PROPLAN team tried to apply the ordering principle 'hardest first' but found that it did not work so well (Jongejan 1986). There were no satisfying criteria for 'hardness' of a pentomino. However, the backtracking algorithm performed within seconds on a VAX-750 computer, which was promising.

The time-surface figure environment

The pentomino environment was extended with a time component. Consider the following pentominos (figure 5.3):

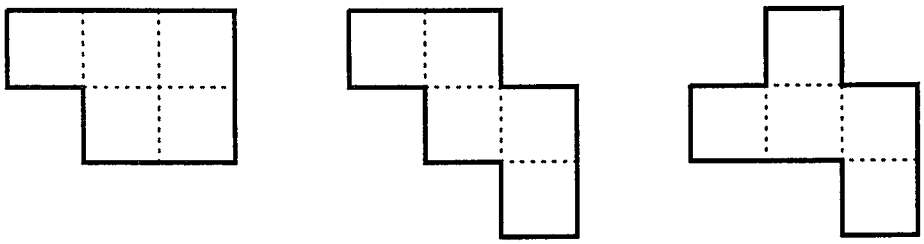


Figure 5.3: Three pentominos

A pentomino has two dimensions whereas the pot plant placement problem has three dimensions: the horizontal and vertical space dimensions in the greenhouse, and the time dimension. If a horizontal and a vertical axis were added in figure 5.3, and if the horizontal axis were taken to be the time axis and the vertical to represent space, collapsed into one dimension, the three different pentominos would stand for different placements of the same space requirement over time.[65]

A place-independent time-space graph of a production lot was termed a *time-surface figure*. A time-surface figure is a relaxation of a pentomino in that its shape is flexible, but a restriction in that the sums of its unit squares per X-coordinate cannot be changed. In the above example the time-surface figure of all three pentominos is (1,2,2): one space unit is occupied in the first (left) time unit, two in the second time unit, and two in the third (right) time unit. The three different placements (i.e., different positions on the Y-axis) in figure 5.3 were called *configurations* of the time-surface figure (1,2,2). A configuration was defined as a set of time-place combinations which together represent the space requirement of one pot plant production lot. Thus a configuration was a time-surface figure with fixed placement.

All in all, in the time-surface figure environment there were extensions on the following points compared with the pentomino environment:

- The space was oriented so that time was displayed on one axis, space on the other.
- The time axis was made semi-infinite at the right-hand side, imitating the open planning horizon of a pot plant grower.
- The pentomino concept was replaced by the time-surface figure concept.
- At each node of the backtracking tree, different configurations of a time-surface figure could be placed interchangeably, imitating alternative placements of a production lot.

The backtracking algorithm still performed within seconds in the modified pentomino environment, which encouraged the PROPLAN team to move on to a less artificial environment.

The PROPLAN environment with fixed lot size

The second surrogate environment was considerably richer than the first. It resulted in VAX-PROPLAN, written in VAX-PASCAL, operating on a VAX-750 computer, with a highly interactive user interface (Van Lemmen and Huijskes 1987). The main screen was a log plan.

An important feature of VAX-PROPLAN was the so-called *transport cost function*. This function computed the costs associated with a particular configuration, using heuristics about the actual tables of origin and destination and assuming that transport to an adjacent table was half as expensive as transport to any other table. This assumption is reasonable for most greenhouses, since transport often takes place by chariot. For

[65] This is a reasonable simplification in most greenhouses. Tables are often placed and numbered as shown in figure 1.1 (p. 3).

transport to an adjacent table the chariot is not needed and the plants are handled only once. For transport by chariot the plants have to be handled twice.

Furthermore, each product had a *cultivation scheme* attached to it, specifying the time-space figure dependent on the season of potting. The user could edit these cultivation schemes. Only one lot size per product could be defined.

Given a certain available space and a semi-open time horizon, the transport cost function, and a cultivation scheme for each product, the problem was to determine when and at which location to pot and cultivate which products. There was a trade-off between transport costs and unused space: the fuller the greenhouse, the more difficult it became to find adjacent tables.

The backtracking algorithm

An interactive backtracking algorithm was central to the first version of VAX-PROPLAN, as well as to all its successors except the very last. This algorithm placed production lots in the greenhouse. The algorithm proceeded by placing one lot at a time. The user could see each lot on the log plan as it was being placed or removed. He or she could also interfere with the algorithm during its execution, e.g. by forcing it to backtrack, or by accepting a table as 'unused space'. Unused space consisted of time / space combinations at which the heuristic did not attempt to place any production lots.

The algorithm started at the upper left-hand corner of the log plan, i.e. at the first table in the first week of the plan. Placement proceeded with time, i.e. from left to right. Each successive week was only considered for potting if there was no untried potting table left in the previous week. Each production lot was placed starting with a so-called *seed table* in its first week of cultivation. The user could install the branching degree of the search tree.

The algorithm accepted candidate lots for the next placement depending firstly on the number of unused spaces in the current potting week, which was tied to a user-given maximum, and secondly on a maximum transport cost value for the candidate lot, derived heuristically from the transport cost function and the actual placement of the configuration. This was an expression of the opinion held by the team that space planning involved balancing unused space against transport cost.

An important task was to prevent the backtracking algorithm from embarking on hopeless searches. A *feasibility check* was therefore carried out after the specification of a planning problem instance, and before starting the backtracking algorithm. The feasibility check basically checked whether the minimum potted quantities required by the user would fit into the planning horizon.

VAX-PROPLAN knew four versions, which were only informally numbered. They will here be called A through D. The main additions to the surrogate environment in each successive version will also be presented.

- Version B introduced *time locks*. Time locks were implemented as a matrix, with the dimensions products and weeks, containing boolean values. If a time lock was set for a particular product PROPLAN did not allow the product to be potted in the specified week.

- Version C introduced *space locks*. These were analogous to time locks, but with products and tables as dimensions of the matrix of booleans. A space lock prohibited placement of a product on a particular table at all times.
- Version D introduced *variable seed tables*, because the obligation to use a single seed table often induced backtracking when accepting the table as unused space and trying a different seed table would have been preferable. The variable seed table allowed to reach acceptable solutions in less time. PROPLAN now tried all tables in the current potting week as seed tables, up to the maximum unused space for the week. This maximum could be set by the user.

5.3 PC-PROPLAN

During 1988 the PROPLAN team got personal computers (PCs) at its disposal. This meant that a conversion of VAX-PROPLAN to PC-PROPLAN was now possible. Such a conversion was desirable given the goal of the project: but for a few exceptions, growers did not possess other management computers than PCs, if they possessed any at all.

PROPLAN 0.0

PROPLAN was first converted with only a minimum of changes, in order to compare the algorithms of the two versions. This conversion turned out to be not a simple thing to do, as the VAX version of the software, written in VAX PASCAL, had become unwieldy and badly structured after the successive changes by six programmers. The conversion to Microsoft C on an IBM-compatible PC yielded PROPLAN 0.0, which became available by the end of 1988 (Wisse 1989). The lot size was still fixed but there were significant improvements to the user interface, which was now similar to popular PC packages in its use of colours, menus and function keys. Specifically the team wanted to make the interface look like PC-INGRES, which was the database package considered for the PROPLAN database.[66]

PROPLAN 0.1

The next version included a further important step from the VAX-PROPLAN version D surrogate environment towards the real-world planning environment. It yielded PROPLAN 0.1, available by March 1989.

PROPLAN 0.1 introduced a number of novelties:

- Variable lot size.

  The size of a production lot was now equal to one or more times the *minimum lot size* given in the PROPLAN data files. The user could set the minimum lot size

[66]Finding a suitable database package was not at all easy. The balance between performance and modifiability was hard to find. After some experiments, among others with Dbase 3^+, INGRES was chosen for the much greater development speed it allowed.

at the desired value. There could be different minimum lot sizes for different greenhouse departments.

- Improved feasibility checking.
  The feasibility checking software routines were re-written and provided with a visual interface which enabled the user to interactively make a rough assessment of available and used space per week and to change space requirements. Some sophisticated heuristics were developed for the feasibility check (Loosveld 1989).
- A prioritizing function.
  This function was used to put the candidate lots for a certain placement in order. The user could provide it with parameter values.
- The dashboard screen.
  One separate screen was introduced on which the user could set a number of parameters to regulate the backtracking algorithm. These parameters included the categories:
  - acceptance criteria for candidate lots
  - prioritizing criteria for candidate lots, by means of the prioritizing function
  - search tree parameters such as branching degree.

  The dashboard screen is depicted in figure 5.10 and described in the accompanying text.

PROPLAN 0.2

PROPLAN 0.2 became available in september 1989. Its main novelties were twofold. The first was an interface with the PROPLAN database in INGRES, which meant that input data for PROPLAN could now be entered through INGRES menus instead of by file editing. However, INGRES did not seem suitable for incorporation into a PROPLAN version distributed to growers, if only because of the size of the package. Therefore INGRES and PROPLAN were not directly interfaced but ASCII files[67] were used as an intermediary. The second novelty was the possibility to store and retrieve previously made plans, the lack of which had been much felt.

With these additions, PROPLAN 0.2 could almost claim to be a usable planning aid in practice, at least as far as the functionality and the usability of the placement planning part were concerned. Its user interface and performance were of sufficient quality. One obstacle remained, viz. the small problem size the system could cope with: not more than 20 tables could be displayed on the screen, whereas growers usually included some sixty to over a hundred tables in a production plan. Also, the PROPLAN software had never been properly tested. The team, however, saw its task as limited to the development of a prototype and did not want to become involved in trying to make a commercial product.

---

[67]ASCII files are a simple type of text files stored according to the American Standard Code for Information Interchange. Most software packages on most hardware can read and generate ASCII files.

In a marketable version of PROPLAN 0.2 a number of requirements would have to be met. They would include providing high-quality input data about cultivation schemes and constraints, interfaces with spreadsheet packages and legible printouts of plans. Links would have to be made with the databases of existing data recording packages.

PROPLAN 0.3

One functional limitation to PROPLAN 0.2 was that it still contained the simple approximation of transport costs originally implemented in VAX PROPLAN. PROPLAN 0.3, ready in summer 1990, contained only one improvement to the previous version: a more elaborate and versatile model of transport costs. Making an adequate, yet computationally acceptable transport cost model turned out to be quite difficult.

PROPLAN HAND 1.0

While working on PROPLAN 0.3 I had become aware that PROPLAN, however interactive, simple and user-friendly from the developers' point of view, was still too complex and esoteric for growers. Furthermore, the backtracking heuristic which was so central to PROPLAN was too rigid. It forced the user to work from left to right on the time axis, which was not always done in practice. In other words, it became apparent that the type of approach which had worked in the case of VLUCHT did not work in the case of pot plant production planning.

Reading literature (dealt with in chapter two) and attending conferences helped me relate this awareness to the experiences of other DSS designers. The realization that DSS development and DSS properties have to suit the situation induced me to engage in the literature study on which chapter three reports.

Not wishing to lose contact with the growers, I decided to suspend the work on PROPLAN 0.3 and focus instead on functions that did not use the backtracking heuristic.

The PROPLAN placement menu contained an item called HAND. This stood for an option in which the user was free to enter production lots himself, without any advice or constraint checking by PROPLAN. This option had not yet been implemented, being poorly compatible with the backtracking algorithm. I decided to develop the option HAND separately from the main PROPLAN software. This would enable rapid development of a prototype HAND option without being hindered by the bulk of the remainder of the PROPLAN software.

From its beginning in August 1990, PROPLAN HAND (HAND for short) was developed in close cooperation with two growers. Version 1.0 turned out to be an immediate success when it became available in October 1990. Its main property is its simplicity. HAND contains no intelligence, from a designers' point of view. It is no more than a means of quickly drawing and manipulating log plans. With a capacity for drawing log plans of 120 tables by 76 weeks it is sufficient for the needs of almost any grower.

Because its functionality is so limited and its interface so easily mastered and visual HAND hardly contains obstacles to practical use. In contrast to PROPLAN 0.2 it does not require recorded data as input. It immediately responds to any user action, and basic practical functionality such as a facility for printing plans is included.

5.4 Problem decomposition model

Before discussing the use in PROPLAN of the problem decomposition model presented in chapter two, let me make a few side remarks. With HAND many features of the previous PROPLAN versions were cast away.[68] This may, however, be only temporary, for PC-PROPLAN 0.2 and 0.3 contain many elements that may be returned to in the future. Moreover, features of PC-PROPLAN may be of interest to readers who are dealing with planning situations that have higher SCAPSIS scores than cultivation planning does.

The problem decomposition model which was presented in section 2.8 has evolved in parallel to the successive PROPLAN prototypes. From the beginning of the project it has been an explicit goal that the software fit into the planning routine of the grower. During the development of functional specifications the researchers became aware that there was no such thing as *the* planning routine of *the* grower. The goal now became that the software should fit reasonably well into the planning routines of most growers.

The researchers wanted to strike a balance between model power and usability. A way to pay attention to the latter was the problem decomposition model. The problem decomposition model has a number of properties which made it suitable as a vehicle to help design the planner - PROPLAN interaction. Perhaps the most important one is the explicit recognition of *context levels* in the model. This allows the system design to have clear concepts, variables and functionality within a context level while at the same time allowing a change of context. Secondly, some concepts in the model which play a role within a single context level are practical. These are notably the distinction between problem formulation and problem solving, the explicit allocation of these tasks to the planner or to PROPLAN, and the twin concepts of refinement versus relaxation when reformulating a problem.

The problem decomposition model was especially useful for assessing the completeness of PROPLAN's functionality and the clearness of its interface. Comparing PROPLAN 0.1 with the model showed that certain functional aspects were missing, and that in some menu options context levels were mixed in undesirable ways.[69]

---

[68]The acronym PROPLAN, if used without version number or other qualifier in the next sections, refers to PC-PROPLAN 0.2.

[69]The problem decomposition model not only served as a tool for the decomposition of the planning problem as a whole. Interestingly, it was found that a very similar representation could be used to describe the backtracking algorithm (Hofstede 1990[a], 1990[b]). The recursive diagram that resulted is given in fig 5.6.

5.5 Formal specification of the PROPLAN problem

Introduction

The essence of a system such as PROPLAN is that no attempt is made at exhaustively modelling the planning situation for which the system is intended. The model incorporated in the planning system is only a partial one, and the user is assumed to supplement a great deal of factual and contextual knowledge. The possibilities for human-computer interaction designed into the planning system are crucial in order to obtain a planner - computer system with optimal capabilities.

Yet, admitting that it is not the whole story, it is important to specify which part of the planning situation is modelled in the system and in which way. For the case of PC-PROPLAN the problem decomposition model is used to describe the view of the planning situation on which the design was based. The automated component of the planning situation is mainly situated at a single context level: the log plan. This context level will be described in detail.

Context levels

The production planning situation at pot plant nurseries can be decomposed in a number of ways, and indeed is treated in different manners by different growers, as we have seen in chapter three. The PC-PROPLAN decomposition is based on the decompositions presented in the reference information model studies (Beers et al. 1985, SITU 1989) supplemented by insights gathered by the PROPLAN team. The context levels distinguished are depicted in fig. 5.4 below.

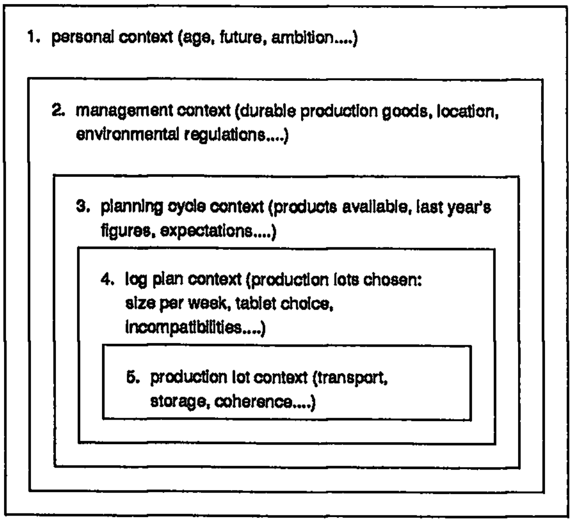


Figure 5.4: Context levels in PC-PROPLAN

The higher context levels overrule the lower ones. This means for example that it makes little sense for a grower whose greenhouse will have to make way for urban development (level 2) to spend much time on next year's log plan (level 4). Most growers have enough trouble to deal adequately with level 3 so that level 4 is less important. For example, they hardly know the relative rentability of their products (level 3) which has a higher priority than optimizing the log plan (level 4). PROPLAN is useful for those growers who have no major overruling problems at higher context levels. The automated part focuses on level 4.

Model of the log plan context

Problem representation
In the log plan context in PROPLAN 0.3 the pot plant placement problem is simplified in a number of ways compared to the real situation at nurseries. In PROPLAN a three-dimensional space has to be filled. The dimensions are space-X, space-Y and time. Each one of these is discrete and consist of identical units. The most detailed association is of one x, one y and one time unit, e.g. (3, 20, 9151). This represents one production location during one week. PROPLAN allows to define several greenhouse departments, but one placement plan is restricted to one department. The visual representation is such that the two space dimensions are collapsed upon one axis.

Operations
The space thus defined is to be filled with *production lots*. A production lot is defined by a product code, a quantity and a potting week. A product has some data associated with it. These are the *product-potting week* data, such as normative costs and profits, and the *cultivation scheme* data, which define the space requirement of a product over time dependent on the week of potting.

The building of *plans* consists of adding production lots to the partial plan. A plan has a number of constraints associated with it, even before any lots have been placed. These are maximum and minimum number of plants allowed to be potted per product, both for the plan as a whole and for each week.

Figure 5.5 gives the conceptual data model of the log plan context. There are some additions compared to VAX-PROPLAN. The most important of these are the following.

- The lot size is no longer fixed but it is a multiple of a user-defined minimum lot size.
- Time locks are replaced by 'weekminmax', i.e. flexible time locks. A weekminimum of zero for a product implies a time lock for that product in that particular week.
- Space locks are made to depend on product, even on cultivation phase of each cultivation scheme per product.

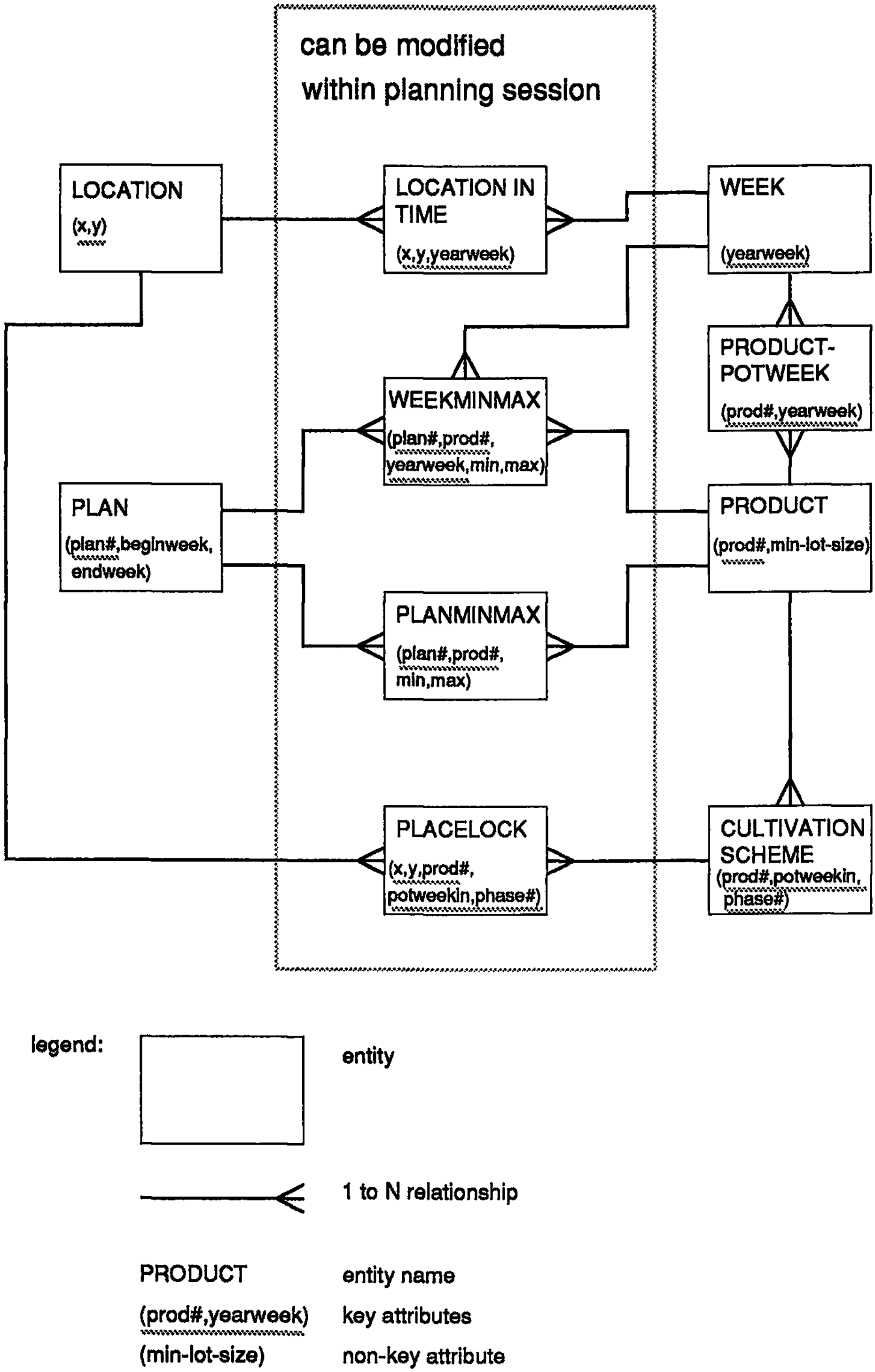


Figure 5.5: Conceptual data model of the log plan context in PROPLAN 0.3
N.B. only a selection of the non-key attributes is shown.

Control structure

The addition of production lots to the plan is controlled by a backtracking algorithm similar to the one in VAX-PROPLAN described above. The algorithm proceeds from left to right on the time axis. It selects candidate production lots, accepts or rejects them, prioritizes the accepted candidates according to a composite measure[70], and offers them in order of decreasing desirability. The control structure of the backtracking algorithm is depicted in figure 5.6.

Figure 5.6 shows that the algorithm is recursive: the entire figure is repeated within box 4. A closer look at the figure shows that the boxes after box 4 are only reached when the algorithm backtracks. As long as each successive candidate lot is placed, only steps 1 to 4 are taken. The index level n counts the number of lots that make up the current plan.

Let us suppose the algorithm starts from scratch, at level n=0.

Action 1: Formulate P_1, the problem at level 1. PROPLAN carries out a feasibility check based on the constraints in the conceptual model. For levels above 1, the extra constraints caused by the partial plan are taken into account. If the check succeeds PROPLAN searches for acceptable candidate lots for the current level n.

Choice 2: If one or more acceptable candidates have been found and not yet tried, proceed to 3. Else go to 8.

Action 3: Place the best remaining untried candidate in the log plan. This solves problem P_n, so that the algorithm can plunge one level deeper into the recursion (box 4).

Action 4: In fact this is a repetition of the entire figure 5.6, which one enters at action 1 for P_{n+1}.

Action 5: If at P_{n+1} box 9 is left at the 'No' side, i.e. if no lot could be placed, this means that the currently placed lot at level n is a blind alley and will therefore have to be removed as well.

Choice 6: After having removed the previous candidate PROPLAN checks whether any untried candidates remain for level n. This is the same choice as in box 2. If no candidates remain, one more lot will have to be removed.

Choice 7: This choice is dependent upon the mode in which the planner works. In the 'Step' menu option PROPLAN offers the user the opportunity to choose which candidate to place at each step. In the 'Automat' menu option PROPLAN proceeds without intervention by the user.[71] If Step is active, PROPLAN will have to regenerate all candidates, in order to allow a full choice. The Automat just takes the next best remaining candidate.

[70]This is referred to as the 'weighting figure' (in Dutch: weeggetal).

[71]An elaborate description of these menu options is given in the next section.

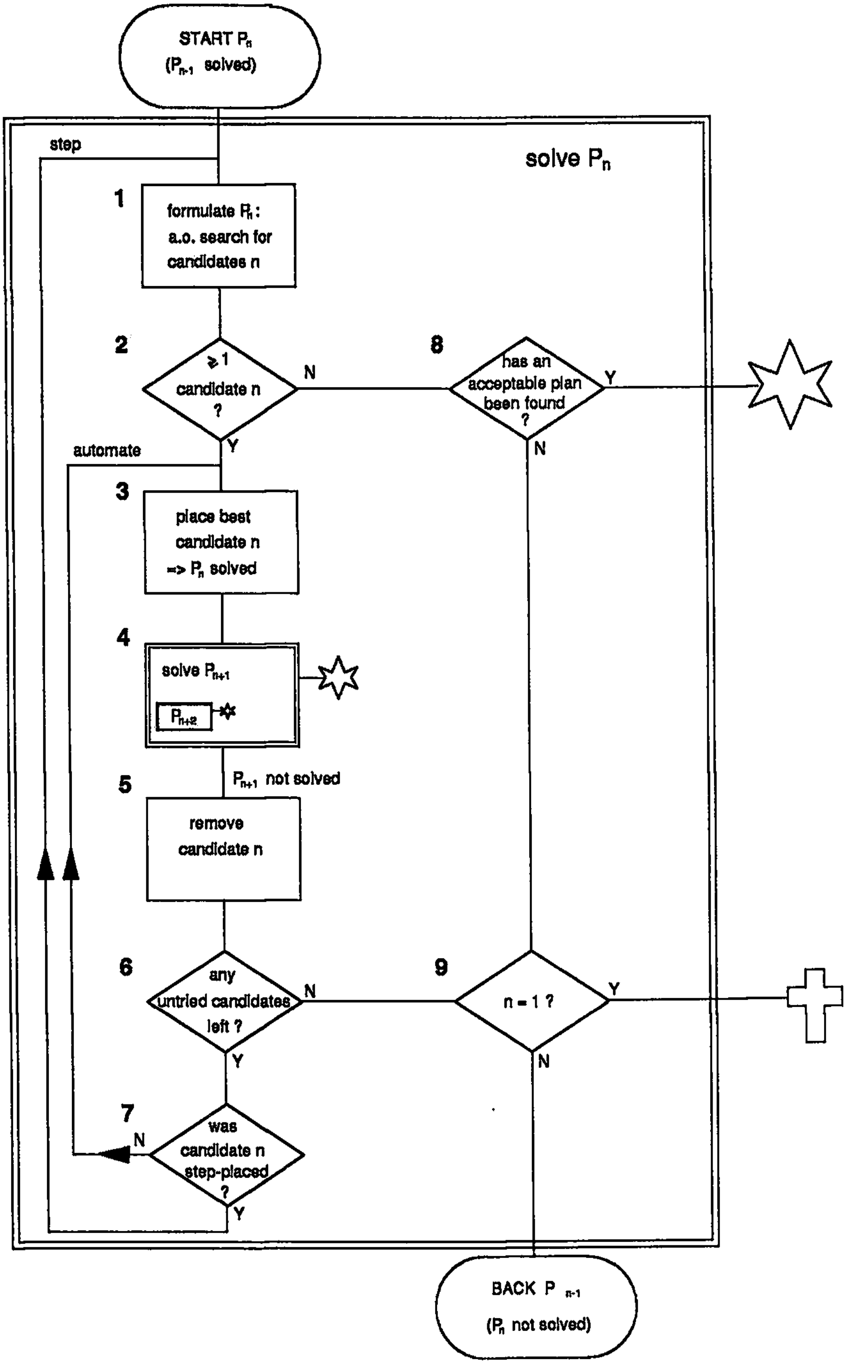


Figure 5.6: Control structure of the backtracking algorithm in PROPLAN 0.2 and 0.3

Choice 8: After failure to find a candidate PROPLAN checks whether the criteria for having attained a valid plan are met. If yes (usually at a high level n, since n represents the number of lots in the plan) then PROPLAN exits with success. If no, the last lot will have to be removed.

Choice 9: One lot has to be removed. If n=1, no lots remain in the plan and PROPLAN exits with failure.

5.6 PROPLAN 0.2 and 0.3 System description

PROPLAN 0.2 is described at length in its manual (Hofstede 1990^{e}).[72] It does not seem appropriate to go into technical details of either the software or the algorithms in the present volume.[73] Instead, I shall pay some attention to the interface, especially to the interface of the heuristics with the user.

PROPLAN 0.3 is identical to PROPLAN 0.2, except for the conceptualization of space and of the transport cost function. PROPLAN HAND has a much looser conceptualization and a slightly different interface, and since it is used as a separate program it will be described separately at the end of this section.

Menus

PROPLAN is menu-driven. All screens are alike, with a scroll bar of menu choices on the uppermost line, explications below the menu options, and working space on the rest of the screen. Keyboard operation is similar to popular software packages, in particular to PC-INGRES. The overall menu structure is shown in figure 5.7.

The database and planning parts are connected by an ASCII file interface, which is why they are depicted in separate areas in figure 5.7. The reason for using this ASCII interface is that the designers wanted the planning part to be easily connectable to different databases.

The PROPLAN user enters figure 5.7 at the left. In the database part, written in PC-INGRES, the user can maintain normative data or recorded data, or select data to be sent to the planning part. The data for the planning part include product data, department data, and data about the plan to be made, including data about space occupation by previous plans.[74] The planning part is accessed via the menu options 'make plan' (*maak afdplan*) or 'modify plan' (*wijzig afdplan*). It can also be directly started, provided the

---

[72]PROPLAN 0.2^{+}, the version described in the manual, contains a few minor changes to PROPLAN 0.2.

[73]As A. Lewandowski put it during a conference presentation in 1988: "Gentlemen do not talk about implementations".

[74]These data are used by PROPLAN to generate 'blocks', the squares marked 'B' in the log plan, as shown in figure 5.8.

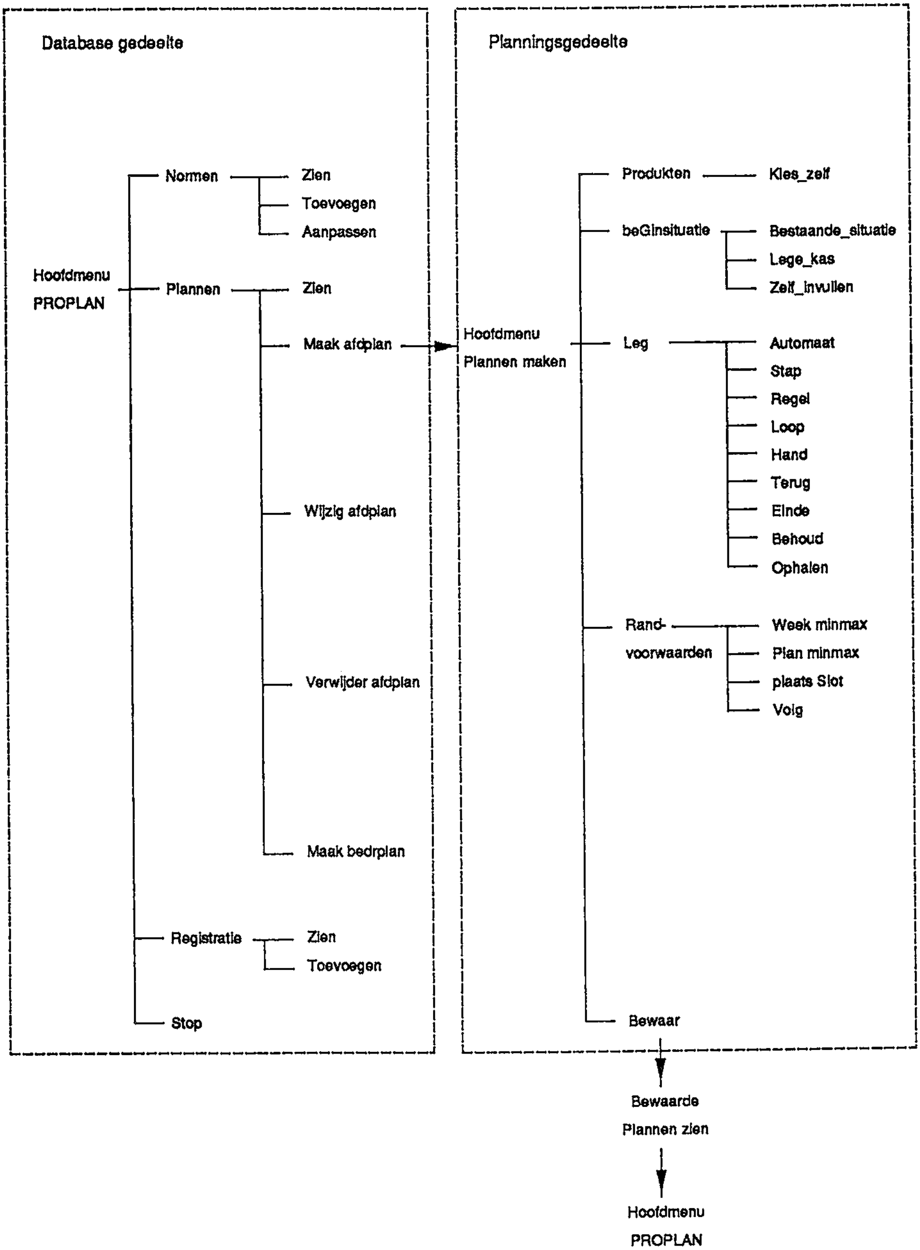


Figure 5.7: PROPLAN 0.2 and 0.3 overall menu structure.

necessary files were edited in some way beforehand. Thus PROPLAN can be used without INGRES.

The planning part consists of five menu options:

1. Products (*Produkten*)
   Choose products for the plan to be made. Optionally look at graphical displays of financial/economic product data per week.
2. Begin (*beGinsituatie*)
   View, define or modify the blocks on the plan to be made. Modifications are made by moving the cursor around the log plan screen. Blocks made here are only valid for the current plan, and they overrule the space constraints defined in the 'Constraints' sub-menu mentioned below.
3. Log plan (*Leg*)
   This is the main PROPLAN screen. It enables a variety of operations on the plan under construction, or on individual production lots. It is treated in more detail in the next subsection.
4. Constraints (*Randvoorwaarden*)
   This sub-menu enables the user to manipulate various constraints before starting a planning session. It is also discussed at more length below.
5. Save (*Bewaar*)
   Permanently save on disk the desired plans from among the plans preserved during the planning session.

Log plan screen

The central screen of PROPLAN is the *log plan screen*. This is the main PROPLAN screen. It enables a variety of operations on the plan under construction, or on individual production lots, or even on single table/week squares. The heuristics behind the operation are treated in the previous section (figure 5.6).

The log plan screen is shown in figure 5.8. The left side of the screen shows the log plan. Tables (numbered 10-200) in the chosen department are depicted vertically, weeks in the chosen planning period (week 43-29) are depicted horizontally. The table/week squares marked with a 'B' are 'blocked', i.e. they cannot be used in the current plan.[75]

The partial plan that is depicted in the picture consists of two production lots: a large Kalanchoe lot is potted at tables 40 and 160 in week 43, and a small Jaconia[76] lot at table 20 in week 45. Each product is indicated by its own colour. Production lots of one product are numbered upwards in order of placement, i.e. from left to right.

---

[75]In fact blocks are a refinement of the time and space locks of VAX-PROPLAN.

[76]Jaconia cafeata 'Diehard' is not an existing pot plant. The name is an homage to the programmer who built PC-PROPLAN 0.0, Jaco Wisse.

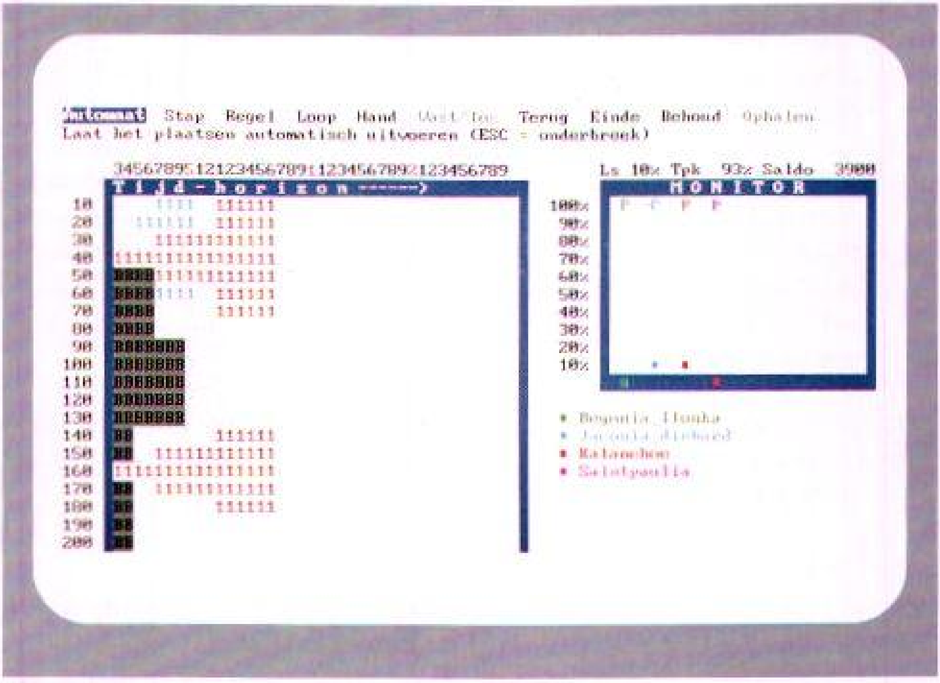


Figure 5.8: Log plan screen

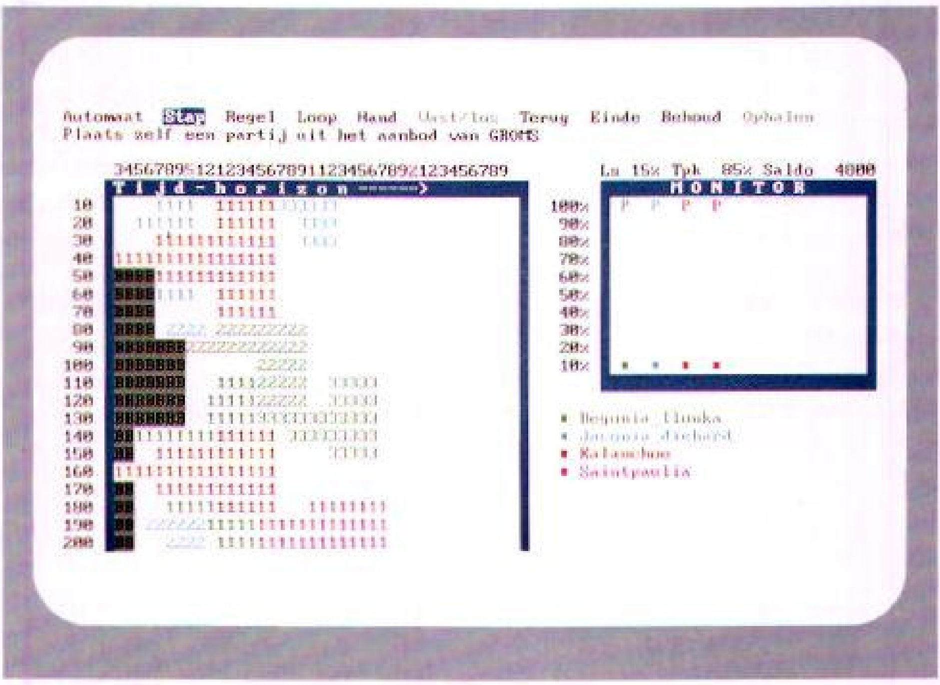


Figure 5.9: Step screen.

The right side of the screen shows a 'monitor'. Its top line displays three figures which give an indication of the quality of the current partial plan:

- 'Ls' is the average empty space so far, in figure 5.8 taken over weeks 43 and 44 only.
- 'Tpk' is the average 'transport quality' of the production lots placed so far. Transport quality is defined as: the costs of transport for spacing a production lot, divided by the same costs if the lot were ideally placed, i.e. with a minimum of transport. In the example (given that the tables are really adjacent in the order in which they appear on the vertical axis in fig. 5.8), the large Kalanchoe lot is ideally placed, but the small Jaconia lot is not. It is spaced in such a way that there is less transport to adjacent tables than would ideally be the case.
- 'Saldo' is the balance of the current plan in Dutch guilders.

Under this top line, the monitor gives a graphical indication of the percentage of the required or (in this case) allowed number of pots per product that has already been placed in the current plan.

Log plan sub-menu: Automat and Step

There are a number of menu options in the log plan screen. The first two are the 'Automat' and 'Step' options, both used for building a plan.

Automat (*Automaat*) is an option in which PROPLAN places production lots according to the backtracking heuristic, until one of three conditions occurs: failure, success, or interruption by the user. The user watches the plan take shape.

Step (*Stap*) works in the same way as the previous option, except that PROPLAN calculates only the candidate production lots for the next placement, displays them and allows the user to choose one of them, possibly after the user has adapted the size or manipulated the placement with the cursor keys. See figure 5.9. The monitor is replaced by a list of candidates, one for each product that can still be potted in the current week. One of the candidates is displayed in reverse video and shown on the log plan. The candidates are listed in order of decreasing weighting figure (*weeggetal*), and the elements of the weighting figure are displayed: 'Transp' for transport quality and 'Graagte' for the expected balance per week and per m^2 relative to its maximum during the plan period. The column 'Vf' gives the multiplication factor of the currently displayed candidate production lot relative to the minimum lot size of the product in question. By manipulating the keyboard, the user can have all the proposed products, lot sizes and placements displayed on the log plan before making a choice.

Log plan sub-menu: Regulate

Regulate (*Regel*), the 'dashboard screen', is a control panel allowing the user to regulate the operation of the heuristic (see figure 5.10). It can be accessed between placement of any two production lots. If the user modifies parameter values, the new values come into effect immediately and remain in effect until the user accesses the Regulate screen again, or until the end of the PROPLAN session, when they revert to their default values.

For each parameter, the screen displays three values. From left to right these are the current value (*Huidig*), the default value (*Standaard*), and the permitted range (*Interval*). This idea was taken from Verbraeck (orally, but see Verbraeck 1990[b]).

The parameters on the Regulate screen are grouped into four categories:

1. Constraints (*Eisen*)
   These are the maximum empty space for the current potting week and the minimum transport quality for the production lots at the next placement.
2. Weighting figure (*Weeggetal*)
   The weighting figure takes into account four elements: expected balance relative to maximum (*Graagte*), minimum number of plant that must be potted in the current week (*Min per week*), *idem* for the entire plan (*Min per plan*), and transport quality (*Transportkosten*). For each of these, a multiplication factor (*Weegcijfer*) ranging between 0 and 100 can be set in the Regulate screen. In this manner the user can exactly determine how much weight PROPLAN gives to each of the constituents of the weighting figure.
3. Lot size (*Partijgrootte*)
   A 'lot expansion factor' (*Uitbreidingscijfer*) can be set which determines how PROPLAN allocates space that is not reserved for 'minima per week', i.e. for mandatory production lots. The larger this number, the more PROPLAN enlarges the candidate lot with the highest weighting figure.
4. Search tree (*Zoekboom*)
   Three parameters of the backtracking algorithm's search tree can be set. On a personal computer with 640 Kb memory, these should not exceed the value of 5 to avoid memory shortage. The three parameters are the maximum number of products considered per step (*Max # kandidaten*), the maximum number of lot sizes per product per step (*Max # partijgroottes*) and the maximum number of different placements per lot size per product per step (*Max # configuraties*).

Log plan sub-menu: other options

- Walk (*Loop*)
  This option for moving the cursor has not been implemented.
- Hand (*Hand*)
  In the triad Automat-Step-Hand, this is the most interactive and least 'intelligent' option. It was implemented as a separate program treated in the next section.
- Fix/Loosen (*Vast/Los*)
  The aim of this option is to fix lots in the partial plan so that the backtracking

heuristic treats them as blocks, or to undo this operation. Due to its low priority, this option has not yet been implemented.

- Back (*Terug*)
  This option enables the user to force backtracking back to a production lot specified with the cursor keys.
- End (*Einde*)
  End the planning session and preserve the plan.
- Preserve (*Behoud*)
  Preserve the plan but do not end the planning session. Sixteen plans can be preserved during a single session.
- Retrieve (*Ophalen*)
  Retrieve one of the plans preserved during the current planning session.

Constraints menu

This menu enables the user to manipulate a variety of constraints. It can only be accessed at the beginning of a planning session. Constraints to be manipulated are

- Minimum and maximum number of pots that can be potted per week, or per plan, which are visualized on a graphical screen (Figure 5.11), screen Weekminmax. Besides the space occupation per product, screen Weekminmax shows the space occupation by blocks. The user can toggle between the Weekminmax screen and a screen in which minima and maxima can be edited.
- Place locks per combination of product, table and season. This is a very fine-grained way of taking into account constraints that are specific to a particular greenhouse. In particular, plants may require special treatment dependent on the season of cultivation, and such treatment may not be possible at all locations.

PROPLAN 0.3

PROPLAN 0.3 is the first version in which the greenhouse is modelled as a two-dimensional space rather than a single row of tables. The user needs to specify some spatial attributes of the greenhouse before starting PROPLAN. PROPLAN converts these attributes to a matrix of (x,y) coordinates. The unit sizes are the table dimensions.

The user also specifies a number of rules for transport costs. Dependent on product and pot size he specifies a value for the cost of transport in directions x and y. The transport cost for a given displacement of a production lot is then calculated according to a formula which takes into account initial cost of installing the transport medium, cost per displacement of the transport medium, plus distance-dependent transport cost per pot. Heuristic assumptions are made about the actual tables of origin and of destination during transport.

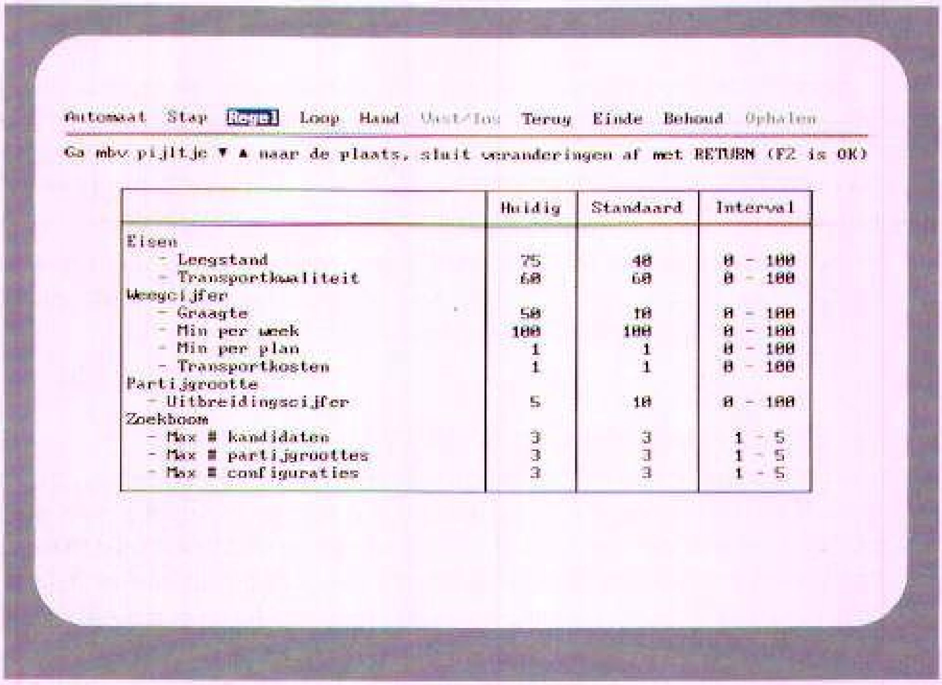


Figure 5.10: Regulate (*Regel*), the dashboard screen.

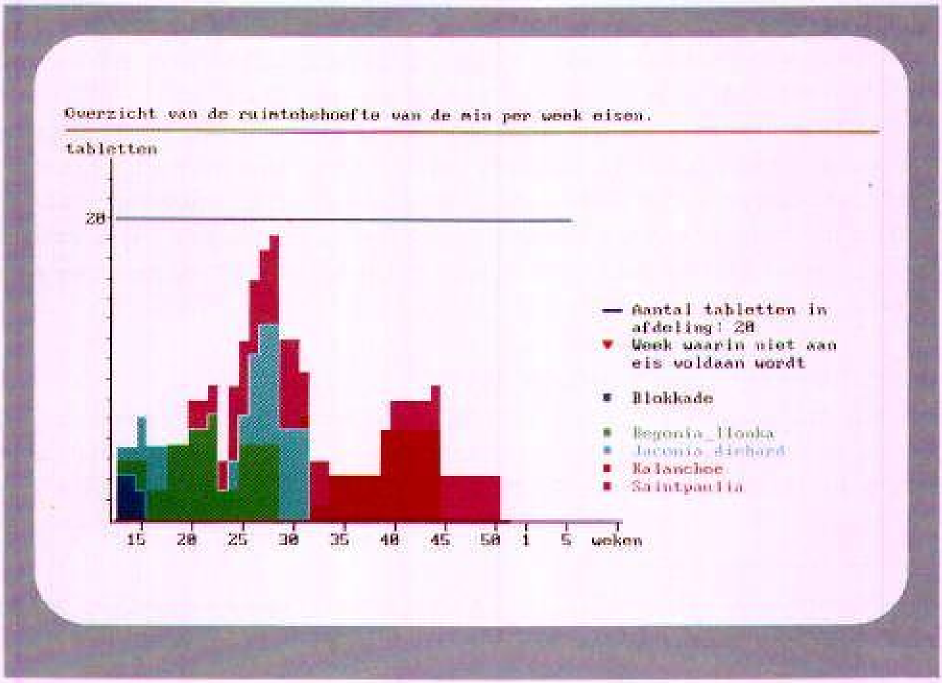


Figure 5.11: Weekminmax screen

5.7 PROPLAN HAND

The interface of the main PROPLAN menu was designed with special concern for consistency. In contrast to this, the main design guideline for HAND has been economy.[77] This meant that the user could get things done with a minimum of keystrokes, memorization, and functionality. It also meant economy of development. The designers wanted to keep the complexity of the software at a minimum level.

A pictorial impression of the main HAND screen, the log plan screen, is given in figure 5.12.[78] Two other screens enable the user to define the way in which HAND displays the floor plan of his greenhouse and the space occupation on the floor plan week by week.

Grower Q. and the development of HAND

The reader will remember grower Q. from chapter three. During the late eighties, grower Q. was satisfied with the degree of automation of the cultivation planning cycle at his nursery. Only one point had been bothering him for several years. Drawing and changing log plans manually is very time-consuming, which had caused him to leave off refining the plan as much as he would like, and sometimes to refrain from updating the plan.

HAND was developed according to the wishes of Q. and another grower. Q. is most pleased to have this automated tool for drawing log plans, which became available in October 1990. The main improvement over pen and paper is that he is now able to try a considerable number of alternatives in a short time. HAND does not include any calculations on the basis of the log plans. However, such calculations do not have priority for Q. because he knows his products so well that he decides about products, dates and lot sizes while working on the log plan, without any calculations.

Q.'s days are busy. On one ordinary afternoon spent with him, 15 interruptions occurred by personnel, by telephone calls and by visitors. Most interruptions required that Q. make some decision. A few took several minutes. This may go to show that Q.'s working day is that of a typical manager, as described in chapter three. He is involved in brief, fragmented, financially consequential person-to-person interaction during most of his time.

A session of Q. with PROPLAN HAND

I closely watched a session of Q. with HAND, after he had had a few days' previous experience, for thirty minutes. He was asked to think aloud. I kept silent most of the time. Data were not collected in a formal way.

---

[77]Hofstede (1990[d]) gives a discussion and some references on the pros and cons of user interface consistency.

[78]The reader is already familiar with the production lots depicted in figure 5.12. as they are the ones described in detail with figure 1.2 (p. 5).

The planning was done for three related departments. The first (6 tables) is quite small and is used for reproduction. Q. collects seed from mother plants and grows seedlings in this department. The second (18 large tables) is larger. It is used for the first potting stages. The third department (68 large tables) is much larger. Transport is by chariot, except for transport to an adjacent table. In the third department there are rail-suspended platforms to move the plants to the central alley, where they are further transported by chariot. Transport by chariot is more time-consuming than by platform.

The planning session started with a log plan version in which Q. had already made some alterations to last year's plan. There were still a few open spaces, mainly during summer. Late summer is a poor selling period, and early summer is a period of rapid growth. As a consequence, summer is a difficult season for planning.

Q.'s strategy was to try and gather as many bits and pieces of empty space as possible together in one place on the log plan, and then to see what could fit in the larger empty area (note that words such as 'place' and 'area' on a log plan correspond to space *over time* in the greenhouse). In order to generate larger, workable empty areas Q. tentatively moved parts of production lots around, altered their size, and sometimes altered the shape of a lot, e.g. deciding that part of a lot could be sold earlier (which implies that its cultivation scheme is changed) or stored in the expedition area.

Quite often a space conflict would arise for certain places on the log plan: a place would be needed that was already taken by an existing production lot. Q. would then make a note on a sheet of paper to remind him, since the current HAND version does not support double space reservations. The other aid he used was a calculator with which he computed the space occupation of a production lot in all its growth phases after altering the lot size.

Q. continually kept enlarging and reducing production lots. Although many of these changes were never carried through on the screen, several tens of changes were made during the session. Each change resulted in a new subproblem to be solved which generated new ideas. The planning session was a shifting puzzle. Q. was continually saying "I could...", "if I were to...", exploring a mental search tree that was not deeper than one or a few nodes but very highly branched.

Three reasons for making changes were most frequently mentioned by Q. The first of these was *increasing expected profit*. Making a lot too large would spoil the price. Many other arguments used by Q. could also be reduced to increasing expected profit. The mention of holiday period, for instance, referred to the fact that during summer holiday, sales are low. Likewise, insiders' arguments like "interesting", "no potential in this lot" boiled down to increasing expected profit. Put in general terms, Q had an intuition about the relation between season, quantity and expected profit for each of his products, and this intuition guided his problem solving.

The second reason was *constraint relaxation*. Q. avoids transporting plants unless there is a need to do so, such as spacing or selling. However, consider the following example. When some tables required by a lot were taken during the first two weeks, Q. considered placing the lot on nearby empty tables, leaving the pots in the trays in which they had stood during the previous period so that they would be easy to transport two

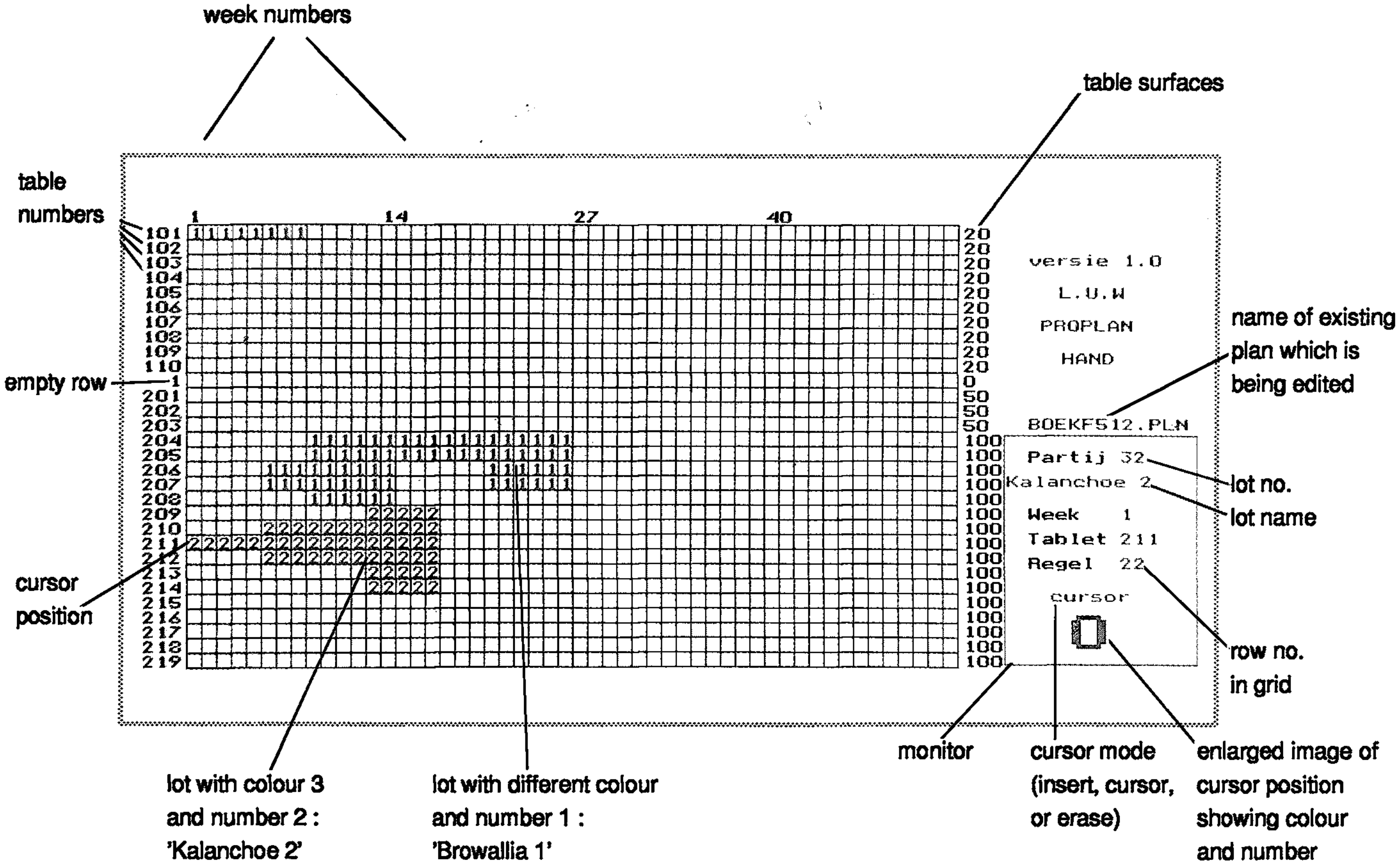


Figure 5.12: PROPLAN HAND version 1.0

weeks later. This would hardly affect quality. The net result is one extra transport step, and a change in cultivation scheme. With PROPLAN 0.2 such a relaxation of the cultivation scheme would not have been possible.

The third reason for making changes to a plan was *risk avoidance*. Q. preferred to disturb the placement or reduce the size of lots with doubtful profitability, rather than to interfere with the placement of lots with stable, predictable prices.

Thus Q.'s general problem solving strategy can be described in heuristic terms as hill-climbing to improve expected profit, starting from last year's plan, avoiding risky maxima, and flexibly relaxing constraints when appropriate. Q. obviously enjoyed the work.

5.8 Experiments with students

During 1988, PROPLAN 0.1, the first version with a 'real-life' model of the placement problem, was conceived. Simultaneously the researchers studied on ways in which to assess the quality of the PROPLAN 0.1 interface. The idea occurred to define quality metrics (Hofstede 1988[a]). What a further pursuing of this line of thought led to is described in chapter four of this volume. At the time I made some attempts at more exact quality measurements. A quality metric was developed on the basis of the problem decomposition model. This metric included a questionnaire in four sections according to the division by Jagodzinski and Clarke (see p. 112). The questions included a quantitative assessment of three of the aspects (see figure 5.13 below). An automatic logging facility was incorporated into PROPLAN. This facility logged most user actions, adding a time stamp: all menu options entered, all settings of the dashboard numbers, and key data of all saved plans. Structured tests with students were then organized. With hindsight it can be said that the tests have been important in furthering my thoughts on interactive planning systems and their quality.

During 1989, 1990 and 1991, students of an advanced planning course[79] were asked to spend an afternoon working with PROPLAN. Aims, procedure and results of the tests were as follows.

The 1989 test of PROPLAN 0.1

Aims

The goals formulated in advance for this first test were to:

1. Assess the PROPLAN 0.1 software and interface.
2. Try out the test procedure, viz. the laboratory setting and the management game context.
3. Assess the quality metric.

---

[79]This was the course 'Planning 2' offered by the department of Agricultural Economics of the Agricultural University.

4. Find out whether the automatic logging facility was useful and satisfactory.
5. Find out whether there was a difference between the quality of the plans made by each of the three groups of subjects: those who could use only Step, only Automat, and both modes.
6. Find out whether the plans made with PROPLAN by different planners showed homogeneity.

Procedure

There were nine subjects. The subjects were divided into three groups. Group A subjects were not allowed to use the 'Step' and 'Back' menu options. Group S subjects were not allowed to use the 'Automat' menu option. Group B subjects could freely use all options.

All the activities of all the groups with PROPLAN were logged automatically. The subjects were given an example nursery with all necessary data about products, including financial and cultivation data. They were asked to make a plan of as high a balance as they could. There was a prize for the best plan.

Results

Referring back to the list of goals, the results were:

1. PROPLAN was seen as promising but not yet mature. The use of keys was counter-intuitive at places. The system was considered complex. Some functionality was lacking: option Hand was not implemented, it was impossible to re-edit plans saved previously. On the whole, PROPLAN turned out to be well mastered within the hour and a half hands-on time that was available.
2. It was instructive for the PROPLAN team to have a group of non-initiates use the system. The students were quite enthusiastic by the end of the day. It was concluded that PROPLAN could possibly be further developed as a management game at some point in time.
3. It was concluded that the questionnaire could be abbreviated, since subjects did not respond to the questions exactly but made their remarks in seemingly haphazard fashion. They did not differentiate clearly between the four aspects of the quality metric. The conclusion drawn from this was that the quality metric would have to be reconsidered.
4. The automatic logging facility worked well. Some data processing might be added to it. The written session transcripts produced by the logging facility were useful. However, processing the data was quite time-consuming, and the time consumption was not proportionate to the result. Had PROPLAN been a riper product, or had the sample group of users been considerably larger, then automatic logging would have been more appropriate. Yet as an experiment the automatic logging facility was successful.
5. Not surprisingly, the Automat was easier to master than Step. Plan quality in the two modes was not different. It seemed that the Automat was more fit for attaining a high balance ('saldo'), whereas Step was more suited to attaining high transport quality. However, given the small sample no statistically valid conclusions could

be drawn.

6. Plans were fairly similar, but this could be attributed to the limited time which led to plans similar to the Automat default plan. The PROPLAN team scored considerably better than the subjects, which indicated that more learning than a few hours was needed to perform well.

The 1990 test of PROPLAN 0.2

Aims

1. Assess the PROPLAN 0.2 software and interface.
2. Test the usability of the manual.
3. Assess the quality metric, more specifically the revised questionnaire.

Procedure

As a consequence of the results of 1989 I decided to simplify the test somewhat. There were no subgroups this time, and there was no automatic logging. It was felt that the data derived from these two elements lacked validity, and that qualitative insights were more important than quantitative conclusions at this stage. The game context and the example nursery were the same as the previous year. The PROPLAN version was 0.2, so that re-editing previously saved plans was now possible. The subjects had to start from the main menu in INGRES and define the plan data themselves. An elaborate manual was available (Hofstede 1990[c]).

Results

1. To judge by the number and quality of the plans, the subjects attained significantly better mastery over PROPLAN than in the previous year. They were unanimously positive about the design idea of Automat and Step. At the same time they were more critical, presumably because they had worked more intensively than those in 1989 and had therefore experienced more limitations. In particular, the absence of menu item Hand was felt to be disturbing, the cut/paste facility in Step was considered time-consuming, there was felt to be insufficient economic information on the screen when judging a candidate lot in Step, and the relation between the parameters in Regulate and the behaviour of the backtracking heuristic was felt to be insufficiently clear.
2. The manual was not much used. More information in the manual about the tuning of the backtracking heuristic would have improved PROPLAN's usability.
3. The questionnaire could further be simplified. On the one hand, entirely open questions could be asked. On the other hand, quantitative ratings could be obtained for software quality, workability and usefulness in practice.

The 1991 test of PROPLAN HAND

In the year 1991 a third test was carried out. This time PROPLAN HAND was the package used by the students. At the time of the test, grower Q. already used HAND.

Aims

The main aim was to find out whether the students reacted to HAND differently than to PROPLAN 0.1 and 0.2. Being agricultural economists they could be expected to favour normative, quantitative support over the computationally empty functionality of HAND. Since they were not familiar with pot plant cultivation it was expected that they would need much time to acquaint themselves with the case, which was less narrowly defined than the previous years' ones.

Procedure

The day before the test, I gave an introductory talk in which the students saw slides of the cultivation of two pot plants, Begonia and Browallia cultivars.

For the test the subjects worked with a different nursery than those in the previous tests. There were three products: the two showed on the slides plus a Kalanchoe cultivar. They were given three pages of written data, containing a greenhouse lay-out and some realistic constraint data, as well as cultivation schemes, expected profits, rough space occupation per product throughout the year. Then they worked for two hours. Both before and after working they filled in a questionnaire which was similar to the 1990 version.

Results

Contrary to previous years, all students had some experience with personal computers. In the allotted time most arrived at a sketchy version of a plan. As expected, none got to fine-tuning the plan. The approach followed by the majority was to start with large production lots, to make a fairly regular scheme of those, and then to fill in the blanks. Two students followed a clearly different approach, concentrating on satisfying specific constraints rather than working largest first. No two plans looked the same.

aspect	1989 (n=9)	1990 (n=8)	1991 (n=7)
software q. avg.	7.6	6.50 (n=8)	6.83 (n=6)
stddev.	–	.76	.75
workability avg.	6.4	5.83 (n=6)	7.00 (n=6)
stddev.	–	1.47	.00
usefulness avg.	7.3	6.58 (n=6)	7.43 (n=7)
stddev.	–	.49	.53

Figure 5.13: 'School grades' showing the users' appreciation of PROPLAN aspects. The grades range from zero (utterly bad) to 10 (perfect). 'avg' is the average school grade, 'stddev' the standard deviation.[80]

[80]Standard deviations of the 1989 test are not known.

Against the pessimistic expectations all students estimated that HAND could be a useful tool for a grower. Arguments were "after all the grower has made the plan himself" and "saves time compared with manual planning".

The average quantitative assessments of software quality, workability and practical usability given in the three years are listed in figure 5.13. Having progressively had more previous experience with personal computers the student population grew stricter each successive year. This was compensated for in 1991 by the positive judgement of HAND, in particular concerning usefulness.

5.9 Field experiences

The PROPLAN team has had frequent contact with pot plant growers and research institutes throughout the project. There have been some contacts with extension services as well, mainly during functional design. Up to and including PROPLAN 0.2, however, the researchers did not work for any specific grower. The argument for this was that growers are so different that it did not seem useful to develop PROPLAN for specific growers. However, after PROPLAN 0.2 I adopted the opinion that developing for a few specific growers also had advantages. For the design of the transport function in PROPLAN 0.3 the team contacted four growers. PROPLAN HAND has explicitly been developed for two of these.

The number of growers who might be interested in a system like PROPLAN amounts to a few hundred. However, the team has ended up working for two growers instead, rather than attempting to satisfy the needs of a large group from the start. In view of the easy adoption by growers of PROPLAN HAND one might say that to work for all growers, as we had done earlier, had been to work for none at all.

There were two reasons to make a new start, and they will be given in the next subsections.

Type of support desired

The first reason for the switch is the type of support wanted by the growers. PROPLAN 0.2 was intended as a versatile tool which the user could adapt to his wishes. The possibility to choose between the Automat and Step option, the possibility to manipulate many constraints in a fine-grained way, and the possibility to choose and weight the criteria used by the backtracking heuristic, were all given with this aim in mind. Yet it turned out that growers still asked for more flexibility and less overhead cognitive load. They did not want to spend time on a system which looked complicated to them and which 'had a will of its own'. Instead, an 'obedient' and very simple system was desired. The growers did not need assistance to reach decisions of a higher normative quality. They needed support that would let *them* do the thinking, and the creating of plans, with a minimum of time-consuming encumbrances. What bothered them most about their non-automated planning was that it took so much time to modify plans.

Incidentally, when the system IPP, which was in a way an ancestor of PROPLAN, was introduced in practice, similar experiences were reported (Saedt 1988). These experiences induced the designers of IPP to add a highly interactive placement module to the optimization part of the system (Annevelink 1990[b]). This module enabled the user to edit a placement plan for an IPP-made cultivation plan.

Communication with the users
The experience with PROPLAN 0.2 made it clear that student tests and contact with growers at public sessions were inadequate as real test environments for usability. Even regular communication with growers at the level of functional specifications was not a guarantee for coming up with a usable system. In fact, too limited contact with growers was probably the main shortcoming of the 'surrogate environments' approach. It is good to note, though, that there was hardly an alternative to such an approach, since intensive contacts with growers were simply not possible during the first years. A grower cannot be expected to spend much time with unknown university persons who ask a lot of time but do not offer anything tangible in return. And indeed the researchers had nothing to offer as long as there was no PROPLAN version on a PC.

Intensive cooperation with a number of growers was actively sought as soon as the first PC version of PROPLAN became available. It was felt that only close cooperation with growers who had a personal interest in the system could guarantee that a new system would be better received than PROPLAN 0.2. Indeed, HAND was well received. In fact grower Q. expressed some surprise when he first became aware that HAND was what he had wanted.

5.10 Applying QUEST to PROPLAN HAND

It is interesting to use QUEST to estimate the quality of PROPLAN HAND. One cannot, however, use the resulting estimate to demonstrate the quality of HAND. After all, since QUEST and HAND issue from one and the same study, one could accuse me of having designed QUEST purposely in such a way that it could be used to show HAND's quality. Yet the exercise is useful, both as a demonstration of the use of QUEST and as an illustration of the essential features of HAND and of its development.

The nursery of grower Q., to whom the reader was introduced in chapter 3 and whom he has met again in earlier sections of the present chapter, will serve as the organization for which the assessment is made. The assessment is structured according to the elements of the quality chain.

Functioning of the organization
According to Q., the throughput time needed to make next year's plan has decreased from a few weeks to a few days. This frees valuable time, and it also makes modification of plans feasible.

Quite a different benefit of the system is that it acts as a status symbol. The log plan has been a central document in the organization for years, known to all employees.

Prominent (potential) customers frequently come to visit Q. to have a look at the nursery and the products. The colourful display of next year's log plan on a PC screen in the grower's office, visible from the passageway, certainly makes a trustworthy and professional impression on both visitors and personnel.

Decision
The efficiency of the planning process has enormously increased. Not much can be said so far about the quality of the plans in a normative sense.

Problem solving
HAND has allowed the grower to be much more creative during planning than he was able to be without the system. He is now able to engage in trial and error, thereby gaining a better insight in the planning situation, especially the spatial dimension. He enjoys the sense of play HAND gives him.

DSS
The data requirement is minimal. Even the few data about table numbers and surfaces, and about product names that HAND uses are not mandatory. This makes the threshold for use very low. The reverse of the coin is that HAND offers no model-based support at all.

The interface is quite familiar to Q. The use of the keyboard is straightforward. Q. was able to make most of next year's plan within the first day of system use.

DSS development
Q. knew none of the PROPLAN team members prior to the project, although he had had some contacts with students in 1986. From the outset, Q., his colleague and the team cooperated closely. The first goal was to develop a simple, manipulative interface and implement it in a usable system version. Activities stopped when this version (HAND 1.0) became available in October 1990. At the time, Q. was positive that he desired nothing more. In September 1991 he spontaneously asked whether any enhancements were forthcoming. Apparently, working with HAND 1.0 has given Q. a taste for more advanced support. The development of PROPLAN HAND has not come to an end yet.

5.11 Future developments

Obviously, the practical success of PROPLAN HAND seems something of a disillusion for the research into heuristics, of which HAND contains none. However, appearances may be misleading.

Firstly, the gap in acceptation between PROPLAN 0.2 and HAND is not that wide. One grower might have adopted PROPLAN 0.2. The grower desisted because he rightly distrusted the technical quality of the software and the continuity of software maintenance, and because a seemingly attractive alternative became available. This package, incidentally, focused on the economic aspects of the cultivation plan and did not include placement planning. The reason why the grower seriously considered investing in the first place in the uncertain adventure that PROPLAN was for him, is the interactive, manipulative design idea in PROPLAN.

Secondly, the expectation I had in 1990 that once PROPLAN HAND was used, a need for more support and intelligence on the part of the system would be felt and would increase, has come true in 1991. At the time of introduction, grower Q. explicitly denied that any more support was needed than what HAND 1.0 could offer. One year later he began to express an interest in further development.

Incidentally, whether and in which direction PROPLAN will indeed be further developed is partly dependent on matters outside the scope of the project itself. Initiatives are ongoing, both outside Wageningen University and at several of its departments.

6 Discussion

It leaves me uneasy, for I am aware of how many people
in our own culture have affective relations
with such mechanisms as automobiles, rolling mills
- and computers.

Herbert A. Simon (1965, p. 43)

This book deals with automated support for tactical planning situations. More specifically, it considers the applicability of interactive planning systems. The related questions of what such a system should look like, for which situations it could be used and how it should be developed are approached from different angles. To make this possible, a variety of literature was studied and a case study was carried out. This mixed deductive / inductive approach has resulted in a number of opinions, as well as in some applied theory and in two prototype interactive planning systems.

In this final chapter the answers to the research questions are first reiterated. Then the literature studies are discussed and the results combined, backing up my argument. In the next sections I summarize my various suggestions for improving the applicability of interactive planning systems, specifically the 'what', 'for which situations' and 'how' aspects. The last sections concern the case study, a discussion of research methods, the generality of the results, and suggestions for further research.

This chapter should be seen as a service to the reader who has read the previous chapters. Since these are diverse in their subject matter and rather lengthy, it seemed a good thing to do to reiterate the main thread of the argument more concisely. An advantage of this repetition is that material from different chapters is combined, so that the mutual relations are made more evident. However, many details of the argument are omitted, as well as references, illustrations and justifications. The conclusions are purposely strongly worded, without the nuances of the main text.

6.1 Research questions

What

The original research question of this book is: *"What should a DSS for tactical planning look like, if it is to be useful in actual practice?"*. The essence of the answer is that the problem is the planner's, not the DSS developer's. A planning problem cannot be snatched from the planner, formalized, and predefined into a DSS. It is an open problem which the planner can choose to alter at any moment. A DSS must reflect this fact; it must leave the planner free with respect to such things as the amount of time spent on planning, the level of normative support, the problem definition, the problem solving strategy, and the definition of constraints and of goals. In brief, the DSS must be modest. It should adapt itself to the way the user perceives the planning problem and to the way he wishes to solve it.

The question then becomes: "*Which functions make a DSS adaptable to a user's problem solving activities?*". Automatic model-based generation of plans is definitely not among these. What is important in all cases is to have basic functions in the DSS. Such functions are presentation, manipulation and administration of objects and plans. In these functions, the user interface is a crucial aspect of the system's usability. More advanced features that can be useful are constraint checking and evaluation. A model for the generation of plans may be usable in certain cases, most probably to obtain tentative solutions of subproblems. Simple functions should always be present, and advanced features should be optional if present.

The user solves his planning problem through various activities. Interaction with the DSS is one of these. Therefore the possibilities for user - system interaction have considerable impact on the usefulness of a DSS. Specifically one can ask: "*In which DSS functions is interaction desirable, and what type of interaction should be possible?*". The main property of good user - system interaction is that it supports, rather than interrupts, the user's flow of ideas. Interaction should not impose unnecessary mental load. The main requirement for the basic DSS functions is that the interface should be familiar to the user, so that system displays mean something to him. With more advanced functions there is the additional requirement that the DSS should make clear what steps it carries out. If the DSS can generate plans, the user should be able to intervene and overrule the DSS whenever he wishes, even if this means violating constraints specified in the DSS. The cooperation between user and DSS should be as close as possible. Ideally the user should be able to watch a plan grow in a familiar representation and to add, subtract or modify objects in a partial plan even while the model is running, without having to exit from the current model run or from the current display. Such possibilities require the development of highly interactive models, and rules out the use of 'monolithic' models with which the user can only interact by modifying the input.

A second question with regard to user - system interaction is: "*What steps in problem solving can be allocated to the user, and what steps to the system?*". The reader will have understood from the above that I have a radical point of view on this question. Anthonisse et al. (1988), who introduced the concept of an IPS (for interactive planning system) remark that (p. 415):

> Due to the practicality to perform intricate computations in real time and to display data and results in an informative way, it is a feasible idea to involve humans throughout the planning process.

It is my conviction that it is perfectly obvious that the planner should be involved, and that the question is whether and how to involve computers. After all, it is the user who solves his real-world planning problem, and whether the DSS solves a formal planning problem need not be relevant to the real-world problem. Incidentally, Anthonisse et al. in their article proceed to mention a number of functional requirements for IPS which are not unlike the ones I come up with.

My conclusions for the functional design of planning systems are that the user should be able to perform all problem solving steps, e.g. define the problem, decide on a strategy, design alternative solutions, choose a plan and modify the plan at a later point in time. By offering a manipulative, familiar interface the system can support these activities. After having acquired some experience with such a simple version, developer and user may decide on additional functionality.

Two weak points of humans as compared to computers are limited memory and inconsistency. These are the areas in which additional support by the DSS should first be sought. A third weak point of humans as compared to computers is constituted by the limited computing capabilities of the former, and this leads many a researcher to expect benefits from DSS-generated plans. However, in real-world settings the human deficiencies are compensated for by a much better sense of overall feasibility, combined with the possibility to change the problem if necessary. A model generating plans behaves like an *idiot savant*: it can do one trick impressively well, but since it has no awareness of context, the result is of no practical use. Therefore models for generating plans should be modest and allow the planner to know better.

For which situations

Part of the controversy about what DSS for tactical planning should look like and how prominent models should be in them is caused by the fact that many an author fails to note that he deals with different types of real-world problems. Authors take the initial translation of a real-world situation into a formal problem for granted and do not seem to observe that this step is in fact crucial and problematic. This is not so surprising, since it is considered more scientific to describe and compare formal problems than real-world situations. But if practical use of a DSS is intended the real-world situation should be taken as a basis, rather than a formal model. To say that a problem is NP-complete (e.g. Verbraeck 1991, p. 43) is not the same thing as to say that it is an open problem, because in the former statement a stable problem definition is assumed.

Now the 'when' question can be meaningfully put: "*For which planning situations is it appropriate to develop a DSS?*".

During the case study it became clear to me that to most growers cultivation planning was very much an open problem in practice. It could hardly be isolated from a great number of other problems and it was ridden with unpredictability. Growers had little time to spare for cultivation planning, and saw no point in trying to optimize the plan. The formal definition of the problem as adopted early in the case study was meaningless to most growers, and too rigid for the few who understood it. So in the pot plant case the maximum level of support was what above I have called 'basic functions':

presentation, manipulation, and administration, and even this only for sophisticated growers. More support may be desirable in the future, but only if the planning situations become more structured.

The pot plant cultivation case is worlds apart from the other two cases treated in chapter three: the KLM case described by Verbeek (1991), and the BARS case. Even this last case is a far cry from the ideal planning situation assumed by many formal models from OR and AI, without a history, with a known object system, with known input data, with deterministic processes and with known goals, and most importantly, without a problem owner who can change his mind.

Taking the extremes of 'total chaos' on the one hand and the 'ideal planning situation' on the other hand as two ends of a scale, I believe that most real-world planning situations are closer to the 'chaos' end of the scale, and that therefore advanced support such as plan generation is only feasible in a minority of situations. In planning situations that are intermediate on the scale, interactive modelling techniques like those described in this study may make it possible to offer usable advanced support.

How

The discussion in the previous paragraphs may be disappointing to some. Strong opinions are presented, but it remains fairly vague what a DSS developer should do in a particular situation. In other words, the question: *"How should a planning system be developed?"* must yet be addressed.

The answer is straightforward, and the key notion is *modesty*. The developer must realize that not he, but the planner is an expert on the planning situation. He must attempt to establish a good working relationship with the planner as soon as possible. If he does not succeed in this the project has fair chances to fail. Once there is commitment and cooperation, the developer should begin by designing an interface that is familiar to the user and easy to use. If possible, he should rapidly deliver a first system that contains only basic support on the basis of this interface and is usable enough to be adopted. Then, on the basis of experiences with this first version, the user may decide that he wants to have more advanced support. If no additional support is asked for, however elegant the models that the developer has in mind, it is no use offering any. The user is likely to feel, and rightly so, that by adopting a model the developer takes the user's problem from him. It may be possible to overcome this bottleneck by adding optional model-based support to the existing interface, so that the user can choose himself if and when to use this support.

6.2 Literature studies

To investigate the 'what' question I studied a body of literature on planning systems. The question 'in which situations' called for a scan of the literature on organization theory and psychology. The 'how' question demanded a study of the literature on quality assessment of DSS. An account of these literature studies combined with my opinions but stripped of all its frills is given in the subsections below.

Research schools on planning systems

I divided the literature on planning systems according to research school. Operations Research (OR), Artificial Intelligence (AI), Decision Support Systems (DSS) and Simulation were dealt with, of which the latter two only very briefly.

OR is especially suited for optimization. The OR literature shows a transition from case-oriented beginnings in the fifties to method orientation in later decennia. From the early eighties, OR authors have professed that the user of OR-based systems is important, but their commitment to that user often does not go beyond this remark. The user interface is frequently treated as a cosmetic element with which the operations researcher need not be concerned.

The AI literature, of which I consider only the symbolic reasoning school, offers techniques which allow for specification of more and of more diverse constraints than OR algorithms do. However, software size and performance of resulting systems are often problematic. As with OR, the predominant view is that the DSS should be intelligent and should solve the problem.

The DSS community is not homogeneous in its use of methods. OR, AI or simulation may be involved in DSS used for tactical planning. Attention to the user is much advocated in the DSS literature. However, often this attention merely takes the form of a call for more research into problem solving or decision making.

Simulation is a useful tool for sensitivity analysis or for 'playing around' with a modelled system. Unfortunately, it takes a lot of work to build a simulation model of a planning situation and validating a model of an organizational system is problematic by definition, so that the applicability of simulation is limited in practice.

The research schools mentioned have a few common characteristics. But for the odd author, all use formal models at an early stage during development. This implies that for all schools the planning problems considered are formal problems rather than open problems. This yields closed planning systems, i.e. systems in which problem formulations are explicit, fully defined, and rigid. As a result a planner is prevented from changing his problem once the problem formulation is given to the system. Secondly, all schools attempt to construct systems that model the planning situation as exhaustively as possible. In other words, they try to make the system intelligent. This may not be in accordance with the wish of a planner to do a creative job, and may hinder a planner from understanding plans and from being committed to them. In the third place all schools use their models in a strict way: the user is often not able to circumvent the models, notably the constraints. Such 'cheating'[81] is usually possible in practice and in non-automated planning. In the techniques used in OR, AI and Simulation, constraints

---

[81]A planner who deals with what he thinks of as an open problem using a planning system that treats it as a formal problem cannot but 'cheat' on the system. As Takkenberg put it into words in Dutch: "*Men zegt dat planners foezelen.*" But who is to blame?

are by comparison hard as nails, although they may be manipulatable. Fourthly, there is not very much literature about systems which are in actual use. All research schools have publication circuits in which research versions or prototypes dominate. This is usually satisfactory for researchers, but it does not necessarily take real-world planning any further.

To sum up, the research schools studied all consider a planning situation as a formal problem, i.e. something which can be isolated and can be formally modelled before it is converted into a planning system. The practical success of this approach cannot be easily assessed for lack of data about whether systems have or have not come into actual use, but it is felt by many a researcher, including myself, that practical impact could be stronger.

Practice-oriented literature on planning systems

The question of what factors contribute to actual use of planning systems developed by researchers was attacked by studying literature that focused on getting a planning system built for a particular case. The purpose was to find out whether there are properties common to planning systems that are in actual use in organizations. However, it turned out that not much of the literature is explicit about the actual use of the planning systems that are presented. Such issues as the communication between users and designers, or the process of introducing a planning system in an organization, or the way in which a system is used, are rarely mentioned in writing.

Upon reflection this is not so surprising. It is plausible that most planning systems developers make money either in a commercial environment, which means they have no time to spare for writing about what they design, nor any interest in informing competitors about their products, or in a scientific community, which means they must publish about the kind of things in which this community is interested. Researchers are more and more put under pressure to 'publish or perish', which does not help improve the quality of publications. In particular, attaining an impact in practice is usually time-consuming and therefore not 'publication-efficient'. Academic journals attach prime importance to internal consistency of a paper, as becomes them, but tend to be less insistent about the practical impact of the work.

Fortunately, there exist authors and editors who have both a scientific and a practical ambition. Besides, developers tell each other things over a drink which they never write down, notably about such subjects as the true reasons why their elegant system was not adopted. All in all I did get a picture, albeit a subjective one. The essence of it is that planning systems used in practice usually contain quite simple models. Their strength lies in aspects other than models. They often contain original features suited to the problem context. They emphasize adaptation rather than fixed planning, e.g. enabling a planner to take a previous plan as the basis for a new plan. The systems are highly interactive and they are integrated with other information systems. Generally the kind of design features mentioned here contributes to improved efficiency and fewer errors. The creative elements of planning, such as problem formulation, goal setting and dealing with constraints, are left to the planner.

Planning as decision making

Because I consider tactical planning as an instance of decision making in organizations I examined a diversity of literature on this wider area. In particular, descriptive rather than prescriptive literature was studied. Since planning is a prescriptive activity it is plausible that those intent on making a planning system might be insensitive to elements of a planning situation that do not fit into a prescriptive perspective.

Three perspectives were adopted in the literature study. The first is called *organizational*. It shows that decision makers generally prefer to manage by taking small steps and awaiting feedback, rather than to plan from scratch and be faced with unpredictable situations. Besides, no decision is left unaffected by the tangle of problems, opportunities, and personal interests in which people in organizations find themselves. Finally, by the nature of their work managers tend to be superficial and adaptive rather than thorough and reflective. Managers spend much of their time talking, and their working day is highly fragmented. They are men and women of action, not of thought.

To sum up, the image of the planner who sits back, concentrates on one single decision situation, collects the data to construct a search space, and searches for a solution within this search space is, to say the least, not likely to be applicable in the case of a planner who is also a manager.

The second perspective is a *psychological* one. It reveals that people are very far from being cool, rational decision makers. To begin with, people are subject to a number of cognitive biases. In the context of planning a bias that can be expected is the illusion of control. Mental effort, in particular the effort of using the imagination, can counteract cognitive biases. Thus, aids to the imagination would seem to be of help in supporting planners. Unfortunately, cognitive biases are highly augmented by excessive stress, and stress is usually present in a tactical planning situation. If nothing else, there is almost always time stress, and if the planner is responsible for results obtained on the basis of the plan additional stress is present. A planning system might help here by alleviating time stress. If a planning system contained a normative model, a planner could use this model to shift responsibility if criticized about the quality of the plan.

Another psychological finding is that people are strongly motivated to do well if they see a task as *play*. This is consistent with the finding that people like to be in decision situations that are neither completely strange nor entirely known. The implication for planning systems seems to be that it is not desirable to take away too many creative tasks from the planner, and that it would be good to have an element of play in the system.

Finally, psychologists have considered decision making from a more normative point of view, asking the question: "What is a rational decision?". It turns out that there are several kinds of rationality. In particular there is a distinction between rationality of the decision in itself, and rationality of the decision in the context of other activities: "Is it rational to spend time on the decision rather than on something else?". This synthetic/analytic rationality can be translated into an argument in favour of planning systems which allow 'quick-and-dirty' planning, perhaps alongside more elaborate planning.

The third perspective is a *systems* perspective. What most catches the eye when one adopts this perspective is that both a person and an organization are systems of a very high level of complexity. A planning situation being intricately linked with the organization, as well as with a particular person or team, one should be cautious to attach too much value to a model of this planning situation which is made with modelling techniques of a much lower level of complexity (as defined by Boulding (1956), see p. 84). Inevitably such a model misses elements of the planning situation, especially those which have to do with the properties of the higher levels of complexity, such as learning and communication. It takes a human, in this case the planner, to deal with matters not modelled in a planning system.

Another understanding provided by the systems perspective is that a system is not something given to a planner but it is something delineated by a planner, and that he may decide to change the boundaries at any time in any which way. This is a flexibility which models of a planning situation cannot match.

The evidence from studying decision making from these three perspectives is overwhelming. At the personal level, people do not attack decisions as isolated entities. Their prime motive is self-interest. They are apt to change their perception of a situation *ad hoc* when confronted with new information. At the organizational level, decisions interact and work is fragmented. Perceptions of those involved in a planning situation cannot but be filtered and biased. There is ambiguity and unpredictability, which means that economic rationality of decision making can at best be conditionally defined. Ultimately the 'rightness' of a decision cannot be quantified, but is a matter of human judgement.

A way to conditionally define economic rationality is to adopt a view on decision making as a series of nested context levels, in which the context of economically rational decision making is embedded in more important contexts -organizational and personal- in which economic rationality is not defined. Such a view is depicted in figure 3.2 (p. 94). In terms of this figure, improving decision making can be attempted at various context levels. It goes without saying that the decision maker should try and obtain a good decision quality at the level of the abstracted decision. Yet, higher context levels are more important.

If a planning system generates plans unassisted by a planner, the rationality behind these plans cannot be richer in context than the models in the system are. If the planner is in control of the interaction with the planning system, he can add his richer but also more whimsical understanding. For this to succeed the planner must be at least willing to work with the planning system, and should preferably be enthusiastic. If the planner does not care for the planning system, it will in all probability not be used effectively or not be used at all.

The organizational level should be taken into account. This means that the system should accommodate changes in the problem formulation, should be adapted to the setting in which it must be used, and should be integrated with related information systems.

The question remains whether these findings apply to tactical planning situations as well as to decision making in general. One could argue that the very essence of planning is to separate a planning problem from the rest of what goes on in an organization. This argument, however, cannot affect the psychological findings, nor the system perspective findings. Neither can it affect the nature of a manager's job. In general it is therefore not correct to assume that a planning situation is an isolated problem situation or that problem delineation and constraints are stable over time. This implies that a planning situation is not properly defined as a formal problem. At least for pot plant growers there is ample evidence that tactical planning is far from being an isolated activity. More generally, planning situations should be studied in their organizational context to find out to what extent it makes sense to consider them as entities that can be separately modelled at all. An attempt to do so, SCAPSIS, was presented in chapter three and shall be discussed later in the present chapter.

Combining bodies of literature

The literature on quality assessment of DSS which was studied to investigate the question: "What should a planning system look like?" turned out to be of limited value, because the scope of evaluation is usually very restricted. For my purposes, the two bodies of literature discussed above are more useful. After a discussion of these two very different bodies of literature it is worthwhile to find out to what extent the literature about planning systems confirms the findings from the organization-theoretical and psychological literature. If these findings are in accordance, then they can be regarded with some confidence, and can possibly be used for generating prescriptions about how to build better planning systems in the future.

The previous paragraphs have shown that the organization-theoretical literature and the planning system literature are indeed in line with each other. At the risk of oversimplifying, I shall recapitulate a number of elements which make this clear.

- Organization theory mentions that decision makers prefer to react to stimuli rather than start from scratch. Planning systems that are used in practice allow modification of previous plans.
- Organization theory stresses that decisions interact. Planning systems in organizations are usually integrated with other information systems.
- Organization theory mentions fragmentation and lack of time. Psychology adds that this can be stressful and therefore detrimental to judgement. Planning systems in organizations are in particular efficiency-oriented, i.e. they save time.
- Psychology stresses the importance of a good human-computer interface. Planning systems in organizations are manipulative and use simple concepts.
- Psychology mentions the advantages of a context of play. Planning systems in organizations allow the user to play, e.g. to manipulate previous plans or partial plans or to make changes in the problem formulation.
- The systems perspective states that formal models are in some respects fundamentally inadequate to describe systems that involve humans in organizations. Method-oriented planning system research is organized around the models used. Yet models play a modest role in planning systems that are in actual use.
- Soft Systems Methodology employs a rich but not very formal set of concepts to

model systems that involve humans. In fact SSM includes the organizational and psychological perspectives in a systems view. Rationality is defined in terms of relationship maintaining. Such a systems view adds a dimension to 'hard' systems views. For instance, the success of many an OR-based DSS project[82] can be explained quite satisfactorily not by referring to the normative quality of the models used but by proposing that because the models were accepted by those involved in the development project, they acted as a successful shared basis for appreciation and action.

There is one single point which recapitulates many elements of the list just given, and which is the reason why formal models will never play more than a modest role in the majority of planning systems that are in actual use. This point bears upon the stability of a planning problem. Organization theory makes it clear that planning problems are open, i.e. they have no fixed delineation. This is in particular true of scheduling- and timetabling-like problems. In system language: the perspective of the planner determines how he defines the boundaries of the system, and these boundaries are not stable. During the fragmented course of time which the planner spends on making a plan, new data arrive, predictions change, 'things happen' or the planner is struck by an idea. During the next planning cycle, a 'changed' planner is at work in a changed world. In brief, there are many things which make the problem change continually. During execution and monitoring of the plan these perturbances are even more pronounced, and partial re-planning is bound to occur in a majority of cases.

In strong contrast with reality, the model-centred approach to building planning systems requires that problem boundaries be stable. It requires that all data about the problem be collected before attacking it. It takes away from the planner the means to change or dissolve, as Ackoff would say, the problem rather than solve it. To conclude, it is obvious that such a model-centred approach to building planning systems brings with it a high risk of producing a planning system which does not fit the practices of the planner and which is therefore not accepted by him. Although the normative quality of the plans produced by the system may be high, this is to no avail if the system is not used.

6.3 Improving the 'what' of planning systems

Interactive heuristics

In the discussion about planning systems and automated support in chapter two the point is made that from a user's point of view, heuristics are among the strongest techniques found in OR or AI. Heuristics need not be complicated. They allow both 'quick-and-dirty' and elaborate use. They can cope with a considerable amount and diversity of constraints.

In the context of an open planning problem the user should be free to change the problem during planning. This freedom should involve both the constraints and the goal

---

[82]The BARS case in chapter three (p. 101) is a case in point.

function if one is present. Therefore, heuristics in an interactive planning system ought to be interactive, so that the user can complement or correct the heuristic while it is at work. As heuristics iterate in small steps, making them interactive need not be difficult provided the steps are understandable to the planner.

Heuristics are treated at some length in chapter two. A heuristic consists of three elements: a representation of the problem and of (partial) solutions as objects, operators that manipulate these objects, and a control strategy that determines the order in which the operators are invoked.

In chapter two it is discussed what requirements these elements should meet in interactive heuristics. It turns out to be essential that the objects used for describing the problem, as well as the operators, be meaningful to the planner. If this is not the case it will be very hard for the planner to exert his domain expertise, because he will then not be able to relate his expertise to what the planning system presents to him. The requirement of a meaningful interface implies that the development and tuning of an interactive heuristic is only worthwhile after a planning interface has been developed that the planner accepts.

Another important requirement is that the problem representation be not too specific. Aspects of the planning problem that are open, i.e. subject to change, unpredictable, or ambiguous, should either not be modelled at all or they should be modelled in such a way that the planner can manipulate them between any two iterations of the heuristic. In this way the planner can decide what to do with such aspects during planning. He can then change his mind several times, or even deal with a constraint implicitly, without having to interrupt his train of thought by leaving the planning interface to edit the problem specification.

A third requirement is that the planner be able to choose the level of support he wishes, starting with a basic level of no normative support. This property is desirable for several reasons: it allows the planner to work in a 'quick-and-dirty' way if necessary, to get a feeling for the system, and to avoid a feeling of loss of control over the planning situation. For the developer it is a way to offer more advanced support unobtrusively. Chapter five recounts how interactive heuristics were used in the PROPLAN project.

Problem decomposition model

Decomposing a problem into subproblems is a powerful help in problem solving, and it is used by every person working on a task of some complexity. Therefore it seems natural in the design of an interactive planning system to adopt the problem decomposition used by the planner and to duplicate it in the planning system. Unfortunately, the communication between subproblems changes when one automates them. A planner without automated aids can switch freely between subproblems and can implicitly or explicitly take into account the constraints of one subproblem while working on another. He can decide *ad hoc* about strategy (i.e. when to switch between subproblems) and about constraint propagation between subproblems.

By contrast, communication between automated subproblems has to be predefined, and is therefore much less flexible. Although there are also obvious advantages, such as error reduction by automatic constraint propagation between subproblems, this loss of flexibility should be taken seriously. The way to overcome it is to give the planner the chance to intervene in two ways when switching from one subproblem to another: he should be able to choose the strategy and he should be able to alter constraints.

In chapter two, a model for hierarchical problem decomposition is described that pays special attention to switching between subproblems. Each subproblem has its own problem formulation and variables. The overall problem is pictured as a recursive hierarchy of subproblems. The problem is solved by first formulating the overall problem, then translating this formulation into one or more problem formulations at the next lower level, etc. until the lowermost level is reached. There, the problem is solved - which in our case means that matches are made between the objects to be planned. After this, the hierarchy is ascended again, this time translating the solution of each subproblem into terms of the next higher level. It usually takes a great many solved subproblems to solve the overall problem, both because, obviously, the subproblems are smaller in scope and because evaluation at a higher level often requires comparison of a number of solutions at the next lower level.

This problem decomposition model has some attractive properties as a guideline for the design of interactive planning systems. Notably, it combines the possibility of describing subproblems as a state space (upon which OR or AI models can act) with the notion of nested context levels, each with their own relevant variables. Also, it promotes consistency in designing a planning system. However, there is as yet but limited experience with its use. In chapter five it is described how the model was used during the development of PROPLAN.

6.4 Building planning systems that fit the situation

Organizations differ widely, and so do tactical planning situations. Above, the conclusion was reached that model-based design of planning systems carries some dangers. But model-based design, resulting in mathematically sophisticated planning systems, has merits that make it suitable in certain cases. The limited success of the approach in practice may simply be due to its use in situations in which it was not appropriate.

In chapter three, a quick-scan tool is described which can be used to characterize tactical planning situations and relate this characterization to possibilities for developing planning systems. The tool's name is SCAPSIS: SCanning Aid for Planning SItuationS. The quick-scan consists of asking eight questions, four of which have bearing upon the context of the planning situation and four of which upon the object system, i.e. the system which is the object of the plan. The questions are the following.

Context characteristics

1 How large is the organization?
2 Is there a specialized planner?
3 How stable is the environment?
4 How often does the planning activity occur?

Object system characteristics

5 How predictable is the object system?

6 Are people unaffected?

7 How controllable is the object system?

8 Are reliable data available?

For each question, a five-point scale is devised and it is indicated how the real-world situation relates to the score on each scale. If a planning situation scores 1 on all questions this means that no automated support is possible. A score of 5 on all questions means that sophisticated support is in principle possible. This is so because a planning situation with scores of 5 throughout can reasonably be viewed as the kind of formal, combinatorial problem on which OR or AI models operate, whereas low scores indicate that the combinatorial aspect of the planning situation is subordinate to 'messy' aspects so that the situation is more properly viewed as an open problem. Intermediate and/or mixed scores, which is what one finds in practice, can be used as a basis for discussing the possibilities for automation.

SCAPSIS has a purely tentative status so far. It has only been used on the examples in chapters three and five of this book. There is no evidence so far about the predictive value of SCAPSIS. Yet, from an organization-theoretical point of view the basis of the tool seems valid. In the hands of planning system consultants or developers, the tool has the potential to grow into a useful instrument. If nothing more, it is at least a useful checklist to use at the outset of a planning system project. It allows developers and intended users to reach an appreciation of the situation. A particularly interesting outcome would be that a diagnosis with SCAPSIS revealed that one had better change certain aspects of the planning situation rather than build a system to support it.

6.5 Improving the 'how' of planning systems

Planning Systems as process and product

The idea behind SCAPSIS is that there is, at least in the imagination, a continuum of planning situations ranging from well-delineated ones which contain no uncertainty or ambiguity, to 'messy' situations which are beset with ambiguity, interfering problems, unpredictability, etc. In practice, almost all planning situations are somewhere in between these extremes. The literature reviewed in chapters two and three suggests that they tend to be close to the 'messy' end of the continuum. Granted that model-based design processes and products are suitable for the well-defined end of the scale, what type of design processes and products are needed for the numerous planning situations that are situated towards the 'messy' end of the scale? In chapter two, some elements of interactive planning systems are suggested. Based on the literature on decision making in organizations, it is concluded in this chapter that for an average planning situation a highly interactive planning system is called for.

An essential property of an interactive planning system as I define it is that design starts with the development of an interface. Designer and planner work closely together, sticking to concepts familiar to the latter, to produce this interface. The interface, used without any 'intelligent' model-based system components, yields a first working prototype. It is only then that a decision is taken as to whether and how to add model-based functionality to the system.

This definition of an interactive planning system intricately links the design process to the end product of this process. This calls for a development process quite unlike model-oriented approaches (such as Mitroff's model shown in figure 4.4) where the designer attempts to construct a valid and complete model of the planning situation early in the design process and where 'user-friendly' interfaces are added later.

In fact, since the very existence of a tactical planning situation presupposes that there also exists a planner with domain expertise, there is no need to exhaustively model domain expertise into the planning system, provided the planner can interactively introduce his domain expertise while working with the system. So to begin with, the system need not be 'intelligent'. Incidentally, an important exception must be made for systems that are developed with a special purpose, e.g. as training aids to novice planners.

The first kind of added functionality that is called for is simple support in constraint checking: the system can perform cross-checks and can perform counts to help signal possible errors or omissions.

In planning situations that are close to the well-defined end of the messiness continuum, one might proceed to add more functionality to the interface, for instance models that suggest (partial) plans. The ideas which are put forward in chapter two could be of use here. As a consequence, cognitive sciences can be expected to play an increasing role in research about DSS design. For this to happen a better integration of these sciences with OR and AI is desirable.

QUEST for quality

Quality assessment of planning systems is the main subject of chapter four. In fact the chapter is about quality of DSS in general, according to the assumption which is made in chapter three that planning is an instance of decision making in organizations. Thus quality is defined using the descriptive stance to decision making adopted in chapter three. More precisely, a 'quality chain' (figure 5.1) is extracted from chapter three (figure 3.3). This chain contains five links: functioning of the organization, decisions, problem solving, DSS, DSS development. These five elements encompass what is important in DSS quality assessment; they cover the time span from the initial development of a system to the ultimate effects of its use on the functioning of the organization.

It turns out that literature on quality of information systems is rather scarce, and much of it focuses on aspects of the 'DSS' link in the quality chain only, e.g. model quality or software quality. A few articles were found which came close to what I was looking for. Using these, and supplementing them with elements taken from my points of view about decision making, I put up a checklist intended to assess the quality of planning systems, and called it QUEST: QUality ESTimator. This quality assessment tool is a very open tool which, to borrow a DSS adage, supports rather than replaces quality assessment. Its merit is that it is rich in diverse aspects, so that less directly tangible benefits of a DSS may become apparent. Both the 'what' and the 'how' of DSS are included. For instance, even if no system in actual use results from a project, the people involved in the project may have learned so many things which enable them to make good decisions in other information system design projects that the project need not be seen as a failure. A consequence of the open character of QUEST is that no unambiguous quality measure results. Its status so far is tentative. It is used on a KLM manpower case in chapter four and on HAND in chapter five.

6.6 PROPLAN, the case study

The case study of developing PROPLAN, PROtotype PLANning system for cultivation planning at pot plant nurseries, has played a very important role throughout this study. It prompted the central research question, and it forced me to acknowledge that in pot plant nurseries, cultivation planning is not properly considered as a formal combinatorial problem.

When the PROPLAN project started in 1986, the aim of the study was to develop a prototype planning system on the basis of interactive heuristics. When I put together a research proposal for a doctoral study in 1988 I was just beginning to get the uneasy feeling that the definition of a planning situation as a formal combinatorial problem could not always be taken for granted. I started reading about planning systems then, and found little to put an end to my uneasiness, which induced me to read more.

Without the case study, the literature studies and theory generation in this volume would not nearly have been as productive as they are. With some malice one might say that the case study provided the bias from which the theoretical part of the research was conducted. The PROPLAN project also yielded some interesting findings in its own right, and these will now be discussed.

Prototyping in surrogate environments

To begin with, there is the design approach of *surrogate environments*. This approach was chosen by John L. Simons in 1986 because the interactive heuristic techniques that were envisaged for PROPLAN were so innovative, and the pot plant placement problem was so complex, that it did not seem wise to try and model the full real-life problem at once using these techniques. The surrogate environment approach is a prototyping approach which works as follows. First, a very simple surrogate environment, in which the problem is stripped to its bare essentials, is modelled and made tractable. This simple environment is the first in a series of surrogate environments of increasing complexity. As soon as one of these turns out to be intractable, a different modelling approach has to be chosen. If a surrogate problem can satisfactorily be dealt with it can be replaced by one with a greater number of realistic elements. There are two hierarchies through which to move: a hierarchy of sophistication of the techniques with which a certain surrogate environment is attacked, and a hierarchy of richness in realistic elements of the surrogate environment.

Originally the idea was that this approach would finally lead to a surrogate environment that was so rich in realistic elements that it could be termed a model of the real problem situation. The modelling techniques used would be such as were suitable for the problem in question.

With PROPLAN, the development so far has been different. Although the surrogate environment approach allowed for the development of a system (PROPLAN 0.2) which was fairly attractive to growers and contained the kind of interactive, manipulative heuristics that were intended, this system was not adopted by growers. The surrogate environment chain was then interrupted to make a system (PROPLAN HAND) which was much simpler in the model sophistication hierarchy.

An important reason to start anew rather than add HAND to PROPLAN 0.2 - which step had been intended in the functional specification of PROPLAN 0.1 and upwards - was that the PROPLAN software had become unwieldy in size, and that there was no programming capacity to cope with this. Another reason was that there were some rigid elements in the design of PROPLAN which impeded the freedom of the user. In particular, PROPLAN worked with fixed cultivation schemes, whereas growers use cultivation schemes as soft constraints. Also, the central backtracking heuristic imposed an order of placement which was not fit for all cases. To put it in other words, the chronological order of placement that had appeared to the designers to be an essential element of the planning problem, and was therefore preserved even in the first surrogate environments, turned out to be of little relevance to the users.

Concluding the discussion of prototyping by surrogate environments, it can be said that the approach is very suitable for experimenting with modelling techniques, but that one cannot count on arriving at a usable system through this approach. With hindsight this is not surprising, since the surrogate environments are not a good medium for communicating with the future users. Yet, it is quite conceivable that one could combine the advantages of user-centredness and surrogate environments. One could start by developing an interface, together with the user, and a first DSS containing only this interface. This procedure would guarantee sufficient richness in realistic elements in the system. Only then one could engage in a succession of surrogate environments of increasing model sophistication on the basis of this interface.

Interactive heuristics

The central screen of PROPLAN 0.2 and 0.3 is the log plan screen. The user has three options for making plans:

- Hand, which behaves like pencil and paper, i.e. it neither imposes nor checks constraints. Option Hand has been developed as a separate module, PROPLAN HAND (fig. 5.12).
- Automat (figure 5.8), in which the system proposes plans. The system works on the basis of a backtracking heuristic of which a number of parameters can be interactively modified during its operation. The heuristic operates in steps of one production lot at a time. The user watches and can interrupt the automat at any point in time.

- Step (figure 5.9), which uses the same heuristic as Automat but proceeds one production lot at a time and has the user choose which candidate, if any, to include at each step. Step also allows the user to edit the size and placement of the proposed candidates.

The backtracking heuristic can be manipulated in a number of ways. There is a dedicated screen, the dashboard screen (figure 5.10), to do this. This screen allows the user to modify the criteria for rejection of candidates, those for prioritizing among accepted candidates, those for establishing lot size if there is space left, and those for determining the branching degree of the search tree. Before starting a planning session, the grower can edit all other data, including detailed space and time constraints.

In terms of management variables, the dashboard screen combined with the automat allow the grower to make a space plan on the basis of anything ranging from a strict potting plan to no constraint whatsoever about product choice. It allows him to choose empty space, transport costs, and/or expected profit as elements of the goal function. By using Step he can include criteria that are not expressed in PROPLAN. Unlike PROPLAN 0.2 and 0.3, PROPLAN HAND can do without any input data on cultivation schemes and financial aspects.

Experiments with students

To supplement the casual information that the PROPLAN team received from users at demonstrations and at nurseries, structured tests with Agricultural Economy students were held in 1989, 1990 and 1991. The students' appreciation of PROPLAN on each of these occasions is displayed in figure 5.13.

In 1989, PROPLAN 0.1 was even supplemented with an automatic logging facility for this test. In itself, automatic logging of users' actions turned out to be a valuable aid. However, it should not be used in a package which is yet immature. In the case of PROPLAN 0.1 there was still so much to change in the software that automatic logging was unnecessarily elaborate.

In 1990, PROPLAN 0.2 was tested. The added functionality did not result in better perceived quality. The main reason for this was that the users had to go through many administrative actions and some waiting before they were able to start a planning session. In 1989 they directly started in the log plan screen. Whereas the plans made in 1990 were significantly better than those made in 1989, the 1990 users were stricter than the 1989 users in their evaluation of PROPLAN.

In 1991, PROPLAN HAND was the test object. Because HAND contains no normative data the students were given a much more open planning task than with the previous tests. Against my expectations, HAND was much appreciated, both for its workability and for its usefulness. To motivate their appreciation of HAND's usefulness, students used arguments such as "If I were a grower I would have confidence in the plans I made with HAND because I would know that I had made them myself".

During the three years in which a test was organized there has been a marked increase in PC experience among the participating students. This can account partly for the tendency to judge software quality more strictly year by year.

Future

As far as can be foreseen there will be two developments around PROPLAN during the years to come. The first of these takes place outside the University. In 1990, a project 'Management Control Pot Plants' was initiated by a number of parties, including extension services, NTS (growers' association), SITU (foundation for automation in horticulture) and PBN (research station for floriculture). As one of the spin-offs of this project, PBN and SITU propose to develop PROPLAN HAND further.

The second development takes place within the Agricultural University. A new research program is currently being defined. This successor of 'DSS in agriculture and horticulture' will focus on 'Tactical and Operational Management Information Systems (TOMIS) at pot plant nurseries and it will be much more strongly coordinated than its predecessor. At least six departments will work on a prototype information system together: Agricultural Economics, Agrotechnics and -physics, Computer Science, Horticulture, Management Theory and Operations Research, and possibly Extension Science. PROPLAN HAND will act as one of the starting points for developing the interface of the information system. Furthermore, the design approach and elements proposed in this volume will be put into practice during the development of the system.

6.7 Discussion of methods

In chapter one the three main research methods used in this study were mentioned. They were: literature investigation, generation of applied theory and a case study. To what extent and in what ways have these methods contributed to the study, and should other methods have been used? In order to answer these questions it is useful to begin by giving a sketch of the research methods that are commonly used in information systems research. In the world of information systems research, extensive methodological discussions take place. This is understandable in a new field with contributions from disciplines with quite diverse research traditions. With deliberate exaggeration, one can distinguish between 'organization interventionists', 'formal modellers', and 'statistical validators'. These are three extremes in a spectrum of research attitudes. A characterization of the three follows.

Organization interventionists

Organization interventionists operate on one or more case studies in organizations, in which they try to change an unsatisfactory initial state of affairs into a satisfactory future state of affairs. Organizations and information systems, they say, belong to the world of human social systems. This world is felt to be so complex that mapping of real problem situations onto formal models of a much simpler systemic level[83] is not sensible. Likewise, they mistrust experiments because experimental settings do not possess the richness of the real world and can therefore seldom be generalized. Therefore they adopt

[83] 'Systemic level' is intended here in the sense of Boulding's hierarchy discussed in chapter three.

informal research methods and regard each case as unique. If an organization interventionist has a taste for theory, he or she may engage in inductive generalization over a number of cases. This yields case-independent generalized theory or generalized intervention tools.

Formal modellers

Formal modellers are of the opinion that clearness is better than vagueness, even if it requires some simplification. They abstract an area of interest (the so-called object system) from its surroundings, model it, and work with the model. Results obtained with the model are fed back into the object system. The first step of abstracting the object system from the real world is carried out with tools of the trade, which are formal and often quantitative in nature and leave no room for ambiguity. The last and crucial step of re-interpretation of the model results in terms of the real-world problem situation is left to the users.

Formal modellers may engage in inductive generalization over a number of cases. This yields generalized models.

Statistical validators

Statistical validators mistrust induction, because it can too easily turn into fantasizing without any proven value. They want to validate or reject given theories. To do so they need large sets of comparable data. They also need some hypothesis or theory to validate. So they introduce a zero hypothesis, set up a questionnaire, find a suitably large set of businesses, managers, DSS, or whatever, collect the data and then 'torture the data long enough to make them confess'[84]. Their approach is deductive.

Strengths and weaknesses

The characterization of these three 'archetypes' makes it clear that each has its strengths and weaknesses. An information systems researcher needs to combine elements of more than one archetype within himself or herself.

Incidentally, there is a marked difference between information systems research in Europe and in the U.S.A. In Europe, formal modelling is popular, whereas in the USA statistical validation is 'the done thing'. On both sides of the Atlantic the organizational interventionist perspective is gaining ground.

In the theory generating part of the present study, the organization interventionist perspective has been most prominent. Formal models have been used as elements of interactive planning systems. I have also carried out some laboratory experiments in the spirit of the statistical validators, but these are relatively unimportant in this study.

The design perspective taken in the case study is a combination of the organization interventionist and the formal modeller perspectives. Such a combination in fact yields what is often called an engineering perspective: formal modelling methods are applied

---

[84]This expression is not mine, but I have not been able to track its origin.

to organizational problems with the aim of 'making' something to improve the situation. In my case, the organization interventionist perspective has been rather more marked than the formal modeller perspective.

In design-oriented research, case studies take the place of experimental data. This necessarily leads to small sample-data so that statistical validation is not possible. The case study in this study was carried out simultaneously with the theoretical research and each has profited from the other. This kind of synergy between theory and experiment is well described by Glaser and Strauss (1967) in their plea for 'Grounded Theory'. The theoretical results could not have been obtained without the case experience, but just as well the case profited from the theoretical findings. My mentioning Glaser and Strauss does not imply that I pretend to have developed 'Grounded Theory': I have not, since I did not start with considerable amounts of empirical data but based myself on a case study and literature only. Notwithstanding this I experienced a feeling of recognition when reading theoretical literature, for instance about Soft Systems Methodology, which is grounded in experience that is similar, although far more massive, than my own.

Concluding it can be said that in this study all three archetypical research approaches have been used, with a strong emphasis on the first two. This is in line with the goals of the study: generating rather than verifying theory.

6.8 Generality of the results

By taking decision making in organizations as a point of departure, it is also defined as a point of return. In other words, it should in principle be possible to generalize the answers to the research questions from tactical planning to decision making in general, and it should be possible to generalize what is concluded about planning systems to information systems in general. An important limitation is that I have only considered individual decision making, so that the additional complexity of group interaction is not taken into account.

Let us consider the central research questions and see whether the answers to them can be generalized. In generalized wording, the research questions become: "*What should an information system look like, if it is to be useful in practice?*", "*For which decision situations is it appropriate to develop an information system?*" and "*How should an information system be developed?*". The conclusions drawn in this study with regard to fragmented decision making, ambiguity, interacting decision processes, and nested context levels, all hold for decision situations in general. In organizations with a separation of functions, problems will tend to be less open than in the small organizations of this book's case study. With appropriate allowances for the type of situation, SCAPSIS can be used for all decision situations, as well as QUEST. It must be noted that group processes carry with them additional complexity, so that information systems for cooperative work are excluded. Most conclusions of this study, for instance those about cooperative development, will probably hold only more strongly in the context of information systems for cooperative work.

Although the predicament for tactical planning is special, because open problems can so easily be mistaken for formal problems, an overly prescriptive orientation among designers of information systems in general, and the neglect of change processes, can lead to problems similar to those that occur with planning systems. Here, QUEST might help designers of information systems realize that there are benefits of various nature to be gained from an information systems development project. Similarly, the discussion in chapter four is relevant to other information systems than planning systems only. The interdependence of system development as a process with the properties of the end product holds for all information systems.

The specific design ideas for interactive planning systems, interactive heuristics and problem decomposition cannot be generalized to fields outside tactical to operational planning. They are specific to problems that are open, have one problem owner, and have a relevant combinatorial component. But the design ideas can readily be generalized to domains other than pot plant cultivation planning.

6.9 Further research

SCAPSIS and QUEST

As a general remark on SCAPSIS and QUEST, it can be said that the empirical basis of these two tools is as yet too slim. Referring again to Glaser and Strauss, SCAPSIS and QUEST can be seen as hypotheses with which to engage in empirical research. Such research could result in empirically grounded modified versions. At the same time, the complex systemic nature -Boulding's level eight- of the subject matter makes it likely that such empirical research would meet with severe conflicts between rigour and relevance. As to the manner in which the tools can be applied, it seems appropriate to use them as instruments in an appreciative system (fig 3.1).

SCAPSIS

SCAPSIS, the quick-scan tool for assessment of chances for automated support, is promising, and if it can be validated, corrected, and extended, it can develop into a useful tool in the hands of designers, which can prevent the wasting of a lot of energy on unpromising projects. A possible research approach is to follow information systems consultants and build their experience into the tool.

Ideally, further development of SCAPSIS could with time lead to a typology of planning situations. This typology would be linked to the type of planning system suited to a particular type of planning situation.

QUEST

The quality assessment checklist is primarily intended as an eye-opener. It is certainly worthwhile to use QUEST on a body of cases and improve it along the way. It may be possible to develop QUEST and collect more experience in the tool. However, one should beware of making too rigid a tool and thereby stifling the creativity which is necessary to obtain a context-specific quality assessment.

Management at pot plant nurseries

At pot plant nurseries, operational and tactical management decision making is a network of tightly interwoven processes. The processes are integrated, and it is a research challenge to build automated support that is also integrated. In fact, the misfit between insular information systems with integrated organizational processes is a problem not at all limited to agriculture.

The sector of pot plant cultivation is a challenging research domain because management is complex, and the sector is very heterogeneous, so that only very flexible and adaptive systems stand a chance of being accepted. Currently, a research programme is being defined to address the question of integrated support for operational and tactical management at pot plant nurseries. The PROPLAN project will serve as one of the sources of inspiration for this new research programme.

Planning system ingredients and user-centred development

Important open questions are whether the concept of an interactive planning system as put forward in this book is valid and whether its implications for systems development hold true. These implications are that the relative importance of user interfaces versus models and of cooperation during design versus functional specifications should be quite different from current practice.

Do systems that are developed as proposed in this book function better in their target organizations than other systems do? This is a type of question that can probably never be answered definitively, but if it were at least established that they function well in certain contexts, then an enormous research agenda would ensue. It would follow an engineering approach. The principles given in this study would have to be applied to case studies and results obtained on a number of case studies could then be generalized.

It is my belief that there is a major research challenge here, not in the least for operations researchers. It is fairly widely felt within the OR community that too much attention has been given to mathematical techniques at the expense of real-world impact. If operations researchers adopt the approach of first developing an interface together with a user, and only then consider enriching the system with interactive heuristics *using the existing interface*, then a rich area for research into interactive mathematical techniques comes into perspective. In many cases it will be useful to embed interactive heuristics in a problem decomposition structure. The mathematical techniques to investigate are 'open models', i.e., models which are not black boxes to the user but which allow interactive manipulation. Until now, both OR and AI have hardly paid attention to the role of the user as a last-minute supplementer of knowledge. They have built closed models that stand and function by themselves. There is hardly any knowledge yet about open models.

I do not propose to discard model-based problem solving but to apply it in the right context at the proper time. The slogan for researchers and developers of planning systems may well become: modesty in modelling.

Abstract

Although I am fully convinced of the truth of the views
given in this volume under the form of an abstract,
I by no means expect to convince experienced naturalists
whose minds are stocked with a multitude of facts all viewed,
during a long course of years,
from a point of view directly opposite to mine.

Charles Darwin (1859 / 1968, p. 453)

This book is a statement about the applicability of automated support for planning. The planning situations considered are those that bear upon the next production cycle of an organization and in which a planner has to allocate 'production goods' to each other. Such situations are for instance tactical planning, production planning, scheduling or timetabling. Drawing from various bodies of literature, a marked position is taken. It is argued that the practical impact of planning systems developed by academics is lower than it could be because formal models play too prominent a part during the development process.

Based on views taken from organization theory, psychology and systems theory, a SCanning Aid for Planning SItuationS (SCAPSIS) is developed that places these situations on a continuum from 'messy' to 'well-structured'. Not only the problem situation in a strict sense but also the organizational context are taken into account. It is argued that in practice the majority of planning situations are closer to the 'messy' end of the continuum than to the other. The planners deal with these situations as 'open problems', i.e., problems of which they can change the definition at any moment.

A planning system for such an open problem will have to grant the user a prominent role. Usability precedes normative support both in importance and in chronology: a user interface familiar to the planner and close to his train of thought is the first and most important element of the system to develop whereas a formal model of the planning problem is definitely not. Such a model, if created at an early stage, can act as a barrier to communication for both parties. As a result, chances are that the resulting system ends up solving the developer's formal problem but not the planner's open one.

To come up with a usable system, developer and user must closely cooperate right from the start of a project and aim to make a first usable system as rapidly as possible. Only then, on the basis of experience with this system, can it be considered whether to proceed, for instance by incorporating model-based components in the system.

A modelling technique with some promise is the category of interactive heuristics, i.e. heuristics that allow the user to intervene at each iteration. Desirable properties of such heuristics are discussed. It is concluded that this category merits more attention as an element of interactive planning systems.

The stance taken towards planning situations in this study is that they are instances of decision making in general, embedded in the functioning of the organization. With this in mind, a tool is created to help evaluate decision support systems. This 'QUality ESTimator' (QUEST) takes into consideration both the process of system development and the resulting system as it functions in the organization.

A case study in which a planning system was developed to support cultivation planning at pot plant nurseries serves as an illustration throughout the book. The development of this system is an example of how design ideas that are attractive from an academic developer's point of view can fail to succeed in practice, and how cooperative development and modesty in modelling can lead to a simple but usable system.

The ideas expressed in the book can be readily generalized to non-agricultural domains.

Samenvatting

Dit boek behandelt de toepasbaarheid van planningssystemen: geautomatiseerde hulpmiddelen voor planning. Beschouwd worden die planningsvraagstukken die de inzet betreffen van de produktiemiddelen van een organisatie gedurende de eerstvolgende produktiecyclus. Zulke vraagstukken worden wel aangeduid als taktische en operationele planning.

Het onderzoek heeft bestaan uit literatuurstudie en een case-studie. De combinatie van deze twee aktiviteiten heeft geleid tot een duidelijk standpunt over de inzetbaarheid en aard van planningssystemen. Ook zijn enkele hulpmiddelen ontwikkeld voor het inschatten van de haalbaarheid van planningssystemen, voor het ontwerp en voor de evaluatie ervan.

De case-studie betrof het ontwikkelen van een planningssysteem voor de teeltplanning op potplantenbedrijven. Zij heeft in het onderzoek een belangrijke rol gespeeld als inspiratiebron voor het theoretisch gedeelte. Alvorens nader in te gaan op de onderzoeksvragen en resultaten zal zij daarom eerst kort worden beschreven.

Teeltplanning op potplantenbedrijven

De nederlandse potplantenteelt is met haar *f* 1,5 miljard jaarlijkse produktiewaarde internationaal vooraanstaand. Het is ook een bloeiende bedrijfstak. Tegelijkertijd is de diversiteit in de potplantenteelt enorm. Geen twee bedrijven zijn hetzelfde. De ontwikkelingen ten aanzien van assortiment, teelttechnieken, automatisering en wetgeving volgen elkaar in hoog tempo op, zodat een potplantenteler alert en flexibel moet zijn om te overleven. Er vindt dan ook een snelle ontwikkeling plaats van eensgezinsbedrijven naar grotere bedrijven waarin meer aandacht mogelijk (en nodig) is voor de besturing.

Een potplantenbedrijf bestaat uit één of meer kassen. De plattegrond van zo'n kas kan er uitzien zoals in figuur 1.1 (p. 3) weergegeven. De planten worden bijvoorbeeld geteeld op vaste tafels, op beweegbare tafels ('rolcontainers') of op betonvloeren. De kleinste oppervlakte waarop wordt geteeld bedraagt vaak tientallen vierkante meters. Oppotmateriaal wordt zelf gekweekt of aangekocht. De afzet verloopt via de veiling of via andere kanalen.

Een van de belangrijke aktiviteiten op een potplantenbedrijf is het opstellen van een teeltplan voor de volgende produktieperiode. In de regel wordt een teeltplan gemaakt voor het eerstvolgende jaar. Bij het opstellen van zo'n plan worden beslissingen genomen over het assortiment, de hoeveelheden, oppot- en afzetdata. Sommige telers leggen tegelijkertijd ook al de plaatsing van de planten in de kas vast.

Plaatsing van planten in een kas is om een aantal redenen van belang. Kasruimte is duur: de afschrijvingen belopen zo'n *f* 100,-- per jaar per m². Een goede ruimtebenutting is daarom wenselijk. Voor een goede groei mogen de planten echter niet te dicht op elkaar staan. Anderzijds is het onrendabel om zaailingen of stekjes direkt op de uiteindelijke afstand te plaatsen, omdat er dan veel ruimte tussen de planten overblijft. De meeste produkten worden dan ook tijdens de teelt één of meer malen wijder gezet. Vaak verloopt de groei van een partij planten niet uniform en worden de planten niet tegelijk afgezet. Soms wordt het restant dan weer ingeschikt. Kortom, het komt nogal eens voor dat de planten moeten worden verplaatst. Bij de meest toegepaste transportsystemen (karretjes, lopende band) is het sneller en dus goedkoper om over korte dan over lange afstand te vervoeren. Bij het maken van een plaatsingsplan kan de teler hierop anticiperen.

Andere argumenten voor een uitgekiende plaatsing zijn: overzicht over de kas, regelmaat in het werk voor het personeel, het representatief zijn van de tuin, beschikbaarheid van teelthulpmiddelen op bepaalde teeltplaatsen (bijbelichting, klimaatregeling).

Geschiedenis van de case-studie

In 1985 kwam het 'Informatiemodel Potplanten' gereed dat was ontwikkeld door een groot aantal betrokken instanties in het kader van het INSP-LO, het Informatica Stimuleringsplan van de overheid. Het eindrapport (Beers et al. 1985) gaf onder meer aan dat er in de potplantenteelt behoefte was aan geautomatiseerde hulpmiddelen bij de teeltplanning. Dit was aanleiding om, voortbouwend op ervaringen op het IMAG en de LU, een projekt te beginnen. Het doel was een planningssysteem te maken voor het ondersteunen van de teeltplanning, in het bijzonder de plaatsingsplanning, op potplantenbedrijven. Het projekt kreeg de naam PROPLAN: PROtotype PLANningssysteem voor potplantentelers. Het woord prototype duidt er op dat het ging om een onderzoeksversie en niet om een vermarktbaar systeem. Dit prototype moest draaien op voor de doelgroep gangbare hardware en het diende in principe praktisch bruikbaar te zijn.

Het projekt PROPLAN is begonnen in 1985. In 1989 was er een versie van PROPLAN beschikbaar die op PC's draaide. Deze versie, PROPLAN 0.2, maakte gebruik van gegevens over teeltschema's en financiële aspekten. De gebruiker kon kiezen of hij PROPLAN een plan liet maken ofwel voor elke geplaatste partij planten zelf het laatste woord wilde hebben. PROPLAN gebruikte voor het maken van plannen eenvoudige rekenregels (heuristieken) waarvan de gebruiker vele instellingen op ieder gewenst ogenblik kon wijzigen.

Toch bleek PROPLAN 0.2 voor tuinders te ingewikkeld en te weinig flexibel te zijn. Om aan deze bezwaren tegemoet te komen werd een van de menukeuzes apart ontwikkeld tot een eenvoudig systeem, dat in 1990 klaar was: PROPLAN HAND. Dit systeem verrichtte geen enkele berekening. Eigenlijk bood het niet meer dan papier en potlood. Het voldeed niettemin aan de behoefte van de telers met wie het was ontwikkeld, vooral omdat het enorm veel tijdsbesparing opleverde en omdat het de mogelijkheid schiep allerlei ideeën te verkennen alvorens een definitief plan te maken.

Onderzoeksvragen

De oorspronkelijke onderzoeksvraag van dit proefschrift was: "*Hoe moet een planningssysteem er uit zien om in de praktijk bruikbaar te zijn?*". Bij het bestuderen van de literatuur over planningssystemen bleek dat deze vraag nader kon worden gespecificeerd. Nogal wat planningssystemen uit de literatuur brachten het niet verder dan onderzoeksversies, terwijl naar praktisch bruikbare systemen werd gestreefd. Er waren nogal eens problemen in de afstemming tussen modellen in een systeem en de wijze van probleem-oplossen van de planner. Naar aanleiding hiervan ontstonden de volgende onderzoeksvragen: "*Welke functies moet een planningssysteem bevatten om te passen bij de wijze van probleem-oplossen van de gebruiker?*", "*In welke functies van een planningssysteem moet er interaktie mogelijk zijn tussen gebruiker en systeem, en waaruit moet die interaktie kunnen bestaan?*" en "*Welke probleem-oplossende stappen moet de gebruiker verrichten, en welke het systeem?*".

Daarnaast ontstonden nieuwe onderzoeksvragen doordat ik begon in te zien dat bepaalde aspekten van planningsvraagstukken in de literatuur werden veronachtzaamd. Organisaties verschillen en datzelfde geldt voor planningssituaties. De relatie ervan met de rest van de organisatie kwam in de literatuur zelden aan de orde. Er werd een duidelijkheid in de afbakening van de planningssituatie gesuggereerd die mij niet realistisch voorkwam. Het leek me aannemelijk dat de aard van een planningssituatie en die van de organisatie waarbinnen deze optreedt gevolgen hebben voor de toepasbaarheid van planningssystemen. Hieruit ontstond de vraag: "*Voor welke planningsvraagstukken is het zinvol, beslissingsondersteunende systemen te ontwikkelen?*".

Een laatste vraag betrof het feitelijk ontwikkelproces van planningssystemen. In de literatuur werd nogal eens de nadruk gelegd op modellen waarin de logica van een planningssituatie was weergegeven. Het feit dat de planners en hun organisatie een veranderingsproces ondergaan wanneer er een planningssysteem komt, werd minder belicht. De onderzoekvraag die dit opleverde was: "*Hoe dienen planningssystemen te worden ontwikkeld?*".

Aanpak

De case-studie en het literatuuronderzoek verliepen gelijktijdig en ondersteunden elkaar. De literatuurstudie betrof drie aandachtsgebieden. Allereerst bestudeerde ik in het kader van de eerste onderzoeksvraag de literatuur over het ontwikkelen van planningssystemen. Voor de vraag naar 'welke situaties' keek ik naar organisatiekundige, psychologische en systeemkundige literatuur. De vraag 'hoe te ontwikkelen' werd door deze beide literatuurstudies deels gedekt en leidde daarnaast to het bestuderen van literatuur over ontwikkeling en kwaliteit van informatiesystemen in het algemeen. Voor alledrie de vragen heb ik niet alleen theoretische antwoorden gegeven maar ook praktische hulpmiddelen ontwikkeld.

Hoe moet een planningssysteem er uit zien?
De literatuurstudies komen in het boek uitgebreid aan bod. Hoofdstuk twee behandelt het ontwerpen van planningssystemen. De behandeling is ingedeeld naar onderzoektraditie. Operations Research (Operationele Analyse) en Artificial Intelligence (Kunstmatige Intelligentie) krijgen de meeste aandacht. Aan heuristische technieken wordt in het bijzonder aandacht geschonken. De eerste onderzoeksvraag wordt besproken. Hierbij wordt in elk geval duidelijk hoe een planningssysteem er *niet* uit moet zien. Veel te vaak staat bij systemen die uit de onderzoekswereld zijn voortgekomen een formeel model van de planningssituatie op de voorgrond. Planners kunnen hiermee vaak in het geheel niet uit de voeten. Eenvoudige systemen die hen de vrijheid bieden te 'sjoemelen' (Takkenberg, mondeling) met de probleemformulering en hen dom werk uit handen nemen vallen beter in de smaak. Met andere woorden, het systeem moet het probleem niet geheel vastleggen maar bepaalde zaken (en daar schuilt een ontwerpprobleem) voor de planner open laten.

Als verder antwoord op de vraag hoe een planningssysteem er dan wel uit zou kunnen zien worden twee ideeën voor het ontwerp van planningssystemen nader uitgewerkt. Het eerste betreft interaktieve heuristieken, waarbij de gebruiker tijdens elke iteratie (herhaling) kan ingrijpen. Zulke heuristieken dienen te werken met begrippen die de planner kent en begrijpt, en deze begrippen moeten zichtbaar en manipuleerbaar worden gemaakt. PROPLAN 0.2 is een poging om dit idee in de praktijk te brengen.

Het tweede idee betreft een model voor probleemdecompositie (het opdelen van een probleem in stukjes). Om een planningsprobleem behapbaar te maken passen planners meestal probleemdecompositie toe. Dit kan ook in een planningssysteem. Daarbij kan de vertaling tussen deelproblemen van probleemformuleringen of van oplossingen moeilijkheden geven, omdat in een geautomatiseerd systeem de deelproblemen minder flexibel met elkaar in verband kunnen worden gebracht dan bij handmatig plannen. Het in dit boek gepresenteerde model kan dienen om deelproblemen flexibel te koppelen en om na te denken over de taakverdeling tussen planner en systeem. Ook voor dit model heeft PROPLAN als proeftuin gediend.

Voor welke situaties is een planningssysteem aan de orde?
De onderzoeksvraag naar toepasbaarheid van planningssystemen voor verschillende typen situaties komt aan de orde in hoofdstuk drie. Dit hoofdstuk gaat in op de plek van planningssituaties binnen het werk van een planner/manager en binnen het functioneren van organisaties. De betreffende literatuur is ingedeeld naar drie gezichtspunten: organisatiekundige, psychologische en systeemkundige literatuur. Op grond van deze literatuur ontwikkel ik eigen inzichten, weergegeven in figuur 3.2 en 3.3, over de aard van besluitvorming en de rol van beslissingsondersteunende systemen in een organisatie. In het kort gezegd komen deze ideeën er op neer dat processen in organisaties veel minder doelgericht, rationeel en duidelijk verlopen dan dat ze achteraf worden voorgesteld. Een perspektief op organisaties, en dus op planningssituaties, waarbij dit

erkend wordt, leidt tot een realistischer inschatting van het nut van planningssystemen. Simpel gezegd: in een rommelige organisatie past een strak planningssysteem niet. Het toepassen van deze inzichten leidde tot het ontstaan van SCAPSIS, een hulpmiddel waarmee een systeemontwerper of andere betrokkene snel inzicht kan krijgen in de toepasbaarheid van beslissingsondersteuning voor een bepaalde planningssituatie.

Bij een analyse met SCAPSIS worden acht vragen gesteld. Over de context van de planningssituatie gaan de volgende vragen:

1. Hoe groot is de organisatie? Is er functiescheiding tussen het strategische, het taktische en het operationele niveau?
2. Is er een gespecialiseerde planner voor de betreffende situatie?
3. Hoe stabiel is de omgeving? Zijn organisatiestruktuur, produkten, markten en regelgeving stabiel?
4. Hoe dikwijls vindt de planning plaats in de huidige vorm?

Dan zijn er vier vragen over de situatie in engere zin:

5. Hoe voorspelbaar is het objectsysteem (het deel van de organisatie waarop het plan betrekking heeft)?
6. Zijn er geen belangen van mensen met de plannen gemoeid?
7. Hoe bestuurbaar is het objectsysteem?
8. Is er aan betrouwbare gegevens te komen?

Voor elke van deze vragen is er een schaal gedefinieerd van 1 tot 5, waarbij 1 overeenkomt met 'chaos' en 5 met 'duidelijkheid'. SCAPSIS kan hierdoor vragen beantwoorden omtrent de mate waarin zowel objectsysteem als context duidelijk afgebakend zijn. Teeltplanning op potplantenbedrijven scoort op alle aspekten laag. Modellen blijken dan ook weinig ingang te vinden in de planning. Het zeer eenvoudige planningssysteem PROPLAN HAND werd wel geaccepteerd.

Een ander geval dat in hoofdstuk drie wordt behandeld, een distributievraagstuk bij een grote voedselproducent (de BARS case) scoort over de gehele linie hoog op SCAPSIS. Inderdaad is in deze situatie met succes een planningssysteem in gebruik genomen dat geavanceerde wiskundige technieken toepast. Een derde situatie, strategische personeelsplanning bij KLM, geeft hoge context-scores maar lage scores op de andere vragen. KLM is dan ook geneigd een ambiteuze oplossing te zoeken.

Hoe moet een planningssysteem worden ontwikkeld?

De resultaten van de literatuurstudies die hierboven zijn gegeven impliceren eigenlijk al een antwoord op de laatste onderzoeksvraag. Zo komt uit de organisatiekunde en psychologie ook het besef dat het ontwikkelen van een planningssysteem een proces van organisatieverandering is. Planners ontkomen er niet aan, anders te gaan werken dan ze gewend waren. Ze zullen hiertoe alleen bereid zijn onder bepaalde voorwaarden.

Het besef dat het proces van systeemontwikkeling zeker zo belangrijk is als het eindresultaat ervan blijkt ook uit de literatuur die in hoofdstuk vier wordt gebruikt. Ik pleit dan ook voor nauwe samenwerking tussen gebruiker en ontwerper bij het ontwikkelen van beslissingsondersteunende systemen. Het eerste doel is een basis voor samenwerking te vinden. Als zodanig kan een zeer eenvoudige versie van een systeem dienen. Zo'n eerste versie behoeft absoluut niet 'intelligent' te zijn. Een vereiste is wel dat de gebruiker zijn zienswijze herkent in het systeem. De concepten en het gebruikersinterface van het systeem moeten hem vertrouwd zijn. Pas indien hij het systeem begrijpt en wanneer hij er daadwerkelijk mee werkt kan hij beoordelen of verdere ontwikkeling wenselijk is. Dan pas komt het gebruik van 'intelligente' technieken aan de orde.

Wat is een goed planningssysteem?

De onderzoeksvragen tezamen kunnen gebundeld worden tot de algemene vraag naar de kwaliteit van planningssystemen. In hoofdstuk vier wordt een 'kwaliteitsketen voor beslissingsondersteunende systemen' geïntroduceerd. Deze keten (figuur 4.1), afgeleid van figuur 3.3, geeft een oorzakelijk verband aan tussen de kwaliteit van (1) het functioneren van een organisatie, (2) de beslissingen die genomen worden, (3) de probleem-oplossingsprocessen die er plaats vinden, (4) de beslissingsondersteunende systemen die er worden gebruikt en (5) de ontwikkeling van deze systemen. Het uiteindelijke doel, verbeteren van het functioneren van de organisatie, is van elke van de overige elementen in de keten mede-afhankelijk. De verbanden zijn overigens niet alleen oorzakelijk. Er is ook toeval in het spel. Ook loopt er een rechtstreekse invloed van het ontwikkelen van systemen naar het functioneren van de organisatie, doordat zo'n ontwikkelproject allerlei veranderingen in mentaliteit, werkwijzen en kennis in gang zet.

De kwaliteitsketen bevat beurtelings een proces (5, 3 en 1) en een resultaat (4 en 2). Ik beargumenteer dat de processen, te weten ontwikkeling van ondersteunende systemen en probleemoplossen, van groter belang zijn dan de resultaten, te weten beslissingsondersteunend systeem en beslissing. Helaas is het bij processen lastiger de kwaliteit te bepalen dan bij produkten.

Om tot een weloverwogen kwaliteitsoordeel te komen is het nodig alle elementen van de kwaliteitsketen in de evaluatie te betrekken. Ik presenteer in hoofdstuk 4 QUEST, een hulpmiddel dat de kwaliteit van alle schakels in de keten in beschouwing neemt om te komen tot een beoordeling van een beslissingsondersteunend systeem.

Slotopmerkingen

In dit onderzoek is gebruik gemaakt van een combinatie van empirie en deductie. Voor het deductieve aspekt is een groot aantal invalshoeken gekozen. In algemene zin kan worden gesteld dat uit het onderzoek de meer formele onderzoeksperspektieven naar voren komen als ondergeschikt aan de inzichten uit de menswetenschappen. De titel van het boek verwijst hiernaar. Modellen van een planningssituatie of van het bestuurde systeem kunnen zeker een plaats hebben bij het ontwikkelen van planningssystemen, maar alleen wanneer ze worden gebruikt op een geëigend tijdstip binnen de context van een veranderingsproces.

Het onderzoek geeft aanleiding tot allerlei vervolgonderzoek. Zoals bij veel onderzoek behoeven de resultaten empirische toetsing. Dit geldt zowel voor de ontwerpideeën uit hoofdstuk twee als voor SCAPSIS en QUEST. Het is mijn vermoeden dat, wanneer heuristieken vanuit het gezichtspunt van gebruikers worden beschouwd, een rijk onderzoeksgebied in het verschiet ligt.

SCAPSIS kan uitgroeien tot een praktisch hulpmiddel voor het opbouwen van gevoeligheid omtrent planningssituaties bij ontwerpers of (aankomende) consultants. Van QUEST verwacht ik dat niet zozeer de feitelijke evaluatiepunten als wel het achterliggende idee van de kwaliteitsketen van invloed kan zijn op het denken over kwaliteit van informatiesystemen. Beide hulpmiddelen lenen zich goed voor het ondersteunen van de samenwerking tussen systeemontwikkelaars en gebruikers.

De resultaten van deze studie laten zich, vanwege het gekozen brede perspectief op besluitvorming in organisaties, zonder meer veralgemenen naar niet-agrarische domeinen.

Bibliography

In de wetenschap is een geluid pas een geluid als het op een echo lijkt.[85]

Kees Fens, 1990

Ackoff, Russell L. (1978)
The Art of Problem Solving. New York: John Wiley.
Ackoff, Russell L. (1979)
"The Future of Operations Research Is Past." *Journal of the Operational Research Society* 30, pp. 93-104.
Ackoff, Russell L. (1981)
"The Art and Science of Mess Management." *TIMS Interfaces* 11:1, pp. 20-26.
Ackoff, Russell L. (1987)
"OR, a post mortem." *Operations Research* 35:3, pp. 471-474.
Alavi, Maryam and H. Albert Napier (1989)
"An experiment in applying the adaptive design approach to DSS development." In Sprague, Ralph H. and Hugh J. Watson (eds), *Decision Support Systems: Putting Theory into Practice*. 2nd ed. London: Prentice-Hall, pp. 82-92.
Alleblas, Joop T.W. (1987)
Management in de glastuinbouw, een zaak van passen en meten. LEI onderzoeksverslag 34. Den Haag: Landbouw Economisch Instituut.
Alter, Steven L. (1980)
Decision Support Systems: Current Practice and Continuing Challenges. Reading, Massachusetts: Addison-Wesley.
Anderson, Norman H. (1986)
"A cognitive theory of judgment and decision." In Berndt Brehmer et al. (eds), *New Directions in Research on Decision Making*. Amsterdam: North-Holland, pp. 63-108.
Angehrn, Albert A. (1991)
Interpretative computer intelligence: a link between users, models and methods in DSS. INSEAD working paper no. 91/12/TM. Fontainebleau: European Institute of Business Administration INSEAD.

[85] In the scientific world a sound is a sound only if it resembles an echo.

Angehrn, Albert A. and Hans-Jacob Lüthi (1989)
Visual interactive modelling and intelligent DSS: Putting theory into practice. Zürich: ETH, Swiss Federal Institute of Technology.

Angehrn, Albert A. and Hans-Jacob Lüthi (1990)
"Intelligent Decision Support Systems: A Visual Interactive Approach." *Interfaces* 20:6, pp. 17-28.

Annevelink, E. (Bert) (1990[a])
Geautomatiseerde produktieplanning, wensen vanuit de praktijk. Report no. 491. Wageningen: Institute of Agricultural Engineering IMAG.

Annevelink E. (Bert) (1990[b])
Handleiding IPP - plaatsingsplan - module versie 1.0. Report no. P511. Wageningen: Institute of Agricultural Engineering IMAG.

Anthonisse, J.M. (Ko), Jan Karel Lenstra and Martin W.P. Savelsbergh (1988)
"Behind the screen: DSS from an OR point of view." *Decision Support Systems* 4, pp. 413-419.

Anthony, Robert N. (1965)
Planning and Control Systems, a Framework for Analysis. Boston: Harvard University.

Anthony, Robert N. (1988)
The Management Control function. Boston, Massachusetts: Harvard Business School Press.

Badie, Catherine (1988)
Contribution à l'étude des problèmes d'affectation de ressources et d'ordonnancement: application au domaine spatial. CERT report no. 26. Toulouse: Centre d'Etudes et de Recherches de Toulouse.

Badie, Catherine, G. Bel, E. Bensana and G. Verfaillie (1990)
"Operations Research and Artificial Intelligence cooperation to solve scheduling problems: the OPAL and OSCAR systems." In *Proc. first International Conference on Expert Planning Systems,* 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 1-5.

Bailey, J.E. and S.W. Pearson (1983)
"Development of a tool for measuring and analyzing computer user satisfaction." *Management Science* 29:6, pp. 519-529.

Ball, Michael and Michael Magazine (1981)
"The design and analysis of heuristics, account of a discussion session." *Networks* 11, pp. 215-219.

Barendse A.N.H. (Dolf) (1986)
Produktieplanning van potplantenbedrijven, plaatsen en verplaatsen van partijen. IMAG report no 240. Wageningen: Agricultural University, department of Mathematics, section Operations Research. co-published by Institute of Agricultural Engineering IMAG.

Baumewerd-Ahlmann, Annegret and J. Kalinski (1990)
"Structuring techniques for knowledge-based planning systems in environmental impact assessment." In *Proc. first Conference on Expert Planning Systems,* 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 257-261.

Beach, Lee R. (1990)
Image theory: decision making in personal and organizational contexts. Chichester: John Wiley.

Beer, Stafford (1981)
Brain of the Firm, The Managerial cybernetics of Organization. 2nd ed., Chichester: John Wiley.

Beers, George et al. (1985)
Informatiemodel Potplanten. Rapport 37, Aalsmeer: PBN.

Beers, George (1990)
Problemen, planning en kennis, een onderzoek naar de processen achter succes en falen van een automatiseringsproject. Ph.D. Series in General Management no. 1. Rotterdam: Rotterdam School of Management, Erasmus University.

Bellow, Saul (1964)
Herzog (rpt. London: Penguin, 1965).

Bemelmans, T.M.A. (1987)
Bestuurlijke informatiesystemen en automatisering. 3rd ed. Leiden: Stenfert Kroese.

Bensana, Eric (1987)
Utilisation de techniques d'intelligence artificielle pour l'ordonnancement d'atelier. CERT report no. 8. Toulouse: Centre d'Etudes et de Recherches de Toulouse, Toulouse.

Bennett, J.L. (ed.) (1983)
Building Decision Support Systems. Reading, Massachusetts: Addison-Wesley.

Beulens, Adrie J.M. (1990)
"More powerful and user-friendly DSS by incorporating ES-technology?" In D. Ehrenberg, H. Krullmann and B. Rieger (eds), *Wissensbasierte Systeme in der Betriebswirtschaft.* Berlin: Ehrig Schmidt Verlag, pp. 95-112.

Biemond, Teunis en Gert Jan Hofstede (1988)
"GROMS, een managementondersteunend systeem voor potplantentuinders." In Cees Huijbers et al. (eds), *Informatica - toepassingen in de Agrarische sector, Proc. VIAS-Symposium 1988, Agro-Informatica reeks 2.* Wageningen: VIAS, pp. 115-124.

Bitner, James R. and Edward M. Reingold (1975)
"Backtrack Programming Techniques." *Communications of the ACM* 18:11, pp. 651-656.

Bjørn-Andersen, Niels (1988)
"Are 'Human Factors' Human?" *The Computer Journal* 31:5, October 1988, pp. 386-390.

Bleijenberg, Peter (1983)
Stap voor stap naar beter management, produktie-proces-besturing op het potplantenbedrijf. Part 1 and 2. Wageningen: Institute of Agricultural Engineering IMAG.

Boersma, Sake K.Th. (1989)
Beslissingsondersteunende systemen: een praktijkgerichte ontwikkelingsmethode. Schoonhoven: Academic Service.

Boehm, B. (1978)
Characteristics of software quality. Amsterdam: North-Holland.

Boland, Richard J. Jr. (1987)
"The In-formation of Information Systems." In Richard J. Boland Jr and Rudy A. Hirschheim (eds), *Critical Issues in Information Systems Research.* Chichester: John Wiley, pp. 363-379.

Bosman, Aart (1977)
Een metatheorie over het gedrag van organisaties. Leiden: Stenfert Kroese.

Bosman, Aart (1986)
"Ontwerpen, procesmodellen en beslissen." In *Informatie,* 28/4, pp. 257-360.

Bots, Jan M. (1991)
De besturing van het primaire agrarische bedrijf. Doctoral Dissertation. Wageningen: Agricultural University, department of Management Studies.

Bots, Jan M. and Eric van Heck (1989)
"Besturingsmodel, informatiemodel, informatiesystemen." In Cees Huijbers et al. (eds), *Informatica - toepassingen in de Agrarische sector, Proc. VIAS-Symposium 1989, Agro-Informatica reeks 3*. Wageningen: VIAS, pp. 127-140.

Bots, Jan M., Eric van Heck, Victor van Swede and John L. Simons (1990)
Bestuurlijke Informatiekunde. Rijswijk: Cap Gemini/Pandata.

Bots, Pieter W.G. (1989)
An environment to support problem solving. Doctoral Dissertation. Delft: Technological University, department of Information Systems.

Boulding, Kenneth E. (1956)
"General Systems Theory - the Skeleton of Science." *Management Science* 2:3, pp. 197-208.

Bullinger, H-J. and B. Shackel (eds) (1987)
Proc. second IFIP conference Human-Computer Interaction INTERACT '87, 1-4 September 1987. Amsterdam: North-Holland (Elsevier).

Carlson, S. (1951)
Executive Behaviour: A Study of the Work Load and the Working Methods of Managing Directors. Stockholm: Strömbergs.

Chandler, J. (1982)
"A Multiple Criteria Approach for Evaluating Information Systems." *MIS Quarterly*, March 1982, pp. 61-74.

Checkland, Peter (1981)
Systems Thinking, Systems Practice. Chichester: John Wiley.

Checkland, Peter (1991)
"Organisations as Process: the Organisational Thinking Needed for Effective Groupware." In P.R.H. Hendriks (ed.), *Groupware 1991: The Potential of Team and Organisational Computing*. Utrecht: Software Engineering Research Centre SERC, pp. 15-20.

Checkland, Peter and Alejandro Casar (1986)
"Vickers' concept of an appreciative system: a systemic account." *Journal of Applied Systems Analysis* 13, pp. 3-17.

Checkland, Peter and Jim Scholes (1990)
Soft Systems Methodology in Action. Chichester: John Wiley.

Churchman, C.W., Russell L. Ackoff and E.L. Arnoff (1957)
Introduction to Operations Research. New York: John Wiley.

Cook, Linda K., D.A. Hinkle and T.W. Bickmore (1990)
"Planning for the manufacturing domain: long-term and reactive scheduling." *Proc. first International Conference on Expert Planning Systems*, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 6-10.

Conway, R.W., W.L. Maxwell and L.W. Miller (1967)
Theory of Scheduling. Reading, Massachusetts: Addison-Wesley.

Cyert, Richard M. and James G. March (1963)
A behavioral theory of the firm. Englewood Cliffs, New Jersey: Prentice-Hall.

Darwin, Charles (1859)
The origin of species. (rpt. Harmondsworth: Penguin, 1968).
Davis, Gordon B. and Margrethe H. Olson (1985)
Management Information Systems. 2nd ed., 2nd printing, Singapore: McGraw-Hill.
De Jong: see Jong
De Leeuw: see Leeuw
De Smit: see Smit
De Waele: see Waele
Dhar, Vasant and Nicky Ranganathan (1990)
"Integer Programming versus Expert systems: An Experimental Comparison." *Communications of the ACM* 33:3, pp. 323-336.
Dickson, Gary W., James A. Senn and Norman L. Chervany (1977)
"Research in Management Information Systems: The Minnesota Experiments." *Management Science* 23:9, pp. 913-923.
Dissel, Han G. van, Hans P. Borgman and Adrie J.M. Beulens (1990)
"Task Allocation between DSS and Problem Owner: The Example of Box & Jenkins Time-series Analysis." *Decision Support Systems* 6, pp. 339-345.

Eiben, A.E. (Guszti) (1989)
"Modeling Planning Problems." In J. Demetrovics and B. Thalheim (eds), *Proc. second Symposium on Mathematical Fundamentals of Database Systems.* Berlin: Springer.
Eiben, A.E. (Guszti) (1991)
A Method for Designing Decision Support Systems for Operational Planning. Doctoral Dissertation. Eindhoven: Technological University, Faculty of Mathematics and Computer Science.
Elzas, Maurice S. (1980)
"Simulation and the Processes of Change." In Tuncer I. Ören, C.M. Shub and P.F. Roth (eds), *Proc. IEEE Conference "Simulation with Discrete Models: A State-of-the-Art View"*, Orlando, Florida, pp. 3-18.
Elzas, Maurice S. and John L. Simons (1987)
"Informatiebeleid in theorie en agrarische praktijk." *Informatie* 29:2, pp. 128-151.

Fayol, Henri (1916)
Administration industrielle et générale. Paris: Dunod (rpt. 1950).
Fens, Kees (1990)
"Levenswerk." *de Volkskrant*, 3 februari 1990.
Fiksel, Joseph and Frederick Hayes-Roth (1989)
"Knowledge Systems for Planning Support." *IEEE Expert*, Fall 1989, pp. 16-23.
Fisher, M.L. (1985)
"Interactive Optimization." *Annals of Operations Research* 5:6, pp. 541-556.
Fisher, M.L. and Alexander H.G. Rinnooy Kan (1988)
"The Design, Analysis and Implementation of Heuristics." *Management Science* 34:3, pp. 263-265.
Fordyce, Kenneth, Peter Norden and Gerald Sullivan (1987)
"AI and the Management Science Practitioner: Links between OR and Expert Systems." *Interfaces* 17:4, pp. 34-40.

Fox, Mark S. (1987)
Constraint-Directed Search: A Case Study of Job-Shop Scheduling. San Mateo, California: Morgan Kaufmann.

Fox, Mark S., Norman Sadeh, Can Baykan (1989)
"Constrained Heuristic Search." In Sridharan N.S. (ed.), *Proc. IJCAI 1989*. San Mateo, California: Morgan Kaufmann, pp. 309-315.

Fox, Mark S. and Stephen F. Smith (1984)
"ISIS - a knowledge-based system for factory scheduling." *Expert systems* 1:1, pp. 25-49.

Freuder, Eugene C. (1989)
"Partial Constraint Satisfaction." In Sridharan N.S. (ed.), *Proc. IJCAI 1989*. San Mateo, California: Morgan Kaufmann, pp. 278-283.

Frowein, J. Chris (1990)
Specificatie van expertise. Doctoral Dissertation. Groningen: University of Groningen, department of Business Economics.

Geraedts, Willy J.H. (1986)
Expert Systeem Technieken en Decision Support Systems: een case-study. Wageningen: Agricultural University, Department of Computer Science.

Gerrity, T.P. Jr (1971)
"The Design of Man-Machine Decision systems: an Application to Portfolio Management." *Sloan Management Review*, 21:2, Winter 1971, pp. 59-75.

Gerstendörfer, Monika and Gabriele Rohr (1987)
"Which task in which representation on what kind of interface." In Bullinger H-J. and B. Shackel (eds), *Proc. second IFIP conference Human-Computer Interaction INTERACT '87*, 1-4 September 1987. Amsterdam: North-Holland (Elsevier), pp. 513-520.

Geuze, M.C. (Rien) (1991)
"Agrarisch IT-beleid in de jaren negentig." In Wil H.G.J. Hennen et al. (eds), *Informatica-toepassingen in de Agrarische Sector, Proc. VIAS-Symposium 1991*. Agro-Informatica reeks 5. Wageningen: VIAS, pp. 134-143.

Ginzberg, Michael J. (1978[a])
"Finding an Adequate Measure of OR/MS Effectiveness." *Interfaces* 8:4, pp. 59-62.

Ginzberg, Michael J. (1978[b])
"Redesign of Managerial Tasks: A Requisite for Successful Decision Support Systems." *MIS Quarterly* March 1978, pp. 39-52.

Glaser, B.G. and A.L. Strauss (1967)
The Discovery of Grounded Theory. Chicago: Aldine.

Glover, Fred and Harvey J. Greenberg (1989)
"New approaches for heuristic search: A bilateral linkage with artificial intelligence." *European Journal of Operations Research* 39, pp.119-130.

Gorry, G.A. and R.B. Krumland (1983)
"Artificial Intelligence Research and Decision Support Systems." In J.L. Bennett (ed.), *Buiding Decision Support Systems*. Reading, Massachusetts: Addison-Wesley, pp. 205-220.

Gorry, G.A. and Michael S. Scott Morton (1971)
"A Framework for Management Information Systems." *Sloan Management Review* 13:1, Fall 1971, pp. 55-70.

Gould, John D. (1987)
"How to design usable systems." In Bullinger H-J. and B. Shackel (eds), *Proc. second IFIP conference Human-Computer Interaction INTERACT '87*, 1-4 September 1987. Amsterdam: North-Holland (Elsevier), pp. xxxv-xli.

Gould, John D., Stephen J. Boies and Clayton Lewis (1991)
"Making usable, useful, productivity-enhancing computer applications." *Communications of the ACM* 34:1, pp. 74-85.

Grünwald, Herbert J. and Leonard Fortuin (1989)
"DSS and ES in the 'information organization' - Back to the roots of OR." *European Journal of Operations Research* 41, pp. 142-150.

Hamilton, Scott and Norman L. Chervany (1981)
"Evaluating Information System Effectiveness - Part 1: Comparing Evaluation Approaches." *MIS Quarterly* september 1981, pp. 55-69.

Hartog, Rob J.M. and Wiebe M. Kroon (1991)
"Ervaringen met het gebruik van expertsysteemtechnieken bij het ontwikkelen van gebruikersvriendelijke beslissingsondersteunende systemen voor roosterproblemen." In Wil H.G.J. Hennen et al. (eds), *Informatica-toepassingen in de Agrarische Sector, Proc. VIAS-Symposium 1991*. Agro-Informatica reeks 5. Wageningen: VIAS, pp. 89-95.

Hayes-Roth, Barbara and Frederick Hayes-Roth (1979)
"A Cognitive model of Planning." *Cognitive Science* 3, pp. 275-310.

Hayes-Roth, Barbara, Frederick Hayes-Roth, Stan Rosenschein and Stephanie Cammarata (1979)
"Modelling planning as an incremental, opportunistic process." *Proc. sixth IJCAI*, Los Altos, California: William Kaufmann, pp. 375-383.

Hee, Kees van (1985)
"Informatiesystemen en beslissingsondersteuning." *Informatie* 27:11, pp. 978-986.

Hendriks, Theo H.B (1990)
"Operationele Research." *Agro-Informatica* 3:2, pp. 7-12.

Hillier, F.S. and G.J. Liebermann (1967)
Introduction to Operations Research. San Francisco: Holden-Day (3rd ed. printed 1980).

Hofstede, Geert H. (1968)
The Game of Budget Control. London: Tavistock.

Hofstede, Geert H. (1970)
"Psychological aspects of manager-computer interaction." Unpublished paper presented at the second Simposio de Logica y Filosofia de la Ciencia, Valencia.

Hofstede, Geert H. (1991)
Cultures and Organizations, Software of the Mind: Intercultural Cooperation and its Importance for Survival. London: McGraw-Hill.

Hofstede, Gert Jan (1988[a])
Kwaliteit van DSS: een terreinverkenning. Wageningen: Agricultural University, Department of Computer Science.

Hofstede, Gert Jan (1988[b])
"Cognitive Biases in Human-DSS interaction." In *Proc. International Conference on designing Support Systems*, Noordwijkerhout, November 14-15. Delft: Technological University, department of Information Systems.

Hofstede, Gert Jan (1990[a])
"Wat heet optimaal? OR, probleemtypen en gebruikers." *Agro-Informatica* 3:2, pp. 20-24.

Hofstede, Gert Jan (1990[b])
"Interactive Planning System design, a model and an application." In *Proc. first International Conference on Expert Planning Systems*, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 175-180.
Hofstede, Gert Jan (1990[c])
Datamodel en Database, theorie in praktijk. Muiderberg: Coutinho.
Hofstede, Gert Jan (1990[d])
"Gebruikersvriendelijkheid: Druk op de muis of kat in de zak?" *Agro-Informatica* 3:3, pp. 4-7.
Hofstede, Gert Jan (1990[e])
Handleiding PROPLAN 0.2$^+$. Wageningen: Agricultural University, Department of Computer Science.
Hofstede, Gert Jan and Adrie J.M. Beulens (1992)
"Optimal decision making or optimal trouble-shooting?". In A.J.M. Beulens and H.J. Sebastian (eds), *Optimization-based computer-aided modelling and design*. Berlin: Springer (forthcoming).
Hofstede, Gert Jan and John L. Simons (1987)
First report on the Growers Management Support System. Wageningen: Agricultural University, Department of Computer Science.
Hogarth, Robin (1987)
Judgement and Choice, the Psychology of Decision. 2nd ed. Chichester: John Wiley.
Hogue, Jack T. and Hugh J. Watson (1983)
"Management's Role in the Approval and Administration of Decision Support Systems." *MIS Quarterly*, June 1983, pp. 15-26.
Holwell, Sue E. (1990)
An investigation of systems ideas in information management. Research abstract for the conference ICIS 1990. Lancaster: Lancaster University.
Horssen, Frits P. van (1989)
"Toepassing van informatietechnologie op tuinbouwbedrijven: verwachtingen en visie van een gebruiker." *VIAS Nieuwsbrief* 2:3, pp. 8-11.
Huber, George P. (1983)
"Cognitive style as a basis for MIS and DSS designs: Much ado about nothing?" *Management Science* 29:5, pp. 567-579.
Huizing G.H. and John L. Simons (1980)
Gebruikershandleiding bij het programma VLUCHT (Van LUchtvaart tot Clusters via een Heuristische Techniek). NLR TR 80.xxx. Amsterdam: National Aerospace Laboratory NLR.
Hulzen, J.J.P. van and R.P. de Moel (1987)
"Gedrags- / prestatiekarakteristieken van informatievoorziening." *Informatie* 29:3, pp. 254-261.

INSP-LO Werkgroep 'Plantaardige Produktie' (1986)
Theorie in Praktijk: Toepassing van Informatica in de sector Plantaardige Produktie. Wageningen: Werkgroep 'Plantaardige Produktie' of the INSP-LO.
Institution of Electrical Engineers (1990)
Proc. first Conference on Expert Planning Systems, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers.
In 't Veld: see Veld

Ives, Blake, Margrethe H. Olson and Jack J. Baroudi (1983)
"The Measurement of User Information Satisfaction." *Communications of the ACM* 26:10, pp. 785-793.

Jagodzinski, A.P. and D.D. Clarke (1988)
"A multidimensional approach to the measurement of human-computer performance." *The Computer Journal*, 31:5, pp. 409-419.

Janis, Irving L. and Leon Mann (1977)
Decision Making, a psychological analysis of Conflict, Choice, and Commitment. New York: the Free Press.

Johnson, D.S. and C.H. Papadimitriou (1985)
"Computational complexity." In Eugene L. Lawler et al. (eds), *The Traveling Salesman Problem, a guided tour of combinatorial optimization*. Chichester: John Wiley.

Jong, Richard de and Henk G. Sol (1991)
"Vehicle routing and scheduling support: coping with strict time windows." In Henk G. Sol and J. Vecsenyi (eds), *Environments for supporting decision processes*. Amsterdam: North-Holland, pp. 277-293.

Jongejan, Peder B. (1986)
Van pentomino tot potplant, een bijdrage aan de heuristische oplossing van de produktieplanning op potplantenbedrijven door middel van prototyping. Wageningen: Agricultural University, department of Computer Science.

Jørgensen, Anker H. (1987)
"The trouble with UNIX MAIL: Initial learning and experts' strategies." In Bullinger H-J. and B. Shackel (eds), *Proc. second IFIP conference Human-Computer Interaction INTERACT '87*, 1-4 September 1987. Amsterdam: North-Holland (Elsevier), pp. 847-854.

Kahneman, Daniel, Paul Slovic and Amos Tversky (eds) (1982)
Judgment under Uncertainty: Heuristics and Biases. Cambridge: Cambridge University Press.

Kampfraath, A.A. and W.J. Marcelis (1981)
Besturen en organiseren. Deventer: Kluwer.

Kartam, N.A. and R.E. Levitt (1990)
"A constraint-based approach to construction planning of multi-story buildings." In *Proc. first International Conference on Expert Planning Systems*, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 245-250.

Keane, David G. (1990)
The strategic information gathering behaviour of chief executive officers. Research abstract for the conference ICIS 1990. Victoria: Victoria University, Information Systems Group.

Keen, Peter G.W. (1975)
"Computer-Based Decision Aids: The Evaluation Problem." *Sloan Management Review* 16, spring 1975, pp. 17-29.

Keen, Peter G.W. (1986)
"Decision Support Systems: The Next Decade." In Ephraim R. McLean and Henk G. Sol (eds), *Decision Support Systems: A Decade in Perspective*. Amsterdam: North-Holland, pp. 221-237 (also in *Decision Support Systems* 3:3, pp. 253-265).

Keen, Peter G.W. and Michael S. Scott Morton (1978)
Decision Support Systems: An Organizational Perspective. Reading, Massachusetts: Addison-Wesley.

Kettenis, Dirk L. (1990)
Simulatie. Leiden: Stenfert Kroese.

Kierulf, Ander, Ken Chen, Jurg Nievergelt (1990)
"Smart Game Board and GO Explorer: a study in software and knowledge engineering." *Communications of the ACM* 33:2, pp. 152-166.

Kramer, N.J. and D. de Smit (1982)
Systeemdenken. 3rd ed. Leiden: Stenfert Kroese.

Kruijk, P.A. (1988)
"Hoe verder in de beschermde teelten?" *Landbouwkundig Tijdschrift* 100:5, May 1988, pp. 43-47.

Larsen, Niels E. and Leo Alting (1990)
"Requirements to Scheduling Simulation Systems." In B. Svrcek and J. McRae (eds), *Proc. 1990 Summer Computer Simulation Conference*. San Diego: Society for Computer Simulation, pp. 231-236.

Lawler, Eugene L., Jan Karel Lenstra, Alexander H.G. Rinnooy Kan and David B. Shmoys (1989)
"Sequencing and scheduling: algorithms and complexity." In F. Du Buisson (ed.), *NFI Notes on Designing Decision Support Systems, NFI 11.89/03*, Eindhoven (to appear in Handbooks in OR and Management Science. Amsterdam: North-Holland).

Lawler, Eugene L. and D.E. Wood (1966)
"Branch-and-bound methods: a survey." *Operations Research* 14:4, pp. 699-719.

Leeuw, A.C.J. de (1974)
Systeemleer en Organisatiekunde. Leiden: Stenfert Kroese.

Leeuw, A.C.J. de (1982)
Organisaties: Management, analyse, ontwerp en verandering, een systeemvisie. Assen: van Gorcum.

Lemmen, Wim P. van and G.J. (Bert) Huijskes (1987)
"PROPLAN; een prototype van een planningsinstrument voor de potplantenteelt." In H.A. Arendse and H.F.C.J. Paulissen (eds), *Informatica-Toepassingen in de Agrarische Sector*, Agro-Informatica reeks 1. Wageningen: VIAS, chapter 4 pp. 1-10.

Leutscher, Klaas Jan (1991)
Potplantenteelt. Wageningen: Agricultural University, Department of Horticulture.

Levin, Bernard (1979)
Taking Sides. London: Cape.

Levitt, R.E. (1990)
"Knowledge-Based Planning Systems: an Engineering Perspective." In *Proc. first International Conference on Expert Planning Systems*, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 181-185.

Lewandowski, A. and I. Stanchev (eds) (1989)
Methodology and Software for Interactive Decision Support. Berlin: Springer.

Lindblom, Charles E. (1959)
"The Science of Muddling Through." *Public Administration Review*, 19, 79-88.

Lindley, D.V. (1985)
Making Decisions. 2nd ed., Chichester: John Wiley.

Loosveld, Serge L. (1989)
Een diagnose van randvoorwaarden voor planning binnen PROPLAN. Wageningen: Agricultural University, department of Computer Science.

Macintosh, Norman B. (1985)
The social software of accounting and information systems. Chichester: John Wiley.

Mack, R., C.H. Lewis and J.M. Carroll (1983)
"Learning to use text processing systems." *ACM Transactions on Office Automation*, 1:3, pp. 254-271.

March, James G. and Johan P. Olsen (1976)
Ambiguity and Choice in Organizations. Bergen: Universitetsforlaget.

McArthur, D.S. (1980)
"Decision Scientists, Decision Makers, and the Gap." *Interfaces* 10:1, pp. 110-113.

McKay, K., J.A. Buzacott and F.R. Safayeni (1989)
"The scheduler's knowledge of uncertainty: the missing link." In J. Browne (ed.), *Knowledge Based Production Management Systems*. Amsterdam: Elsevier, pp. 171-189.

McLean, Ephraim R. and Henk G. Sol (eds) (1986)
"Decision Support Systems: A Decade in Perspective." *Proc. IFIP Wg. 8.3 conference June 1986*, Amsterdam: North-Holland.

Meyn, Niels (1921)
Den Lykkelige Ø. København: Jespersens Forlag.

Mietus, Dieta M., René J. Jorna and John L. Simons (1991)
Cognitive strategies in scheduling nurses, an analysis of knowledge components for building decision support systems. Paper presented at the first IFORS Specialized Conference on DSS, 27-29 March 1991. Publication forthcoming.

Migliarese, Piero G. (1985)
"Knowledge representation and organizational noise: an organizational framework about decision support systems." In Leif B. Methlie and Ralph H. Sprague (eds), *Knowledge Representation for Decision Support Systems*. Amsterdam: IFIP / North-Holland.

Millman, Zeeva and Jon Hartwick (1987)
"The Impact of Automated Office Systems on Middle Managers and Their Work." *MIS Quarterly*, Dec. 1987, pp. 479-491.

Milne, A.A. (1926)
Winnie-the-Pooh. (rpt. London: Methuen, 1978).

Minsky, Marvin L. and Seymour A. Papert (1988)
Perceptrons. Expanded ed. Cambridge, Massachusetts: the MIT Press.

Mintzberg, Henry (1973)
The Nature of Managerial Work. New York: Harper and Row.

Mintzberg, Henry (1975)
"The Manager's Job: Folklore and Fact." *Harvard Business Review*, July-August, pp. 49-61; also in Pugh (1984), pp. 417-440.

Mitrani, I. (1982)
Simulation techniques for discrete event systems. Cambridge Computer Science Texts 14, Cambridge: Cambridge University Press.

Mitroff, Ian I., F. Betz, L.R. Pondy and F. Sagasti (1974)
"On managing science in the systems age: two schemas for the study of science as a whole systems phenomenon." *Interfaces* 4:3, pp. 46-58.

Mitroff, Ian I. and Tom R. Featheringham (1974)
"On systemic problem solving and the error of the third kind." *Behavioral Science* 19, pp. 383-393.

Morgan, Gareth (1986)
Images of Organization. Beverly Hills, California: Sage.

Moscarola, Jean (1981)
"l' Efficacité de la décision et de l'aide à la décision dans les organisations." In *Proc. ninth IFORS Conference.* Amsterdam: North-Holland, pp. 91-103.

Moss-Jones, John (1987)
Automating managers: the implications of information technology for managers in five manufacturing companies. Ph.D. thesis. Leeds: Open University.

Müller-Merbach, Heiner (1981)
"Heuristics and their design: a survey." *European Journal of Operational Research* 8, pp. 1-23.

Narula, Subhash C. and H. Roland Weistroffer (1988)
"Algorithms for multi-objective nonlinear programming problems: an overview." In A.G. Lockett and G. Islei (eds), *Improving Decision Making in Organisations,* Lecture Notes in Economics and Mathematical Systems, no. 335. Berlin: Springer.

Näslund, Torbjörn (1992)
Evaluation as a means for improving development of decision support systems. Research report LiTH-IDA-R-92-01, Linköping: Linköping University, Department of Computer and Information Science.

Neelamkavil, F. (1987)
Computer simulation and modelling. Chichester: John Wiley.

Olle, T. William, Henk G. Sol and Alexander A. Verrijn Stuart (eds) (1986)
Information systems design methodologies: Improving the practice. Proc. IFIP WG 8.1 Conference. Amsterdam: North-Holland.

Open Universiteit (1985)
Oriëntatie in de bedrijfskunde. artnr 3014, Heerlen: Open Universiteit.

Ören, Tuncer I., Maurice S. Elzas and Grant Sheng (1985)
"Model reliability and software quality assurance in simulation of nuclear fuel waste management systems." In *Proc. Waste Management Conference 1985,* Tucson, Arizona, pp.381-396.

Ow, Peng Si and Stephen F. Smith (1986)
Towards an opportunistic scheduling system. Unpublished paper presented at the Hawaiian International Conference on Systems Sciences.

Pearl, Judea (1984)
Heuristics: Intelligent Search Strategies for Computer Problem Solving. Reading, Massachusetts: Addison-Wesley.

Pearson, S.W. (1977)
Measurement of Computer User Satisfaction. Ph.D thesis. Tempe: Arizona State University.

Peters, Thomas J. and Robert H. Waterman Jr (1982)
In Search of Excellence. New York: Harper and Row.

Pirsig, Robert M. (1974)
Zen and the art of motorcycle maintenance. New York: Bantam Books.

Ponder, Q.D. (1957)
"The Effective Manufacturing Foreman." In E. Young (ed.), *Proc. of the tenth annual meeting of the Industrial Relations Research Association*. Madison, Wisconsin: Industrial Relations Research Association, pp. 41-54.

Pugh, D.S. (ed.) (1984)
Organization Theory. 2nd ed., Harmondsworth: Penguin.

Rasmussen, Jens (1987)
"Cognitive engineering." In Bullinger H-J. and B. Shackel (eds), *Proc. second IFIP conference Human-Computer Interaction INTERACT '87*, 1-4 September 1987. Amsterdam: North-Holland (Elsevier), pp. XXV-XXX.

Ready, Colin M., W.H. Simmons and J.C. Taunton (1990)
"A knowledge based planning and scheduling toolkit for the process industry." In *Proc. first Conference on Expert Planning Systems*, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 110-113.

Rees, J.R. van (1982)
"De methode doet het niet." *Informatie* 24:2, pp. 81-93.

Rich, Elaine (1983)
Artificial Intelligence. Singapore: McGraw-Hill.

Rich, Elaine and Kevin Knight (1991)
Artificial Intelligence. 2nd Edition. New York: McGraw-Hill.

Rinnooy Kan, Alexander H.G. (1976)
Machine Scheduling Problems: Classification, complexity and computations. Doctoral Dissertation. Amsterdam.

Robert, Jean-Marc (1987)
"Learning a computer system by unassisted exploration." In Bullinger H-J. and B. Shackel (eds), *Proc. second IFIP conference Human-Computer Interaction INTERACT '87*, 1-4 September 1987. Amsterdam: North-Holland (Elsevier), pp. 651-656.

Saarinen, Timo (1990)
"System development methodology and project success: an assessment of situational approaches." *Information & Management* 19, pp. 183-193.

Saedt, Anton P.H. (1982)
"Decision support by linear programming under changing circumstances in a potplant nursery." In C-H. Hanf and G.W. Schiefer (eds), *Proc. of the third Symposium of the European Association of Agricultural Economists (EAAE)*. Kiel: Kieler Wissenschaftsverlag Vauk, pp. 134-144.

Saedt, Anton P.H. (1988)
IMAG Productie Planning (IPP) in de praktijk. Report no. 381. Wageningen: Institute of Agricultural Engineering IMAG.

Saedt, Anton P.H., Theo H.B. Hendriks and F.M. Smits (1986)
Production planning with a microcomputer in potplant nurseries. Report no. 197. Wageningen: Institute of Agricultural Engineering IMAG.

Savelsbergh, Martin W.P. (1988)
Computer aided routing. Amsterdam: CWI.

Sayles, L.R. (1964)
Managerial Behavior: Administration in Complex Organizations. New York: McGraw-Hill.

Schaik, Frans D.J. van (1988)
Effectiveness of Decision Support Systems. Delft: Delft University Press.
Schiefer, G.W. (1991)
"Information technology in agriculture - Situation and perspectives." In Wil H.G.J. Hennen et al. (eds), *Informatica-toepassingen in de Agrarische Sector, Proc. VIAS-Symposium 1991*. Agro-Informatica reeks 5. Wageningen: VIAS, pp. 9-15.
Scott Morton, Michael S. (1971)
Management Decision Systems: Computer Based Support for Decision Making. Cambridge, Massachusetts: Harvard University Press.
Serafini, Paolo and Walter Ukovich (1989)
"A Decision Support System Based on the Job Shop Problem in Real Manufacturing." In A. Lewandowski, I. Stanchev (eds), *Methodology and Software for Interactive Decision Support*. Berlin: Springer, pp. 202-210.
Shneiderman, Ben (1987)
Designing the User Interface. Reading, Massachusetts: Addison-Wesley.
Silver, Edward A., R. Victor. V. Vidal and Dominique de Werra (1980)
"A tutorial on heuristic methods." *European Journal of Operational Research* 5, pp.153-162.
Simon, Herbert A. (1947)
Administrative Behavior. New York: the Macmillan Company.
Simon, Herbert A. (1960)
The New Science of Management Decision. New York: Harper and Brothers.
Simon, Herbert A. (1965)
The Shape of Automation for Men and Management. New York: Harper and Row.
Simon, Herbert A. (1981)
The Sciences of the Artificial. 2nd ed. Cambridge, Massachusetts: the MIT Press.
Simon, Herbert A. (1987)
"Two heads are better than one: the collaboration between AI and OR." *Interfaces* 17:4, pp. 8-15.
Simons, John L. (1987)
"Expert systemen: Mode of node?" In H.A. Arendse and H.F.C.J. Paulissen (eds), *Informatica-toepassingen in de Agrarische Sector, Proc. VIAS-Symposium 1987*. Agro-Informatica reeks 1. Wageningen: VIAS, chapter 10 pp. 1-10.
Simons, John L. (1988)
"Integratie van Expert System technieken in Decision Support Systemen." In Cees Huijbers et al. (eds), *Informatica-toepassingen in de Agrarische Sector, Proc. VIAS-Symposium 1988*. Agro-Informatica reeks 2. Wageningen: VIAS, pp. 17-25.
Simons, John L. (1989)
Over chaos en orde in de bestuurlijke informatiekunde. Inaugural Lecture. Groningen: University of Groningen, Faculty of Management Theory.
Simons, John L. and Roel J. Wieringa (1986)
The place of expert systems in a typology of information systems. Wageningen: Agricultural University, Department of Computer Science.
SITU (1989)
Het gedetailleerde informatiemodel glastuinbouw, cluster taktische en strategische planning. Honselersdijk: SITU.
SITU (1991)
Jaarverslag over 1990. Honselersdijk: SITU.

Smith, Gerald F. (1988)
"Towards a heuristic theory of problem structuring." *Management Science* 34:12, pp. 1489-1506.

Smith, Stephen F., Mark S. Fox and Peng Si Ow (1986)
"Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems." *AI Magazine*, Fall 1986, pp. 45-61.

Sol, Henk G. (1982)
Simulation in information systems development. Doctoral Dissertation. Groningen: University of Groningen, department of Business Economics.

Sol, Henk G. (1987)
De gebruiker in model. Delft: Technical University, Department of Information Systems.

Sol, Henk G. (1989)
"Het Ontwikkelen van Informatiesystemen: Een probleem oplossende benadering." In Cees Huijbers et al. (eds), *Informatica - toepassingen in de Agrarische sector, Proc. VIAS-Symposium 1989*. Agro-Informatica reeks 3. Wageningen: VIAS, pp. 9-21.

Sol, Henk G. (1991)
"Information systems to support decision processes." In Henk G. Sol and J. Vecsenyi (eds), *Environments for supporting decision processes*. Amsterdam: North-Holland, pp. 115-127.

Sprague, Ralph H. Jr and Hugh J. Watson (eds) (1989)
Decision Support Systems, Putting Theory into Practice. 2nd ed. Englewood Cliffs, New Jersey: Prentice-Hall.

Stabell, Charles B. (1983)
"A Decision-oriented Approach to Buiding DSS." In J.L. Bennett (ed.): *Buiding Decision Support Systems*. Reading, Massachusetts: Addison-Wesley, pp. 221-260.

Stainton, R.S. and D.B. Papoulias (1984)
"Heuristics - the relational approach." *European Journal of Operational Research* 17, pp. 16-20.

Starbuck, William H. and Jane Webster (1991)
"When is play productive?". *Accounting, Management and Information Technologies* 1:1, pp. 71-90.

Streitz, Norbert A., Will A.C. Spijkers and Lidewij L. van Duren (1987)
"From novice to expert user: a transfer of learning experiment on different interaction modes." In Bullinger H-J. and B. Shackel (eds), *Proc. second IFIP conference Human-Computer Interaction INTERACT '87*, 1-4 September 1987. Amsterdam: North-Holland, pp. 841-846.

Takkenberg, Cees A.Th. (1983)
Planning en methode van onderzoek. Doctoral Dissertation. Groningen: University of Groningen, department of Business Economics.

TIPSY (1990)
De Contouren van een OnderzoeksNetwerk: verslag derde promovendidagen informatiesystemen. Den Haag: Werkgemeenschap Informatiesystemen SION.

Todd, Peter and Izak Benbasat (1987)
"Process Tracing Methods in Decision Support Systems Research: Exploring the Black Box." *MIS Quarterly* dec. 1987, pp. 493-512.

Tolstoy, Leo (1878)
Anna Karenin (rpt. Harmondsworth: Penguin, 1954 / 1978).

Tyszka, Tadeusz (1986)
"Information and evaluative processes in decision-making: the role of familiarity." In B. Brehmer et al. (eds), *New Directions in Research on Decision Making*. Amsterdam: Elsevier, pp. 151-161.

Van der Ven: see Ven

Van Dissel: see Dissel

Van Hee: see Hee

Van Horssen: see Horssen

Van Hulzen: see Hulzen

Van Lemmen: see Lemmen

Van Rees: see Rees

Van Schaik: see Schaik

Vegter, B. (1990)
"Pas echt nut van PC bij bedrijfsvergelijking." *Vakblad voor de Bloemisterij 45:3*, pp. 92-95.

Veld, J. in 't (1981)
Analyse van Organisatieproblemen, een toepassing van denken in systemen en processen. 3th ed. Amsterdam: Elsevier.

Vellekoop, Arie H. (1989)
Decision Support for the analysis of economic risk. Doctoral Dissertation. Rotterdam: Erasmus University, Econometrical Institute.

Ven, Michael B.M. van der (1989)
A group decision support system for transfer pricing in the pharmaceutical industry. Doctoral Dissertation. Delft: Technological University, department of Information Systems.

Verbeek, Peter J. (1991)
Learning about Decision Support Systems, two cases on manpower planning in an airline. Research Series no. 7. Rotterdam: Tinbergen Institute / Amsterdam: Thesis Publishers.

Verbraeck, Alexander (1990[a])
"Adaptive production planning environments using decision rules and simulation." *Proc. first International Conference on Expert Planning Systems*, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 147-151.

Verbraeck, Alexander (1990[b])
"A Decision Support System for timetable construction: automated lesson planning using a special timetabling algorithm." *Proc. first International Conference on Expert Planning Systems*, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 207-211.

Verbraeck, Alexander (1991)
Developing an adaptive scheduling support environment. Doctoral Dissertation. Delft: Technological University, department of Information Systems.

Verbraeck, Alexander and Henk G. Sol (1988)
Software ontwikkeling bij BOS-bouw. Report no 88-76. Delft: University of Delft, department of Information Systems.

Vickers, sir Geoffrey (1961)
"Judgement." *The Manager* January 1996, pp. 31-39 (Rpt. In D.S. Pugh (ed.) (1984), *Organization Theory*. 2nd ed. Harmondsworth: Penguin, pp. 183-201).

Vlek, Charles A.J. (1989)
"Modellen en methoden voor het begrijpen en verbeteren van beslissingen." In Cees Huijbers et al. (eds), *Informatica - toepassingen in de Agrarische sector, Proc. VIAS-Symposium 1989.* Agro-Informatica reeks 3. Wageningen: VIAS, pp. 207-224.

Von Winterfeldt: see Winterfeldt

Waele, Wim de (1990)
"ReDS: A planning and scheduling environment for manufacturing." In *Proc. first International Conference on Expert Planning Systems*, 27-29 June 1990. IEE Conference Publication 322. London: the Institution of Electrical Engineers, pp. 11-15.

Wagner, H.M. (1969)
Principles of Operations Research with applications to managerial decisions. Englewood Cliffs, New Jersey: Prentice-Hall.

Wassink, J.G.J. and T.J.F. van Rossum (1990)
"Managementinformatie: inventariseren of ontwerpen?" In *Informatie*, 32:3, pp. 264-275.

Waterman, D.A. (1986)
A Guide to Expert Systems. Reading, Massachusetts: Addison-Wesley.

Weber, E. Sue and Benn R. Konsynski (1987)
"Problem Management: Neglected Elements in Decision Support Systems." *Journal of Management Information Systems* 4:3, pp. 64-81.

Werkgroep MBP (1990)
Vooronderzoek Management Beheersysteem Potplanten. Report no. P87129. Den Hoorn: Agro-Adviesburo Bram Marrewijk.

Wilkins, D.E. (1988)
Practical Planning: Extending the Classical AI Planning Paradigm. San Mateo, California: Morgan Kaufmann.

Willmott, Hugh, Jan Mouritsen, Per Flensburg, and Bente Elkjær (1990)
"Systems developers: preoccupations, knowledge and power." *Proc. International Conference on Information Systems.* Baltimore, Maryland: ACM, pp. 257-264.

Winston, Patrick H. (1977)
Artificial Intelligence. Reading, Massachusetts: Addison-Wesley.

Winterfeldt, Detlof von and Ward Edwards (1986)
Decision Analysis and Behavioral Research. Cambridge: Cambridge University Press.

Wisse, Jaco (1989)
PC-PROPLAN versie 0.0. Wageningen: Agricultural University, department of Computer Science.

Ziggers, Gerrit W. (1991)
"De tuinder: 'Een informatiesysteem, wat kan ik er mee?'." *Agro-Informatica* 4:4, pp. 10-13.

Author index

Subject index

About the author

Gert Jan Hofstede was born in Hengelo (Overijssel), the Netherlands, on 29 September 1956. He attended secondary school in Switzerland and in the Netherlands. He then read population biology at Wageningen Agricultural University. In 1983 he graduated with hydrobiology, plant ecology, philosophy and ethology as his main subjects. A qualified high school teacher of biology, he then joined Volmac Toptraining b.v. and worked as a COBOL programmer.

In 1985 he accepted a position with the department of Computer Science of Wageningen Agricultural University, his present employer. There he developed a course on data modelling and SQL and authored a textbook on the subject (*Datamodel en database: theorie in praktijk.* Muiderberg: Coutinho, 1990). He has for a number of years now acted as a secretary to the multidisciplinary research program 'Decision Support Systems in agriculture and horticulture' of Wageningen University. From 1988 to 1990 he was editor of *Agro-Informatica*, the journal of the Dutch Society for Informatics in Agriculture (VIAS). Since 1990 he has been a member of the research committee of his department. He authored and co-authored a number of publications in various media. These cover the subject of this book as well as others.

He is a member of the Dutch Decision Support Systems group, in particular the Team Interactive Planning Systems (TIPSY), an informal working group which he co-founded. He is also a member of the working groups on Human-Computer Interaction and on information systems for management (EIS/MIS/DSS) of the Dutch Society for Informatics. He participates in the Euro Working Group on DSS. Having been elected to participate in the Doctoral Consortium of the International Conference on Information Systems in 1990 he was a Doctoral Fellow of the Society for Information Management during that year.

He is married with three daughters.