DISTRIBUTED CASE MANAGEMENT -
A CONCEPT FOR DECISION SUPPORT SYSTEMS

by

THOMAS HUBERT HOFBAUER

M.Sc., Technical University Munich, 1988

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
(Department of Commerce and Business Administration)

We accept this thesis as confirming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

July 1992

© Thomas H. Hofbauer, 1992
In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

(Signature)

Department of Commerce and Business Administration

The University of British Columbia
Vancouver, Canada

Date August 11th, 1992
ABSTRACT

This thesis suggests a new perspective for Decision Support Systems (DSS) that is guided by the dominant role of experience in decision making. Evidence from cognitive research supports the view that organizational problem solvers rely to a large extent on using episodic knowledge gained from similar problem solving experiences, rather than by starting from first principles every time. In addition, people tend to cooperate and seek other's experience, especially as task domains become more complex and relevant knowledge becomes more sparse. Case-Based Reasoning (CBR) has gained much appeal by utilizing previous decision making results to aid in current problem solving activities. However, existing models do not support the exchange of case-based experiences (i.e. learning from others' experience) among organizational workers. Humans are also more flexible in problem solving than existing CBR models in that they can draw analogies from various, related domains, rather than just from within one domain. Derived from both analogical reasoning (AR) and CBR methods, a DSS model based on the concept of Distributed Case Management (DCM) is proposed that would facilitate the exchange of computer-mediated experiences among organizational workers. The feasibility of this approach is demonstrated by implementing a distributed retrieval mechanism based on an analog retrieval algorithm called Analog Retrieval by Constraint Satisfaction (ARCS).
# TABLE OF CONTENTS

**ABSTRACT**

**TABLE OF CONTENTS**

**LIST OF FIGURES**

**ACKNOWLEDGEMENTS**

**CHAPTER ONE**
**INTRODUCTION**

**CHAPTER TWO**
**ADVANCED DECISION SUPPORT FOR ORGANIZATIONAL ACTIVITIES**

2.1 Cognitive Aspects of Decision Making and Problem Solving

2.1.1 A Reasoner's Problem Solving Behavior

2.1.2 Typical Shortcomings in Problem Solving Behavior

2.1.3 Supporting People's Problem Solving Behavior

2.2 Decision Support Systems (DSS)

2.2.1 Traditional Decision Support Concepts

2.2.2 Alternative Decision Support Concepts

2.2.3 A new DSS Design Perspective

**CHAPTER THREE**
**ANALOGICAL TRANSFER**

3.1 Cognitive Principles

3.2 Case-Based Reasoning (CBR)

3.2.1 Indexing and Memory Organization

3.2.2 Case Relevance and Matching

3.3 Analogical Reasoning (AR)

3.3.1 The Similarity Concept in AR

3.3.2 Analogical Mapping

3.3.3 Analogical Concept Transfer

3.3.4 Analogical Retrieval
# TABLE OF CONTENTS

## CHAPTER FOUR
**DISTRIBUTED CASE MANAGEMENT**

4.1 The Distributed Case Management (DCM) Concept 30
   4.1.1 Operating Principles 30
   4.1.2 Easing Case Access through Indexing Support 31
   4.1.3 Advantages of Case-Based Experience for Decision Aiding 31

4.2 Targeted Support Characteristics of the DCM Concept 33
   4.2.1 Enhancing Creativity 33
   4.2.2 Support of Transactive Memory Behavior 33
   4.2.3 Learning Through the Exchange of Experiences 34
   4.2.4 Domain Flexibility 35
   4.2.5 Intra- and Inter-Domain Transfer 35
   4.2.6 Assumptions and Limitations 35

4.3 IS Project Management: An Example Application 36
   4.3.1 The Challenge of IS Project Management 37
   4.3.2 Distributed Case-Based Experience for IS Project Management 38

## CHAPTER FIVE
**A DECISION SUPPORT SYSTEM MODEL BASED ON DISTRIBUTED CASE MANAGEMENT**

5.1 Overview of the Distributed DSS Model 40

5.2 The Case Facility 42
   5.2.1 The Case Base 42
   5.2.2 Selective Case Index 46
   5.2.3 Domain Vocabulary (Semantical Domain Knowledge) 47

5.3 The Central Index Facility 48
   5.3.1 Semantical Knowledge Base 48
   5.3.2 Hierarchical Domain and Case Index 51
   5.3.3 Organizational Knowledge 51

5.4 Case Management Processes of the Decision Support Cycle 53
   5.4.1 Case Representation 53
   5.4.2 Retrieval 56
   5.4.3 Interpretation and Adaptation 57
   5.4.4 Testing and Evaluation 58
   5.4.5 Case Base Maintenance 59
# TABLE OF CONTENTS

## CHAPTER SIX
**D-CARE**

**DISTRIBUTED CASE-BASED RETRIEVAL**

6.1 Basic Concept 62  
6.2 Knowledge and Data Representation 64  
6.2.1 Episodic Knowledge Representation 65  
6.2.2 Semantic Knowledge Representation 68  
6.2.3 Domain Case Index 71  
6.3 Analog Retrieval Strategy 71  
6.3.1 Adaptation of ARCS for Distributed Retrieval 72  
6.3.2 Retrieval Phase I - Semantic Similarity Assessment 73  
6.3.3 Retrieval Phase II - Partial Mapping 74  
6.4 User Preference Selection 76  
6.5 Retrieval Initiation 78  
6.5.1 Retrieval Strategy 78  
6.5.2 Retrieval Parameters 79  
6.5.3 ARCS Parameters 81

## CHAPTER SEVEN
**AN EXAMPLE CONSULTATION WITH D-CARE**

7.1 Case Scenario 83  
7.2 Example Consultation Session 85  
7.3 Description of Retrieved Case 93

## CHAPTER EIGHT
**CONCLUSION AND FUTURE RESEARCH**

8.1 Conclusion 94  
8.2 Future Perspective 96

## REFERENCES

98

## APPENDICES

APPENDIX A: Examples of Semantical Concepts 106  
APPENDIX B: Semantic Weights used in D-CARE 107  
APPENDIX C: Internal Case Representation Format 108  
APPENDIX D: D-CARE Application Cases 109  
APPENDIX E: Complete Mapping Result of D-CARE Consultation 120
LIST OF FIGURES

Figure 3.1: Learning by experience 16
Figure 3.2: The basic CBR cycle 18
Figure 3.3: Similarity classes based on the kinds of predicates shared 23
Figure 5.1: The functional components of a case facility 43
Figure 5.2: Conceptual model of the case base 45
Figure 5.3: Key features of the central index facility 49
Figure 5.4: Semantical concept hierarchy (example) 50
Figure 5.5: Domain index hierarchy (example) 52
Figure 6.1: Retrieval process flow 63
Figure 7.1: User interaction during retrieval process 84
ACKNOWLEDGEMENTS

This work owes much to the advice and help of several people. Foremost among them I want to express my appreciation and personal thanks to Dr. Carson C. Woo, my thesis advisor, for encouraging me in the development of my ideas while showing the professional and personal guidance as well as the continuous attention to detail that helped to improve this research substantially.

I wish to thank Dr. Albert Dexter and Dr. Robert Goldstein for their efforts while serving on my thesis advisory committee. I would also like to thank Dr. Kenneth MacCrimmon and Dr. David Morosan for helpful comments regarding the cognitive underpinnings of human problem solving. Further, I am indebted to Dr. Paul Thagard, Princeton University, who unreservedly provided the ARCS prototype implementation as a platform for "research on thinking", and Andersen Consulting, Vancouver, for providing some factual IS case information.

Thanks from my heart to Barbara for her emotional support and proofreading of most of this work. Finally, thanks to my parents Inge and Otto: without their backing participation in this program, and this research in particular, would have been not achievable. As a member of the Institute of Robotics and Intelligent Systems (IRIS) I also wish to acknowledge the support of the Networks of Excellence Program of the Government of Canada, the Natural Science and Engineering Research Council (NSERC), and the participation of PRECARN Associates Inc.
CHAPTER ONE
INTRODUCTION

"The modern organization will be a man-machine organization where humans and machines will interact in highly sophisticated ways. Man and machine will be more like cooperating partners than master and slave." [Yu 1988]

In recent years, Office Information Systems (OIS)\(^1\) research has advanced to a stage that provides a variety of methods and tools for addressing the much varied and unstructured aspects of organizational work. In the Decision Support Systems (DSS) area, where the underlying goal is to increase the efficiency and effectiveness of decision making in organizations, this trend is reflected in the advancement of a series of new approaches. One type of approach seeks to extend traditional DSSs to a higher level of the organization in order to deal with cooperative and interdependent aspects of organizational decision making. The development of this approach, as, for example, reflected in Distributed Decision Support Systems (DDSS) [Chung 1991], is primarily driven by the organization's structure and the nature of its decision making tasks. Another approach attempts to improve the underlying decision model of traditional DSSs through knowledge-based technology. By adapting DSS principles to traditional rule-based expert systems, a semi-expert system [Keen 1987] can be developed that provides suggestions to the decision maker using domain experts' knowledge. Regardless of the benefits of these approaches, cognitive behaviors of organizational decision makers are not supported by any of them.

\(^1\) Although the term "Office Information System" is widely acknowledged, OIS are commonly termed "Organizational Information Systems" in order to de-emphasize the fact that such systems are constrained to the office area (see [Hewitt 1986]).
Studies of DSS usage indicate that the inspiring and interactive decision making and problem solving environment, as envisioned by DSSs, remains to be realized [Elam & Mead 1990]. For example, El Sawy [1986] points out that DSSs do not provide sufficient features to assist executives in the assessment of how technology support can give a competitive advantage, aid in strategic development, or marketing opportunities. There appears to be a major caveat between the skills and the knowledge needed for organizational decision making in many complex tasks and those actually supported by traditional DSS features. While several DSS researchers (e.g. [Stohr 1983; Bosman 1987]) have stressed the importance of cognitive aids in the design of DSSs, basic features that assist people's problem solving behavior are still lacking. One important aspect regarding the behavior of organizational decision makers is the link between experience, learning, and decision making. Although widely acknowledged, it has received little attention in the DSS research community.

Objective of the Thesis

In response, the main goal of this thesis is the development of the Distributed Case Management (DCM) concept that emphasizes the dominant role of experience in organizational decision making and problem solving. This concept differs from traditional DSS approaches in two ways. First, it is based on the assumption that organizational workers tackle problems by greatly relying on episodic knowledge learned from similar problem solving episodes, rather than by starting from first principles every time. The process of adopting similar, past episodes directly to the current problem, which is associated with experiential learning [Kolb 1976], is referred to as analogical transfer. Cognitive research has demonstrated that analogical transfer is an important factor in human problem solving [Holyoak & Koh 1987; Gentner 1989; Caplan & Schooler 1990] and that the use of analogues is preferred by decision makers over abstract principles and rules (e.g. [Klein & Calderwood 1988]). Second, the concept adheres to the apparent
fact that knowledge and experience in organizations is distributed among individuals. When relevant knowledge is sparse and one's experience is insufficient in providing insight to a problem, an organizational worker seeks the advice and experience of others. Thus in organizational environments, problem solvers not only learn from their own experience but, more importantly, from others' experience.

One way to model these aspects of organizational problem solving behavior is through Case-Based Reasoning (CBR), a pragmatic approach to the more general concept of Analogical Reasoning (AR). CBR is an effort to interpret new problem situations, make a decision or explain a causal relationship by utilizing cases of previous decision making experiences rather than abstracted concepts like rules. Various CBR methods have been proposed and employed in different application domains for performing problem solving (for applications see [Kolodner 1991]); and the idea has been accepted as an appealing approach to support decision making [Duncan et al. 1991]. Current CBR models, however, do not support the exchange of work experiences among individuals and thus, learning from other organizational members. Also, in contrast to the broader and more flexible role of analogical transfer in human reasoning, CBR has pursued a too pragmatic problem solving orientation supporting the user only in a specific domain.

The concept of distributed case management therefore incorporates an analog retrieval and mapping mechanism adopted from the work of Thagard and colleagues [1990] (Analog Retrieval by Constraint Satisfaction (ARCS)). This mechanism provides problem solvers with the opportunity to look outside their specific domain for feedback and inspiration in related fields which is often required for creative tasks, thus emphasizing not only an intra-domain but also an inter-domain transfer of case-based knowledge [Reed et al. 1974; Holyoak & Koh 1987; Bassok & Holyoak 1989; Thagard & Holyoak 1989].
Based on the proposed concept, this thesis presents a generic DSS model which operates in an environment that consists of individual case bases, connected via a common platform representing different organizational units; it therefore facilitates the transfer of case-based experiences between users, allowing an individual to up-date his case memory with those retrieved from other sources within the network. Learning primarily takes place as an individual's case base is extended by newly, well-solved cases leading to improved future decision making. In addition, the proposed approach offers the potential for the discovery of experiential knowledge within the network of distributed case bases.

Although the system offers learning capabilities it is not proposed as an automatic learning system, rather the system guides the user by suggesting solutions or warning of failures from one's cases or that of other users, leaving complex reasoning and decision making to the organizational worker. Reasoning from old cases not only suggests a once proven solution for the current problem, but may also help to critique solutions, interpret situations, or explain an anomalous event. The system also employs an interactive mode for consultations, allowing the user to gradually narrow decision constraints while monitoring the decision progress during subsequent consultation steps. Another important aspect of this thesis is to demonstrate the feasibility of the purported decision support concept. This is achieved through a prototype implementation of the proposed distributed retrieval mechanism. In addition, an IS project management application was chosen to show the potential usefulness of the DCM concept.

Thesis Overview

The remainder of this thesis contains seven chapters. Chapter Two discusses characteristics of existing DSS approaches and motivates the need for a new DSS design that supports people's problem solving behaviors. Chapter Three outlines the framework
of past and related research in CBR and AR. Together with Chapter Two, it provides the theoretical background for the concepts that are proposed in subsequent chapters. In Chapter Four, the Distributed Case Management (DCM) concept is introduced as a new design perspective for DSS. This chapter also discusses the targeted support characteristics of the DSS approach. Chapter Five presents a DSS model that is based on the proposed DCM concept in Chapter Four. In Chapter Six, the prototype implementation of D-CARE (Distributed Case Retrieval) is described, supporting the feasibility of the DCM concept and the distributed DSS model. Chapter Seven presents an example consultation with D-CARE, followed by conclusive arguments and an outlook on aspects of future research in Chapter Eight.
"Effective decision systems should concentrate on assisting the decision maker to gain insight into the decision problem at hand rather than on merely supplying a somehow 'right' answer. This implies that to help individuals facing difficult decision problems, a decision system should aid in understanding and interpreting the problem clearly." [Holtzman 1989]

There are many cognitive characteristics of people's decision making and problem solving in organizational activities that are significant but not adequately addressed in existing DSSs. This chapter will present some of these characteristics that point at the limitations of traditional DSS concepts. Furthermore, it will discuss the need for a new DSS design perspective that is inspired by experiential reasoning in distributed organizational environments. This discussion is justified and motivated by research from both cognitive psychology and the DSS domain.

2.1 Cognitive Aspects of Decision Making and Problem Solving

Organizational tasks often involve activities which correspond to labels like decision making, problem solving, planning, or idea creation\(^2\). From a cognitive perspective, these

\(^2\) A useful classification for such activities in an organizational environment is provided by the typology of McGrath [1984], in particular Planning Tasks (Type 1), Creativity Tasks (Type 2), and Decision-Making Tasks (Type 4).
activities are associated with a knowledge intensive reasoning process that depends upon considerable skill and mental effort on the part of the human reasoner. Supporting the reasoner in this process requires the identification of factors that contribute to the strengths and possible weaknesses of the reasoner's problem solving behavior.

2.1.1 A Reasoner's Problem Solving Behavior

Following a study of the problem solving behavior of system analysts (APB model [Vitalari 1981]), three major components are identified in a problem solving process: mental behaviors, a set of problem solving modes, and classes of domain knowledge. According to Vitalari, a reasoner who attempts to make a complex decision or solve a problem may employ various patterns of mental behaviors such as

- search for problem clues
- recall of knowledge
- gathering of information
- hypotheses generation
- goal-setting
- application of heuristics
- strategy formulation

A specific pattern of mental behaviors forms the reasoning process by which the problem is solved. While performing the task, the reasoner may also exhibit various states or modes of behavior. A problem solving state is defined as a particular focus in time in the reasoning process, and may be associated with several mental behaviors. Problem identification, problem classification, and problem reformulation are among those states that are considered crucial for task accomplishment. In addition, all reasoning processes depend on the availability of several different knowledge sources such as organizational or domain knowledge. Difficulties in applying one or more of the above mental behaviors may lead to difficulties in essential problem solving states and, in consequence, result in the failure to accomplish the task. The next section delineates some of the
possible causes that lead to dismal problem solving behavior.

2.1.2 Typical Shortcomings in Problem Solving Behavior

From research in problem solving, decision making, and creativity, several causes of poor problem solving performance are identified and discussed. Causes are attributed either directly to a deficiency in mental problem solving behaviors (e.g. difficulties in identifying problem clues) or to a lack of essential resources (e.g. lack of experience).

Failure to Attain Problem Insight

Insight has been cited as crucial factor in many accounts of problem solving and is considered significant in identifying problem clues [Ellen 1982; Weisberg & Alba 1981, 1982]. It plays a similarly important role in complex decision tasks where an insightful understanding of the underlying situation and decision consequences leads to a better selection. Failure to attain insight can be associated with the problem solver's inability to identify relevant problem clues (Section 2.1.1). While there are other causes, attaining insight is often the result of discovering an effective problem representation that helps the reasoner to interpret the problem in a more familiar context [Kohler 1956; Weisberg & Alba 1981; Kaplan & Simon 1990].

Failure to Access Appropriate Knowledge

Although certain problem tasks can be accomplished on a common sense basis, in most organizational tasks the individual is highly dependent on various sources of domain knowledge in order to succeed. For example, Vitalari [1981] determined that system analysts utilize up to eight different categories of domain knowledge. Yet identifying or discovering relevant knowledge and alternatives from external sources is usually not easy.
The decision maker also requires the experience that guides him in utilizing abstract knowledge concepts appropriately and in handling exceptional situations. The importance of the role of domain knowledge in skilled performance, task categorization and strategy selection is also reflected by the extensive investigation in cognitive studies on this topic [Chase & Simon 1973; Mayer 1975; Chi et al. 1981; Lancaster & Kolodner 1987].

**Difficulties in Recovering from Problem Bottlenecks**

In determining task requirement individuals often impose unnecessary constraints that hamper their ability to define alternative strategies or apply appropriate heuristics. This difficulty is commonly described as a perceptual constraint or conceptual block [Adams 1986]. For example, imposing the result of an established use of a method or an object in a problem situation is known as functional fixedness, a block that is counterproductive in creative thinking [Duncker 1945]. The decision maker may perceive such situations during task performance as a severe bottleneck. Alternative representations are required in order to break out of a set way of viewing a problem (*perceptual set breaking*) and to solve a problem (*cognitive set breaking*) [Elam & Mead 1990]. Divergent thinking and delayed judgement are also good strategies to break established choice patterns.

**2.1.3 Supporting People’s Problem Solving Behavior**

Although there is little that can be done to improve the *innate* skills and characteristics of an organizational decision maker, eliminating potential causes for shortcomings in problem solving behavior is likely to have immediately observable effects on the decision performance. Since experiential knowledge support has been shown to be effective in stimulating essential behavioral factors such as set breaking, reformulation, generating alternatives, and the like [Adams 1986; Glass & Holyoak 1986; Lancaster & Kolodner 1987], taking experience and learning in problem solving into consideration [Kolb & Fry
1975; Kolodner 1983; Lancaster & Kolodner 1988; Caplan & Schooler 1990] is therefore an important design feature of an advanced decision aid.

With respect to organizational work, the role of experience can be viewed in two ways. First, a person's experience plays the role of a decision aid. Experience acquired in organizational tasks helps to improve the decision maker's task proficiency. This improvement is a result of learning from one's previously experienced successes and failures in performing certain tasks, implementing decisions, or adopting a solution. Second, experience plays a role within the larger context of organizational knowledge transfer. Work experiences are shared among individuals or groups with the intention to provide the expertise needed by others or to invite feedback from them. People's tendency to seek others' experience or opinion is thought of as an essential reflective activity in problem solving (see [Grundy 1982; Boud et al. 1985]). For example, consultants often obtain advice from colleagues to acquire knowledge in which they are lacking. Even when a consultant holds experience in a certain domain he still may reflect his idea or interpretation regarding a difficult problem against that of another domain experts' experience.

2.2 Decision Support Systems (DSS)

2.2.1 Traditional Decision Support Concepts

A DSS is viewed as a computer-mediated tool that assists managerial decision making by presenting interpretations and facts for various alternatives. Traditional DSSs, which are accepted as an integral part of management [Keen 1987], rely primarily on previously identified, quantitative decision models. They fall short of supporting important environmental decision factors such as domain knowledge access, creativity, or novelty.
Significant cognitive principles, as discussed in the previous section, are barely considered in the usual contingent of DSS features. The highly structure-oriented DSS model is basically designed for a particular problem class in a specific domain. Furthermore, traditional DSSs are based on the paradigm of single decision makers in the organization. The cooperative working environment, which has a significant impact on organizational decision making, is not reflected in such DSSs, thus neglecting important issues such as the exchange of opinions and transfer of knowledge.

2.2.2 Alternative Decision Support Concepts

Alternative Settings

In recent years research in the decision support area has cumulatively pointed towards advanced concepts that support cooperative decision making. The most prominent result of this trend has been the emergence of Group Decision Support Systems (GDSS). GDSSs are designed to aid in the joint effort of a group of organizational members involved in the decision making process. The group setting for GDSSs is typically characterized by the joint responsibility of all group members for the decision they arrive at [DeSanctis & Gallupe 1987]. Although targeted decision making groups for a GDSS setting are usually homogenous, GDSSs also be applied to negotiation groups involving conflict [Jelassi & Foroughi 1989]. A step further in reflecting organizational decision processes is the concept of Distributed Decision Support Systems (DDSS)\(^3\). Associated with the term distributed decision making [Burns et al. 1987], DDSSs convey the idea of cooperative decision making across the functional and hierarchical structure of an entire organization. This concept can be attributed to the fact that many organizational

\(^3\) Conceptually similar is the idea of Organizational Decision Support Systems (ODSS) [e.g. Watson 1990, King and Star 1990]. However, due to the novelty of this research area attempted classifications of these concepts are still divergent. For example, while Chung [1991] clearly differentiates ODSS and DDSS along the dimension of task independence, Walkers [1990] definition is not as clearcut.
decision making processes need to be distributed among multiple, contributing participants in an organization. As such, the essential types of DDSSs support among a network of decision making nodes in the organization are communication, cooperation and coordination [Chung 1991].

The transition from individual DSSs to more complex DSSs forms (i.e. GDSSs, DDSSs) obviously effects the kind of support that is required. According to Chung [1991] GDSSs and DDSSs can be differentiated from conventional DSSs by the type of communication, computer, and decision support provided to aid in problem identification and decision finding. Network and communication characteristics of such systems show considerable improvements over basic DSS designs. However, progression on the computer and decision support side are often dismal. Computer support is basically adopted from traditional DSSs (e.g. database access, analysis tools) while decision support is only enhanced by certain structured group methods such as Delphi or group brainstorming. From a cooperative work perspective, measures that support important cognitive aspects of organizational decision making such as feedback, exchange of alternative opinions, and knowledge transfer among individuals are neglected or not adequately addressed.

Alternative Kinds of Support

Deviating from the traditional approach are DSSs inspired by expert system technology. By incorporating a rule-based concept, DSSs are designed to aid the decision making process through a set of recommendations reflecting domain expertise. Clearly, rule-based concepts provide useful features for the application of domain knowledge in decision making. On the other hand, decisions in rule-based systems are derived without involving the user in the reasoning process, therefore such systems tend to be more of a replacement for the decision maker [Duncan et al. 1991]. In addition, since rule-based
concepts are usually highly domain-oriented, relatively small changes in the problem domain require extensive intervention by the expert. A partial improvement can be found in what Holtzman [1986] terms as an Intelligent Decision System: a DSS using rule-based concepts to give decision analysis guidance to the user and provide domain knowledge. Although such a DSS certainly provides more interactive and individual decision support following the suggestions of a semi-expert system [Keen 1987], it is still bound to the problem of domain rigidness.

A relatively new approach to DSS design is the use of Case-Based Reasoning (CBR) methods. The CBR paradigm has evolved as a research undertaking over the last decade, but it has only recently been applied to the DSS area [Duncan et al. 1991]. While not as assertive as a rule-based system, a DSS based on CBR methods assumes the role of a consultant to the decision maker rather than an assistant. However, current CBR approaches still fall short in several aspects; among these aspects are the restricted domain dependence and the inability to exchange experiences between other individuals.

2.2.3 A new DSS Design Perspective

Given the DSS concepts discussed in Sections 2.2.1 and 2.2.2 and the arguments for supporting human's problem solving behavior in Section 2.1.3, a perspective for a new DSS design is advocated. This work proposes a DSS design that combines the use of Case-Based Reasoning (CBR) and Analogical Reasoning (AR) in order to utilize experiences available in a distributed network of DSSs. Like in DDSSs, this design perspective supports the distributed characteristic of organizational decision making. However, unlike DDSSs, the focus here is not so much on the individual who contributes in part to a more complex, global organizational decision, but rather on the individual who benefits from the experience of decisions that were made by others in the organization. Furthermore, experiential knowledge is used to guide decision making but,
Unlike rule-based DSSs, there is no a priori knowledge (i.e. knowledge base) provided. Instead, this DSS design provides mechanisms that help the users to collect, organize, and utilize experiential knowledge. Details of this perspective will be discussed in Section 4.1. The following chapter outlines the theoretical basis for this perspective, which is known as analogical transfer.
CHAPTER THREE
ANALOGICAL TRANSFER

"The ability to perceive similarities and analogies is one of the fundamental aspects of human cognition. It is crucial for recognition, classification, and learning and it plays an important role in scientific discovery and creativity. Human reasoning does not always operate on the basis of content-free general inference rules but, rather, is often tied to particular bodies of knowledge and is greatly influenced by the context in which it occurs."

[Vosniadou & Ortony 1989]

Chapter Three examines the use of analogy in decision making and problem solving, which is known as analogical transfer. The first section introduces the cognitive principles that differentiate analogical transfer from rule-based inference. In this sense the skill of analogical thinking is reflected as the interaction among problem solving, experience, and induction. The remaining two sections review the two main approaches in artificial intelligence and cognitive science that model cognitive processes involved in analogical thinking. Case-Based Reasoning (CBR) is introduced in the second section, while the third section discusses Analogical Reasoning (AR) mechanisms with particular focus on analog retrieval. Thus, together with Chapter Two, Chapter Three provides the theoretical background for the concepts proposed in Chapter Four and subsequent chapters.
3.1 Cognitive Principles

Human problem solvers usually have to cope with a considerable amount of new information and knowledge. In order to process it, they relate it to previous knowledge and experiences of their own or obtain additional external feedback. Reasoning about current problem situations in light of previous problem solving episodes, often described as a reflective activity, is closely connected to learning from experience as suggested by existing learning models [Kolb and Fry 1975; Boud et al. 1985]. Associating new events with existing experiences can be operationalized in two ways (e.g. [Caplan & Schooler 1990]) (Figure 3.1):

(a) by abstracting general concepts and rules for application to the problem (rule-based processing) or

(b) by directly adopting similar, previous episodes to the requirements of the problem (episode-based processing).

![Diagram of Learning by Experience](image)

Figure 3.1: Learning by experience.
Episode-based processing describes a typical trait of human reasoning: remembering a more familiar, seemingly related situation and using the recalled knowledge to make predictions about the present situation. Faced with novel, uncertain information or more complex and less tractable problem situations, cognitive science research suggests that people prefer to draw on holistically similar episodes of past experiences instead of retrieving abstracted characteristics of those episodes (e.g. rules, concepts, schemata, etc.) [Gick & Holyoak 1983; Fried & Holyoak 1984]. This behavior not only dominates traditional areas which lack a strong domain model, such as law and business planning, but is also significant in everyday situations that call for a heuristic judgement in light of uncertainty or too many unknowns.

With reference to problem solving literature, *episode-based processing* corresponds to *analogical transfer* in human problem solvers. Analogical transfer is a central form of induction to generate inferences in pragmatically important situations [Holland et al. 1986]. Predictions, generalizations, and inductive projections from known instances are all examples that involve the transfer from the known to the unknown, a process that is central to learning. Studies have demonstrated that analogical transfer improves people's problem solving performance [Holyoak & Koh 1987; Bassok & Holyoak 1989] and the use of analogy is widely observed in various states of human problem solving and decision making [Ross 1989a; Read & Cesa 1990]. For example, Klein and Calderwood [1988] suggest that in real situations expert decision makers perceived analogues as being far more important than the application of abstract principles or rules.

In the development of computational models for analogy-based processing two related directions, each having a different emphasis, have been pursued: one, *Case-Based Reasoning (CBR)* as adopted for AI problem solving, the other, *Analogical Reasoning (AR)* as envisaged by cognitive science.
3.2 Case-Based Reasoning (CBR)

Accepted as a major paradigm in machine learning research [Rissland et al. 1989], case-based reasoning is a method that adopts principles of analogical transfer to problem solving and decision aiding. Instead of starting from scratch in a new problem task, in CBR one is guided by previous problem solving experiences stored as cases. Thus a reasoner using CBR to interpret or solve a new problem situation can derive shortcuts and anticipate problems in new situations as a result of having dealt with similar problems before. As depicted in Figure 3.2 the CBR process can be summarized as follows:

1. Analyzing and accepting a current problem
2. Retrieving best case(s) from which to generate a solution
3. Adapting the solution to the new problem
4. Testing and evaluation of the applied solution
5. Update memory by storing the new case

![Figure 3.2: The basic CBR cycle.](image-url)
It follows from the above description that learning occurs in a natural way when the case memory is updated by newly adopted cases (Step 5). Directed by the user's input and judgement, this process leads to the continuous extending and improving of the case-base. To date, CBR applications have been employed in domains such as planning, legal reasoning, medical diagnosis, industrial design, explanation finding, and adversarial problem solving (for an overview see Kolodner [1991]). Central to CBR applications is the retrieval and selection of relevant cases from memory. Retrieval is considered to be a massive search problem where the major issue appears to be efficiency. However, unlike data retrieval, the retrieval of cases from memory is not simply a question of perfect matches but rather of partial matches since a new case (as retrieval probe) is unlikely to match exactly with those already stored. The central goal in case retrieval is to avoid an exhaustive search by determining which cases are relevant, where relevancy is measured in terms of pragmatic similarity.

3.2.1 Indexing and Memory Organization

Traditionally, CBR utilizes indexing to make the retrieval process more selective. Indexing is the assignment of a label to a case that, under a certain retrieval criterion, is significant enough to identify relevant cases. Most existing models, which are influenced by Schank's [1982] theory of reminding, choose indexes that have causal relevance to goal accomplishment, i.e. indexes that list successful solutions as well as failures of cases. At retrieval time, features of the problem on hand are used as a probe structure to be matched against these case indexes. Efficient search procedures, operating with a probe structure (i.e. set of problem features) on carefully structured case indexes, are provided as part of a CBR system. Case index organizations are mostly based on discrimination nets (d-nets) as, for example, in Hammond's CHEF [Hammond 1989] and Kolodner's JULIA [Kolodner 1987].
On the other hand, indexing is also the major bottleneck. Once indexes, on the basis of a certain set of goal features, are employed in a case-base, case access is limited to those indexes chosen. Unfortunately, a case can never be indexed under all potentially relevant aspects, a basic requirement in order to provide flexibility in analogical transfer. Alternatively, a case index can be more abstract so as to cover many different aspects, but then retrieval becomes imprecise due to the retrieval of too many irrelevant cases. For this reason, indexing is traditionally domain-restricted, rendering the retrieval of out-of-context cases, such as those needed for legal justification or technical applications, more difficult. Yet cases from other domains can be particularly useful for complex and creative problem work where acquisition of domain-external knowledge is encouraged in order to attain insight or adopt new ideas.

3.2.2 Case Relevance and Matching

In determining the relevance of a case, CBR typically focuses on usefulness [Kolodner 1991], where the term useful refers to shared goal orientation. Rather than selecting a relevant case by judging its similarity to the problem on hand, CBR examines whether a case provides features that are beneficial for the attainment of a problem’s goal. Indexes are the means to mark cases at case storage time for their usefulness. Cases are then selected on the basis of a match of problem features or a subset of those with the indexed case features. Thus the problem of similarity matches between the problem and potential candidate cases is reduced to a search for appropriately indexed cases in the case-base. On the other hand, this adds to the indexing problem previously described since one cannot predict all the useful aspects of a case. In addition, feature indexing lacks important context information, and the goal-orientation in case retrieval is relatively constrained. In situations where no matching features are found or where too many cases are retrieved due to the indifference of indexed features, appropriate similarity judgements are required.
3.3 Analogical Reasoning (AR)

Analogical reasoning is an effort to develop models of analogical transfer that are not only consistent with but can explain controlled psychological experiments [Thagard & Holyoak 1989]. In modeling human memory processes, AR attempts to devise and verify adequate algorithms for analogy-based processing that go well beyond the capabilities of CBR models. Thus, while CBR and AR share the same cognitive principles, CBR is more perceived as an engineering approach to problem solving applications. Nevertheless, central to this work is the idea that analogical algorithms, embedded in an otherwise complete CBR system, can be used to overcome some of the obstacles of traditional CBR indexing mechanisms. An extensive overview of computational models of analogy is provided by Hall [1989]. The AR process, which like CBR reflects experiential learning, has been commonly described as follows:

1. Retrieving of a plausible source analogue
2. Mapping between the source and the target (problem)
3. Transferring knowledge from the source and to the target
4. Subsequent learning

As in CBR, retrieval of potential analogues is the key issue in AR. It is also the most challenging since, in the absence of any external information of what is relevant in memory, noticing, or being reminded of an analogue is difficult. Once potential source analogues are identified (i.e. by retrieval) a mapping process between the target and each of the analogues is performed to match possible correspondences and determine the similarity distance. Based on established correspondences, knowledge can then be conveyed from the source to the target. Learning takes place if relevant new knowledge has been adopted in this transfer process. Beyond that, the abstraction of common principles from mapped analogues is also possible as a more abstract learning step.
3.3.1 The Similarity Concept in AR

Similarity is widely accepted as a key determinant for analogical transfer [Gentner 1983]. Both the accessing of an appropriate analogue and the setting up of correspondences between problem and analogue rest upon some form of similarity assessment. One way to assess similarity is by testing whether a situation shares a proper sub-set of the features with the other (e.g. [Carbonell 1986]). Yet complexity and context dependence of the similarity concept makes this difficult. In succession to the work of Kahneman and Tversky on human judgement and choice (e.g. [Kahneman & Tversky 1972; Tversky & Kahneman 1981]), research in recent years has pointed towards a variety of kinds of similarity that are associated with analogy-based processing. Among these, three classes of similarity tend to emerge in numerous accounts as highly indicative (e.g. [Vosniadou & Ortony 1989]): structural, goal-oriented (pragmatic), and semantical. A closer determination of these similarity classes is crucial for an understanding of the analog retrieval and mapping concepts.

Structural Similarity

Structural similarity, which in a purely formal sense has been a dominant component of most analogical models [Gentner 1983; Falkenhainer et al. 1986; Holyoak & Thagard 1989; Thagard & Holyoak 1989], is also the most important characteristic for determining an analogy. Gentner [1989] differentiates kinds of similarity by identifying whether the resemblance between two situations is based on the relational structure, on the object characteristics, or on both (similarity space, Figure 3.3)\(^4\). In this sense, a \textit{true analogy} is classified as a correspondence where only object relations are mapped (e.g. "Atoms are like the solar system", "The brain is like an information processing unit").

\(^4\) Gentner's separation of similarity features is based on simple syntactical characteristics, where \textit{attribute} is a one-place predicate and \textit{relations} are multi-place predicates [Gentner 1983].
Mere appearance is primarily characterized by corresponding object attributes (e.g. "The office building is shaped like a pyramid"), while in literal similarity both relational and object characteristics are mapped. This is based on the assumption that in order to assess the similarity between two situations, not only the number of shared features matters [Tversky 1977], but also the kinds of features (i.e. relations, object attributes) that match.

Figure 3.3: Similarity classes based on the kinds of predicates shared [Gentner 1989].

Pragmatic Similarity

Unlike Gentner’s structural feature distinction, many researchers in the problem solving area focus more on the pragmatics of analogy-based processing. Corresponding to the goal-oriented CBR approach, pragmatic similarity denotes the causal relevance to goal accomplishment of the analog. It has been accepted as being a major factor in
influencing the retrieval of potential analogies [Winston 1980; Holyoak & Koh 1987; Ross 1987; Carbonell 1986; Veloso & Carbonell 1991], although empirical evidence is still limited. With respect to causal relevance, Holyoak and Koh [1987] distinguish between surface and structural features\(^5\). A surface similarity is defined as a match between features that play no causal role in determining the possible solution of the analog. Such features are said to be "goal preserving", making them less predictive in determining a relevant analog. In contrast, a structural (or goal-directed) similarity is based on features that influence goal attainment (i.e. these features are good predictors with respect to goal relevance). However, while structural similarities are context independent, pragmatic similarities only hold true in context of a specific problem situation. In addition, applications of analogy may be useful without necessarily showing causal relevance such as in creative solutions.

**Semantic Similarity**

Derived from its importance in human memory retrieval processes, the semantic similarity aspect is given similar status for analog processing. Semantic similarity between two entities is determined by the degree of overlap in semantical meaning. Lexical relations such as synonym, sub/super-ordinate, whole-part, kind-of, or antonym represented in a thesaurus, can be used to determine degrees of semantic similarity between predicates in analog descriptions. For example, the similarity between synonyms (car:automobile) is more highly defined than it is between whole-part (car:tire) relations. Regarding analog retrieval, there is ample work that suggests the importance of semantic similarity in accounts of noticing or being reminded of a potential analogue [Gentner 1987; Gick & Holyoak 1983; Holyoak & Koh 1987; Ross 1987, 1989b]. It shows that

---

\(^5\) Note that the term "structural feature" is purely problem goal-related and has nothing in common with other similarity terms. In contrast, structural similarity, as used by Gentner [1983], is based on formal relational and object correspondences between two situations. In later work, to avoid confusion, Holyoak termed goal-related features as "pragmatically central" or simply "important". 
humans are sensitive to the degree of semantical overlap in words between a problem and a potential analog.

**Multiple Similarities for Analog Processing**

Analogical reasoning models, although mainly concerned with true analogies, principally provide the capability to deal with any kind of similarity. They also take on the three similarity classes as an integrative set instead of focusing on one particular similarity. In contrast, CBR is basically restricted to literal similar cases with a strong pragmatic similarity influence. In other words, while CBR is merely concerned with retrieving similar cases from within one domain (*intra-domain transfer*) by using goal-oriented features, AR focuses on analogical transfer from disparate or remote domains (*inter-domain transfer*)\(^6\). In the context of this work, it is important that for models of AR the same mechanisms of inter-domain transfer are also applicable to intra-domain analogies. Thus AR is not only a more flexible approach to analogical transfer than CBR, but also has more potential with respect to interpretation, discovery and application of relevant analogies.

**3.3.2 Analogical Mapping**

Analogical mapping is the process that establishes correspondences between elements of a potential analogue and those of a target problem. In most analogical models, mapping is understood to be the matching of relational structures, while at the same time preserving structural consistency [Burstein 1986; Falkenhainer et al. 1986; Holyoak & Thagard 1989]. The consistency requirement conveys the idea that a system of relations that holds true for analog (*source*) objects should also hold true among problem

---

\(^6\) Carbonell's work [1986] is one of the few exceptions to this, since his derivational analogy was only intended to be *intra-domain*.
ANALOGICAL TRANSFER

(target) objects [Gentner 1983]. Consistent matching is attempted by placing objects in the analogue and the problem in one-to-one correspondence based solely on the virtue of their relational role. Correspondences of object attributes are irrelevant when it comes to satisfying this requirement. Holyoak and Thagard [1989] interpret the consistency requirement as an approximation of a formal isomorphism constraint. However, contrary to the mathematical definition, isomorphy is used as a degree of constraint for successful mapping, noting that useful analogues might violate a one-to-one or relational correspondence.

Several computational models were developed for the mapping process. With the emphasis on structural consistency this, of course, required more sophisticated mechanisms than indexing. Gentner's Structure Mapping Engine (SME) (see [Falkenhainer et al. 1986]), for example, is able to compute correspondences under strict obedience of structural consistency. However, the model does not account for other important aspects of similarity. In contrast, Holyoak and Thagard [1989] offer an analogical mapping model (ACME - Analogical Constraint Mapping Engine) that incorporates all three basic aspects of similarity - structural (isomorphism), goal-dependent (pragmatic centrality) and semantical as discussed in Section 3.3.1. For example, pragmatic facts about the potential use of an analogue can not only identify a relevant analogue, but can also identify which aspects of an analogue are useful for transfer. Key to ACME is a parallel constraint satisfaction mapping algorithm that applies all three similarities collectively.

3.3.3 Analogical Concept Transfer

The correspondences established in the mapping process provide the basis for a subsequent transfer of knowledge from the analogue to the target. In addition, problem context and knowledge regarding the problem domain often direct the potential transfer
of knowledge and use of inferences that can be made about the target. Last but not least, the amount of new knowledge actually conveyed across the established correspondences depends upon the problem solver's existing knowledge. According to Gentner [1984], most of the knowledge transfer takes place between two extremes:

(a) **Pure Matching**

A match of the relational structures of two situations in which the domains, context, and knowledge concepts are known. There is no transfer of new structures or essential information from one situation to the other; rather the focus is on corresponding knowledge concepts, serving as a reformulation, metaphor or re-interpretation of a case.

(b) **Pure Carry-Over**

A partial match of relational structures occurs when only the domain and the context of one situation is well known while little is known about the other domain. In this case, maximum knowledge is conveyed from one situation (source) to the problem situation (target). The focus here is on solving or interpreting a problem situation.

### 3.3.4 Analogical Retrieval

Retrieval is the primary requirement for the success of analogical transfer. Without having access to potential analogues and context information, mapping would not be feasible. In spite of an already known analogue (e.g. provided by a teacher), the spontaneous access ("reminding") of a relevant analogue becomes the central issue. Numerous cognitive studies converge on the findings that certain surface features, e.g.
similar attributes, are important for accessing analogues [Gick & Holyoak 1983; Reed 1987; Ross 1987]. With respect to the similarities identified in Section 3.3.1, access to an analogue is primarily dominated by semantical similarity while structural and pragmatic aspects support this process. Interestingly, while computational models for mapping and transfer are abundant, there exists only one complete model for analogical retrieval which was developed by Thagard and colleagues called Analog Retrieval by Constraint Satisfaction (ARCS) [Thagard et al. 1990]. As in ACME, ARCS incorporates all basic similarity classes, i.e. **semantical**, **structural** and **pragmatic similarity**, by treating them as equal constraint pressures in a parallel constraint satisfaction algorithm. For this research ARCS provides the basis for the analog retrieval mechanism (Section 6.3).

ARCS uses the semantic concept representation of WordNet, a computer-based lexical reference system derived from psycholinguistic theories of human lexical memory [Miller et al. 1988]. The main concept of WordNet is that semantic information is encoded as a set of lexical relations, with kind and whole-part relations forming a hierarchical structure (see Appendix A for examples of WordNet concepts). Isomorphism in ARCS is interpreted in the sense of Gentner's structure-mapping. In propositional representation, structures are *isomorphic* when there is a one-to-one correspondence between existing propositions. Mapping propositions imply that there is a map between constituent predicates as well as arguments.

**Constraint Satisfaction**

*Constraint satisfaction* or *cooperative algorithms* are derived from vision research in AI. They are characterized by a set of elements to be interpreted and a set of constraints between adjacent elements. Given these two characteristics, an algorithm is cooperative if it repeatedly adjusts the interpretation of each set of elements so it is in greater harmony with its neighbors until global harmony, or constraint satisfaction is obtained.
[Charniak & McDermott 1985]. By using a constraint satisfaction method, ARCS searches in a parallel fashion by considering many potential retrieval candidates competitively.
CHAPTER FOUR
DISTRIBUTED CASE MANAGEMENT

"Many ideas grow better when transplanted into another mind than in the one where they sprang up." (in [Gentner 1989])

The basic idea of the Distributed Case Management (DCM) concept is to support individuals through a distributed DSS platform for the maintenance and transfer of corporate case knowledge. The discussion in this chapter focuses first on important characteristics and the functionality of the concept, then these issues are demonstrated in the context of the IS project management domain to underline the practical relevance of this concept.

4.1 The Distributed Case Management (DCM) Concept

Distributed Case Management (DCM) is proposed as a new DSS design perspective to support the use of experience in organizational problem solving behavior, as discussed in Section 2.1.3. The adoption of mechanisms from AR and CBR to operate in a distributed organizational environment is the most significant aspect of this concept. Details of the conceptual idea behind the proposed approach are discussed in the following.
4.1.1 Operating Principles

Individually gathered work experiences in the context of performing organizational tasks are maintained by an organizational worker or by a larger organizational unit (e.g. task group, department) as cases. Cases of organizational workers are stored in the workers' individual case repositories which are at different, interconnected locations. The exchange of experiential knowledge to aid decision making is user request-oriented, in that an underlying system assists the user in locating and accessing his own experiential knowledge and that of others. By means of this concept the user can retrieve cases from other sources provided that he has the access rights to those sources (Section 5.3.3). Parameters for retrieving relevant cases are determined by the problem on hand. Furthermore, cases that are stored at distributed locations across the organization adhere to the concept of a "corporate memory" [Kolodner 1991].

4.1.2 Easing Case Access through Indexing Support

Analog retrieval serves as the primary mechanism to detect and identify relevant knowledge within the distributed environment. Once a relevant case is identified, it can be indexed by the decision maker according to its use. Indexing, in this situation, serves as a convenient access mechanism to manage the identified cases. This facilitates repeated access to cases that have already been proven useful to the decision maker in previous tasks.

4.1.3 Advantages of Case-Based Experience for Decision Aiding

A case-based approach aids experiential reasoning in many ways. Since experiential knowledge is typically incomplete or unstable as a result of dealing with new, unfamiliar problem situations, it is difficult to represent in a declarative form. Until more evidence
DISTRIBUTED CASE MANAGEMENT

is gathered that positively confirms a unique decision making experience, a
geneneralization of this knowledge in the form of stringent rules is hardly appropriate. On
the other hand, a case description of the same decision making experience can be used
to suggest a solution or give advice in a similar situation. Cases are rich collections of
contextual facts of an event or situation, incorporating subtle details that are normally
eliminated by abstraction in rule representations\(^7\). Furthermore, a system approach
enhances the human's reasoning capabilities as it collects case-based experiences more
reliably, can store a vast amount of cases, and does not "forget".

Another advantage of cases is that it is a conflict free representation of
inconsistent experiences and thus alleviates the traditional knowledge acquisition
bottleneck. In line with principles of human cognitive behavior [Harman 1986], episodes
of human experience may correspond to conflicting sets of beliefs. This implies that
experience-based knowledge often reflects a variety of slightly conflicting viewpoints in
different contexts of a problem situation. For example, persons with different frames of
reference may experience situations differently and even one's own perception of the
same situations may be inconsistent. The fact that a collection of cases can reflect
experiences with conflicting viewpoints is favorable in many applications, providing a
greater variety of alternatives for the decision maker. Selecting the appropriate case
during the decision process is then left to the reasoner's evaluation. Knowledge
acquisition is then much easier since, unlike rules, cases require almost no debugging of
the interactions between them and initial knowledge acquisition can be "rote" [Rissland
et al. 1989].

---

\(^7\) Nevertheless, cases are not a substitute for declarative knowledge; rather cases are understood to be
a complement to rule-based knowledge, paralleling the view that most of the knowledge gathered in complex,
new domains is first processed and encoded in the form of episodic memory. Consequently, for a realistic
system it is important to see case-based or analogical reasoning as only one component of a general problem
solving mechanism, next to a rule-based component. However, cognitive research suggests that a substantial
set of previous episodes sharing similar structures in a complex domain is required before rule and concept
abstraction can take place [Fried & Holyoak 1984; Gick & Holyoak 1983].
4.2 Targeted Support Characteristics of the DCM Concept

The proliferation of experiences from previous problem solving tasks facilitates a range of problem solving behaviors (Section 2.1.2):

- Identifying and accessing of relevant knowledge
- Interpreting unfamiliar problems on the basis of familiar ones
- Reformulating existing problem representations through analogical reasoning
- Gaining insight to complex problem situations

Other prominent features of the targeted DSS environment are outlined in the following.

4.2.1 Enhancing Creativity

Creativity is a desirable but often neglected characteristic in supporting organizational problem solving tasks. As pointed out by Elam & Mead [1990], conventional decision support concepts fail to address the creative and intuitive aspects of decision making. Evidence from creativity research (e.g. [Amabile 1983]) suggests that a key to aiding creativity-relevant skills is to encourage divergent thinking through the breaking of perceptual blocks (Section 2.1.2) and the production of a variety of alternative solutions. With the availability of past experiences from different domains, creativity will be enhanced.

4.2.2 Support of Transactive Memory Behavior

According to the group theory of shared memory and distributed knowledge behavior, called Transactive Memory Theory [Wegner 1987], people in cooperative settings make extensive use of external memories, as well as the communication that takes place between them. While an individual regards external memory as any kind of storage which is not part of the individual's mind, it is central to this theory that other
individuals may be regarded as external storage as well. Simulating a transactive memory via a distributed CBR environment is one way to enhance an individual's problem solving capabilities by facilitating knowledge access among organizational problem solvers.

4.2.3 Learning Through the Exchange of Experiences

The transfer of experiential knowledge (Section 2.1.3), based on analogy, is facilitated between different tasks and domains with the intent to support the obtaining of domain alternatives, the discovering of new knowledge, and the like. Facilitating the exchange of experience and expertise among organizational units (individuals, groups, etc.) is a key factor in successfully supporting organizational problem solving work, since it contributes to the experiential learning process in that individuals learn from each other and thus improve their decision making.

On the other hand, learning from each other by exchanging experiences relies on the migration of experiential knowledge among organizational members. Communication channels are essential for knowledge migration, but without the appropriate knowledge distribution channels, knowledge migration will remain an idealistic issue. The inability of individuals to cope with huge amounts of information is partly responsible for this. Many imperfections are due to administrative and functional short cuts and the failure to assist or enhance cooperative efforts between subsidiaries, departments, or task groups. Even people in the same department often are not informed of their colleagues projects let alone benefit from their experience. Traditional information systems only account for the proliferation of data but do little to deliver the expertise held by a few but needed by the many. Traditional DSSs and expert systems, on the other hand, take static knowledge approaches, fail to break the brittleness of single domain expertise, and need substantial intervention from specialists in order to change their knowledge base.
As pointed out in Section 2.1.2, the failure to access appropriate knowledge is a shortcoming in problem solving behavior which needs to be supported. A case-based concept for the storage, maintenance, and transfer of experiential knowledge renders itself as a crucial factor in such an effort.

4.2.4 Domain Flexibility

Organizational decision making and problem solving is rarely constrained to one domain. Organizational tasks are often interdisciplinary, dependent upon multiple, coherent domains that show various degrees of overlap. It is a landmark characteristic of the case-based approach to decision aiding that cases can be invariably applied to multiple domains. Obviously, a model that supports a coherent set of domains provides more flexibility for the problem solver than one that supports each domain as if it was an individual system.

4.2.5 Intra- and Inter-Domain Transfer

A main incentive in learning and problem solving is not only to support transfer within domains (intra-domain), but also across domains (inter-domain) where a problem or explanation in one area is conveyed to provide a solution or explanation in another area. The inter-domain knowledge transfer also encourages the abstraction of category prototypes and common generalities between domains for interpretive applications.

4.2.6 Assumptions and Limitations

To ensure compatibility in the transfer of case-based knowledge, it is assumed that cases across the distributed system are stored using the same underlying reasoning formalism and that cases in a user’s case base are accessible from the outside. Nevertheless, cases
can still differ to the extent that two people storing the same case use different case descriptors (i.e. predicates for objects and relations). Note that while decision support is provided by cases in the form of suggestions for solving current problems, it still leaves the actual decision process, i.e. the choosing and modifying of the conveyed solutions, to the organizational decision maker.

With respect to the learning capabilities, experiential learning is limited to the extent that it requires interactive feedback of the decision maker. An improvement of the underlying decision base can only be achieved if the new case provides sufficient variation. Consequently, learning by experience is not self-sufficient, there must be external experiential input other than the exchange of experiences among participating members. A newly solved case is usually the result of a set of retrieved cases and additional input from the user, who adopts the suggested solutions to the requirements of the problem case. This input constitutes an eminent part of a new experience, which is fed back to the system. As such, learning by experience depends on reciprocity: the user is less likely to generate a solution without adequate support and the system cannot learn without receiving new case examples. In addition, subsequent concept learning (learning by analogy) can take place as a result of the mapping process, but would be limited by the usefulness of what is being learned. For example, the learning of semantic concepts in the context of a decision support system may be more useful than the learning of semantic weights (Section 8.2). In any case, user judgement is absolutely advocated with regard to any learning result.

4.3 IS Project Management: An Example Application

The IS project management is a typical domain where proficiency and success of the management task rests largely upon the experience gained in previous projects. In fact,
DISTRIBUTED CASE MANAGEMENT

project experience influences all related project tasks, such as planning, budgeting, and especially the carrying through of a project.

4.3.1 The Challenge of IS Project Management

The management of IS projects is characterized by the striving towards a precise goal (e.g. development of a new product) within well defined time, budgetary, and other resource constraints. The primary challenge of the management task lays in the planning, controlling, and coordinating of efforts that are required to accomplish a project within its constraints. Support in such tasks is usually provided by development methodologies (e.g. IS life-cycle models) and planning tools. Nevertheless, the practice of project management requires much more than these supports. A project leader is, like any manager, constantly challenged by the unexpected: things that go wrong despite rigorous planning efforts and continuous controlling measures. Although certain IS projects are more routine in nature (e.g. maintenance, standard applications), generally they tend to be rather complex and often problematic. Issues such as novelty, non-standard technology, large system size, functional complexity, research effort, multi-partner cooperation, large and heterogenous teams, etc. challenge many IS projects.

For example, control measures are sufficient to check for constraints and development requirements but fail to identify more intricate problems. The difficulty is to recognize the underlying conflicts as well as to discover potential complications at an early stage in the project in order to react within project limits. Long before clear signs signal a project crisis, "soft" indicators like team acceptance problems, unsuccessful project meetings, dismal client cooperation, and so on hint at critical conditions. These indicators are hard to measure by clear criteria, rendering not only the evaluation of the project progress, but project management activities in general as a highly experience-based task.
In addition, IS project management not only requires input from the IS domain but from various related business fields. Thus, an important aspect in the acquisition of new experience is that project managers look beyond their core domain and its development methodologies, since the transfer of knowledge from different domains (inter-domain transfer) is an important source of knowledge input.

4.3.2 Distributed Case-Based Experience for IS Project Management

Project managers typically draw on experiential knowledge from previous projects in an effort to use results and learn from the difficulties and failures encountered in these projects. Although certain management skills rely on conceptual models and methods, the application of these models is primarily based on practical experience, for example, when participating in other projects by exchanging of information with other project leaders through external consultation work or through cooperation with clients, etc. Accumulating experiences and making it available for novices or for those less experienced colleagues is crucial in the guidance of the project and its members.

On the other hand, project efforts are characterized by cooperative team work. The exchange of ideas and experience among participants is an essential part of successful group work. Consequently, the flow of the experiential exchange is not one-sided but mutual: acquired experience is not only provided by the project manager to participating members, but also from individual project members to the project manager and other participants. With increasing complexity and size, the coordination of IS projects becomes quite difficult. Obviously, in such a situation, a project manager does not have a perfectly global view and therefore does not have all the required knowledge, rather knowledge and experiences gathered by individuals remain distributed across the project team. The mutual sharing of such work experiences is essential for all participants in converging towards the project goal and influencing the dynamics of
project management.

In addition, the case-based approach serves as a source for the transfer of IS domain related knowledge. For example, the potential of reusing existing system designs and specifications is substantially enhanced through the case-based paradigm in combination with powerful analog retrieval mechanisms. This is especially beneficial for requirements analysis in the context of system specification (for an application see [Maiden & Sutcliffe 1992]).
CHAPTER FIVE
A DECISION SUPPORT SYSTEM MODEL BASED ON DISTRIBUTED CASE MANAGEMENT

This chapter introduces a DSS model that captures the idea of the Distributed Case Management (DCM) concept as outlined in Chapter Four. In the first section, an overview of the proposed DSS model and its main components is provided. These components, in particular the case facility and the central index facility, are described in more detail in the second and third section, respectively. The fourth section explains each of the functional processes of the model as reflected by the DCM concept. Note that the model and its functionality presented in this chapter are conceptual. A prototype implementation of part of this model is discussed in Chapter Six.

5.1 Overview of the Distributed DSS Model

Central to the distributed DSS model is the idea of a case facility as a decision maker's repository and working environment for experiential knowledge represented in the form of cases. Similar to the data base concept, a case facility incorporates a case base, also referred to as the case memory, as well as case management functions which provide for the representation, storage, organization, retrieval, and interpretation of cases. As will become clear later, case management functions differ from conventional database...

---

8 Interpreted in terms of the transactive memory theory [Wegner 1987], the user refers to cases stored in his case facility as "internal memory" and to cases in all other case facilities as "external memory" (Section 4.2.1).
management functions in several ways, since cases are not only complex, compounded data structures but also knowledge structures containing factual knowledge.

A client/server network concept is adopted in order to link individual case facilities (clients) together with a central index facility (server). With every case facility acting as an independent DSS unit, the entire network of connected case facilities constitutes a distributed, multi-user case-based DSS. Through a set of global case management functions, mutual access of case-based knowledge among collaborating organizational members is provided. In this respect, the central facility's main purpose is to guide and support the global case management functions, particularly the case retrieval process.

The network of case facilities is intended to link any set of inter-organizational and remote organizational units that strive for common experiential knowledge exchange. However, the user who connects with the network through a case facility requires no knowledge of the logical or physical data organization in order to retrieve cases from anywhere within the network. Access requests for cases across the case network are handled via the centrally maintained case index facility. Nevertheless, background information is provided where appropriate to facilitate the decision support process, e.g. the user is informed where cases are retrieved from. In this context, some of the traditional data retrieval issues like concurrency and update are relevant although not discussed in this work. On the other hand, following Section 4.1.3, the issue of data consistency and integrity is not relevant with respect to distributed case management.

---

9 The network of case facilities can be interpreted in terms of traditional network concepts such as LAN or WAN architectures. However, this work does not discuss any of the related networking concepts that are required for the adoption of the distributed DSS model to a specific architecture.

10 This approach corresponds in principle to that of a central-index distributed database system [Laudon & Laudon 1988].
5.2 The Case Facility

A case facility provides functions that are bundled under a set of functional components referred to as case management processes. Derived from the traditional CBR model, they constitute the basis of the case-based decision support cycle (DSC). Accordingly, these processes account for the case representation, the retrieval, the interpretation and adaptation of the retrieved cases, the testing and evaluation of new cases in light of retrieved ones, and the case base maintenance which manages cases in the case base. However, case management processes incorporate expanded retrieval and case maintenance mechanisms and thus differ from that of traditional CBR processes (Sections 5.4.2 and 5.4.5). To support case management processes, in particular retrieval and case base maintenance, a case facility contains a selective case index which is part of the case base, and a domain vocabulary representing semantical knowledge. Figure 5.1 depicts the main components and their functional relationships in the case-based decision support cycle.

5.2.1 The Case Base

Overview

The case base is the core of a case facility. A case facility's case base (i.e. the case memory) is as conceptually adequate as a problem solver's episodic memory. That is, it reflects the computational representation of a decision maker's experience (episodic knowledge). Corresponding to the traditional modal model of cognitive theory [Atkinson & Shiffrin 1968], the case memory can be further differentiated: the long term case memory (i.e. the case base) representing the entire volume of a problem solvers stored experiences, and the short term case memory (i.e. working memory) exhibiting recently remembered experiences and those that are still in processing stage. Cases that are
Figure 5.1: The functional components of a case facility
Figure 5.1: The functional components of a case facility
retrieved from the case base into the system ("recalling or remembering a case") are said to be **activated** or being held in **active case memory**. The process of recalling an experience parallels the retrieval of the episode from the long term memory and its activation in the short term memory. Cases can also be **deactivated** ("forgetting a case") for reasons like increasing the processing efficiency and avoiding the cluttering of the active case memory with cases that are not currently needed. For internal case processing the system utilizes highly specific data structures to represent activated cases which are different from those in the long term memory (see Section 6.2 for more details). With respect to the maintenance of a case in a case facility three stages can be identified:

1. **Case is not in case facility**, meaning the case is neither in the case base nor in active case memory.
2. **Case is in case facility**, meaning the case is loaded from an external case facility or encoded from the representation unit and then stored in the case base but the case is not activated.
3. **Case is activated in case facility**, meaning a case is retrieved from the case base or loaded from another case facility directly into the active case memory without storing it intermediately. Thus, the activation stage of a case does not automatically imply that the case is stored in the case base.

**The Case Base Interface**

The case base in the DSS is thought of as an underlying database and its interface to the case base maintenance process\(^{11}\) (Figure 5.2). In this case, the interface reflects the

---

\(^{11}\) For the underlying data base both relational and object-oriented models are plausible. Relational data bases provide efficient access methods but cases are more scattered. In contrast, an object-oriented approach provides more integrity for all case related features, an argument that is particularly appealing with respect to the transport of cases among distributed case facilities (see for example MOAP architecture [Woo & Lochovsky 1992]).
A decision support model based on distributed case management

The interface provides the conceptual case schema and the functionality needed to transform the data schema of the underlying DB into the case data structure. Since cases are complex data structures, procedures are required that decompose the case into a simpler format so that it can be stored in the DB. Similarly, case access functions must track all decomposed parts of a case stored in the DB and forward the completely recomposed case structure to the case facility. Key to an efficient connection between case base maintenance and DB processes is a tight coupling concept [Vassilou 1985]. In this configuration, all transformation and DBMS access functions are encapsulated in a
lower layer, hidden from the actual application program, i.e. the case base maintenance process. To facilitate performance increases in the interaction between DSS and DB, the interface also features a case cache and an intelligent look-ahead mechanism [Härder et al. 1987], being directly triggered by case management processes. This allows access and the intermediate storage of a bundle of related cases when needed together, e.g. during the network processing activity in active memory. In addition, an index serves as a reference for cases in active memory that were selectively indexed by the decision maker.

5.2.2 Selective Case Index

For retrieval purposes, a case can be indexed before being stored in the case base. As implied by the name, the use of a selective case index is optional and utilizes user preference-based index criteria. That is, criteria for selective case indexing are determined by what the decision maker thinks is predictive for later retrieval of specific cases from his case base. "Useful" is described as what best addresses the problem solver's goal in a certain problem solving context [Kolodner 1989], meaning his needs, expectations or preferences. Index labels should be indicative for a case's special features or characteristics, e.g. highlight what makes a case worth remembering when compared to other cases. For example,

- new sub-field of specialization
- prototypical case for a class of problems [Hunter 1989]
- exceptional case as counter-example
- personal success or value
- frequency or recency of use

Provided that a useful label is chosen, the retrieval of indexed cases is more convenient and efficient than analog retrieval. On the other hand, given the drawbacks of traditional CBR indexing (Section 3.2.1) a case index is only used for a bundle of important or
useful cases in the decision maker's case base. In line with the transactive memory theory (Section 4.2.1), selective indexing is also more flexible than traditional indexing in that it allows the decision maker to refer to cases in other case facilities by directly indexing them in his case facility.

5.2.3 Domain Vocabulary (Semantical Domain Knowledge)

Analogous to the operation and maintenance of a data dictionary in an organization, the DSS model foresees the maintenance of semantical knowledge in the organization. A case facility's **domain vocabulary** contains semantical knowledge which is highly domain-oriented and not part of common semantical knowledge (**general concepts**, Section 5.3.1). Semantical domain knowledge is defined as **semantical domain concepts**, similar in structure to WordNet concepts [Miller et al. 1988] (see ARCS, Section 3.3.4). Domain concepts, which are used to support analog retrieval, cover semantical definitions for nouns, verbs, and adjectives. In situations where a word has both general concept and domain concept definitions, the latter is strictly applied in context of the particular domain associated with the domain concept.

The domain vocabulary is not an active knowledge base, rather it serves as a mere working repository to define semantical domain concepts (i.e. by the decision maker or directly through a concept learning process) which are then used to update the semantical knowledge base in the central index facility. In other words, semantical domain knowledge is actually acquired and managed in several distributed case facilities but utilized (e.g. for analog retrieval) only through the use of the central semantical knowledge base. On one hand, this concept reflects the distribution of domain expertise in the organization, and on the other hand, it accounts for a more efficient way of utilizing decentralized semantical domain knowledge. Specific organizational members are assigned maintenance responsibility for a particular domain either by means of their
expertise in the field or simply by task assignment. Domain concept definitions which are updated by the responsible member are derived through direct input from the decision maker's experience or by concept learning during the decision support cycle (Section 5.4.4). For support of the analog retrieval process, domain vocabulary concepts from individual case facilities are regularly transferred to the central index facility to update the domain semantical part of the semantical knowledge base (Section 5.3.1).

5.3 The Central Index Facility

Besides serving as a network controller, the central index facility is primarily a platform for commonly accessible index and knowledge sources (Figure 5.3). It features a global domain index hierarchy and a global case index, both of which support selective index retrieval strategies. Analog retrieval, on the other side, is based on the semantical knowledge represented in the central index facility (semantical knowledge base). Organizational knowledge modeled on top of the domain hierarchy is also used in addition to locate certain case bases or relevant semantic information for case management. One should note that the central index facility does not provide a case base since case storage is decentralized and only directly associated with user's personal case facilities, nor does it provide any of the main case management functions incorporated in a case facility; it merely provides utility functions serving externally initiated requests to utilize index or knowledge sources in the course of global case operations.

5.3.1 Semantical Knowledge Base

The semantical knowledge base of the central index facility is the main source of knowledge support for analog retrieval which is used to determine case similarity.
Based on semantic word associations such as synonym, superordinate, or part-of relationships between words, semantic knowledge is represented in the form of concept definitions following the format used in WordNet (Section 3.3.4). In particular, the semantical knowledge base consists of two kinds of concepts for nouns, verbs, and adjectives: common language terms defined as general concepts and domain specific
terms defined as **domain concepts**. Because domain expertise is distributed across the organization, domain concepts are defined in selected, individual case facilities (domain vocabulary, Section 5.2.3) and regularly transferred to the central index facility in order to update the knowledge base. Both concept types are hierarchically interlinked, forming a semantical concept hierarchy from the most general terms to the most specific, domain related terms (Figure 5.4). The semantical knowledge base, present in the active memory of the central index facility, is consulted for any analog retrieval process, no matter if for retrieval within a case facility (internal retrieval) or for external retrieval across the network (for more details see Chapter Six).

---

**Figure 5.4:** Semantical concept hierarchy (example).
5.3.2 Hierarchical Domain and Case Index

The domain and case index is implemented mainly to serve domain oriented case retrieval. The domain hierarchy reflects existing case domains within the entire system, as defined by the users of the system. Any new case that is stored in the distributed system is indexed under the appropriate domain in the central index facility. If a new case is associated with a domain that is not included in the domain hierarchy then the hierarchy is required to be updated first. Figure 5.5 exhibits an example of a domain hierarchy index. Nevertheless, though indexing cases via domains is central to this DSS model, other indexing criteria, such as case creators, projects, clients, etc. are used as well.

5.3.3 Organizational Knowledge

Organizational knowledge in this context is understood to be administrative or domain relevant knowledge. It can be modeled with respect to the represented domain hierarchy, either directly in the form of domain associations or in the form of rules that become activated when cases from a domain or facility are accessed (e.g. through trigger functions). Domain associations allow for the specification of responsibilities for certain domain areas or the identification of domain expertise within the distributed environment (see Figure 5.5. for an example). It can also be used to represent task or project relationships with respect to participating facilities or covered domains. Note that the character of organizational knowledge can be normative (e.g. a person is assigned responsibility for a particular domain vocabulary) as well as descriptive (e.g. a person has managed a project in this domain).

Rules, on the other hand, become essential for case management when the access and transfer of case-based knowledge needs some form of security control among
Figure 5.5: Domain index hierarchy (example)
different members and different levels in the organization. Substantial work on dealing with issues such as security policy, obligation, and integrity in distributed systems is, for example, provided by Glasgow and MacEwen [1990]. However, issues of data security go beyond the scope of this thesis and will not be discussed further.

5.4 Case Management Processes of the Decision Support Cycle

This section presents the case management processes in light of the case-based decision support cycle (Figure 5.1). Compared to traditional CBR systems, the integration of analogical reasoning mechanisms, which are used for the retrieval and mapping of cases with conventional indexing techniques, is a novelty. This includes the use of an extended similarity concept based on semantical, structural, and pragmatic factors in order to determine the relevance of a case. Nevertheless, the assessment of case relevance is highly domain- and context-dependent; a case might be prototypical in one situation but not particular useful in another situation despite structural similarity in both problem situations. Besides semantic knowledge consultations, the decision maker’s judgement plays an important role in the case selection and thus in the overall decision process.

5.4.1 Case Representation

The computational representation of the decision maker’s experience in the form of cases is one of the key issues in the case-based decision support cycle. The case representation accounts for two main aspects: the appropriate representational format (case data structure) to capture the required case semantic and the encoding process to transform natural language descriptions into the internal case data structure.
Representation Format

In this model cases are considered rich descriptions of previously solved problems or decisions made in the context of organizational work experience (e.g. during assignments, projects, group tasks, meetings, etc.). Derived from Section 3.2, the representation of a case is based on a profile or schema incorporating a subset of the most salient features of the episode or problem experienced. For example, problem solving situations suggest a case schema that includes a brief description of the context (initial state, resources, operators, etc.), characterization of the problem situation (including constraints), a statement of the goal or objective, and a description of the solution and how it was achieved (for a formal representation see [Carbonell 1986]). Business cases may cover various related subjects like finance, accounting, marketing, management policy, etc. For this purpose a flexible format consisting of a basic set of case sections and identified by descriptors is used. For the same reason, a case is thought to describe only a particular problem situation in a business related task or project. Consequently, the description of tasks or projects may consist of many cases that are encoded under the same heading (e.g. project related facts) even though those cases are not necessarily causally related. The format is adaptable to any specific case situation by further adding appropriate sections. Inspired by the propositional data structure of Holyoak and Koh [1987] the following basic set of case descriptors (sections) is used:

1. **Case overhead** containing information about the associated case domains, the case access right (i.e. whether the case is accessible for other organizational members), causal links with other cases, case history (i.e. in what context and for what kind of problem was the case used as decision support previously).

2. **Initial state description** (facts) depicting descriptors like resources, operators and participants with the option to add previously defined domain specific descriptors when needed.

3. **Problem description** outlining the situation, the causal relevance of the problem,
important relationships, problem constraints, etc.

(4) **Goal description** stating the objectives that should be achieved; directives for solving the problem.

(5) **Solution description** including important steps of the solution in achieving the problem goal.

Note that the domain descriptor in the case overhead, which marks the associated domain(s) for a particular case, enables the system to assign the appropriate frame of reference for interpretation of the semantic concepts detailed in a case. The propositional format used for the case representation provides a convenient way to express relational structures without having to deal with too much context information, e.g. attributes of the objects involved. Nevertheless, with an additional frame structure associated at a deeper level, such information can be provided when needed. Basic propositions are also encapsulated in a frame data structure. A frame schema fulfills the requirements for a detailed object representation with the semantical context generically derived, for example, from a domain concept hierarchy. For a more detailed description of the representational format and data structures used for case representations see Chapter Six.

**Representation Process**

In a real system a case representation process would support the decision maker in the formulation and encoding of case descriptions\(^\text{12}\). It would be based on an **intelligent encoding interface** for the transformation of text descriptions into a system's internal representation format that is used for the storage and further processing of the case. This transformation is considered a semi-automatic process: based on the result of a natural

\(^{12}\) Note that the case representation process is not part of the D-CARE implementation (Chapter Six).
language parsing process, decomposed text is encoded into the new data structure under the advice of the decision maker. While the parser process is supported by a semantical knowledge base, the decision maker is asked to resolve and adjust semantical ambiguities that arise during the analysis. The entire representation process would be designed as an interactive, expert system like, consultation. Case representation takes place when cases are encoded as probe cases for retrieval (Section 5.4.2) or directly used to update the case base with any external case material such as documented business cases, legal files, or meeting notes.

5.4.2 Retrieval

The case retrieval process determines the most relevant case to a given case by applying analogical and/or indexing retrieval strategies. Going well beyond traditional CBR approaches, these strategies employ different measures to determine the relevance of a case. In terms of case locations, the retrieval strategies also differentiate between internal cases (i.e. cases in the facility's case base) and external cases (i.e. cases in other case facilities) (see Section 6.5 for retrieval specification details).

**Analog retrieval** is based on a complete similarity metric between two cases including semantical, structural and pragmatic features with the semantical features having predominant status in this process (Section 3.2). Since analog retrieval in general requires all the available semantical concept information, the process utilizes the semantical knowledge base in the central index facility for both the retrieval of internal and external cases. Once the semantical overlap between cases is determined, analog retrieval employs a partial mapping process to assess the correspondence with respect to all three similarity aspects (i.e. semantical, structural, pragmatic).

---

13 For details on the retrieval process, the reader is advised to refer directly to Chapter Six, particularly Sections 6.3 and 6.5.
Index retrieval, on the other hand, determines relevance only by an overlap metric regarding indexed case features and utilizes different indices depending on the location of cases. While using the domain and case index (Section 5.3.2) in the central index facility when retrieving external cases, the selective case index (Section 5.2.2) is primarily used for index retrieval applied to internal cases. As mentioned before, selective case indexing does not attempt to provide a complete index, rather it is intended for a set of cases selected by the decision maker on the basis of personal preference. Selective indexing is particularly useful once the relevance of a case is determined through an analog retrieval process. Moreover, retrieval results arising from either analogical or index retrieval can be further narrowed by the user by applying criteria based on preference and usefulness in context of a specific problem situation (Section 6.4).

5.4.3 Interpretation and Adaptation

The prime objective of this process is the use of cases as a basis

(a) in which to develop an interpretation (i.e. provide analysis or argumentation based on similarities why a problem is classified or treated like a precedent case) and/or

(b) from which to adapt existing solutions to generate a solution for the new problem.

A causal interpretation can be conveyed from a relevant case to a problem case if there exists an overall correspondence (i.e. semantical, structural, and pragmatic) with emphasis on structural features between elements of these two cases. For example, a current economic situation can be explained on the basis of the outcome of a previous situation with similar features and causal relationships. This process, which is primarily based on the user's input, is closely related to the traditional CBR interpretation and adaptation process [Rissland et al. 1989] but differs in the support provided.
If cases are retrieved by analog retrieval (Section 5.4.2), the system provides additional, quantitative similarity information that helps the user to determine where interpretations and explanations can be derived from those cases and for which parts of the problem they can be applied to. In particular, for this purpose the preceding mapping process (as part of the retrieval process, see also Section 6.1) emphasizes the type of correspondence (relation, object) and lists the numeric weight for each established correspondence between cases (see Section 7.2 for an example). If index retrieval was applied then an additional mapping process is often useful to obtain information about existing correspondences.

Usually the adopted solution for the problem is an amalgamation of a set of relevant case solutions adjusted to the specific problem situation at hand. The decision maker is responsible for the adjustment of the new problem by adding to or modifying the existing solution(s), thus providing the variation needed in the learning cycle. In addition, the decision maker not only profits from previous successful solutions, but is also warned by cases of problems that have been encountered in previous adaptation attempts, thereby avoiding potential failures. For this purpose, cases may be associated with information about success/failure as well as a history-of-use template identifying when and in which context situations a case (solution) was used.

5.4.4 Testing and Evaluation

Due to the nature of organizational decision making and problem solving, there is hardly a unique right or correct solution to a problem, rather a solution is highly dependent on context and usefulness. In fact, what was once useful might be again, although not necessarily. In order to justify an adopted solution or interpretation for a problem case the new case must be tested together with its proposed solution. Again, this process corresponds directly to the one in traditional CBR applications (Section 3.2). One way
is to test the robustness of the adopted case interpretation against counter-examples or hypotheticals [Rissland et al. 1989]; another way is to use the newly created solution as a memory probe and see whether it was already applied in other case instances. Both test strategies may themselves require the retrieval of additional cases.

Beyond that, the impact of a new case solution needs to be evaluated in light of its actual use before it can be fed back into the system for future decision support. For example, the decision maker might interrupt the decision support process and return at a later time to report upon the success or failure of the use of the case solution (i.e. the decision or solution applied in the case). Should the outcome substantially differ from what was expected, the adoption of alternative solutions may be considered before the case is stored as a failure. Note that feedback and judgmental evaluation by the decision maker are important inputs not only for the current decision process but for the learning progress and thus for the subsequent improvement in the system's decision support capabilities.

5.4.5 Case Base Maintenance

The case base maintenance process covers both the required case base updating functions to complete the Decision Support Cycle (DSC) as well as the basic utility functions of managing the access and organization of the case base.

Knowledge Acquisition through Case Base Updating

Encoding and storing cases is a basic form of knowledge acquisition. Updating the case base with a new case completes this process. Case-based knowledge can be added to the case base in either of three ways:
(a) as cases that were directly encoded from external sources such as files, books, notes, etc., without further processing (i.e. representation process only),

(b) as cases that were encoded and processed during the complete DSC, or

(c) as cases retrieved from external sources (other case facilities) during the DSC.

While (a) and (c) resemble simple knowledge acquisition processes, only (b) involves a learning step. Most cases, as a result of the decision making/problem solving process, are worth remembering. For example, a case of type (b) can be novel, well solved, exceptional, a failure etc. One may note that even multiple versions of a case often provide very useful sources in decision making, e.g. when different viewpoints, aspects, emphasis etc. result in several case versions by different organizational members across the network. However, within one's case base a decision maker should avoid cluttering with too many cases which provide insufficient variation for future decision making. Hence, before a new case is added to the case base the degree of overlap with cases already in the case base is checked to ensure the usefulness of the newly acquired case.

Index Update

Index information across the entire case network requires frequent updating. When a new case is stored or an existing case moved to a new location two kinds of indexes are affected: local index information in the case facility and index information in the central index facility. For every case to be inserted into a local case base, both domain and case index in the central index facility are automatically updated. The information for the case index update contains the case ID and the facility where the case is based, allowing later reference for external retrieval operations. Since a case can be associated with more than one domain, each relevant domain needs to be updated in the domain index hierarchy using the case ID as entry. When a case is associated with a domain which is not included in the index hierarchy, the index hierarchy itself must be updated first.
Case Base Access Control

By definition of the distributed case-based DSS model, a case base is accessed by internal processes as well as external access requests, i.e. from other facilities. Concurrency problems are primarily caused by situations when an external case request interferes with internal case base access operations, making concurrency control of access operations eminent.

In the same way, case base maintenance needs to control selective access rights. As mentioned in Section 5.2.1, not all cases in a decision maker's case base might be intended for organizational wide access. Access information is encoded as additional overhead with a case (Section 5.4.1) and checked at every external access request. General aspects of security policy, which can be modeled as organizational knowledge, are mentioned in Section 5.3.3.
D-CARE is a prototype implementation of the distributed case-based DSS model that, with respect to the scope of this thesis, focuses only on the case retrieval operation as discussed in Section 5.4.2. Given the key role of case retrieval in the decision support cycle the implementation is intended to offer important insight regarding performance, user interaction, and the feasibility of the projected case-based DSS. D-CARE is a single workstation simulation of a DSS configuration that incorporates a central index facility and two case facilities. This chapter provides a synopsis of the system's features and explains the implemented retrieval process with focus on the realization of the retrieval strategies and the supporting data structures in detail. The system was implemented in Common Lisp (Lucid Inc./Version 4.1) on a SUN Sparc IPC workstation. The distributed retrieval mechanism was developed on the basis of a pre-implemented version of the ARCS algorithm as described in [Thagard et al. 1990]. To render the functionality of the distributed case-based retrieval mechanism ARCS was significantly modified and enhanced by an interactive user interface.

6.1 Basic Concept

Case retrieval is implemented as an interactive, iterative process providing various opportunities for the decision maker to direct the progress and outcome of the retrieval. The decision maker is able to influence the retrieval process primarily in two ways:
Figure 6.1: Retrieval process flow.
through the retrieval initiation, that is when strategy, scope, probe, and other retrieval parameters are specified, or during the actual retrieval run when the system interactively consults the decision maker for his judgement or directions. Figure 6.1 exhibits the basic retrieval process flow reflecting functional relationships between different sub-processes and user's inputs.

Five main retrieval processes can be identified: the retrieval initiation process, index-based retrieval, analogy-based retrieval, user preference selection, and full mapping (optional). Initially, the retrieval process is specified by various parameters during retrieval initiation. The two major pathways for retrieval are determined by index and analog retrieval. Analogy-based retrieval is further segmented into semantic similarity assessment (Phase I) and partial mapping (Phase II). User preference selection and full mapping are optional steps following retrieval. For example, certain situations require one-to-one mapping to evaluate the correspondence between a probe case and retrieval outcome in more detail$^{14}$. Such mapping focuses not only on semantical concept correspondences but also attempts to find a mapping for every case feature based on structural correspondences. However, full mapping as well as index retrieval are currently not implemented.

6.2 Knowledge and Data Representation

Basically, three kinds of knowledge concepts are represented:

(1) Episodic knowledge in the form of case structures

(2) Semantic knowledge defining semantic word associations and used for

\[14\] In a future implementation full mapping will be based on the ACME (Analogical Constraint Mapping Engine) algorithm [Holyoak & Thagard 1989].
determining similarity between cases

(3) Organizational knowledge incorporated in the domain index

Furthermore, knowledge and data structures appear in different computational forms. That is, knowledge stored in active system memory (internal or temporary memory) is, for reasons of access efficiency, represented differently than knowledge stored in a data base (external or permanent memory).

6.2.1 Episodic Knowledge Representation

The representational format for cases is derived from the structure concept in ARCS [Thagard et al. 1990]. Relationships between objects are expressed as propositions in predicate calculus format with single- or multi-place predicates. Cases are stored in a frame-like format based on such propositions and are associated with an external as well as internal representation format.

External Knowledge Structure (Storage)

When cases are encoded in LISP and stored, e.g. in a data file, they appear as structured lists of the form

```lisp
'(case-name
   (TEXT ("text description"))
   (DOMAINS (domain_1, ... , domain_n_1))
   (ACCESS (facility_1, ... , facility_n_2))
   (HISTORY ((facility_1, case_1), ... ))
   (FACTS (p_f1, ... , p_fn))
   (PROBLEM (p_p1, ... , p_pn))
   (CONSTRAINTS (p_c1, ... , p_cn))
   (GOAL (p_g1, ... , p_gn))
   (SOLUTION (p_s1, ... , p_sn))
)
```
where TEXT provides a text description of the case, DOMAINS denote associated case domains, ACCESS lists those facilities that have access to the case (all facilities by default), HISTORY provides a history-of-use template of the case represented by tuples of the form (requested facility, associated case). This is followed by a variable set of case sections describing the case content, e.g. FACTS, PROBLEM, CONSTRAINTS, GOAL and SOLUTION. $p$ is a proposition of the type $p_{ki}$ where $k = \{f, p, c, g, s\}$ denotes the case section and $i = 1, \ldots, n$ serves as an enumeration of the propositions. Each proposition is of the form

$$(predicate \ (arg\_list) \ truth\_value \ label)$$

where predicate can be one of a set of special predicates (i.e. cause, if, conjoin-event, required) or any one-, two- or multi-place predicate, arguments in arg_list are either objects or propositions, truth-value is of type (true, false, unknown) and label identifies a proposition. Proposition labels appearing as arguments in arg_list are used to denote causal relations between propositions, sometimes associated with a propositional truth value. For example, proposition is3s-10

$$(cause \ (is3s-9 \ (is3p-4 \ t)) \ t \ is3s-10)$$

states that proposition is3s-9 causes proposition is3p-4 to become true. Note that every case and every proposition is uniquely labeled not only within a specific case facility but within the entire DSS. Further, every proposition is also uniquely associated with the case where it appears. In a real system this is automatically warranted by an algorithm at case encoding time.
Internal Knowledge Structure (Memory)

When a case is activated in active memory of a case facility (i.e. loaded from the case base or an external facility into the system memory) it is for reasons of access efficiency stored as a set of LISP property-lists. This internal format is basically of the form

\[
<\text{SYMBOL: case-name}>
\]

\[
\text{DATA-TYPE: case}
\]

\[
\text{SECTIONS: (list-of-case-section-labels)}
\]

\[
\text{PROPOSITIONS: (list-of-case-proposition-labels)}
\]

\[
\text{DOMAINS: (list-of-domains)}
\]

\[
\text{ACCESS: (list-of-facilities)}
\]

\[
\text{HISTORY: (list-of-facility-case-tuples)}
\]

\[
\text{section\_1: (list-of-propositions)}
\]

\[
\ldots
data_{section\_n: (list-of-propositions)}
\]

where \(<\text{SYMBOL: case-name}>> indicates that the property-list is stored under the LISP symbol \text{case-name}. In the same way, a property-list is installed for every proposition and for every predicate of a case. To keep track of this internal case representation schema overhead information, contained in additional lists such as SECTIONS and PROPOSITIONS, is used (see Appendix C for the complete internal case representation format). Note that the internal case data structure is of a temporary nature, only accessible as long as the case is maintained in memory. When not needed any more, the case structure (i.e. all related property lists) is deleted from memory and the case is said to be deactivated.

Part of the case information is used to update existing index information in the central index facility. In particular, the \text{case-name} and the list PROPOSITIONS are used to update BELONGS-TO and FROM-PROPNS, respectively, for each of the appropriate semantical concepts (Section 6.2.2) in the semantical knowledge base. In addition, the \text{case-name} is also used to update the domain case index and other relevant indices in the
central index facility. In the prototype implementation, this is done during the case activation process (since the actual case base is not part of the implementation). In a realistic implementation, however, the central index facility would be updated whenever a new case is stored the first time in any case base within the entire system.

6.2.2 Semantic Knowledge Representation

As outlined in Section 5.3.1, semantical similarity during analog retrieval is measured utilizing general and domain concepts, both available in the central index facility. As with case structures, there is an external definition format for semantical concepts as well as an internal memory format for facilitating access in memory. Analog to the external case format, semantic concepts are externally defined and encoded as structured lists. Internally, as presented below, they are stored as LISP property lists augmented by additional overhead information.

General Semantic Concepts

General concepts are internally represented under a property-list schema of the form:

```
<SYMBOL: concept-name>
  DATA-TYPE: concept
  BELONGS-TO: (list-of-case-names)
  FROM-PROPNS: (list-of-proposition-labels)
  SUPERORDINATES: (list-of-concepts)
  SUBORDINATES: ...
  SYNONYMS: ...
  PART-OF: ...
  SUB-PARTS: ...
  ANTONYMSS: ...
  PLURAL: concept-plural *
  TENSES: (list-of-tenses) *
```

* for nouns *
* for verbs/adjecitves *
where BELONGS-TO and FROM-PROPNS provide references to all the concept's appearances in cases and propositions of cases. This information in the central index facility is updated whenever a new case is stored somewhere in the distributed system. SUPERORDINATES, SUBORDINATES, SYNONYMS, PART-OF, SUB-PARTS, and ANTONYMS denote semantic relationships between the defined concept and other concepts. PLURAL and TENSES are used to identify related forms of the same concept's word stem. Each semantic relationship is associated with a specific weight as a measure for the semantic distance between the two words. Semantic weights, which are utilized by the constraint satisfaction method in determining the overall similarity between two cases, were adopted from ARCS [Thagard et al. 1989] with a range\textsuperscript{15} between + 1.0 and - 0.4. For general concepts in D-CARE these weights were the same as in ARCS; for domain concepts, additional weights were defined (see Appendix B for a list of semantic weights used in D-CARE).

**Semantic Domain Concepts**

Complementary to general concepts, domain concepts cover domain dependent vocabularies. By linking domain concepts to general ones, a hierarchical concept structure among concept definitions is achieved. The internal storage schema is derived from the general concept schema as shown.

\[
\text{<SYMBOL: concept-name>}
\begin{align*}
\text{DATA-TYPE:} & \quad \text{concept} \\
\text{BELONGS-TO:} & \quad (\text{list-of-case-names}) \\
\text{FROM-PROPNS:} & \quad (\text{list-of-proposition-labels}) \\
\text{FACILITY:} & \quad \text{facility}_n
\end{align*}
\]

\textsuperscript{15} The range for semantic weights is determined by criteria for the optimal processing of the constraint network algorithm (see also Section 6.3.3).
SUPERORDINATES:  (list-of-general-concepts)
SYNONYMS:  ...
DOMAINS:  (list-of-domains)
domain_1:  ((list-of-domain-synonyms) (list-of-instances))
...  ...
domain_k:  ((list-of-domain-synonyms) (list-of-instances))

FACILITY shows the facility where the concept and related concepts of the same domain are originally defined before being transferred to the central index facility, SUPERORDINATES provide the linkage to general concepts and DOMAINS exhibits the domains for which the concept is defined. In addition, for each domain a list of domain specific synonyms and a list of instances (if the concept is a noun) is provided. Constituting a form of organizational knowledge, instances are used to define a scope of possible values for a domain concept. For example, despite an abundance of synonyms for the concept consulting personnel, only a set of specific terms may be used for case descriptions within a department or a corporation.

EXAMPLE

consulting-personnel

FACILITY:  facility_1
SUPERORDINATES  (personnel staff)
SYNONYMS:  (consultant)
DOMAINS:  (IS-project-mgmt)
IS-PROJECT-MGMT:  ((information-consultant IS-mgmt-consultant IS-consultant IS-project-manager systems-programmer)
(IS-manager senior-consultant consultant programmer-analyst programmer))
6.2.3 Domain Case Index

The domain case index is hierarchically structured by linked domain fields. Every domain field is based on a property-list of the form

\[
\langle \text{SYMBOL: } \text{domain} \rangle \\
\text{SUPERDOMAINS: } \text{(list-of-domains)} \\
\text{SUBDOMAINS: } \text{(list-of-domains)} \\
\text{RELDOMAINS: } \text{(list-of-domains)} \\
\text{CASES: } \text{(list-of-cases)} \\
\text{FACILITIES: } \text{(list-of-facilities)} \\
\text{facility}_1: \text{ (list-of-cases)} \\
\ldots \\
\text{facility}_n: \text{ (list-of-cases)}
\]

where SUPERDOMAINS, SUBDOMAINS, and RELDOMAINS are references to other domains (i.e. multiple hierarchy possible). CASES is an identifier for all cases associated with this domain and \text{facility}_1, \ldots, \text{facility}_n show where these cases are located. This information is updated whenever a new case is stored anywhere in the distributed system.

6.3 Analog Retrieval Strategy

As shown in Figure 6.1, the analog retrieval procedure is basically designed as a two phase process:

**Phase I**  
Semantically similar candidate cases are identified on basis of common word associations and retrieved from internal and/or external case bases into the retrieval initiating facility (\textit{semantical similarity assessment}).

**Phase II**  
Then the most relevant case(s) are selected through assessing the relevance of the candidate cases in a competitive analogical mapping procedure (\textit{partial mapping}).
The segmentation of the retrieval process and the pre-assessment of the preliminary retrieval result are crucial in monitoring the retrieval progress as part of the entire decision support process. By evaluating the preliminary retrieval result the user is able to direct the retrieval outcome more efficiently. As a result of the initial similarity assessment, the user can opt whether to proceed with the retrieval process or to re-initiate the retrieval with new parameters in order to further limit the number of candidate cases. Since the second phase (analogical mapping) becomes increasingly elaborate and time consuming with a larger number of candidate cases, this option makes retrieval more efficient.

6.3.1 Adaptation of ARCS for Distributed Retrieval

The implementation of the analog retrieval procedure is based on the ARCS (Analog Retrieval by Constraint Satisfaction) algorithm (Section 3.3.4). Corresponding to retrieval Phase II, ARCS is a parallel constraint satisfaction mapping algorithm that applies semantical, structural and goal-oriented constraints. The mapping algorithm was augmented by a semantical similarity assessment procedure (Phase I), incorporating the global retrieval of potential candidate cases. In order to guide global retrieval within a distributed system, this procedure required additional knowledge sources and index information. In particular, for the semantical knowledge base in the central index facility semantical concept definitions were adopted from ARCS; existing ARCS concepts (about 1200) were used as general semantic concepts and augmented by additional 70 concepts definitions, and 30 concepts were defined as semantic domain concepts (Section 6.2.2).

Furthermore, in compliance with the functionality of a decision support system, stronger user involvement was emphasized in D-CARE. For this purpose, several ARCS procedures were modified and new ones developed to make the retrieval process more transparent to the decision maker. For example, support functions in the central index
facility were designed to generate preliminary retrieval information during Phase I to help the user decide the next step, i.e. proceeding with Phase II or re-initiating retrieval Phase I (see Section 7.2 for retrieval information provided by Phase I).

6.3.2 Retrieval Phase I - Semantic Similarity Assessment

Central to this phase is the identification of cases which are semantically similar to the probe case. The case similarity between two cases is based on the semantic overlap of key words, measured by the number of common word associations between these two cases. In order to be considered as a potential candidate case for the problem case on hand, a minimum number of associations is required. A semantic association is either a perfect match (identical words) or any semantic relationship between a word in one case and a word in another case. In the current implementation, only predicates are used to determine semantic overlap. For example, the predicate "submit" in case A is associated with the predicate "propose" in case B on the basis of a synonym relationship:

(submit (obj-peter obj-report) t Ap1) * case A *
(propose (obj-john obj-paper) t Bp1) * case B *

In a real system, objects would also be tested for semantic associations, e.g. showing that "paper" is a superordinate to "report". However, currently the argument list is only checked for matching objects in order to enhance the weighing on the association.

Associated predicates between cases are determined via the semantic knowledge base (Section 6.2.2) in the central index facility. Based on the concepts defined there, predicates from the probe case are used to probe into the knowledge base and collect all associated predicates with reference to other cases. This task of finding all associated predicates and their related cases is delegated by the case facility that initiated the retrieval to the central index facility (function send_request). In the central index facility,
the function \texttt{get\_assoc} identifies for each probe predicate a set of semantic associations and their relevant weights. It returns a paired list in the form

\[
((\text{assoc-pred}_1 \ \text{weight}_1) \ (\text{assoc\_pred}_2 \ \text{weight}_2) \ \ldots)
\]

to the retrieval facility where the list is stored under the probe predicate's proposition label to be used in Phase II. For example, probing with the one-place predicate "office" might result in the list

\[
((\text{office} \ 1) \ (\text{room} \ .3) \ (\text{work-place} \ .3) \ (\text{business} \ .6) \ (\text{position} \ .6) \ (\text{task} \ .6) \ (\text{department} \ .2) \ (\text{headquarter} \ .2)
\]

In order to assess a case as a potential candidate, the number of associations per case is counted during this process. Among all the cases identified only those that provide a minimum number of word associations are selected as potential candidates. The criterion \texttt{min-assoc} is defined by the decision maker as one of the retrieval initiation parameters. Next, for cases not already in active memory of the retrieval facility, storage locations are identified in order to load the cases into active memory for further processing. Note that before selected cases are loaded from their case bases into the retrieval facility, the decision maker can decide to restart the retrieval process again based on the preliminary result of the selected cases. For example, this would be advisable in case the number of pre-selected cases is too large.

\textbf{6.3.3 Retrieval Phase II - Partial Mapping}

In this process the selection of the most relevant case(s) from the set of potential candidate cases (Phase I) takes place. The relevance of a candidate case, a similarity
measure, is determined by mapping the probe case against all candidate cases in a parallel fashion (parallel constraint network). Using ARCS, case mappings are represented in a constraint network as hypothetical correspondences between propositions, predicates, and arguments of the cases mapped [Thagard et al. 1990]. The actual similarity assessment is performed by adjusting the weight activation pattern of the constraint network until a stable and continuous activation state is reached. In particular, having all relevant candidate cases loaded into the case facility the ARCS constraint network is accomplished in 3 steps:

(1) Utilizing the lists of weighted associations (Phase I), hypothetical correspondences between the probe predicates (P-pred-i) and associated candidate predicates (C-pred-j) are represented as units P-pred-i = C-pred-j. These predicate units are linked to a semantic unit by excitatory links, assigning each link the semantic weight that corresponds to the degree of the semantic similarity between the unit's predicates (i.e. semantic concept weight for the association P-pred-i = C-pred-j). The semantic unit itself is a special unit with the weight 1. In the same way, units are constructed for argument, proposition and case correspondences that are related to the predicate units. These units are connected with excitatory links to units for each probe/candidate correspondence hypotheses, assigning to each link a standard excitatory weight $excit^{16}$, typically 0.01.

(2) Inhibitory links are constructed between units representing incompatible hypotheses, e.g. between Probe = Cand-1 and Probe = Cand-2. These links make the similarity evaluation competitive in that the activation of one case will suppress the activation of alternatives.

---

16 The value for $excit$ is adopted from ARCS and follows Grossberg's [1978] suggestions for the constraint network calculation.
Pragmatic features are represented by marking elements (arguments, predicates, propositions) as important and hypothetical units as presumed to hold. Marking is achieved by linking units to a special pragmatic unit with the weight 1, assigning each link the standard excitatory weight. Note that pragmatic features can be set automatically (e.g. goal related predicates) or interactively determined by the decision maker in the context of the retrieval process. The decision maker also has the option to adjust any other semantic weight individually, according to his specific preference.

Once the network is constructed and initially activated with a minimum excitation level, the network is calculated in an iterative process. That is, each unit's activation is adjusted on the basis of the input of those units that are directly linked to it. One network cycle includes the activation of all network units. This process is repeated until the network has settled into a stable activation pattern (i.e. all units have reached asymptotic activation), usually after fewer than 150 cycles (Section 6.5.3). The activation of unit $j$ on cycle $t+1$ is calculated by Grossberg's formula [1978]. The highest unit activation for corresponding probe/candidate cases indicates the most relevant case(s). Instead of one specific unit the network often renders several units with high activation levels, separated by marginal differences. In such situations, the most relevant case can not simply be determined by the activation level, but the decision maker's judgement is required in a separate process. One way to choose from a set of relevant cases is by user preference selection.

6.4 User Preference Selection

Often selection of the relevant case for the problem on hand cannot be justified by quantitative retrieval criteria alone. For example, a numerical degree of similarity
between probe and candidate cases is a precise indicator for a case’s relevance only as long as it allows one to clearly isolate one case from other, similar cases. If results for several cases are numerically close, e.g. with less than 0.1 difference, then these cases can be considered to be almost equivalently relevant. In such situations, the selection of one or two cases from the retrieved ones is better decided in light of what the decision maker thinks is more useful and predictive for solving his problem. Useful is described as what addresses the decision maker’s goal best [Kolodner 1989], meaning his needs, expectations, or preferences.

The preferred selection of relevant support cases shares many of the criteria applied to the selective indexing strategy of cases. Following Kolodner [1989], several preference heuristics can be applied by the user:

- **Goal-Directed Preference** \(\text{Cases that help to support the decision maker's current goal.}\)
- **Salient-Feature Preference** \(\text{Cases that share outstanding and salient features with the current problem.}\)
- **Frequency Preference** \(\text{Cases that have been utilized more often than others show an established support record and should be preferred.}\)
- **Recency Preference** \(\text{Cases that have been used more recently than others are considered as more up to date than others.}\)

While the application of the first two heuristics is based on the goal section and the predicates of a candidate case, a special history attribute is used to record usage for the application of the last two heuristics. The heuristics themselves are thought of as rule-based functions that are evoked by the decision maker if the outcome from the main retrieval process (analog, index-based) needs further selection.
6.5 Retrieval Initiation

The retrieval initiation process serves as a menu driven platform for specifying the case retrieval request. As part of the interactive D-CARE user interface, the decision maker is provided with the options and the functionality primarily to define and direct the retrieval but also to monitor progress during subsequent retrieval runs (see Section 7.2 for an example session with D-CARE). Retrieval is defined in three main steps: initiation of retrieval strategy, specification of the retrieval parameters, and specification of the mapping algorithm's parameters (ARCS) for performance adjustment.

6.5.1 Retrieval Strategy

The retrieval strategy, which predetermines the retrieval behavior most significantly, is defined by the retrieval method and the retrieval scope.

- **Retrieval Scope**
  With the retrieval scope the decision maker indicates whether he wants to focus on cases from (a) his own case base (internal), from (b) other case facilities (external) or from (c) both (internal & external). Further specification of the retrieval focus is achieved by defining a probe data view (Section 6.5.2)

- **Retrieval Method**
  The decision maker chooses among (a) analog retrieval, (b) selective index-based retrieval or (c) a combination of both. While index retrieval is the more efficient and convenient method, analog retrieval offers a more effective search by encompassing a greater variety of cases as well as domains, but at the expense of larger time and processing complexity.
Unless a specific situation advocates a clear choice between one of these methods, (c) offers a strategic compromise: index-based retrieval starts with a narrow focus (i.e. in the decision maker's local case base) and, only if unsuccessful or not satisfying, widens the search spectrum with the analog approach to include also external cases.

6.5.2 Retrieval Parameters

In principle, retrieval is specified by a retrieval probe and the data spectrum that is probed into. Specifying the probe case is the primary step in this process. Additional parameters allow one to define these two factors in detail.

- **Probe Case**
  The probe case can either be selected directly from existing cases (internal), be created by modifying an existing case, or defined particularly for retrieval purpose. Note that a case does not have to be complete in order to serve as a probe, rather a probe is often an incomplete, unsolved problem case.

- **Probe Tailoring**
  Probe tailoring allows for the specification of parts such as the problem description or certain constraints of a case for using it as a retrieval probe. However, in order to avoid cluttering due to retrieving too many cases, the probe should be detailed enough, thereby restricting the retrieval outcome to a reasonable number of cases.

- **Importance**
  Further specification of the probe is possible by marking
important elements in the case. Marked arguments, predicates, and propositions are used as pragmatic constraints in retrieval Phase II.

- **Case Domain**

  The associated domain of a probe case is a parameter that determines the scope of semantic binding during retrieval. For example, a problem case from a domain, such as IS project management, utilizes a higher semantic similarity weighing when comparing cases from the same domain than with cases from other domains. If, however, aspects of other domains in this case are considered more important, for example human relation problems, they would not qualify for the same similarity measure, thus are discriminated in the retrieval of relevant support cases. To lend more weight to other case aspects in retrieval, relevant case domains can be added to a case probe.

- **Probe Data View**

  In addition to selecting the retrieval scope, the retrieval spectrum can be narrowed by specifying a view (i.e. subset) of the cases to be probed. Although a view can be principally defined by any set of constraints the current implementation only allows a selection by one of three criteria: (a) individual cases, (b) domain, and (c) facility. Since case retrieval is based on similarity rather than exact match and retrieval in an open system can involve too many case sources, defining a view beforehand is a recommended strategy to limit the number of potential candidate cases and thus the retrieval processing complexity.
• Concept Associations  A significant indicator for the relevance of a candidate case during analog retrieval is its number of shared concept associations with the probe (variable min-assoc). In effect during retrieval Phase I, this parameter is used for two reasons: it provides for the evaluation of potential candidates in terms of their relevance to the probe and it can be used as a constraint with a certain value, for example considering only cases with 15 or more concept associations as candidates (default 10).

6.5.3 ARCS Parameters

The parameters discussed in this section are related to performance details in constructing and adjusting the ARCS constraint network used in the retrieval Phase II.

• Excitation Level  The excitation level determines the basic weight used for standard excitatory links in the constraint network. The default value provided by the variable excit is 0.01 (Section 6.3.3). The variable can be adjusted (suggested range in ARCS: between 0.1 and 0.001) in order to emphasize or de-emphasize correspondences between non-predicate elements (i.e. objects, propositions, cases).

• Concept Significance  The concept significance, i.e. the semantic concept weighing, can be individually adjusted for selected case predicates. Derived from the decision maker's adjustment, a real system will also be able to learn context dependent concept weights which can be used in future analog retrieval.
• **Number of Cycles**

The number of cycles indicates the number of completed network processing runs used to adjust the network activation pattern. During one network cycle every unit is adjusted exactly once. Test results of Thagard and colleagues [1990] showed that 150 cycles were sufficient in most applications to reach an asymptotic activation state. However, trial retrieval runs showed that 50 cycles are often satisfying in order to evaluate the retrieval result.
Shifting attention to the user's role is an important factor in characterizing the functionality of a decision support model. The material provided in this chapter is intended to enlighten the user's interaction with the proposed distributed case-based DSS from a user's perspective. The interaction with the DSS is exemplified through a consultation with the D-CARE system that incorporates the typical steps of a decision maker during retrieval of relevant cases. Because IS project management is an interdisciplinary area, cases from the IS and related domains (i.e. human resource management, accounting, and the like) are predestined for the demonstration of both intra-domain as well as inter-domain knowledge transfer. For this application, cases were elicited from the IS consulting area (see Appendix D for a formal representation); part of the case frame definition was inspired by cases from Andersen Consulting. The consultation is based on a case scenario reflecting a particular problem situation during an IS consulting project. Note that the flow of the consultation session, supported through menu screens and reflected in Figure 7.1, also indicates points of the user's judgement and decision steps.

7.1 Case Scenario

A leading consulting firm was contacted by a former client to conduct a follow-up IS project. The consulting firm was asked to develop and install a data dictionary application interfacing with the already installed data base system at the client's site in
AN EXAMPLE CONSULTATION WITH D-CARE

INDEX-BASED

- INDEX RETRIEVAL

ANALOGY-BASED

- SEMANTIC SIMILARITY ASSESSMENT
  - PRELIMINARY RESULT
  - PARTIAL MAPPING (Phase II)
  - PRELIMINARY RESULT

USER PREFERENCE SELECTION

- PRELIMINARY RESULT
  - FULL MAPPING
  - FINAL RESULT

LEGEND
- Process
- Decision Step
- Retrieval Process Flow

Figure 7.1: User interaction during retrieval process.
Vancouver. The project was managed by Mr. Petersen while most of the development and actual implementation at the client's site was executed by Mr. Hill, a junior consultant. Although this consultant is knowledgeable in data modeling and has particular experience with using this data dictionary system, he had to rely on the support and expertise of the two older, more experienced client employees who were responsible for the data base system maintenance. However, the two employees did not acknowledge his competence, suggesting that they should take responsibility for the implementation. Consequently, the expected cooperation turned out to be rather dismal and a conflict surfaced as a result of the lack of supportive communication between Hill and the two employees. In order to get the project back on track, Petersen, who was informed about the on site problems, had to consider several alternatives. For this decision task he decided to consult the case-based DSS for some advice given in similar situations.

7.2 Example Consultation Session

At this point, the case description is assumed to be encoded and stored in the case base under the index label "Dismal Cooperation with Client during System Installation". This case is tagged as IS-Case-3-14\92, which is associated with the IS Project-14\92. Before specifying the retrieval parameters, Peterson ("the user" hereinafter) makes some changes in the case representation via the "CASE EDIT" command, then he selects "INITIATE RETRIEVAL" and the system responds with the D-CARE Retrieval Menu screen:
The user opts for a complete retrieval initiation (Selection 1). Following this selection, the user is guided through the retrieval initiation stage (Figure 7.1, Point A) in a sequence of three phases, i.e. retrieval strategy, retrieval parameters and ARCS parameters (for details see Section 6.5). Corresponding to the fact that one may usually wish to start from his own experience before consulting others for their advice, the user decides to scan his own case base for relevant cases first (Selection 1):
This is intended to provide an initial overview. Also, for efficiency reasons, this search is, by default, based on selective index retrieval using the case label as an index feature. The user specifies case IS-Case-3-14\92 as the retrieval probe. The data, which is to be probed, is defined by cases in the IS-Project-Mgmt domain (domain view) corresponding to the domain of the retrieval probe. Before the actual retrieval is started, all retrieval settings, as chosen by the user, are summarized for confirmation in a separate screen:

*** D-CARE RETRIEVAL SETTINGS ****************************************** D-CARE R8 ***

Retrieval Run: 1

*** RETRIEVAL STRATEGY ***********************************************
Retrieval Scope: internal
Retrieval Method: index-based

*** RETRIEVAL PARAMETERS **********************************************
Probe Case: IS-CASE-3-14\92
Probe Domains: (IS-PROJECT-MGMT)
Data to be Probed: Domain> (IS-PROJECT-MGMT)

Press ENTER to start retrieval, or (n) to change settings > y

This feedback allows the user to make corrections or alterations before the retrieval process is actually started. Note that the user does not have to pass through all retrieval specification stages again, rather he can now make changes with respect to particular parameters. Finally, after the first retrieval run (selective index retrieval), the system reports one case back where conflict arose between participating client employees instead of between client and consultant. Since the user is able to identify the recalled case as being irrelevant to his problem (Figure 7.1, Point B), he restates the retrieval setting for a externally focused, analogically-based retrieval run (again point A):
The retrieval specification required additional parameters as a result of the analog retrieval method, as shown in the retrieval setting. For example, the field Probe Case Sections (i.e. case tailoring) shows that the entire probe case is involved in the retrieval, although a partial selection of sections would have been possible. No domains other than the core domain (IS-Project-Mgmt) are used for the retrieval probe, and no important objects or predicates are marked for this run. The minimum number of concept associations that is chosen to determine a relevant case is 10 (default), while the selected default setting for the ARCS network excitation level is 0.01. Further, the user finds that 50 calculation runs are sufficient for a preliminary determination of relevant cases.
The central index facility, which might be stationed at some other, remote location, reports intermediate retrieval results to the requesting case facility after the completion of the semantic similarity assessment (i.e. Phase I). In the screen message below, the Number of Probe Predicates indicates the number of unique probe predicates which are used for finding associated concepts (see 6.3.2). Particularly important is the Number of Associated Predicates per Case: these figures show the semantic similarity between each of these cases and the probe, and thus indicates beforehand the most promising candidates for Phase II. Note that every case with at least one associated predicate is listed as an associated case. Based on these figures, the user can adjust the min-assoc criterium, which determines the class of candidates for the second retrieval phase.

*** PRELIMINARY RETRIEVAL STATISTICS 1 ****************** D-CARE RUN ***

Probe Case: IS-CASE-3-14\92
Number of Probe Predicates: 30
Domains Probed into:
IS-PROJECT-MGMT
HUMAN-RESOURCE-MGMT
ACCOUNTING

Number of Associated Cases: 8
Total Number of Associated Predicates: 32
Number of Associated Predicates per Case:

<table>
<thead>
<tr>
<th>Associated Cases</th>
<th>Associated Predicates</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 ACCOUNTING-CASE-16\91</td>
<td>Facility_28</td>
</tr>
<tr>
<td>14 IS-CASE-2-06\92</td>
<td>Facility_2</td>
</tr>
<tr>
<td>15 IS-CASE-14-34\91</td>
<td>Facility_2</td>
</tr>
<tr>
<td>18 IS-CASE-4-05\91</td>
<td>Facility_8</td>
</tr>
<tr>
<td>19 IS-CASE-11-37\90</td>
<td>Facility_6</td>
</tr>
<tr>
<td>16 IS-CASE-8-23\90</td>
<td>Facility_8</td>
</tr>
<tr>
<td>7 HUMAN-RES-CASE-28\91</td>
<td>Facility_17</td>
</tr>
<tr>
<td>5 MARKETING-CASE-2-08\91</td>
<td>Facility_34</td>
</tr>
</tbody>
</table>

Number of Cases above Min-Assoc: 6
Cases to be loaded: NIL

************************************************************************************
By indicating the facility name (e.g. Facility_28) for each associated case the user is also informed from where the case came from. Given this intermediate result, the user recognizes that too many IS-cases were selected with relatively indifferent amounts of associated predicates. After viewing some of the IS case representations, he concludes that many associations result from a match in the relatively uniform, initial state sector of the IS cases (Appendix D). Since this section is not relevant for his problem he decides to use a reduced retrieval probe (i.e. problem description, constraints and goal description) and run Phase I again (Figure 7.1, Point C). He also changes the min-assoc requirement from 10 to 5 for this retrieval. These changes are reflected in the following retrieval output:

*** PRELIMINARY RETRIEVAL STATISTICS 1 ****************** D-CARE RUN ***

Probe Case: IS-CASE-3-14\92
Number of Probe Predicates: 17
Domains Probed into:
   IS-PROJECT-MGMT
   HUMAN-RESOURCE-MGMT
   ACCOUNTING

Number of Associated Cases: 4
Total Number of Associated Predicates: 16
Number of Associated Predicates per Case:
   11   ACCOUNTING-CASE-16\91 (Facility_28)
   5    IS-CASE-14-34\91  (Facility_2)
   6    IS-CASE-8-23\90   (Facility_8)
   4    IS-CASE-11-37\90  (Facility_6)

Number of Cases above Min-Assoc: 3
Cases to be loaded: NIL

Press ENTER to continue, (n) to change retrieval settings > n
This result is more differentiated, with case ACCOUNTING-CASE-16\91 emerging as a clear favorite (11 associated predicates). Altogether three cases pass the min-assoc criterium and are applied in the partial mapping process (i.e. Phase II). After ending Phase II, the system responds with the final retrieval result:

*** FINAL RETRIEVAL STATISTICS *****************************************************

Results of probing with IS-CASE-3-14\92 after 52 cycles:

<table>
<thead>
<tr>
<th>Case Match</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCOUNTING-CASE-16\91</td>
<td>0.5344</td>
</tr>
<tr>
<td>IS-CASE-8-23\90</td>
<td>0.2335</td>
</tr>
<tr>
<td>IS-CASE-14-34\91</td>
<td>0.0978</td>
</tr>
</tbody>
</table>

-----------------------------------------------------------------------------

This result presents, for each hypothetical match, the calculated network activation, an indicator of the strength of the hypothesis or, in other words, the degree of similarity between the two cases. With the possible values being between +1.0 (max) and -.4 (min), this activation confirms ACCOUNTING-CASE-16\92 as the most similar case with an activation of 0.5344. To interpret the similarity between the problem and this case (individual correspondences), a more detailed listing of the mapping results can be requested.

*** MAPPING RESULT *************************************************************

<table>
<thead>
<tr>
<th>Mapping</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCEPT ACKNOWLEDGE</td>
<td>0.2758</td>
</tr>
<tr>
<td>COMPETENCE AUTHORITY</td>
<td>0.2422</td>
</tr>
<tr>
<td>COMPLEX COMPLEX</td>
<td>0.1849</td>
</tr>
<tr>
<td>COOPERATE COOPERATE</td>
<td>0.4111</td>
</tr>
<tr>
<td>HILL BENDER</td>
<td>0.1041</td>
</tr>
<tr>
<td>Best mapping of object</td>
<td>Corresponding object</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>MANAGER</td>
<td>CONTROLLER-MANAGER</td>
</tr>
<tr>
<td>MOTIVATED</td>
<td>MOTIVATED</td>
</tr>
<tr>
<td>NEW-TECHNOLOGY</td>
<td>NEW-TECHNOLOGY</td>
</tr>
<tr>
<td>PETERSEN</td>
<td>OBJ-CONTROLLER-MANAGER</td>
</tr>
<tr>
<td>OLDER</td>
<td>OLDER</td>
</tr>
<tr>
<td>IS1C-2</td>
<td>ACC1C-2</td>
</tr>
<tr>
<td>IS1C-7</td>
<td>ACC1C-6</td>
</tr>
<tr>
<td>IS1C-5</td>
<td>ACC1C-1</td>
</tr>
</tbody>
</table>

In particular, the mapping result shows correspondences between relationships (e.g. ACCEPT and ACKNOWLEDGE), objects (e.g. PETERSEN and OBJ-CONTROLLER-MANAGER) and attributes (OLDER and OLDER) (see Appendix E for a list of the complete mapping result). Activation values indicate individual strengths for the correspondences. If these results are not satisfying the user always has the option to apply a full mapping procedure; on the other side, he can also interrupt the session and proceed at a later time (Figure 7.1, Point D).

### 7.3 Description of Retrieved Case

The retrieved case ACCOUNTING-CASE-16 is similar to IS-CASE-3-14 in that it reflects the age/competence conflict but in a different setting. An assisting accountant/manager is responsible for some of the company’s controlling functions and reports directly to the corporate controller. With new equipment and software installed, a younger accountant with state-of-the-art skills in the field of computer analysis and cost controlling is assigned to support the accountant/manager. However, the older and more experienced accountant does not readily accept the "rookie", resulting in reduced cooperation. To solve the conflict the controller temporarily decides to directly supervise
the activities of the new accountant thus confirming the newcomer’s role in implementing new controlling systems. The key aspect in the solution is that the controller pursues a bridging role in fostering the cooperation between these two employees.
CHAPTER EIGHT
CONCLUSION AND FUTURE RESEARCH

8.1 Conclusion

The presented thesis emphasized a new perspective for the design of organizational decision support systems. Central to this effort was the concept of a distributed case-based DSS model that guides decision making by reference to case-based experiences. Chapter Two and Three provided the motivational and theoretical background of this work. Chapter Two discussed some of the shortcomings of existing DSSs regarding the support of human problem solving behavior while Chapter Three outlined the principles of analogical transfer, in particular Analogical Reasoning (AR) and Case-Based Reasoning (CBR). The major contributions of this work are presented in Chapters Four to Six. These can be summarized as follows:

- Extending the case-based reasoning paradigm to support decision making in a distributed environment was the first contribution. This was achieved through the development of the Distributed Case Management (DCM) concept, which is used as a basis for organizational decision support. The DCM concept foresees the facilitation of experiential knowledge transfer among cooperating organizational workers. It also provides basic learning capabilities that allow for domain independent maintenance and improvement of an experiential knowledge base in the organization (corporate memory).
• Based on the DCM, a series of new DSS capabilities was designed by incorporating an advanced analog retrieval and mapping mechanism. This was the second contribution. In contrast with traditional DSSs, the proposed DSS model is conceived as a decision support environment that primarily assists human problem solving behavior through the storage and recall of work experiences. Also, in contrast with current approaches to case-based decision aiding, the analog approach not only supports the transfer of experiential knowledge within a specific domain but across domains. The acquisition and maintenance of semantic domain knowledge and the semantic similarity assessment process support this capability. In addition, this approach provides the means for discovering new experiential knowledge within the distributed organizational DSS environment.

• The third contribution of this work was to demonstrate the feasibility of the DCM concept and the analogical transfer approach. This was done in two ways. First, a prototype implementation simulated the distributed retrieval concept on a single workstation (D-CARE). A significant part of this implementation is the user's participation model that allows a gradual focusing on retrieved cases. Second, the use of D-CARE was demonstrated through an example application in the domain of IS project management with focus on the user's interaction.

In general, what has been presented within the boundaries of this work is intended to provide new perspectives towards the goal of designing more advanced organizational (decision) support tools. Although the material incorporates some significant contributions and "food for thought", more research is required to achieve this goal.
8.2 Future Perspective

Completion of the Proposed DSS

The retrieval of relevant analogues, as realized in D-CARE, is only one part of a case-based DSS. Other components of the decision support cycle (Section 5.4), such as interpretation/adaptation, testing/evaluation, and the case base maintenance are equally significant in the overall decision support process. Of particular importance for the interpretation of retrieved cases is the mapping process that places priority on structural constraints. Mapping will be realized on the basis of ACME (Analogical Constraint Mapping Engine) [Holyoak & Thagard 1989], a model that is technically analogous to ARCS. On the other hand, pragmatic constraints play a dominant role in the adaptation and evaluation of a case solution. The design of a suitable and sophisticated user interface is also an essential part of successful decision support. Integration and implementation of all these components are deferred to future projects. This will also include the development of an actual DSS implementation that takes system related operating and network functions into account. The MOAP architecture [Woo & Lochovsky 1992] would be a suitable platform for such a system. Finally, a solid empirical evaluation that investigates the adequacy of the purported decision model under realistic conditions in an organizational environment will also be an important contribution to these projects.

Future Research

Furthermore, this work provides inspirations for a variety of subsequent research efforts:

- The development of a case representation component that incorporates a natural language front-end (Section 5.4.1) is viewed as essential to elevate the system's user
acceptance in an organizational environment. Text-oriented case encoding and editing are key factors for the conception of the DSS user interface. The primary aim here is to provide a convenient support tool that is easy to use without burdening organizational personnel with unnecessary encoding formalisms.

- The augmentation of more learning capabilities in addition to the basic case-based learning functionality is a move towards more intelligent decision support. For example, extraction and generation of prototypical cases from a collection of case experiences is a useful method to prevent the cluttering of the database with too many, marginally different cases. Cases with special features are then identified as special exemplars of prototypical cases. Inductive learning methods, such as conceptual clustering [Michalski 1983] or concept formations through generalizations, are possible techniques that can be applied here. Another aspect of learning is the automatic elicitation of semantic concepts as a result of case mappings and the adjusting of semantic weights based on frequency of occurrence of semantical matches.

- Since case-based reasoning is only one part of human reasoning behaviors, integrating it with methods of rule-based reasoning is required for a comprehensive (cognitive adequate) decision support approach. In the context of learning enhancement, rules can be extracted from a cluster of cases once episodic knowledge becomes more evident and commonly established among organizational users. The evolution from case- to rule-based knowledge is a form of knowledge level learning [Dietterich 1990]. Again, since rules usually express generally valid knowledge, this is also another example of organizational knowledge migration.
REFERENCES


REFERENCES


REFERENCES


APPENDICES

APPENDIX A: Examples of Semantical Concepts

A semantical concept is a frame-like representation used to encode semantic information associated with a word. In ARCS and in D-CARE (Chapter 6), semantical concepts are modeled after WordNet [Miller et al. 1988], an electronic reference system based on psycholinguistic theories of the organization of human lexical memory. For a noun, semantic information is represented as kind-of, part-of, and synonym relations forming lexical hierarchies. For example, the noun ANIMAL is a kind-of ORGANISM which makes ORGANISM a superordinate of ANIMAL and ANIMAL a subordinate of ORGANISM. Semantic information for verbs and adjectives is represented in a much simpler form. In the following, samples for all three word forms are listed.

ANIMAL (noun)
- SUPERORDINATES: organism, living-thing
- SUBORDINATES: prey, mammal, primate, reptile, amphibian, fish, bird, insect, vertebrate, game
- PARTS: voice, tooth, tail, claw, antler
- PLURAL: animals
- SYNONYMS: beast, creature, fauna
- ANTONYMS: plant, flora

INVOLVED (adjective)
- SYNONYMS: complex, intricate, committed, dedicated, engrossed, immersed, absorbed, rapt
- ANTONYMS: simple, distracted, absent

RESIGN (verb)
- TENSES: resigns, resigned, resigning
- SYNONYMS: leave, retire, quit, cede, abdicate, surrender, reconcile, submit
- ANTONYMS: remain, keep-position, maintain
APPENDICES

APPENDIX B: Semantic Weights used in D-CARE

The semantical similarity between words can be differentiated based on the kind of semantical relationship between them. For example, a synonym is interpreted to be more similar to a word than a superordinate. By assigning individual weights to different kinds of semantical relations, semantic similarity can be expressed quantitatively and used in the constraint network calculation. Weights for D-CARE were essentially those used in ARCS. According to Thagard and colleagues [1990], the allocation of weights in ARCS is supported by empirical data on psychological word association tests. However, there is no strong theory for a particular allocation of weights.

Weights were defined for the following semantical relations:

<table>
<thead>
<tr>
<th>Relation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDENTITY</td>
<td>1.0</td>
</tr>
<tr>
<td>SYNONYM</td>
<td>0.6</td>
</tr>
<tr>
<td>SUPERORDINATE</td>
<td>0.3</td>
</tr>
<tr>
<td>SUBORDINATE</td>
<td>0.2</td>
</tr>
<tr>
<td>PART-OF</td>
<td>0.1</td>
</tr>
<tr>
<td>SUB-PARTS</td>
<td>0.0</td>
</tr>
<tr>
<td>ANTONYMS</td>
<td>-0.4</td>
</tr>
</tbody>
</table>

PLURALS are treated like identities.

For semantic domain concepts in D-CARE additional weights were defined to express context dependent similarities:

- For SYNONYMS in key domains = 0.8
- For SYNONYMS in other domains = 0.5
APPENDIX C: Internal Case Representation Format

In ARCS as well as in D-CARE, cases utilize a special, internal representation format in order to maximize efficiency of the constraint network process. The format takes primarily advantage of LISP property-lists which are internally managed by hash indices, thus allowing efficient access to all case relevant features. In addition, lists are used to maintain case format relevant overhead information.

A particular case is represented by a property-list:

```
<SYMBOL: case-name>
   DATA-TYPE: case
   SECTIONS: (list-of-case-section-labels)
   PROPOSITIONS: (list-of-case-proposition-labels)
   DOMAINS: (list-of-domains)
   ACCESS: (list-of-facilities)
   HISTORY: (list-of-facility-case-tuples)
   section_1: (list-of-propositions)
       ...
   section_n: (list-of-propositions)
```

Each proposition of this case is represented by a property-list:

```
<SYMBOL: proposition-label>
   PROPOSITION: (predicate (arg-list) truth-val label)
   BELONGS-TO: ((case-name section) ( ... ) ... )
```

Each predicate (i.e. predicates of the propositions) is represented by a property-list:

```
<SYMBOL: predicate>
   BELONGS-TO: (list-of-case-names)
   FROM-PROPNS: (list-of-proposition-labels)
```

The following lists store information collectively for all cases:

```
all_cases = (list-of-case-names)
all_propositions = (list-of-proposition-names)
all_predicates = (list-of-predicates)
```
APPENDICES

APPENDIX D: D-CARE Application Cases

CASE-1 describes the typical age/acceptance conflict among cooperative workers. The younger, external consultant who is responsible for the installation and has the necessary knowledge for the new database implementation is not readily accepted by the two older, more experienced employees at the client’s site. Consequently, cooperation is reduced and motivation of the workforce low. Despite the higher cost (time, traveling, etc.), an additional experienced manager is brought in for a critical interim period.

(make_case 'IS-project-1
  '(IS-project-mgmt human-resource-mgmt)
  '(facts
    ((office (Minneapolis) t is1f-1)
     (project (obj-database-implementation) t is1f-2)
     (client (Amer-Hospital-Corp) t is1f-3)
     (client-location (Florida) t is1f-4)
     (client-industry (obj-health-care) t is1f-5)
     (client-position (obj-committee) t is1f-6)
     (contract (obj-cooperative-prime) t is1f-7)
     (contract-status (obj-follow-up) t is1f-8)
     (project-status (obj-closed) t is1f-9)
     (office-personnel (Petersen) t is1f-11)
     (office-personnel (Hill) t is1f-12)
     (client-personnel (Smith) t is1f-13)
     (client-personnel (Koch) t is1f-14)
     (equipment-type (obj-dbms) t is1f-15)
     (equipment-vendor (Online-Corp) t is1f-16)
    )
   )
  )
  )
  '(problem
    ((experience (obj-experience) t is1p-1)
     (competence (Hill) t is1p-2)
     (responsible (Hill obj-database-implementation) t is1p-3)
     (new-technology (obj-database) t is1p-4)
     (complex (obj-database-implementation) t is1p-5)
     (cause (is1p-4 is1p-5) t is1p-6)
     (younger (Hill Smith) t is1p-7)
    )
   )
  )
(younger (Hill Koch) t is1p-8)
(older (Smith Hill) t is1p-9)
(older (Koch Hill) t is1p-10)
(conjoin-event (is1p-7 is1p-9) t is1p-11)
(conjoin-event (is1p-8 is1p-10) t is1p-12)
(accept (Smith is1p-2) false is1p-13)
(accept (Koch is1p-2) false is1p-14)
(cause (is1p-11 is1p-13) t is1p-15)
(cause (is1p-12 is1p-14) t is1p-16)
(cooperate (Smith Hill) false is1p-17)
(cooperate (Koch Hill) false is1p-18)
)
)
'(constraints
((far-away (Minneapolis Florida) t is1c-1)
 (require (obj-database-implementation obj-experience) t is1c-2)
 (cause (is1p-5 is1c-2) t is1c-3)
 (cause (is1c-2 is1p-3) t is1c-4)
 (required-for ((is1p-17 t) obj-success) t is1c-5)
 (required-for ((is1p-18 t) obj-success) t is1c-6)
 (motivated (Hill) false is1c-7)
 (cause (is1p-17 is1c-7) t is1c-8)
 (cause (is1p-18 is1c-7) t is1c-9)
 (required-for ((is1c-7 t) obj-success) t is1c-10)
)
)
'(solution
((manager (Petersen) t is1s-1)
 (competence (Petersen) t is1s-2)
 (accept (Smith is1s-2) t is1s-3)
 (accept (Koch is1s-2) t is1s-4)
 (cooperate (Smith Petersen) t is1s-5)
 (cooperate (Koch Petersen) t is1s-6)
 (support-until (Petersen Hill (is1p-17 t)) t is1s-7)
 (support-until (Petersen Hill (is1p-18 t)) t is1s-8)
 (cause (is1s-7 (is1p-27 t)) t is1s-9)
 (cause (is1s-8 (is1p-27 t)) t is1s-10)
)
)
CASE-2 deals with a design problem. Users were not satisfied with the existing user interface for a complex online information system. Initial tests showed that users had difficulties adapting to the new user interface and that graphical icons were not supportive in this process.

(make_case 'IS-project-2
   '(IS-project-mgmt)
   '(facts
      ((office (Chicago) t is2f-1)
       (project (obj-user-interface-enhancement) t is2f-2)
       (client (Maryland-Port) t is2f-3)
       (client-location (Maryland) t is2f-4)
       (client-industry (obj-government) t is2f-5)
       (client-position (obj-senior-official) t is2f-6)
       (contract (obj-cooperative-prime) t is2f-7)
       (contract-partner (IBM) t is2f-17)
       (contract-status (obj-follow-up) t is2f-8)
       (project-status (obj-closed) t is2f-9)
       (office-personnel (obj-consultants) t is2f-11)
       (client-personnel (obj-users) t is2f-13)
       (equipment-type (obj-mainframe) t is2f-15)
       (equipment-vendor (IBM-4300) t is2f-16)
    )
   )
   '(problem
      ((traditional-system (obj-info-system) t is2p-1)
       (functionality (obj-funct1) t is2p-2)
       (user-interface (obj-tui) t is2p-3)
       (has-feature (obj-info-system obj-tui) t is2p-4)
       (has-feature (obj-info-system obj-funct1) t is2p-5)
       (support (obj-tui obj-funct1) t is2p-6)
       (used-to (obj-users obj-funct1) t is2p-7)
       (functionality (obj-funct2) t is2p-8)
       (modified-to (obj-funct1 obj-funct2) t is2p-9)
       (complex (obj-funct2) t is2p-10)
       (used-to (obj-users obj-funct2) false is2p-11)
       (satisfied-with (obj-users obj-tui) false is2p-12)
       (support (obj-tui obj-funct2) false is2p-13)
       (cause (is2p-13 is2p-12) t is2p-14)
       (motivated (obj-users) false is2p-15)
    )
   )
(cause (is2p-14 is2p-15) t is2p-16)
)
)
'(constraints
 ((required-for (obj-info-system conc-work) t is2c-1)
 (dependen-on (obj-users obj-info-system) t is2c-2)
 (cause (is2c-2 is2c-1) t is2c-3)
 (required-for (obj-funct2 conc-work) t is2c-4)
 (sufficient (obj-tui (is2p-13 t)) false is2c-5)
 (limited (obj-funds) t is2c-6)
 (redesign (obj-info-system) false is2c-7)
 (cause (is2c-6 is2c-7) t is2c-8)
)
)
'(solution
 ((augment (obj-tui) t is2s-1)
 (change (obj-tui obj-gui) t is2s-2)
 (user-interface (obj-gui) t is2s-3)
 (more-user-friendly (obj-gui obj-tui) t is2s-4)
 (support (obj-gui obj-func2) t is2s-5)
 (cause (is2s-4 is2s-5) t is2s-6)
 (design (obj-consultants obj-gui-features) t is2s-7)
 (co-design (obj-users obj-gui-features) t is2s-8)
 (implement (obj-consultants obj-gui) t is2s-9)
)
)
)

*******************************************************************************
CASE-3 depicts a similar situation to CASE-2. For a new desktop application a graphical user interface (GUI) with several new images and icons was developed. Unfortunately, users were less enthusiastic about the new design than the developers, since they interpreted icons inconsistently. In the revision, a set of alternative icons and image frames were prototyped so users could arrange and create their own frames. The solution was to let users actively take part in the GUI design process (co-design, create, etc.) in order to achieve satisfaction with the outcoming result.

(make_case 'IS-project-3
   '(IS-project-mgmt)
'(facts
  ((office (New-Jersey) t is3f-1)
    (project (obj-gui-design) t is3f-2)
    (client (Fibertel-Inc) t is3f-3)
    (client-location (New-York) t is3f-4)
    (client-industry (obj-consumer-products) t is3f-5)
    (client-position (obj-senior-official) t is3f-6)
    (contract (obj-sole-contractor) t is3f-7)
    (contract-status (obj-follow-up) t is3f-8)
    (project-status (obj-closed) t is3f-9)
    (office-personnel (manager) t is3f-10)
    (office-personnel (designer) t is3f-11)
    (client-personnel (users) t is3f-12)
    (equipment-type (obj-gui-environment) t is3f-13)
    (equipment-vendor (MultiVision-Inc) t is3f-14)
  )
)
)
')(problem
  ((user-interface (obj-gui) t is3p-1)
    (develop (obj-designer obj-gui) t is3p-2)
    (icon (obj-icon) t is3p-3)
    (frame-image (obj-frame-image) t is3p-4)
    (has-feature (obj-frame-image obj-icon) t is3p-5)
    (misinterprete (obj-users obj-icon) t is3p-6)
    (has-many (obj-gui obj-frame-image) t is3p-7)
    (design (obj-frame-image) t is3p-8)
    (hard-to-separate (obj-frame-image) t is3p-9)
    (satisfied-with (obj-users obj-gui) false is3p-10)
    (cause (is3p-9 is3p-10) t is3p-11)
    (design (obj-gui-process) t is3p-12)
    (influence (obj-users obj-gui-design) false is3p-13)
  )
)
')(constraints
  ((support (obj-gui obj-functionality) unknown is3c-1)
    (required ((is3c-1 t)) t is3c-2)
    (complex (obj-functionality) t is3c-3)
    (suit (obj-design obj-user-needs) unknown is3c-4)
    (required ((is3c-4 t)) t is3c-5)
    (selection (obj-design (is3c-4 t)) unknown is3c-6)
    (many (obj-frame-image) t is3c-7)
    (know (obj-users obj-user-needs) t is3c-8)
  )
CASE-4 describes a funding problem that jeopardizes an ongoing IS-project. The project (an AI application) bears more risk than traditional projects; consequently, the client management is more cautious than usual about the progress of the project. As costs increase beyond budgetary limits, without the client's consent, the client management decides to halt the project, because it was believed that accomplishment of the project was in doubt. As a solution, the consultants prepared a new interim prototype that persuades the client management to continue the project since the problem bottleneck was just temporary in nature.
(project-status (obj-in-progress) t is4f-8)
(office-manager (Lee) t is4f-9)
(office-personnel (obj-consultant) t is4f-10)
(client-personnel (Johnsen) t is4f-11)
(equipment-type (obj-mainframe) t is4f-12)
(equipment-type (obj-PC) t is4f-13)
(equipment-vendor (IBM) t is4f-14)
)
)
'(problem
((in-progress (obj-project) t is4p-1)
 (cost-increase (obj-project-costs) t is4p-2)
 (encounter (obj-consultants obj-technical-bottleneck) t is4p-3)
 (exceed (obj-project-costs obj-project-budget) t is4p-4)
 (cause (is4p-3 is4p-4) t is4p-5)
 (expected (obj-consultants (is4p-4 t)) t is4p-6)
 (inform (obj-consultant obj-client) false is4p-7)
 (prepared-for (obj-client (is4p-4 t)) false is4p-8)
 (promising (obj-results) unknown is4p-9)
 (believe (obj-client (is4p-9 false)) t is4p-10)
 (halt (obj-client obj-project) t is4p-11)
 (cause (is4p-8 is4p-11) t is4p-12)
 (cause (is4p-10 is4p-11) t is4p-13)
 (conjoin-event (is4p-12 is4p-13) t is4p-14)
 (terminated (obj-project) unknown is4p-15)
 (if ((is4p-9 false) (is4p-15 t)) t is4p-16)
 (jeopardized (obj-project) t is4p-17)
 (cause (is4p-14 is4p-17) t is4p-18)
)
)
'(constraints
((limited (obj-project-budget) t is4c-1)
 (estimate (obj-cost-estimate) t is4c-2)
 (cause (is4c-2 is4c-1) t is4c-3)
 (pilot-project (obj-project) t is4c-4)
 (risky (obj-project) t is4c-5)
 (increase (obj-client obj-budget) unknown is4c-6)
 (required ((is4p-7 t)) t is4c-7)
 (convinced-of (obj-client obj-project-success) unknown is4c-8)
 (if ((is4c-8 t) (is4c-6 t)) t is4c-9)
 (if ((is4c-6 t) (is4p-15 f)) t is4c-10)
)
CASE-5 describes a funding problem that in contrast to CASE-4 is caused by a shift in the client's objective and strategy. The project originally required extensive funding, much of it allocated to research efforts for system development. When the company placed priority on other endeavors, sufficient funding, particularly for the research effort, was withdrawn which caused the project to stall. In order to solve the problem, the primary concern was to convince the client of the necessity for research. In addition, the cooperation of another client in which to share research efforts was also pursued.

(make_case 'IS-project-5
  '(IS-project-mgmt)
  '(facts
    ((office (Boston) t is5f-1)
     (project (obj-logistic-routing-system) t is5f-2)
     (client (Amer-Barrytech-Inc) t is5f-3)
     (client-location (New-Hampshire) t is5f-4)
     (client-industry (obj-automobile-supplier) t is5f-5)
     (client-position (obj-senior-official) t is5f-6)
     (contract (obj-prime-contractor) t is5f-7)
     (contract-status (obj-new) t is5f-8)
     (project-status (obj-closed) t is5f-9)
     (office-personnel (obj-manager) t is5f-10)
     (office-personnel (obj-consultant) t is5f-11)
     (client-personnel (obj-users) t is5f-12)
     (equipment-type (obj-AI-dbms) t is5f-13)
     (equipment-vendor (obj-multiple) t is5f-14))
'(problem
  ((project-objective (obj-project-objectives) t is5p-1)
   (project-terms (obj-project-terms) t is5p-2)
   (determine (obj-client obj-project-objectives) t is5p-3)
   (accept (obj-client obj-project-terms) t is5p-4)
   (include (obj-project-objectives obj-research-effort) t is5p-5)
   (is-long (conc-project-time) t is5p-6)
   (cause (is5p-5 is5p-6) t is5p-7)
   (change (obj-client obj-strategy) t is5p-8)
   (shift (obj-financial-resources) t is5p-9)
   (erased (obj-project-funding) t is5p-10)
   (cause (is5p-8 is5p-9) t is5p-11)
   (cause (is5p-9 is5p-10) t is5p-12)
   (halted (obj-project) t is5p-13)
   (expected (obj-consultant is5p-13) false is5p-14))
)
)
'(constraints
  ((new-technology (obj-new-technology) t is5c-1)
   (involve (obj-project obj-new-technology) t is5c-2)
   (complex (obj-project) t is5c-3)
   (cause (is5c-2 is5c-3) t is5c-4)
   (limited (obj-project-funding) t is5c-5)
   (expensive (obj-research-effort) t is5c-6)
   (dependent (obj-project obj-research-effort) unknown is5c-7)
   (believe (obj-manager (is5c-7 t)) t is5c-8))
)
)
'(goal
  ((convince (obj-client dependent) t is5g-1)
   (desire (obj-manager (is5p-13 false)) t is5g-2)
   (extend (obj-project-funding) t is5g-3)
   (share (obj-research-effort) t is5g-4)
  )
)
)
'(solution
  ((contact (obj-manager obj-other-client) t is5s-1)
    (cooperation (obj-cooperation) t is5s-2)
    (cooperate (obj-client obj-other-client) t is5s-3)
    (support (obj-cooperation obj-project) t is5s-4)
  )
)
CASE-6 is similar to CASE-1 in that it reflects the age/competence conflict but in a
different setting. An assisting accountant is responsible for some of the company's
controlling functions and reports directly to the corporate controller. With new
equipment and software installed, a younger accountant with state-of-the-art skills in the
field of computer analysis and cost controlling is assigned to the accountant manager.
The older and more experienced accountant feels threatened by the "rookie" who is
knowledgeable but inexperienced. Again, in this situation the managing controller plays
a bridging role in fostering cooperation between these two employees.

(make_case 'accounting-case-1
   '(financial-accounting IS-project-mgmt)
   '(facts
       (office (Boston) t acc1f-1)
       (project (obj-controlling) t acc1f-2)
       (company (Valtech-Inc) t acc1f-3)
       (company-location (Chicago) t acc1f-4)
       (company-industry (obj-electronic-supplier) t acc1f-5)
       (office-personnel (obj-controller-manager) t acc1f-6)
       (office-personnel (Meyer) t acc1f-7)
       (office-personnel (Bender) t acc1f-8)
       (equipment-type (obj-PC-mainframe) t acc1f-9)
       (equipment-vendor (IBM) t acc1f-10)
     )
   )
   '(problem
       (skilled-with (Bender obj-controlling-IS) t acc1p-1)
       (system (obj-controlling-IS) t acc1p-2)
       (advanced (obj-controlling-IS) t acc1p-3)
       (manage (Bender obj-IS-controlling-support) t acc1p-4)
       (manage (Meyer obj-controlling-support) t acc1p-5)
       (older (Meyer Bender) t acc1p-6)
       (acknowledge (Meyer acc1p-1) false acc1p-7)
       (cause (acc1p-6 acc1p-7) t acc1p-8)
     )
   )
(support (Meyer Bender) false acc1p-9)
(cause (acc1p-8 acc1p-9) t acc1p-10)
)
)
'(constraints
 ((required (obj-IS-controlling-support) t acc1c-1)
 (require (obj-IS-controlling-support obj-controlling-IS) t acc1c-2)
 (skilled-with (Meyer obj-controlling-IS) false acc1c-3)
 (dependent-on (Bender Meyer) t acc1c-4)
 (like (Bender conc-work) false acc1c-5)
 (motivated (Bender) false acc1c-6)
 (cause (acc1p-9 acc1c-6) t acc1c-7)
)
)
'(solution
 ((controller-manager (obj-controller-manager) t acc1s-1)
 (authority (obj-controller-manager) t acc1s-9)
 (support (obj-controller-manager Bender) t acc1s-2)
 (acknowledge (Meyer acc1s-9) t acc1s-3)
 (cooperate (Meyer obj-controller-manager) t acc1s-10)
 (demonstrate-trust (obj-controller-manager Bender) t acc1s-4)
 (cause (acc1s-2 acc1s-4) t acc1s-5)
 (cause (acc1s-4 (acc1p-7 t)) unknown acc1s-6)
 (support-until (obj-controller-manager (acc1s-6 t)) t acc1s-7)
 (cause (acc1s-5 (acc1p-9 t)) t acc1s-8)
)
)
)
APPENDIX E: Complete Mapping Result of D-CARE Consultation

*** FINAL RETRIEVAL STATISTICS for IS-PROJECT-1 ***************

Results of probing with IS-CASE-3-14\92 after 52 cycles:
Structures retrieved:

IS-CASE-3-14\92 = ACCOUNTING-CASE-16\91 with activation: 0.5344458019154804
IS-CASE-3-14\92 = IS-CASE-8-23\90 with activation: 0.2334928128796523
IS-CASE-3-14\92 = IS-CASE-14-34\91 with activation: 0.09783209379965338

Factors involved were:

Best mapping of ACCEPT is ACKNOWLEDGE. 0.27581652553227587
Best mapping of COMPETENCE is AUTHORITY. 0.24222262823088525
Best mapping of COMPLEX is COMPLEX. 0.18499635950760524
Best mapping of COOPERATE is COOPERATE. 0.4111103914852594
Best mapping of HILL is BENDER. 0.10408444217730754
Best mapping of IS1C-10 is ACC1C-1. 0.045590826941671214
Best mapping of IS1C-2 is ACC1C-2. 0.11079694011118437
Best mapping of IS1C-5 is ACC1C-1. 0.04563123404182394
Best mapping of IS1C-6 is ACC1C-1. 0.04563123404182394
Best mapping of IS1C-7 is ACC1C-6. 0.11594994542210924
Best mapping of IS1P-10 is ACC1P-6. 0.08446219497033117
Best mapping of IS1P-13 is ACC1S-3. 0.037909022450032824
Best mapping of IS1P-14 is ACC1S-3. 0.037909022450032824
Best mapping of IS1P-17 is ACC1S-10. 0.032122681271910425
Best mapping of IS1P-18 is ACC1S-10. 0.032122681271910425
Best mapping of IS1P-2 is ACC1S-9. 0.07744779367080916
Best mapping of IS1P-4 is IS5C-1. 0.057616538613448044
Best mapping of IS1P-5 is ACC1P-2. 0.05211881624395256
Best mapping of IS1P-7 is ACC1P-6. -0.009770992933504957
Best mapping of IS1P-8 is ACC1P-6. -0.009770992933504957
Best mapping of IS1P-9 is ACC1P-6. 0.08446219497033117
Best mapping of IS1S-1 is ACC1S-1. 0.0938423567151717
Best mapping of IS1S-2 is ACC1S-9. 0.0917635796168896
Best mapping of IS1S-3 is ACC1S-3. 0.04018679390553376
Best mapping of IS1S-4 is ACC1S-3. 0.04018679390553376
Best mapping of IS1S-5 is ACC1S-10. 0.05650745078476609
Best mapping of IS1S-6 is ACC1S-10. 0.05650745078476609
Best mapping of IS1S-7 is ACC1S-2. 0.05385912292464157
Best mapping of IS1S-8 is ACC1S-2. 0.05385912292464157
Best mapping of KOCH is MEYER. 0.0616218972536266
Best mapping of MANAGER is CONTROLLER-MANAGER. 0.11574863771309316
Best mapping of MOTIVATED is MOTIVATED. 0.2451396880785248
Best mapping of NEW-TECHNOLOGY is NEW-TECHNOLOGY. 0.2040824838954436
Best mapping of OBJ-DATABASE is OBJ-NEW-TECHNOLOGY. 0.0098096084989735
Best mapping of OBJ-DATABASE-IMPLEMENTATION is
OBJ-IS-CONTROLLING-SUPPORT. 0.01592231225726276
Best mapping of OBJ-EXPERIENCE is OBJ-CONTROLLING-IS. 0.01673522578036004
Best mapping of OBJ-SUCCESS is NIL. 0.03532897469408564
Best mapping of OLDER is OLDER. 0.29665951956912573
Best mapping of PETERSEN is OBJ-CONTROLLER-MANAGER. 0.0939475592461132
Best mapping of REQUIRE is REQUIRE. 0.24504107109644635
Best mapping of REQUIRED-FOR is REQUIRED. 0.3459583094705036
Best mapping of SMITH is MEYER. 0.0616218972536266
Best mapping of SUPPORT-UNTIL is SUPPORT. 0.23367839470179933
Best mapping of YOUNGER is OLDER. -0.08620653793080892

Network Processing Time:

Init Time: 2.51 seconds.
Probe Time: 90.590002 seconds.
Run Time: 116.009998 seconds.
Total Time: 206.6 seconds.