Titles in the Series

M. Shaw, R. Blanning, T. Strader and A. Whinston (Eds.)
Handbook on Electronic Commerce
ISBN 978-3-540-65882-1

J. Blazewicz, K. Ecker, B. Plateau and D. Trystram (Eds.)
Handbook on Parallel and Distributed Processing
ISBN 978-3-540-66441-3

H. H. Adelsberger, B. Collis and J.M. Pawlowski (Eds.)
Handbook on Information Technologies for Education and Training
ISBN 978-3-540-67603-8

C. W. Holsapple (Ed.)
Handbook on Knowledge Management 1
Knowledge Matters
ISBN 978-3-540-43527-3

Handbook on Knowledge Management 2
Knowledge Directions
ISBN 978-3-540-43527-3

J. Blazewicz, W. Kubiak, T. Morzy and M. Rusinkiewicz (Eds.)
Handbook on Data Management in Information Systems
ISBN 978-3-540-43893-9

P. Bernus, P. Nemes, G. Schmidt (Eds.)
Handbook on Enterprise Architecture
ISBN 978-3-540-00343-4

S. Staab and R. Studer (Eds.)
Handbook on Ontologies

S. O. Kimbrough and D.J. Wu (Eds.)
Formal Modelling in Electronic Commerce
ISBN 978-3-540-21431-1

P. Bernus, K. Merlins and G. Schmidt (Eds.)
Handbook on Architectures of Information Systems
ISBN 978-3-540-25472-0 (Second Edition)

S. Körn, O. Herzog, P. Lockemann, O. Spaniol (Eds.)
Multiagent Engineering
ISBN 978-3-540-31406-6

J. Blazewicz, K. Ecker, E. Pesch, G. Schmidt, J. Weglarz
Handbook on Scheduling

Frada Burstein
Clyde W. Holsapple
(Editors)

Handbook on Decision Support Systems 2
Variations
We are sure the chapter authors also appreciate the time and effort these people spent to referee manuscripts submitted for potential inclusion in this book, offering insights, critiques, and suggestions for improvements. They undoubtedly share the pride in the value of this final product. We extend a special thanks Ms. Kate Lazarenko for her assistance in keeping this project on track and attending to details essential for achieving the quality of this publication. Finally, we acknowledge Dr. Werner Mueller and Springer's publishing team for their encouragement, patience, and assistance in completing this sizable project.

Frada Burstein and Clyde W. Holsapple (Editors), *Handbook on Decision Support Systems*

Table of Contents

VOLUME 2: VARIATIONS

Preface ................................................................. VII

Contributors to Volume 2 .......................................... XXXI

PART VI. TIME AND SPACE ISSUES FOR DECISION SUPPORT

CHAPTER 37
Decision Support in Turbulent and High-Velocity Environments .......... 3
Sven A. Carlsson and Omar A. El Sawy

CHAPTER 38
Supporting Decisions in Real-Time Enterprises:
Autonomic Supply Chain Systems ........................................ 19
Daniel E. O'Leary

CHAPTER 39
Decision Support for Emergency Situations ................................ 39
Bartel Van de Walle and Murray Turoff

CHAPTER 40
Geographic Information and Analysis for Decision Support ............... 65
Peter B. Keenan

CHAPTER 41
Support for Real-Time Decision Making in Mobile
Financial Applications .................................................. 81
Frada Burstein, Julie Cowie, Arkady Zaslavsky
and Jocelyn San Pedro

CHAPTER 42
Context-Sensitive Decision Support Systems in Road Safety .......... 107
Patrick Brézillon and Juliette Brézillon
### PART VII. SCOPES OF DECISION SUPPORT

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>Title</th>
<th>Authors</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>43</td>
<td>Personal Decision Support Systems</td>
<td>David Arnott</td>
<td>127</td>
</tr>
<tr>
<td>44</td>
<td>Developing Practical Decision Support Tools Using Dashboards of Information</td>
<td>Frédéric Adam and Jean-Charles Pomerol</td>
<td>151</td>
</tr>
<tr>
<td>45</td>
<td>Business Intelligence</td>
<td>Solomon Negash and Paul Gray</td>
<td>175</td>
</tr>
<tr>
<td>46</td>
<td>Competitive Intelligence Systems</td>
<td>Vicki L. Sauter</td>
<td>195</td>
</tr>
<tr>
<td>47</td>
<td>Process-Based Decision Support</td>
<td>Dina Neiger and Leonid Churilov</td>
<td>211</td>
</tr>
</tbody>
</table>

### PART VIII. DEVELOPING AND MANAGING DECISION SUPPORT SYSTEMS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>Title</th>
<th>Authors</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>DSS Systems Analysis and Design: The Role of the Analyst as Change Agent</td>
<td>Kenneth E. Kendall and Julie E. Kendall</td>
<td>293</td>
</tr>
</tbody>
</table>

### PART IX. DECISION SUPPORT CASES AND APPLICATIONS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>Title</th>
<th>Authors</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>Evaluation of Decision Support Systems</td>
<td>Cheul Rhee and H. Raghav Rao</td>
<td>313</td>
</tr>
<tr>
<td>52</td>
<td>Planning a Portfolio of Decision Support Systems</td>
<td>Albert L. Lederer</td>
<td>329</td>
</tr>
<tr>
<td>53</td>
<td>Decision Support System Evolution: Predicting, Facilitating, and Managing Knowledge Evolution</td>
<td>Daniel E. O'Leary</td>
<td>345</td>
</tr>
<tr>
<td>54</td>
<td>Systems for Supporting Operations Management Decisions</td>
<td>Anita Lee-Post and Chen H. Chung</td>
<td>371</td>
</tr>
<tr>
<td>55</td>
<td>Systems for Supporting Marketing Decisions</td>
<td>Mike Hart</td>
<td>395</td>
</tr>
<tr>
<td>57</td>
<td>Flying High with Real-Time Business Intelligence</td>
<td>Ron Anderson-Lehman, Hugh J. Watson, Barbara H. Wixom and Jeffrey A. Hoffer</td>
<td>443</td>
</tr>
<tr>
<td>59</td>
<td>DSS in Healthcare: Advances and Opportunities</td>
<td>Rajiv Kohli and Frank Piontek</td>
<td>483</td>
</tr>
<tr>
<td>CHAPTER</td>
<td>Title</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Data Quality and Decision Making</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Decision Makers and Their Need for Support</td>
<td>83</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Decision Support Through Knowledge Management</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Decision Support Systems: A Historical Overview</td>
<td>121</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Reference Disciplines of Decision Support Systems</td>
<td>141</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>DSS Architecture and Types</td>
<td>163</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Integrated Document Management for Decision Support</td>
<td>191</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Databases and Data Warehouses for Decision Support</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Model Management and Solvers for Decision Support</td>
<td>231</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Online Analytical Processing (OLAP) for Decision Support</td>
<td>259</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Spreadsheet-Based Decision Support Systems</td>
<td>277</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Multi-Criteria Decision Support</td>
<td>299</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Web-Based Decision Support</td>
<td>315</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Collaborative Technologies</td>
<td>341</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>A Motivational Model of Knowledge Sharing</td>
<td>355</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>The Nature of Group Decision Support Systems</td>
<td>371</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>GDSS Parameters and Benefits</td>
<td>391</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>The Nature of Organizational Decision Support Systems</td>
<td>415</td>
<td></td>
</tr>
</tbody>
</table>
# Table of Contents

## PART IV. INTELLIGENT DECISION SUPPORT SYSTEMS

### CHAPTER 24
Advisory Systems to Support Decision Making
Brandon A. Beemer and Dawn G. Gregg

### CHAPTER 25
Using Autonomous Software Agents in Decision Support Systems
Traci J. Hess, Loren Paul Rees and Terry R. Rakes

### CHAPTER 26
Artificial Neural Networks in Decision Support Systems
Dursun Delen and Ramesh Sharda

### CHAPTER 27
Data Mining and Data Fusion for Enhanced Decision Support
Shiraj Khan, Auroop R. Ganguly and Amar Gupta

### CHAPTER 28
Decision Support via Text Mining
Josh Froelich and Sergei Ananyan

### CHAPTER 29
Decision Support Based on Process Mining
Wil M.P. van der Aalst

### CHAPTER 30
Adaptive Decision Support Systems
via Problem Processor Learning
Clyde W. Holsapple, Varghese S. Jacob, Ramakrishnan Pakath, and Jigish S. Zaveri

## PART V. EFFECTS OF COMPUTER-BASED DECISION SUPPORT

### CHAPTER 31
A Learning-Enhanced Adaptive Decision Support System Framework
Michael Shaw and Selwyn Piramuthu

### CHAPTER 32
Benefits of Decision Support Systems
Roger Alan Pick

### CHAPTER 33
Involvement and Decision-Making Satisfaction with a Decision Aid: The Influence of Social Multimedia, Gender, and Playfulness
Traci J. Hess, Mark A. Fuller and John Mathew

### CHAPTER 34
Decision Support Systems Failure
David Arnott and Gemma Dodson

### CHAPTER 35
Could the Use of a Knowledge-Based System Lead to Implicit Learning?
Solomon Antony and Radhika Santhanam

### CHAPTER 36
Group and Organizational Learning Effects from Multiparticipant DSS Usage
M. Shane Tomblin

## Keyword Index
CHAPTER 49
On the Design Features of Decision Support Systems:
The Role of System Restrictiveness and Decisional Guidance

Mark S. Silver
Fordham University, New York, NY, USA

This chapter is about the substantive design of DSS features. It begins with a set of five premises that are fundamental for designing DSSs yet are often neglected in the prescriptive literature. Because a DSS is an intervention into the processes by which decisions are made, and because the ultimate outcome of DSS design is not the system itself but the system's consequences, the key question for designers to keep in mind is this: What will the decision maker do with the system? Contemplating this question leads to two key design features of DSS: system restrictiveness and decisional guidance. System restrictiveness refers to how a DSS limits decision makers who rely on it to a subset of all possible decision-making processes. Decisional guidance refers to how a DSS enlightens, sways, or directs decision makers as they choose and use its functional capabilities. Together these two features play a significant role in determining whether a DSS will successfully achieve its design objectives while avoiding undesirable side-effects. The chapter explores how a system's restrictiveness and decisional guidance can be defined by designers to achieve their design objectives as well as how DSS features can restrict and guide.

Keywords: Decision support system; Decision-making process; Restrictiveness; Decisional guidance; Design; Consequences; Constraints; Features

1 Introduction

This chapter addresses the design of computer-based information systems intended to support human decision makers, commonly referred to as decision support systems (DSSs). The chapter is written with those who develop such systems in mind. Most of what is written here is not inconsistent with conventional thinking on DSS design, but the emphasis here is not what one typically finds in the prescriptive DSS design literature and my perspective differs from that which is most commonly adopted in practice. Since the discussion here is limited by the constraints of a single chapter, interested readers – practitioners or researchers – can turn to Silver (1991b) for a more formal and extensive treatment of this material.
Design is both a process and a product (Walls et al. 1992, Hevner et al. 2004). The design process consists of those activities that system builders perform while developing systems, whereas the design product is the DSS that results from those efforts. Although process and product are certainly connected, this chapter concentrates on the product. To distinguish it from the design process, the design product might be termed the system’s substantive design. The substantive design of a DSS comprises the system’s design features.

Any use of the terms features or design features requires some qualification, because these words mean different things to different people. Some use the term feature to refer to a system’s functional capabilities — its information-processing features. Others would include among a system’s features the elements of its human-machine interface, such as its menu structures and dialog styles. Some use the term features narrowly to refer to what others (DeSanctis et al. 1994, Griffith 1999) more specifically call optional or tangential features, functions that might or might not be present in a given system of a given type. Still others think of features as the bells and whistles that implementers and vendors use to embellish the systems they produce. Additional meanings abound. Given these many meanings, I use the terms features and design features in the broadest sense possible, to refer to all of a decision support system’s components, characteristics, and properties, including, but not limited to, its information-processing functions and elements of its user interface.

Much of the prescriptive literature on DSS design addresses the process, not the product of design. Such classic works as those by Keen and Scott Morton (1978), Sprague and Carlson (1982), and Bennett (1983) concentrate on distinguishing how systems that support decision makers should be designed from how other systems of the time were being developed. But these works, and those that followed in the subsequent decades, tell DSS developers little about the substantive design of DSS — about the appropriate design features for a given DSS. Over the years we have also seen, and we continue to see, a great volume of design research focused on specific decision-aiding technologies. While this literature does address substance rather than process, it too offers little substantive advice for developers on which design features — functional capabilities, interface elements, and so forth — are appropriate for a given system to produce. Additional meanings abound. Given these many meanings, I use the terms features and design features in the broadest sense possible, to refer to all of a decision support system’s components, characteristics, and properties, including, but not limited to, its information-processing functions and elements of its user interface.

The chapter rests on several premises that discussions of DSSs often neglect. Most of these premises are based on the process view of decision making, the recognition that decision making is not a point event but a sequence of decisional activities. Each premise has important implications for the substantive design of DSSs.

Premise 1: A DSS is an intervention into the process through which decisions are made (Silver 1991b).

Unlike those computer-based systems intended to replace people, DSSs are constructed to assist human decision makers. Indeed, the defining element of a DSS for most commentators is that the system supports, rather than replaces, the human decision maker. But a DSS is not just an information-processing assistant. Because system use will likely change the way decisions are made, the DSS is an intervention into the process through which decisions are made. The computer-supported process will differ — perhaps dramatically — from the unaided one. The implication of this observation for DSS design is that designers must contemplate System restrictiveness has implications for various substantive design decisions, including which functional capabilities to include in a DSS, which options to provide with each of those capabilities, and how to package the capabilities into a system. The second attribute, decisional guidance, reflects those aspects of the system that influence (intentionally or not) the selections decision makers make as they employ the discretionary power afforded them by the DSS.
how the features they design are likely to affect the path decision makers will follow in arriving at a decision.

Consider, for example, decision makers confronting a multi-attribute decision-making task, such as locating a warehouse, renting an apartment, or buying a vehicle. This classic decision problem is characterized by the need to choose among a (possibly large) set of alternatives, each described by a set of attributes. Many solution strategies — sometimes called choice rules — for tackling this problem have been observed and classified. DSSs can intervene in the multi-attribute decision-making process by affecting which of the many strategies users employ (Todd and Benbasat 2000).

Premise 2: The ultimate outcome of the DSS design process is not the computer-based system but its effects.

Contrary to widespread belief, the DSS itself is not what should be of greatest importance to DSS designers. After all, DSSs are not built to be admired or exhibited; they are constructed to be used by decision makers. Of ultimate interest, therefore, are the system’s consequences for its users and for others. These consequences include the system’s effects on the decision-making process, the decisions made through that process, and the ramifications of those decisions. These consequences need to be the true foci of DSS design. The substantive design implication is that designers must focus on system features not for their own sake but for their anticipated consequences.

Consider, once again the multi-attribute decision-making problem and the many choice rules that could be used to solve it. Some of these choice rules are compensatory, allowing high scores on one attribute to compensate for low scores on another, whereas others are non-compensatory. Employing a compensatory approach would likely lead to a different choice than would a non-compensatory rule — for instance, a different warehouse location. And this decision could have many ramifications, such as how delivery trucks are routed and how well retail outlets are stocked. Since the mix of decision aids included in a DSS would likely affect which choice rule (or combination of rules) the user of such a system adopts (Todd and Benbasat 2000), this design decision can have significant consequences. These consequences need to be the ultimate outcome of the design process.

Premise 3: The consequences of a DSS are not necessarily those intended by the designer.

System designers may try to engender a given set of consequences when they construct features, but since decision makers may not behave as designers expect (Griffith 1999), the intended consequences may not be realized. Moreover, even if the designer’s intended effects are realized, the decision-maker’s behavior may also lead to unanticipated side-effects. The substantive design implication is that focusing only on achieving planned consequences is not sufficient.

One of the early DSSs documented in the literature (Gerrity 1971, Stabell 1974) was a portfolio management system intended to improve the performance of portfolio managers by enabling them to look at buy/sell decisions from the perspective of the entire portfolio rather than that of individual securities. The system was considered successful, although the system did not succeed in changing the decision-making behavior of the account executives. The system’s success was an unplanned side-effect, as the company obtained a competitive advantage from the system for other reasons.

This premise reminds us that the consequences of a DSS are not necessarily positive. Not only might the hoped-for positive effects not be realized, but system use may even lead to negative consequences. Various studies (for instance, Fripp 1985, Kottemann and Remus 1987, Elam and Mead 1990) have found evidence of degraded or dysfunctional performance using a DSS. So the downside risk of building a DSS is not just that it may fail to achieve its intended benefits; the greater risk is that intervening in a decision-making process may be deleterious. When one thinks of design objectives, one tends to think of the positive accomplishments pursued by the designer. But more completely, the designer wants to achieve the intended effects without incurring negative side-effects. While designers understandably may tend to focus on achieving the effects they desire, paying attention to only these consequences is ill-advised. Designers must also anticipate and avoid undesirable consequences.

Premise 4: DSSs can affect the structure or the execution of the decision-making process.

Since many decision-making tasks can be performed in more than one way, an important element of decision making is deciding how to decide — choosing among the different decision-making processes that can be followed. This strategy selection might be made at the outset or it might evolve as the decision maker confronts the problem. Either way, DSS designers must differentiate this structuring of the decision-making process from the execution of the process — actually engaging in the information-processing activities that lead to the decision. For instance, when confronting a multi-attribute decision task, structuring the process consists of selecting a choice rule or constructing a hybrid of such rules. Applying the chosen rule(s) — for instance, selecting attributes, setting cutoff values, and eliminating alternatives in the elimination-by-aspects approach — constitutes executing the process. Each of these activities — structuring and executing the process — can be the source of degraded decision-making behavior and each can be amenable to computer-based support. Many designers take the structuring for granted and focus only on process execution, which can lead to unexpected effects as decision makers use the DSS to follow strategies not anticipated by the designer.

2 Sometimes DSSs have consequences of interest that are not directly related to the decisions that emerge from their use, such as shifts in organizational power, structure, or culture.
Premise 5: The design features of a DSS — or any computer-based information system, for that matter — are not limited to the technical properties of the artifact.

While system builders typically concentrate on a system’s technical features, such other elements of the DSS as usage policies, training, ongoing support, and cost are also design features of the system that can be defined by the design process. Considering this more complete set of features matters, because consequences follow not just from the technological properties of the system but also from the social and economic factors that surround it. For instance, if using a system is voluntary, some decision makers might use it and others might not. Training and ongoing support might have an effect on whether or not people use the system and how they perform if they use it. Alternatively, a policy of mandatory use might address the problem of non-use but might lead to misuse or abuse of the same technical system.

Table 1 summarizes these five premises and their implications for DSS design. Taken together, they lead us to see DSS design this way:

DSS design is a process wherein designers define a system’s (technical and non-technical) features in an attempt to affect the structure and execution of the decision-making process so as to achieve desired design objectives while avoiding undesirable side-effects. In short, the DSS design process is about producing a substantive design whose consequences will conform to the project’s design objectives.

Table 1. Key premises and their implications for DSS design

<table>
<thead>
<tr>
<th>Premise</th>
<th>Implications for substantive design</th>
</tr>
</thead>
<tbody>
<tr>
<td>A DSS is an intervention into the process through which decisions are made.</td>
<td>Designers must contemplate how the features they design are likely to affect the path decision makers follow in arriving at a decision.</td>
</tr>
<tr>
<td>The ultimate outcome of the DSS design process is not the computer-based system but its effects.</td>
<td>Designers must focus on system features not for their own sake but for their anticipated consequences.</td>
</tr>
<tr>
<td>The consequences of a DSS are not necessarily those intended by the designer.</td>
<td>Designers must contemplate how substantive design features are likely to affect whether or not design objectives are achieved as well as how those features may lead to undesirable side-effects.</td>
</tr>
<tr>
<td>DSS can affect the structure of the execution of the decision-making process.</td>
<td>Designers must distinguish the structuring of the decision-making process from its execution and consider how the system features will likely affect each.</td>
</tr>
<tr>
<td>The design features of a DSS are not limited to its technical properties. They include social and economic elements as well.</td>
<td>Designers must pay attention to these non-technical features, recognizing that, in combination with the technical features, they can influence consequences.</td>
</tr>
</tbody>
</table>

3 DSS Design Features and Decision-Making Behavior

When considering a system’s technical features, it is natural for DSS designers and users alike to focus on the system’s functional capabilities, asking, what can the system do? But because what ultimately matter are the system’s consequences, the real question is this:

What will the decision maker do with the system?

This question decomposes into a sequence of two more focused ones:

1. What is the decision maker capable of doing with the system?
2. Given those capabilities, what will he or she do?

The answers to these questions are tightly linked to the system’s design features. The first question reminds us that the system’s features affect both what a decision maker can and cannot do with the system. We tend to think of DSSs as enabling, because their functional capabilities augment the human decision maker’s limited information-processing capabilities. But DSSs are both enabling and constraining. Since any given system has a finite set of functional capabilities, when decision makers rely on a given system to support their decision making that system’s feature set limits what they can do. Designers who are concerned about the consequences of the systems they build — and that should be all DSS designers — need to look at the design from both perspectives — from that of empowerment and from that of restrictiveness.

The second question highlights that, despite the system’s constraints, decision makers are still likely to have significant opportunities for exercising discretion. For instance, decision makers can typically choose among functional capabilities, supply inputs to those capabilities, and select display formats for viewing the outputs. Indeed, any system so limiting as to allow the decision maker no discretion would not qualify as a support system. So, just as some design features affect what the decision maker can do, other features affect what the decision maker does do while exercising the discretion he or she is granted. Designers who are concerned about the consequences of the systems they build must also consider, therefore, how the system’s features may influence the way that decision makers choose among the uses and use the system’s functional capabilities.

Taken together, the answers to these two questions bring us to the central relationship between substantive design features and decision-making behavior:

The design features of a DSS can play a role: (1) in restricting what a decision maker can do when employing a given system, and (2) in guiding what he or she opts to do within the limits of those restrictions.

This reasonably intuitive conclusion conflicts with the typical designer view of a system. Designers typically focus on providing useful functionality — that is, defining things the decision maker can do with the system — without paying
much attention either to what the decision maker cannot do or to what the decision maker will opt to do. This overly narrow analytic focus can lead to systems whose consequences are not those intended by the designer, are undesirable, or both. Treating the system’s restrictiveness and its guidance as key design variables is therefore a better developmental approach.

4 System Restrictiveness

Because DSSs enhance the human’s own limited information-processing capabilities with a set of computer-based capabilities, one can easily contemplate DSSs only in terms of their enabling power. But since DSSs provide decision makers with a given set of functional capabilities, DSSs enable some decision-making processes while restricting other processes. For instance, a DSS might implement only a few of the many algorithms for time-series forecasting. Or a DSS might support currency conversions among dollars, euros, and yen, but not pounds sterling. Similarly, a DSS might accept only non-negative input values for growth rates, disallowing the possibility of contraction.

Since enablement and constraint are essentially inverses — what is constrained is not enabled and what is enabled is not constrained — considering both sides of this coin explicitly might seem unnecessary when designing a DSS. By this logic, focusing on one will take care of the other. In particular, since defining what the system can do is a necessary design task, the functional capabilities are, after all, the sine qua non of the system — one could develop a DSS without paying any attention to how it constrains. Although this exclusive focus on enablement is often seen in practice, several factors suggest that paying design attention to a system’s restrictiveness is also essential.

Contemplating a system’s restrictiveness encourages designers to ask key questions that might otherwise go unasked, such as, “Should this capability be restricted?” or “Would it be a mistake to restrict that function?” But differently, focusing on the constraints makes the designer explicitly consider what should and what should not be constrained, which may be essential for achieving desired consequences or for avoiding undesirable side-effects. Since the designer is actively building features into the system, he or she might very well fail to think in terms of what has been, or should be, excluded. Focusing on restrictiveness can help avoid these type I and type II design errors (omitting capabilities that should be included or including capabilities that should be omitted).

Consider a DSS that is used as part of a planning process and that embeds a simulation model. Since decision makers often fall prey to the wishful thinking bias, whereby their predictions are overly optimistic, one design objective might be to produce more realistic predictions by preventing such bias. A system designed to meet this objective might restrict the allowable values of some inputs and might also limit the number of times the model can be run (to prevent gaming of the system). Even if such debiasing were not an explicit objective of the system, designers might want to build these restrictions into the system to prevent the wishful thinking bias from cropping up as an undesirable side-effect of model use. In this example, the system’s constraints — what cannot be done with the system — are critical design features.

Restrictiveness can directly affect system consequences, since the absence of a function or an option may force decision makers to follow different decision-making processes than they would otherwise have selected. But system restrictiveness can also affect behavior and outcomes by influencing a decision maker’s attitude toward the system. On the one hand, a decision maker might choose not to use a highly restrictive DSS if he or she finds it overly constraining. On the other hand, a decision maker might abandon a minimally restrictive DSS if its many functions and options make it difficult for him or her to use.

System restrictiveness is a design variable — a system feature under the control of the designer — that must be carefully set. Much of the early literature on DSS development prescribed flexibility (Sprague and Carlson 1982) — the opposite of restrictiveness. Later work (Silver 1990, 1991b) argued that a balance between flexibility and restrictiveness is required, where the balance depends on the objectives of the specific DSS being developed. Some design objectives favor greater flexibility (enablement) whereas others favor greater restrictiveness.

Exploring the role of system restrictiveness as a design variable has two parts: (1) considering how a given DSS design objective can be met by enabling or restricting decision makers, and (2) contemplating how the system’s technical properties can accomplish the desired enablement or restriction. But before engaging either of these issues, a formal definition of system restrictiveness is in order.

System restrictiveness: the manner in which a decision support system limits its users’ decision-making processes to a subset of all possible processes.

Restrictiveness thus defined is a multifaceted quality of a DSS, because systems can restrict in many ways and because many aspects of decision-making processes can be restricted. Indeed, one often cannot say which of two DSSs is the more restrictive, except in those cases where the processes supported by one system are a proper subset of those supported by the other. Consider, for example, one DSS that supports time-series forecasting and another that supports multi-attribute decision making. The processes supported by the two systems are so different that it might not make sense to refer to one system as more restrictive than the other. Even given two DSSs for the same task — say time-series forecasting — one cannot generally identify the more restrictive of the two. Suppose that one DSS supports several forecasting methods while the other supports several different

---

3 I defer until later in this section the matter of a given DSS restricting different decision makers differently.

4 The technology acceptance model (Davis et al. 1989), for example, posits that perceived usefulness and perceived ease-of-use will affect intention to adopt an information system.
methods. It would not be appropriate to characterize either as more restrictive than the other. As it turns out, however, DSS designers commonly find themselves in situations where one proposed design is strictly more restrictive than the other because designers frequently are in the position of choosing between more restrictive and less restrictive versions of the same system. Given an initial design, for example, the designer may contemplate adding or removing functional capabilities or options, thereby making the design less or more restrictive, respectively.

4.1 Designing System Restrictiveness

In many — perhaps, most — contexts, restrictions are considered bad. People generally do not like blackout dates on discount airline travel or 15-minute limits on parking meters. One might conclude, therefore, that system restrictiveness is an undesirable quality and that DSS designers should avoid restricting. Indeed, users of early DSSs rebelled against the inflexibility of those systems. But the claim that restrictiveness is inherently bad is mistaken. Sometimes restrictiveness is bad, in the sense that it works against a designer's objectives, and sometimes it is good, in the sense that it promotes those objectives. Moreover, whether we like it or not, whether we choose to think about it or not, systems are inherently restrictive. Any computer-based system — other than a Turing machine — has a limited set of capabilities. So the design question is not so much whether or not to restrict, but how, and how much, to restrict. Table 2 summarizes the factors that favor lesser restrictiveness and those that favor greater restrictiveness. The following discussion introduces these factors; more extensive discussions can be found in Silver (1991b).

Table 2. Design objectives favoring greater and lesser restrictiveness

<table>
<thead>
<tr>
<th>Design objectives favoring greater restrictiveness</th>
<th>Design objectives favoring lesser restrictiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Promoting use</td>
<td>Promoting use</td>
</tr>
<tr>
<td>Prescription</td>
<td>Meeting unspecified needs</td>
</tr>
<tr>
<td>Prescription</td>
<td>Supporting changing decision-making environments</td>
</tr>
<tr>
<td>Providing structure</td>
<td>Supporting multiple decision makers and tasks</td>
</tr>
<tr>
<td>Promoting ease of system learning and use</td>
<td>Allowing users discretion</td>
</tr>
<tr>
<td>Fostering structured learning</td>
<td>Fostering creativity</td>
</tr>
<tr>
<td></td>
<td>Fostering exploratory learning</td>
</tr>
</tbody>
</table>

On the Design Features of Decision Support Systems

4.1.1 Objectives Favoring Lesser Restrictiveness

The groundbreaking work on DSSs by Keen (1980), Sprague and Carlson (1982), and others offers several reasons for designing less restrictive DSSs. Much of this work recognized that the rigidity of the failed management information systems (MISs) and even the early DSSs was not conducive to supporting human decision makers engaged in problem-solving activities. Effective support for human decision makers in many cases favors less restrictiveness.

Meeting unspecified needs. Decision makers often are unable to express their needs adequately during the analysis and design stages of system development. They may only be able to determine which functional capabilities are useful and which are not by actually using the DSS. This difficulty may be due, in part, to the underspecified nature of decision-making tasks. One way that system builders can cope with underspecified needs is to build a generalized and flexible system that provides a broad spectrum of capabilities from which decision makers can choose. This approach leads to minimally restrictive systems. After all, designers would be unable, or unwise, to produce a very limited system in a situation where specific needs cannot be defined a priori.

Supporting changing decision-making environments. Over time, decision makers, the problems they confront, and the organizational and industrial settings within which they operate all change. Indeed, the DSS itself may contribute to some of these changes as decision makers' understanding of their tasks and situations evolves through use of the system (Keen 1980). If the DSS is intended to have longevity — to continue to be useful as the elements of the decision-making environment change — the DSS must be sufficiently robust to support both the current and future decision-making environments. Such durability will require more flexibility and less restrictiveness in the system's design.

Supporting multiple decision makers and tasks. Some DSSs are custom designed for a given task and, possibly, for a small set of specific individuals. But such highly tailored DSSs — once a hallmark of computer-based decision support — are increasingly giving way to more generalized systems intended to support multiple tasks and decision makers. If a DSS is to support a wide range of tasks, each with its own information-processing requirements, and a wide range of decision makers, each with his or her own individual characteristics, the system will likely need to be less restrictive rather than more.

Allowing discretion. Computer-based support systems attempt to combine the information-processing power of the computer with the strengths of the human decision maker. Taking advantage of these human strengths requires granting the decision maker opportunities to exercise his or her discretion. This discretion may relate to structuring the decision-making process — deciding how to decide — or to executing that process. Since increased restrictiveness reduces the opportunities for exercising discretion, designers may want to create less restrictive systems to
allow for the expression of more human judgment and preferences. For instance, they might build systems that support more than one approach to solving the problem or that provide more opportunities for human input into the process as it proceeds.

**Fostering creativity.** For many decision-making tasks, especially those where decision makers formulate their own solutions as opposed to choosing among ready-made alternatives, fostering the creativity of the human decision maker is an important objective. Highly restrictive systems are likely to stifle such creativity, so this design objective, too, favors more flexibility and less restrictiveness.

**Fostering exploratory learning.** Since the early days of DSSs (see, for instance, Alter 1980), gaining a better understanding of the tasks they confront has been recognized as a potentially valuable byproduct of DSS use for decision makers. Allowing users to experiment with a system’s functionality can foster such understanding through exploratory learning. For example, giving a decision maker a free hand to examine the contents of a data warehouse—slicing, dicing, drilling down, and so forth—may be a useful way to help him or her understand the business better. Running a broad range of scenarios through a simulation model similarly can help someone get a better feel for the dynamics of the business. So flexibility, rather than restrictiveness, may be most appropriate when builders desire to promote such exploratory learning.

### 4.1.2 Design Objectives Favoring Greater Restrictiveness

If one views a decision support system merely as an information-processing assistant, one might have difficulty seeing why designers would want to restrict the decision-making process. But recognizing that a DSS is also an intervention into the process through which decisions are made, we find several significant reasons designers might want, or need, to restrict those processes.

**Prescription.** DSS interventions are often intended to impose a given decision-making process. Several different reasons for prescribing can be identified. Sometimes the intent is to prescribe a normative or preferred approach. For instance, a portfolio management system might prescribe processes consistent with normative portfolio theory or a DSS for multi-attribute decision-making tasks might prescribe the normative linear additive model. Similarly, some managers may have their own preferred way of performing a task—say, predicting sales or selecting products to promote—and those managers may require that their subordinates follow the preferred approach. Even in the absence of a normative or preferred process, prescribing a single process may benefit the organization by creating consistency or standardization of decision making. A bank may want to ensure that all of its many loan officers follow the same, standard procedure for granting or denying loans, thus reducing risk. The given method may not be deemed superior to other, equally acceptable methods, but the uniformity of decision making is beneficial to the bank.

**Proscription.** Some DSSs restrict for the opposite reason, not to require but to preclude various forms of decisional behavior—especially, current behavior. For instance, a DSS might be intended to move managers away from the flawed process they are currently following, or it might be designed to break managers out of their routines and make them more innovative and creative. In both these cases, a DSS could be constructed that constrains its users from following the current process. Proscription, however, is not always for the sake of changing current behavior; it may be intended to prevent detrimental use of a system’s capabilities—that is, system use that would degrade decision-making performance. A DSS might try to prevent its users from stumbling into some of the many pitfalls to which human decision makers are vulnerable. Human decision makers, for instance, are known to fall prey to numerous systematic cognitive biases (Tversky and Kahneman 1974) that distort their judgment. A system could limit its users’ opportunities for making such biased judgments and choices.

**Promoting structure.** Deciding how to decide is often one of the most difficult aspects of decision making. Given a powerful DSS with many functions and options, decision makers may struggle to structure an effective decision-making process. They may suffer from information overload if too many information sources are available. They may similarly be overwhelmed if too many functional capabilities are available. A more-restrictive DSS that provides the decision maker with a specific path to follow may sometimes be more effective than a less-restrictive system that requires the decision maker to structure his or her own process.

**Promoting ease of system learning and use.** One of the tenets of DSS development is that DSSs should be easy to learn and use. But highly flexible, minimally restrictive systems tend to be more complex with more features that the decision maker needs to learn, remember, and be able to use effectively. One way to make such DSS easier to learn, remember, and use is to make them more restrictive, limiting the functions, options, and so forth.

**Fostering structured learning.** While exploratory learning, where decision makers gain knowledge of their environment by experimenting with a system, is sometimes an added objective of DSS design, other times DSS are intended to help decision makers in a more structured way to understand their decision-making environment and the decisions they confront. Greater restrictiveness may be valuable in providing users with a structured learning experience. For instance, a restrictive system could march decision makers through a given solution approach. Or it could take decision makers on a structured tour of a database, helping them understand how to analyze the data that are available to them.

### 4.1.3 Restrictiveness and Promoting Use: A Double-Edged Sword

One obvious objective common to virtually all information systems is that the system be used. Promoting use is especially significant for DSSs because, unlike transaction-processing systems, use of DSSs is often discretionary (Bennett 1983). In terms of promoting use, restrictiveness can be a double-edged sword. Too little or too much restrictiveness might inhibit use. Too much restrictiveness may make the decision maker feel overly constrained and reject the system. For instance, the system might preclude the decision maker from following his or her favorite
process. Or the many constraints imposed by the system might frustrate him or her. Since one can easily see how great restrictions may inhibit use, one might assume that great flexibility would promote use. Indeed, this idea was a theme of the early DSS literature. However, unlimited flexibility also has its shortcomings. A decision maker given a very powerful and flexible system may be overwhelmed by its capabilities. Just as decision makers can suffer from information overload, they might suffer from an overload of functional capabilities or an excess of options while using those capabilities. Learning, remembering, and using the system may be made more effortful by the complexity that follows from the lack of restrictiveness. Here, too, users may choose not to use the system.

Table 2 therefore categorizes promoting use both as a factor favoring greater and as a factor favoring lesser restrictiveness. Positioning the restrictiveness properly to promote use may be challenging. Setting the restrictiveness appropriately may be even more challenging if promoting use conflicts with other design objectives. The more specific decisional objectives of the system may call for more, or less, restrictiveness than is desirable to promote use and may, consequently, inhibit use. For instance, building a restrictive system to prescribe a given behavior may be fruitless if that restrictiveness inhibits use. This conundrum creates a difficult situation for the designer, to which I shall return later.

4.2 How do DSS Design Features Restrict Decision Makers?

Deciding which process restrictions and how much process restrictiveness are appropriate to achieve design objectives is one thing; designing the system to restrict the decision-making process as desired is another. How do DSS design features restrict decision makers? In general, constraints imposed on the technological properties of the DSS restrict the decision-making behavior of those who use the system. A system’s technological properties can be constrained in various ways.

Information-processing functions (information-processing capabilities). Any given DSS makes available to its users some set of information-processing capabilities, such as searching for information, analyzing data sets statistically, solving optimization problems, calculating financial indicators, graphing trends, and so forth. This set of functional capabilities—functions, for short—is constrained in the sense that it is a proper subset of the infinite set of information-processing capabilities that could possibly be built into a system. And since these constraints on the set of functional capabilities restrict what the decision maker can do with the system; they restrict his or her decision-making process. If the system does not support a given information-processing function, then decision makers who rely on the system cannot follow a process that requires that capability. In the case of time-series forecasting, for example, many forecasting methods have been proposed, but only a subset of the many methods might be included in the system. In the case of multi-attribute decision-making tasks, only a subset of the many possible choice rules might be included. An electronic spreadsheet package might be provided to users without such add-ins as linear regression.

In addition to limiting the set of functional capabilities available, the design of a DSS may also impose constraints on how those functional capabilities are used in combination one with another. Such restrictions may prohibit some uses or require others. The most restrictive systems—relatively uncommon today—require users to march through the system’s capabilities in a predefined order as a fixed sequence of steps. Other, less-restrictive systems constrain in various ways how decision makers combine functions. Some functions may be prerequisites for others; use of the second function may depend on the successful completion of the first. Some functions may be alternatives; if the decision maker uses one, he or she cannot use the other. Some functions may be incompatible; here, too, decision makers who use one cannot use the other. In many cases, the dependencies between functions may be conditional. Decision makers who project unbalanced budgets may be required to run the projection again or to invoke a budget-balancing procedure. Decision makers who want to run a multiple regression analysis on autocorrelated data may be required to transform the data first. Even researchers, practitioners, and users rebelled against the inflexibility of the first interactive systems, this design dimension (restrictions on combining functions) has largely been neglected. But such restrictions can play significant roles in how the DSS affects decision-making behavior and, therefore, in the ultimate success or failure of the system.

Data sets. The data processed by the functional capabilities come from essentially three sources. The data may be provided by the system, the data may be entered directly by the user, or the data may be imported by the user from elsewhere. Each of these data sources can be restricted. The most restrictive systems in this regard provide decision makers with no choice concerning the data to be

5 Understanding the constraints of a system’s functional capabilities is made more complicated by the distinction between low- and high-level functions. If a system provides access to users to low-level functions that provide basic computational power—such as a built-in macro language—or matrix algebra processor—then an enterprising user might employ those functions to implement higher-level capabilities not directly supported by the system. For instance, matrix algebra could be used to perform a least-squares regression in the absence of a dedicated regression function and an electronic spreadsheet formula could be built to calculate multi-attribute utility in the absence of such a built-in function. Designers who intend to restrict high-level functions need to consider whether decision makers can work around these constraints by employing lower-level functions.

6 When the DSS provides the data it may get these data from various sources. Where a DSS gets these data raises a complex set of design issues that does not matter for our purposes here. All data that are provided by the system, rather than the user, are therefore grouped together in this discussion.
employed, limiting them to a single data set provided by the system. Most DSSs, however, are not so restrictive. Systems can be less constraining by including multiple data sets, by allowing decision makers to enter their own data, or by enabling them to import external data. When decision makers are allowed to enter or import data, restrictiveness may still be operative. For instance, decision makers might be limited in the number of datasets they can create, the size of those datasets, or even their contents.

Systems that limit data sources, DSSs can also constrain data use. A DSS might not allow all data to be used with all functional capabilities. Some data sets might be local to particular functions — for their use only — while other data sets might be global to the entire system. Conversely, some functions might accept data from any of the three sources, while others might be limited to a single source, such as data embedded in the function. Restrictions can be even more detailed; some specific data sets might be allowed with one function and not another. Specific restrictions might relate to the conformance between the data structure and the function, but decisional reasons may also drive such restrictions. For instance, a system intended to foster forward-looking, rather than backward-looking, analyses might allow certain data to be used by forecasting models but not to be perused directly by users. To encourage more environmental scanning, some functions might work only with external data as opposed to internal corporate data. For different tasks, data with different attributes (for instance, different levels of aggregation or time horizons) may be appropriate (Gorry and Scott Morton 1971). Data access might also be limited for organizational reasons. Some data may be deemed confidential and viewable by some decision makers and not others. For instance, as corporate executive support systems were migrated to divisional levels, issues arose concerning who should be allowed to see which data (Houdeshel and Watson 1987).

Models. Many DSSs offer model-based functions, such as solvers for linear programs or forecasters of time-series data. DSS models vary widely, including optimization models, simulation models, statistical models, and choice models (rule choices), among others. Discuss how model-based functions can be restrictive it is useful to distinguish model types from a specific instance of a model type. For our purposes, a model type refers to a class of models that have similar mathematical properties and, in particular, can be operated upon by the same functional capabilities. For instance, linear programs are a type of model, whereas a given linear program is an instance of that model type. An implementation may be different in terms of an algorithm designed to work on any linear program or even to provide a new one. Similarly, a spreadsheet model would be highly restrictive if its formula cells were protected so that users could change inputs but not the structure of the model. By itself, however, an electronic spreadsheet package provides a highly non-restrictive modeling capability.

Parameters. Parameters that provide option settings or values for key variables on the way important roles in determining the information-processing behavior of a functional capability. For instance, a statistical function that performs hypothesis testing needs to be given the level at which to reject the null hypothesis. A stepwise multiple regression is driven by the critical r-value. Database searches are controlled by a set of search options. The elimination by aspects choice rule requires an ordered set of attributes and associated cutoff values. Highly restrictive systems may predefine these values and options, granting the decision maker no opportunity to change them. Less-restrictive systems may empower the decision maker to set or change these parameters. The setting of options and key variables is often an integral part of the decision-making process, since human decision makers often bring their judgments and preferences to bear through the setting of these parameters. Constraining a function’s parameterization limits the decision maker’s discretion and his or her ability to make judgments and express preferences. Such constraints are likely to affect the decisions made and their consequences.

Visual representations. A large body of research has found that how the information provided by a DSS is displayed visually can have a significant impact on the behavior of decision makers and on the decisions they make. Much of this research has focused on the differential effects of tabular versus graphical display of information, but within each of these two broad categories one finds much constraints. Other model types — spreadsheet models come to mind — are less structured. Two spreadsheet models can be very different one from another.

Model-based functions can vary greatly in their restrictiveness. The most restrictive embed a specific instance of a model, allowing the decision maker to provide, at most, a few parameter values. A decision maker is allowed to change these parameters, but only within the range of acceptable values for a numeric variable. Such restrictions are likely to affect the decisions made and their consequences.

See Geoffrion (1989) for a more formal discussion of model classes and Silver (1991b) for a more formal discussion of models in the context of DSSs.
room for variability of display formats. Microsoft Excel, for example, offers not only a large selection of chart types but a great many variants and options for each of those types. The consequences of a given display format — for instance, the performance of the decision maker — may depend on such factors as the characteristics of the task and of the decision maker.

DSSs vary in terms of how they constrain the visual representations available to decision makers. Some DSSs are highly restrictive, predefining displays that decision makers cannot change. This approach typified the early executive information systems, which enabled senior executives to peruse hundreds of unalterable information-packed predefined displays. Other DSSs give users greater flexibility, allowing them to select among alternative displays or even to create their own. For instance, someone who is not satisfied with any of Excel’s many formats for displaying numerical values can devise his or her own format. Since the fit between information-packed predefined displays. Other DSSs give users greater flexibility, allowing them to select among alternative displays or even to create their own. For instance, someone who is not satisfied with any of Excel’s many formats for displaying numerical values can devise his or her own format. Since the fit between display, the task, and the decision maker may affect performance (Vessey 2006, Goodhue 2006, Te’eni 2006), one can easily see how limitations on visual representations can be consequential. On the one hand, allowing decision makers to select their own representations might increase the fit of the system with the person. But on the other hand, restricting the display to those that fit the task might be more beneficial. Visual representations present a good example of both the significance and the challenge of restrictiveness as a design variable.

Table 3 summarizes the principal ways in which a system’s technical properties can be constrained. These constraints on the system’s technical features might have the effect of restricting the structure of the decision-making process, its execution, or both. Constraints on the system’s information-processing capabilities generally restrict the structure of the decision-making process — that is, how the decision maker goes about making the decision. Similarly, constraints on the data sets and models that the functions employ, as well as constraints on the visual representations they produce, also tend to limit the process structure. Constraints on parameters, however, such as limits on setting values and selecting options, which essentially restrict a decision maker’s discretion when using a functional capability, usually restrict the execution of the process — that is, what the decision maker does as he or she proceeds along the decision-making path.

4.3 Absolute Versus Actual Restrictiveness: A Paradox

Thus far, I have treated system restrictiveness in absolute terms as a property of the system without considering what a given user experiences. But a given DSS might restrict different decision makers differently. For instance, a given decision maker might be unaware of some of the system’s capabilities. Another decision maker might be incapable of operating a given function. Still another might find learning or using a given function too effortful to be worthwhile. The decision maker in each of these cases is more restricted than another decision maker who takes full advantage of the system’s capabilities. So when discussing system restrictiveness, absolute restrictiveness, what the hypothetically least-restricted individual experiences, must be distinguished from actual restrictiveness, what a given individual experiences.

The relationship between absolute and actual restrictiveness can be paradoxical. As one decreases the absolute restrictiveness of a system, by reducing the constraints on its technical properties, one may be increasing the system’s complexity. This increased complexity may degrade usability, as the effort required to learn and use the system is increased. This decrease in usability may increase the restrictiveness of the system as experienced by many users, as these users may be unable to operate some of the system’s capabilities. So reducing absolute restrictiveness may, in some cases, increase actual restrictiveness. We might refer to this as the restrictiveness paradox.

Figure 1 captures somewhat abstractly the likely relationship between absolute and actual restrictiveness, although the precise relationship would vary from one person to another. A highly restrictive system in absolute terms (on the far right)
will also be experienced as highly restrictive by most people. After all, if the system has highly constrained capabilities, all users will be greatly limited in what they can do. Reducing the absolute restrictiveness (moving to the left) should, up to some point, be expected to reduce the restrictiveness actually experienced by the system's constraints are loosened. But at some point — which could well differ from person to person — the effect is likely to be reversed. Beyond that point (shown, for simplicity, as the midpoint in the diagram), further decreases in absolute restrictiveness will lead to so much increased effort for the decision maker that he or she will experience a return to a higher level of actual restrictiveness. The restrictiveness paradox produces a likely U-shaped relationship between absolute and actual restrictiveness. These observations may help explain why promoting use is an objective that can favor lesser as well as greater restrictiveness (Table 2). Since either too much or too little absolute restrictiveness can lead to substantial actual restrictiveness, either can inhibit use.

The restrictiveness paradox makes designing DSSs especially challenging, because designers may think they are enabling decision makers with enhanced power and flexibility but may actually be restricting those decision makers by degrading usability. Designing a system to support many decision makers may be even more challenging given differences in their perceptions of the system’s restrictiveness. Such non-technical features of the system as training and ongoing technical support may be useful for bringing absolute and actual restrictiveness more in line with each other. For instance, training can be used to make decision makers aware of the full set of system capabilities as well as to reduce the effort required to learn and use the system. So a system’s actual restrictiveness may be influenced by non-technical features as well as the system's technical properties.

While restrictiveness can obviously play a significant role in determining effects — since it defines what a decision maker can and cannot do with the system — it is not the only design variable that matters. It leaves open the question of what the decision maker will actually do (given what the system allows). To address this question we need to consider another design feature — decisional guidance.

## 5 Decisional Guidance

One way to think of restrictiveness is that it limits the decision maker’s ability to exercise discretion while using the system. But all support systems grant decision makers some discretion — otherwise they would not be support systems — and many systems grant substantial discretion. DSSs grant decision makers discretion in structuring their decision-making processes when they allow them to choose functional capabilities, data sets, models, and visual representations. DSSs grant decision makers discretion in executing those processes when they allow them to interact with the functional capabilities — for instance, by supplying parameter values and selecting options. The system’s effects on the decision-making process, on the decisions that follow from it, and on the repercussions of those decisions, will be influenced by how decision makers use the discretion they are granted, which, in turn, will be influenced by elements of the system’s design. Collectively, those elements that play a role in this influence are referred to as the system’s decisional guidance. Like system restrictiveness, decisional guidance is a multifaceted design feature. It is defined formally as follows:

**Decisional guidance:** The manner in which a DSS, intentionally or not, enlightens, sways, or directs its users as those users exercise the discretion the system grants them to structure and execute their decision-making processes (adapted from Silver 1991a, 1991b, 2006).

The definition of decisional guidance is noteworthy for several reasons. First, note the modifier *decisional*. Its purpose is to distinguish decisional guidance, which plays a substantive role in the choices decision makers make, from such mechanical guidance as help screens and other interface features that assist users with the mechanics of operating the system. Second, note the role of intention — or lack thereof. Decisional guidance need not be intentional. Like system restrictiveness, decisional guidance is a design variable. However, just as systems restrict even if the designer does not contemplate the system’s restrictiveness, systems may guide even if the designer does not intend to. For instance, menu items are often arranged alphabetically to enhance usability, but this order might increase the likelihood of decision makers selecting those items that appear earlier in the alphabetical listing. Some systems place the most recently used menu items at the top of the
list — again to enhance usability — but this design might have the unintended effect of reinforcing the human tendency to use the same approach repeatedly rather than to consider alternatives. So designers must acknowledge both inadvertent and deliberate guidance.

Third, even when decisional guidance is deliberate, it does not necessarily direct decision makers. Decisional guidance might be directive, strongly pointing the decision maker toward a given action, or it might be milder, intending to sway the decision maker, or it might be neither of these. The purpose of the guidance might be to enlighten decision makers, allowing them to make a more-informed selection without trying to bias them. This, too, is a form of decisional guidance because it can affect how a decision maker exercises discretion. In common usage, guidance is often thought of as directive, but the term is used here more broadly to include the informative as well as the suggestive.

Lastly, the relationship between a system’s restrictiveness and its decisional guidance is well defined. Restrictiveness defines what decision makers can do. Subject to those restrictions, decisional guidance affects what decision makers actually do. One sometimes hears people — even information system (IS) designers and researchers — speak of system constraints guiding behavior. Strictly speaking, however, constraints restrict rather than guide — they limit what is possible. Guidance influences how decision makers behave subject to those constraints. Failing to heed the distinction between restrictiveness and guidance can lead to great confusion in discussions of system design. Figure 2 shows the relationship; in short, restrictiveness limits the opportunities for guiding.

Although system restrictiveness and decisional guidance are two different design dimensions, decisional guidance raises a pair of questions analogous to those raised by restrictiveness: “What objectives would motivate a DSS designer to deliberately try to guide decision makers?” and “How can a system’s design features guide decision makers, deliberately or inadvertently?”

### 5.1 Objectives that Motivate Providing Deliberate Decisional Guidance

The design objectives that might motivate designers to build deliberate decisional guidance into a DSS, summarized in Table 4, are quite varied. Notice that these objectives include, but are not limited to, most of the design objectives that favor greater restrictiveness, since guidance is sometimes an alternative means of achieving the same design objectives as restrictiveness.

#### 5.1.1 Providing Greater Support

The human-machine partnership that characterizes DSS use is often viewed as follows: The human decision maker controls the information processing, provides judgments, and expresses preferences whereas the computer-based system processes information in support of those human activities. One purpose of deliberately providing decisional guidance may be to provide greater support than just that of a powerful information-processing assistant. A DSS can help guide decision makers as they control the system, choosing among its various functional capabilities and options. It can help guide decision makers as they form judgments, such as assessing probabilities or identifying causal relationships. And it can even help guide decision makers as they clarify and express their preferences — for instance, preferences for one alternative over another. Whenever decision makers are called upon to interact with the DSS — to exercise discretion — the system might provide decisional guidance that either tenders information to enlighten their behavior or offers suggestions as to how to behave. Of course, designing such decisional guidance can be quite challenging for the system builder.
5.1.2 Prescribing, Proscribing, Structuring, and Fostering Structured Learning

We have already identified a variety of design objectives that favor greater restrictiveness. In particular, restrictiveness is often worth considering when a DSS is intended to influence the decision-making process by prescribing some behavior or proscribing some other behavior. Similarly, designers may restrict decision makers to help them structure their decision-making processes or to foster structured learning. But we have also encountered a variety of factors that discourage such restrictiveness. Too much restrictiveness can, for example, inhibit use, reduce the system’s longevity, limit the environments to which it can be applied, remove the decision maker’s discretion, and so forth. Decisional guidance can be a means of getting the best of both worlds. Rather than restricting decision makers, a DSS could attempt to sway or direct them via suggestive guidance. Since decision makers would not be compelled to follow the system’s suggestions, this approach would likely be somewhat less effective than restrictiveness at accomplishing some of these design objectives. But it would allow for a less-restrictive system that could simultaneously accomplish objectives associated with lesser restrictiveness.

5.1.3 Promoting Usability

As I have noted several times already, less-restrictive systems can be difficult to learn, remember, and use. While extreme flexibility sounds good in theory, in practice too much flexibility can also be debilitating. Indeed, the restrictiveness paradox suggests that as systems become less and less restrictive in absolute terms, they become increasingly restrictive in actuality for a given decision maker. Decisional guidance — either informative or suggestive — can help reduce the effort associated with minimally restrictive systems. By providing decision makers with additional information about their options, or even suggesting how to proceed, the usability of the system can be increased. This may even help to bring actual restrictiveness more in line with absolute restrictiveness, thus resolving the restrictiveness paradox.

5.1.4 Avoiding Inadvertent Guidance

Designers often fail to recognize that, even if they do not intend to influence how users behave, the features they design may do so, often in subtle ways. For instance, saliency effects of menu items — such as the primacy effect, where people opt for the first items listed — can be the source of systematic biases in the choices decision makers’ make with a DSS. In some sense, inadvertent guidance is the default when designers do not include deliberate guidance in a system. Avoiding the inadvertent consequences of a guidance vacuum therefore constitutes another reason for deliberately building guidance into a system.

5.2 How a System’s Design Features Can Deliberately Guide Decision Makers

The concept of deliberately providing decisional guidance is fairly intuitive to understand, but designing the guidance mechanisms can be challenging and complex. Decisional guidance varies along several dimensions. One such dimension is the source of the substantive content of the guidance. Specific information or suggestions might be predefined by the designer and built into the system. Or the content might be created dynamically by the guidance mechanism based on the decision maker’s behavior with the system. A DSS might even interact with the user to produce more tailored and sophisticated guidance. We might refer to these as the modes of guidance: predefined, dynamic, and participative. Guidance mechanisms also vary with respect to how the guidance is invoked. It might be provided automatically by the DSS or it might be offered on-demand by the decision maker.

Table 5 summarizes the various dimensions that characterize deliberate decisional guidance. Here I focus on the two dimensions that stand out as the keys to understanding decisional guidance mechanisms: the targets and the forms of the guidance. See Silver (1991a, 2006) for a more complete discussion of all the dimensions.

Table 5. Dimensions of deliberate decisional guidance (Silver 2006)

<table>
<thead>
<tr>
<th>Targets</th>
<th>Directivity</th>
<th>Modes</th>
<th>Invocation styles</th>
<th>Timing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choosing functional capabilities</td>
<td>Informative guidance</td>
<td>Predefined</td>
<td>Automatic</td>
<td>Concurrent</td>
</tr>
<tr>
<td>Using functional capabilities</td>
<td>Suggestive guidance</td>
<td>Dynamic</td>
<td>On-demand</td>
<td>Prospective</td>
</tr>
<tr>
<td>Quasi-suggestive guidance</td>
<td>Informative guidance</td>
<td>Participative</td>
<td>Hybrid</td>
<td>Retrospective</td>
</tr>
</tbody>
</table>
5.2.1 Targets of Guidance: Structuring Versus Executing the Decision-Making Process

Both the structuring and the executing of decision-making processes may be amenable to decisional guidance. When using a DSS, decision makers structure their decision-making processes by choosing which functional capabilities to employ as well as which data sets, models, and visual representations to use with those functional capabilities. Decision makers are sometimes challenged to choose one of many possible functions that serve the same purpose but will likely lead to different outcomes. For instance, when decision makers need to create forecasts from time-series data, they may find numerous forecasting methods available. Similarly, when choosing among alternatives in a multi-attribute decision-making task, they may be empowered to select from among various choice rules. In meta-choice situations such as these — where the decision maker is choosing a solution technique — the designer may want to provide guidance that either informs the decision maker of the relative merits of each approach or even suggests to the decision maker the most appropriate function given the specific circumstances. Guidance could similarly be offered for selecting among data sets, models, and visual representations.

In less-structured DSSs, the challenge may not be to choose among the solution approaches made available by the DSS but to fashion a decision-making process from the functional resources available. The decision maker may need to figure out what sequence of information-processing activities will solve the problem best. What should he or she have the DSS do first? Next? Should the most recent step be repeated? For instance, after projecting a budget that is out-of-balance, should the decision maker invoke a budget-balancing routine? Or should the decision maker revise his or her assumptions and create a new projection from scratch? Decisional guidance could help with these structural choices, as well.

While executing the decision-making process, decision makers are called upon to make various parameters that reflect judgments or preferences. What is an acceptable return on investment? What is the relative importance of location and cost when renting a warehouse? What will the inflation rate be next year? Decisional guidance might provide additional information, perhaps even additional functional capabilities, to help decision makers arrive at these judgments and preferences. Alternatively, guidance mechanisms might suggest appropriate values to the decision maker.

5.2.2 Forms of Guidance: Informative Versus Suggestive

The various examples just discussed included two very different forms of deliberate decisional guidance mechanisms. Some of the guidance was purely informative, intended to enlighten decision makers so they could arrive at their own choices. When choosing a forecasting method, for instance, a decision maker might be told the mathematical properties of the method, its strengths and weaknesses, and the types of data sets and circumstances for which it is best suited. Similarly, choice rules could be described in terms of how they work, whether they are compensatory or non-compensatory, whether they require more or less effort, and so forth. Informative guidance for judgments, such as predicting future economic indicators or assessing probabilities of future events, might include historical data that could be useful as points of reference. Charts of historical trends might also be of value. Such guidance could also include judgments made by others and even a record of previous judgments made by this decision maker together with the actual results. This information might help calibrate the decision maker. Informative guidance might also include qualitative information — for instance, news stories — that shed light on the judgments being made. Informative guidance for preferential inputs might include historical information concerning the decision maker’s prior preferences or descriptive statistics that provide a better understanding of the alternatives confronting the decision maker. For instance, if choosing among a large number of warehouses to rent, descriptive statistics could help the decision maker get a feel for the range of warehouses on the rental market and their attributes.

In contrast to these examples of informative guidance, a DSS might also offer suggestive guidance. For instance, the DSS could recommend which forecasting method best suits the data or the situation. It could recommend which choice rule to follow. It could propose values for the various judgmental inputs that the decision maker must provide. The substance of the various suggestions might be built into the system or the guidance mechanism might contain information-processing capabilities that generate the recommendations dynamically, perhaps with the active participation of the decision maker.

Table 6 provides examples of the various combinations of targets and forms of guidance.

<table>
<thead>
<tr>
<th>Form of guidance</th>
<th>Suggestive guidance</th>
<th>Informative guidance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Target of guidance</strong></td>
<td><strong>Forming the decision (group or individual)</strong>*</td>
<td><strong>Descriptive/analytical outputs</strong></td>
</tr>
<tr>
<td><strong>Form of guidance</strong></td>
<td><strong>Recommended operators</strong></td>
<td><strong>Comparison of operators</strong></td>
</tr>
<tr>
<td><strong>Target of guidance</strong></td>
<td><strong>Selecting the functional elements</strong>*</td>
<td><strong>Map of relationships among operators</strong></td>
</tr>
<tr>
<td><strong>Form of guidance</strong></td>
<td><strong>Set of recommended operators</strong></td>
<td><strong>Record of behavior in similar contexts</strong></td>
</tr>
<tr>
<td><strong>Target of guidance</strong></td>
<td><strong>Using the functional elements</strong>*</td>
<td><strong>History of activity this session</strong></td>
</tr>
<tr>
<td><strong>Form of guidance</strong></td>
<td><strong>Ordered list of recommended operators</strong></td>
<td><strong>Set of operators not recommended</strong></td>
</tr>
<tr>
<td><strong>Target of guidance</strong></td>
<td><strong>Calculating the outputs</strong>*</td>
<td><strong>Recommended values</strong></td>
</tr>
<tr>
<td><strong>Form of guidance</strong></td>
<td><strong>Set of operators not recommended</strong></td>
<td><strong>Set of values not recommended</strong></td>
</tr>
<tr>
<td><strong>Target of guidance</strong></td>
<td><strong>Evaluating the outputs</strong>*</td>
<td><strong>Definition of required input values</strong></td>
</tr>
<tr>
<td><strong>Form of guidance</strong></td>
<td><strong>Record of behavior in similar contexts</strong></td>
<td><strong>Descriptions of how inputs will be used</strong></td>
</tr>
<tr>
<td><strong>Target of guidance</strong></td>
<td><strong>History of activity this session</strong></td>
<td><strong>Tables, graphs, or analyses of data</strong></td>
</tr>
</tbody>
</table>

Note: Reprinted by permission from Silver, M.S., "Decisional Guidance for Computer-Based Decision Support," MIS Quart., 15(1), 1991. Copyright © 1991, Regents of the University of Minnesota. All rights reserved.
6 Design Objectives and Philosophies of Change

One of the themes of this chapter has been that a system’s restrictiveness and decisional guidance need to reflect its design objectives. That is, designers need to set these design variables in a way that will produce consequences consistent with the design goals. Any given system will have its own unique set of specific objectives, but overall a system’s objectives generally fall into one of two broad categories reflecting the philosophy of change underlying the development project. Some systems subscribe to a philosophy of directed change, where the DSS is intended to move decision makers from their current decision-making behavior in a specific direction, toward preferred behavior. Other systems adopt a non-directed philosophy which views the DSS as providing a set of information-processing tools that can enable change whose direction is determined by the decision maker through use of the system. These approaches to change have a differential impact on the choice of system restrictiveness and decisional guidance.

In the case of directed change, one might assume that designers should build highly restrictive systems. This approach is certainly worth considering, since the restrictions can be used to prescribe or proscribe decision-making behavior. But, as we have seen, excessive restrictiveness runs the risk of creating a DSS that inhibits use. So that fails to achieve other objectives associated with lesser restrictiveness. Another design possibility, therefore, would be to combine some restrictiveness with some suggestive decisional guidance to direct the change. These technical features might be complemented with such other features as training and coaching to help produce the desired effects.

In the case of non-directed change, designing a minimally restrictive system seems to be in order. But this approach can also be problematic, because such DSSs can suffer from usability problems due to the large number of functions and options they provide. In this case, combining minimal restrictiveness with either informative or suggestive guidance specifically aimed at increasing usability and helping decision makers cope with the lack of restrictions might be valuable. A more-extensive discussion of change philosophies, system restrictiveness, and decisional guidance can be found in Silver (1990).

7 Conclusion

This chapter focused on the substantive design of decision support systems, arguing that DSSs must be seen not just as information-processing assistants but as interventions into the processes through which decision are made. The chapter also asserted that ultimately designers are — or need to be — concerned not with system features but with system consequences. The challenge for DSS designers is to design a set of system features that are likely to achieve the design objectives — that is, the desired consequences — without producing undesirable side-effects.

System restrictiveness and decisional guidance are two multifaceted system features that play essential roles in how a DSS functions as an intervention, and in the consequences that follow from that DSS intervention. Carefully setting these two design variables is a vital but often overlooked element of DSS design, which too often focuses on creating individual functional capabilities. Whether designers think about it or not, their designs will restrict. And whether they think about it or not, their designs will guide. Successful design therefore requires thinking about it — paying careful attention to system restrictiveness, decisional guidance, and their relationship each with the other.

With the great popularity of personal computing and, increasingly, browser-based applications, many people today interact with computer-based systems that afford discretionary opportunities. Some of these systems — for instance, recommender agents for e-commerce — can easily be seen as instances of DSSs. Others — e-mail, for example — cannot. While the design issues raised here may be richer for those applications that deal more explicitly with decision making, system restrictiveness and decisional guidance can play a valuable role in designing any of these interactive systems.

Acknowledgements

I appreciate the helpful comments of M. Lynne Markus, Leon Schwartz, and an anonymous reviewer.

References


