

**STELLINGEN  
(PROPOSITIONS)**

**behorende bij het proefschrift**

**A PATTERN DIRECTED APPROACH TOWARDS  
AN OBJECT ADAPTIVE DECISION SUPPORT ENVIRONMENT FOR  
WATER RESOURCES MANAGEMENT**

**van**

**Ying LIU**

**19 February 1996**

1. A lack of software effectiveness in decision support for water resources management could have two causes: requirements inappropriately collected, or non-adaptiveness to changes in the requirements.

This dissertation

2. A decision maker's decision cannot be described in terms of properties of all his/her brain cells, the function of a church cannot be found through studies of the bricks, the columns, etc.

This dissertation

3. "If we improve our understanding of ordinary talk of physical things, it will not be by reducing that talk to a more familiar idiom; there is none. It will be by clarifying the connections, causal or otherwise, between ordinary talk of physical things and various further matters which in turn we grasp with help of ordinary talk of physical things".

*Word and Object*, Quine, W. v. O. The Technology Press of The MIT, 1960, pp. 3

4. GIS applications require object-oriented methods, which belong to the field of software engineering, evolving towards managing the diverse knowledge domains, while still remaining advantageous to object-oriented software constructions.

This dissertation

5. In art, there is a fondness for contradiction.
6. In public speaking, saying nothing should be just as acceptable as saying much.
7. Patience is one of the most essential requirements for understanding something.
8. Giving a child his leisure means allowing him to make a mess of everything.
9. Saying no to everything does not gain much self-confidence.
10. You will never do anything if you wait for clear water in the Yellow River.

645393  
3190347  
TR diss 2718

**A pattern directed approach towards**

**An Object Adaptive Decision Support Environment for  
Water Resources Management**

Cover design:  
Aone. Tj. Postma

Published and distributed by:  
Delft University Press  
Stevinweg 1  
2628 CN Delft  
The Netherlands

CIP-DATA KONINKLIJK BIBLIOTHEEK, DEN HAAG

LIU, Ying

A Pattern Directed Approach towards An Object Adaptive Decision Support Environment for  
Water Resources Management

Ying LIU. - Delft: Delft University Press.  
Dissertation Delft University of Technology  
With ref. - With appendix - With summary in Dutch  
ISBN 90-407-1311-1

Subject heading: decision support system / geographical information system / object oriented design /  
pattern-directed analysis / knowledge engineering /water management

Copyright © 1995 by Ying LIU

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or  
transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise,  
without the prior permission of the author.

Printed in The Netherlands

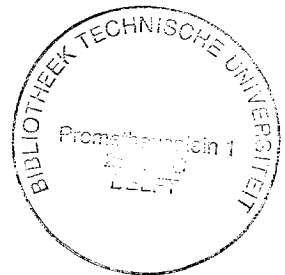
**A Pattern Directed Approach Towards  
An Object Adaptive Decision Support Environment for  
Water Resources Management**

**Proefschrift**

Ter verkrijging van de graad van doctor aan de  
Technische Universiteit van Delft  
op gezag van de Rector Magnificus,  
Prof. ir. K. F. Wakker,  
in het openbaar te verdedigen  
ten overstaan van een commissie  
aangewezen door het College van Dekanen  
op maandag 19 februari 1996 te 10:30 uur

door

Ying LIU,  
geboren te Beijing, China





Dit proefschrift is goedgekeurd door de promotoren:

Prof. dr. H. Koppelaar, en  
Prof. dr. ir. P. van der Veer

De leden van de promotiecommissie:

Rector Magnificus, voorzitter

Prof. dr. M. B. Abbott

Prof. dr. H. J. van den Herik

Prof. dr. H. Koppelaar

Prof. dr. ir. W. W. H. Thissen

Prof. dr. ir. H. J. Sips

Prof. dr. ir. P. van der Veer

Dr. ir. R. A. Vingerhoeds

IHE Delft (Int. Inst. for Hydrau. Env. and Eng.)

Rijksuniversiteit Limburg

Technische Universiteit Delft

Technische Universiteit Delft

Universiteit van Amsterdam

Technische Universiteit Delft

Technische Universiteit Delft

*To those willing to try new approaches of computer applications  
to cope with problems of traditional engineering*

## Preface

Since 1992, I have been, as a computer applicationer, studying 'computer-based decision support for water resources management'. I soon found that the complexity involved in this subject is, to a great extent, attributable to the problems of water resources management themselves which are here regarded as - 'external environments' of the computing system that one often claims to be DSS (decision support system). This argument leads me to search a broad interdisciplinary approach, because the systems studied in water resources management are in many respects different from business management which DSS was originally designed for.

Foremost I thank my promotor Prof. Dr. Henk Koppelaar who introduced me to The Netherlands; With the support given by Dr. Ir. R. Sommerhalder, and Mrs. Trudie Stoute in the group of theoretical computing at TU Delft, I was able to work for the DIAC project.

Since then, I have had numerous discussions with professor Koppelaar on a large variety of topics. He introduced me to Prof. Dr. Ir. Peter van der Veer, and continues to guide my PhD studies. He supervised me, in a very creative, constructive and therefore never-boring manner, in writing this dissertation.

It was Prof. Dr. Ir. Peter van der Veer who offered me the position of research assistant at the civil engineering informatica centre (CTI) which he newly found at TU Delft. He was also the first to lead me into water related research. Subsequently, he became my second promotor. He patiently shared with me something of his knowledge about water management, and above all, encouraged me in tackling various difficult issues.

I thank Ir. Ijsbrand Haagsma and Ir. Remco Johans. We worked together for the same project called IDEE. Our co-operation has been most challenging, because the disciplines enabling us to co-cooperate are something we are all searching as our research progresses. Their interests have stimulated my work.

I would like to express my great gratitude to Prof. Dr. Micheal B. Abbott at Delft International Institute of Hydrological and Environmental Engineering. Through his book - hydroinformatics, and his numerous other writings related to this topic, he awakened in me a long dormant fascination of the matters *existing* between computing and the aquatic environment.

I thank Dr. J. Gerard Wolff, at University of Wales, UK, for his many publications and discussions about bridging cognition with computing. His SP model (Simple and Power) has been largely adopted for designing effective conceptual schema in my work.

I thank Prof. A. Tzonis, at TU Delft, for introducing to me many researchers in the field of artificial intelligence applied for architecture design; and Prof. Teng-Rei Long, at Chongqing Jianzhu University, UK, for introducing his expertise to me in environmental engineering. From them, I learned to look at things from different perspectives.

I am pleased to acknowledge the support given by my fellow country men, some of long standing friendships, others of more recent acquaintance: Ms. Shi-Rong Li, Dr. Hao-Ming Zhou, Dr. Xiao-Dong Li, Dr. Hai-Kuan Li, Dr. Wen-Jun Zhang, Mr. Zhi-Gang Wei, Mr. Jun Lu, Mr. Xiao-Jiang Wang, and Mr. Xiao-Xiang Xu. They always put their work aside to talk with me whenever I needed them. They inspired me methodologically in many ways.

Mr. Brian McDowell has exerted a gigantic influence on my life, ever since we met some eight and a half years ago in Belfast. He used to listen to my very broken English and tried very hard to figure out what I said. He coaches me on how to speak, write and understand English. Besides his working knowledge of sociology, music, commerce, humour, and so on, which he makes readily accessible, he shows me consistently how to keep good spirits in the face of many difficulties. Above all, he, together with his brother Ronald McDowell, have spent many hours reading and correcting the manuscript of this dissertation, after which Brian and I had lengthy conversations on the telephone just to make some of my most difficult sentences readable. If there are still English errors in this dissertation, they must be newly made by me.

It is so fortunate that I met Mr. Aone Tj. Postma, an artist living in Delft. He has shown me a world which is totally different from the things I deal with in my office. Through discussions we often hold on subjects such as the relations between rationality and emotion, he makes me realise there is a fondness for paradox and contradiction which has something to do with the notion 'intelligence'. He brought me to Friesland from time to time to visit his family, where Mrs. Klaaske Postma-Demmer and Mr. Meint Postma always make me feel at home. Aone taught me how to ride horse and justify my knowing about horse through 'real-time control'. He has introduced so many friends to me and has helped me in overcoming the difficulty caused by cultural differences. In particular, I thank Ms. Mary Koppenol, Ms. Debby Nieuw Coop and Mr. Bert de Klerk with their musical band called 'cry out', and Mr. Orlando Barboza with his blues-music band called 'second hand man'. They cheer me up whenever I see them and whatever mood I am in.

I truly owe my family in China a great debt, specially my parents and my brother Fan-Han Liu, who originally made me viewing things objectively. They always believe me and I can always rely on them.

In a way, I have worked by my enthusiasm and reverence for certain ideas, and if readers feel these, this is the best I could ask for.

Again, I personally responsible for all the errors generated in this dissertation.

## TABLE OF CONTENTS

### Preface

INTRODUCTION	1
The research background	2
Conceptual issues to be addressed	7
Technical developments to be reported	10
Structure of the dissertation	11
Note	13

### Part I: on the externals of decision supports

1	A CRISIS OF SOFTWARE-EFFECTIVENESS IN DECISION-SUPPORT	
1.1	The diverse approaches to decision support	17
1.1.1	The artificial intelligent approach	19
1.1.2	The software environmental approach	19
1.1.3	The mixed approach	21
1.2	Concluding remarks on the approaches to decision support	22
1.3	An imposition of system engineering disciplines	22
1.3.1	System	23
1.3.2	Engineering	23
1.3.3	System' s observer	24
1.4	The fundamental problems	24
1.4.1	Complexity attributable to the externals	24
1.4.2	A lack of software effectiveness	25
1.5	Viewing water resources management as the application domain	26
1.5.1	The knowledge domains in terms of systems	26
1.5.2	A case of water policy analysis in the Netherlands and the geographical presentation	28
1.6	Remarks on the applications of decision-support technologies	32
1.7	Coping with the crisis	33

## 2 STUDIES OF DECISION MAKING PRIMITIVES

2.1	An instance of a need for data	35
2.2	Remarks on some concepts in treating decision problems	36
2.2.1	Under a model-driven approach	36
2.2.2	Under a data-driven approach	38
2.3	Introducing system engineering disciplines	40
2.3.1	The system of nature	42
2.3.2	The states of the system	42
2.3.3	The states of certainty	43
2.4	A decision making environment	44
2.4.1	Objective modeler	44
2.4.2	Observer	44
2.4.3	Decision maker	45
2.5	About P-D (physics-descriptive environment)	46
2.6	About O-O (objective-optimal environment)	46
2.6.1	Multi-objective and attribute	46
2.6.2	Proxy attribute	48
2.6.3	The properties of an attribute	48
2.6.4	The properties of a set of attributes	48
2.6.5	The decision rule	49
2.7	About P-R (politics-rational environment)	49
2.7.1	Quantitative or non-quantitative models?	50
2.7.2	Rationality	50
2.7.3	Types of conflicts	51
2.7.4	Modelling conflicts	51
2.8	Conclusions	52

## 3 CONNECTIONS TO COOPERATIVE AND DISTRIBUTED EXPERT SYSTEMS

3.1	Being cooperative and distributed	53
3.2	An architecture based on centralised control	54
3.3	Approaches to cooperation between expert systems	56
3.3.1	Approaches to cooperation in the front-end processors	57
3.3.2	Efficiency enhancer	59
3.3.3	Dynamic scheduler	59
3.4	Some critical bottlenecks in the applications of cooperative and distributed expert systems	60
3.5	Strong AI or weak AI?	61
3.6	The notion 'intelligence'	64

3.7	Intelligent function and intelligent service	65
3.8	Cases of the needs in altering knowledge-domains pragmatically	66
3.8.1	Constraints of spatial properties	66
3.8.2	Constraints of broad objectives	67
3.9	The notion 'object'	67
3.10	Pattern as a general representation of conceptual synthesis	69

## **Part II: object-oriented techniques vs. the externals**

### **4 A DOMAIN-DRIVEN OBJECT MODEL BASED ON GIS TECHNOLOGIES**

4.1	The adopted principles	75
4.2	Domain driven object-oriented modelling	77
4.3	Introducing object-dimensions and a causative dimension	79
4.4	Spatial dimension of data model and digitised maps	82
4.5	Data models	83
4.5.1	Spatial data model on the spatial-dimension	83
4.5.2	Topological operations on the attribute-dimension	84
4.5.3	Spatial-operations on the attribute-dimension	85
4.6	A simple object-calculus	85
4.6.1	Basic notations	85
4.6.2	Atoms	86
4.6.3	Well-formed formulas (wff)	87
4.7	Concluding remarks	87

### **5 DATA ENTITY AND A FUNCTION OF MULTIPLY-INDEX-IDENTIFIER**

5.1	The use of data entity	89
5.2	An example of allocating a data entity in a window	91
5.3	Identifiers of data-entities accessing databases	92
5.4	Multiply indexing an identifier of a data entity	93
5.5	A pattern-based identifier-access	94
5.5.1	Multiple subscripts	95
5.5.2	Keys and sequences of binary numbers	96
5.5.3	Subscripted attributes	99

6	KNOWLEDGE ACQUISITION FOR EMBEDDED MEANINGS OF OBJECTS	
6.1	A repertory grid-method	101
6.2	Embedded meanings and uncertainty	102
6.2.1	Attribute ordering table (AOT)	102
6.2.2	Eliciting embedded meanings	103
6.3	Knowledge validation in eliciting embedded meanings	104
6.4	A consistency test for knowledge validation	107
7	KNOWLEDGE REPRESENTATIONS BASED ON EFFECTIVE CONCEPTUAL SCHEMAS	
7.1	A 'trade-off' between domain- and implementation-driven modelling	111
7.1.1	Effectiveness supported by efficiency plus redundancy	112
7.1.2	Software as a compressed representation of inputs and outputs	112
7.1.3	Schema plus correction	114
7.2	SP language	114
7.3	Connections to object-oriented software constructions	116
7.3.1	In class-inclusion relation, part-whole relation and inheritance of attributes	116
7.3.2	Class, meta-class, instance and evolution of classes	117
7.3.3	In logic and logical inference with uncertainty	118
7.4	Concluding remarks	118

### **Part III: the applications in water resources management**

8	ORIENTORS vs. INDICATORS	
8.1	Orientors or indicators?	123
8.2	Water-use and water-demanding	126
8.2.1	Schemas	126
8.2.2	Other constructions for policy-assessment	127
8.2.3	The concept of the value in the non-consumptive uses	128
8.3	The policies and spatial attributes	129
8.3.1	Representing some spatial properties	130
8.3.2	Remarks on the objects in the schemas for representing spatial properties	133



9	IMPLEMENTATIONS	
9.1	AML, a menu and conceptual schema	135
9.2	Dynamic data-paths	138
9.3	Naming variables for a pattern-based path	139
9.4	Other examples for organising AML programs	140
9.4.1	Main program	141
9.4.2	A program of thread management	142
9.4.3	A program for managing drawings	143
9.4.4	A sub-routine of analysis	144
9.4.5	Zoom menu	145
9.4.6	Quit	146
	SUMMARY	147
	EVALUATION	
	Conclusions	149
	Limitations and recommendations to the future work	149
	APPENDIX	
1.1	Visualised Scenario - alternative 2a/b	153
1.2	Visualised Scenario - alternative 3a/b	155
1.3	Visualised Scenario - alternative 4/5	157
2	Tables of Survey	159
3.1	Selected Concepts of Decision Problems	161
3.2	Model and Modelling - a Philosophical Perspective	165
4	Explanations of Knowledge Domains in Terms of Systems	169
5	Data-entities in Pragmatic Terms	171
6	An Example of Parsing a SP Program	175
	BIBLIOGRAPHY	177
	EXPLANATION OF CONVENTIONS	197
	SAMENVATTING (summary in Dutch)	199
	CURRICULUM VITAE	201



## INTRODUCTION

This dissertation presents an approach to expanding applicability of a so called *decision support environment* in water resources management.

Functionally, the term decision support environment (DSE) can be referred to as a set of computer software tools, enabling users to share the knowledge, information and data<sup>+</sup> in order to understand a decision-problem better, and thus to improve their decision making.

Elements of DSE concept are coined in literature as distributed environments [Dewire94], distributed decision-support systems [Schlickmann92], heterogeneous co-operative distributed expert systems [Bell90], collaborative environment or even hypermedia [Bieber92], to mention but a few. In water resources management (WRM), it is not uncommon to address the applications of these advances, see, e.g., [Loucks91].

However, in respect to the fact that

- a) WRM is primarily related to *spatial* and dynamic processes,
- b) WRM has therefore invoked decisions being made quite differently from the industrial-development decisions [Manne67], and yet,
- c) tools are, in essence, computer programs characterised by interactive menu-driven techniques,

we then have to be concerned with the issues underlined by two inter-related questions which seem to be less often discussed in depth, but are of importance. The first question is:

- In which sense do tools support the decision-making in WRM?

The use of tools to execute models was the first step in controlling programs interactively between the end-user and computer, but not the last. The creation of tools was not directly affected by the *introduction* of computer. In the beginning, a computer program was fully written by programmers, and running rather blindly to users. Later, computer science has developed various methods, such as new language paradigms and new tools, to make programming activities more effective to meet users' requirements. To this extent, one of the most fundamental computing-problems has been, and will continue to be, shifted to how to achieve control over the creation of linguistic models of knowledge, or even existing programs. This shift has been regarded as a turning point in creating many opportunities to widen computer applications in general [Adler92], as well as interactive modelling in WRM in particular, e.g., [Loucks85] [Garrote95], and yet, to what extent the methodologies emerging thereby associate with the application for *decision making* in WRM is still a problem worth investigating. Therefore we come to the second question, i.e.,

<sup>+</sup> Here, *knowledge* simply refers to a set of instructional representations of knowing in a generalised manner; while *information* is a view of something given by a sender, and got by a receiver - a human being or a machine, and it appears always in a form of *data* as codes.

- How to encapsulate existing and newly appearing tools into the relevant application-domains?

For instance, as we have mentioned, WRM is primarily related to *spatial* and dynamic processes. Therefore the fast progress in computer technology has promoted the development of *GIS*. GIS is an acronym for *geographical information systems* - the tool driven technology for managing, manipulating and analysing geographical spatial data. Researched for decades, practical use of GIS has been made possible just in the last few years by astounding advances in computers. GIS is much more advanced than computer-aided design (CAD) or any other spatial data system - its applications are expected to appear wherever geographically distributed information is used and that is almost everywhere in business and government, e.g., [Kovar93] [Bieber95].

However, by studying GIS in a computing perspective, we find that GIS is *spatial data based* [Koppelaar94] with the facilities of object-related map manipulation routines supported by high resolution visualisation techniques. These emerging facilities, in turn, require object-oriented methods, which belong to the fields of software engineering, evolving towards managing the diverse knowledge domains, while still remaining advantageous to object-oriented software constructions [Meyer88] for achieving tool-encapsulations for example. In short, these matters put the current object-oriented methods into the frontier with which we shall deal, and allow us to develop our approach towards a better management of the various object-orientations to improve DSE.

In order to state our research issues more specifically, let us first of all get an impression of how we take WRM as our application domain and why we consider GIS as an instance of encapsulated tools.

## The research background

Man's request for a better use of the available water is as old as mankind itself [Chow64]. The problems of WRM are derived from issues such as: how to simulate full water employment, to achieve part of certain intangible and/or non-quantifiable objectives, the promotion and support of economic growth, etc. etc.

The development of modern thought contributed to WRM can be found in any up to date and relevant elementary books. It relates to the hydrological cycle increased due to the process of hydrology and hydrologic (as science and engineering), and is associated with almost every area of scientific endeavour (such as the social sciences). The need has even been felt by the descriptive sciences with elements of decision theory.

The most advanced association between WRM and computer science is *hydroinformatics* [Aboot91]. Hydroinformatics stands, because a computer-based system has to manage the knowledge of the water as standing reserves on one hand, and on the other, to impart the knowledge that the information conveys to us [Abbott94a] [IAHR94]. This draws our attention more than ever to the place where the reliability and calibration of models are validated by software-modelling as well as prospects in the context of information and telecommunication technology [Cunge95].

As Abbott remarkably points out,

"the numerical model now becomes the domain knowledge encapsulator, for it is this which now encapsulates all that is known about the physical system that can be taken as given in any particular situation. This encapsulator must then itself be capable of using all the data that are available and of processing them in such a manner and with such rapidity that they can be efficiently assimilated by the human decision maker" [Abbott93].

Suppose what was originally 'the numerical model' now appears as the nexus of a hydroinformatics system as Abbott would say. Then, what are the problems which DSE is facing? Are they just some matters of system-modelling?

### **Decision problems in water resources management**

The decision making in WRM is neither well structured, nor unambiguous having multiple objectives and socio-political aspects. Hence, there is no unique, objective way to find the 'best' solution [Fedra90].

Whatever the interpretation given to prime objectives, decision making in WRM is always involved with solving problems in conjunction with each other. The treatments of these problems have to embrace a broad spectrum of issues associated with many analysis factors, such as water demands, water supply, available technologies, and data gathering facilities [Loucks89].

If we shift our concerns onto another side of WRM, which is, managing human activities, what further arises is that, any interference with water resources can cause irrecoverable damage. And yet, maintaining water resources essentially entails many isolated events which are not so easily grouped, unless we *constantly* associate each with an object, and inject the overall goal to control them together.

The collaborative relations have been, for many years, actively stimulated between many groups, such as the managers, system analysts, policy-makers, decision-making bodies and even farmers. Later, we shall generalise these factors as *external environments* of DSE. But for now, let us consider GIS a little bit further.

### **GIS - a tool driven technology**

GIS is a set of computer programs for storing, retrieving, analysing, and displaying cartographic data.

In a GIS, the earth's features are not only represented in pictorial form, as in conventional paper maps, but as information or data. This data contains the spatial information of conventional maps, but when stored in a computer, is much more flexible in the way it can be represented. Spatial data in a GIS can be displayed just like a paper map with roads, rivers, vegetation and other features represented as lines on a map complete with legend, border and titles [Burrough86] [Aronoff89] [Star90] (also, see the following figure).









There are tools, such as ARC/INFO [ARC92], available for digitising maps and thus storing maps in a computer; they make it possible for us to facilitate the maps through geometric intersection in vectors by the arithmetic and Boolean operations on the maps; they even enable us, in a greater extent, to move information around in a digital form [Abbott93]. By all together, we seem to have won the battle of computerised 'decision support'. But on the contrary, the informational connectivity in any GIS does not necessarily mean the informational integration. These two things are often mistaken for one another.

At the first place, it is known that GIS conveys the nature of all tools - they are computer programs characterised by interactive menu-driven techniques. In turn, it is these techniques, as well as the approaches to develop them, that conclude the *use* of graphics through data input, editing, verification, and processes of controlling the sequence and parameters of a model and program operation; referring a spatial database by a structured format; representing a set of statistical tables which can be converted to charts and graphs, and *then*, providing meaningful displays of input and output data in both graphical and pictorial manner. After all we may realise how easy it is for the user to store, retrieve and analyse the information.

At the second place, map generation is a process of transforming the real world into a map that is considered as the cartographical conceptualisation and visualisation of reality. It is also known as cartographical abstraction. *It is the physical reality of our world compressed in a symbolic way.* In an artistic sense, the abstraction of reality is like a caricature in which certain features are emphasised and others are not in an attempt to present some particular aspect of geographical environment [Muehrcke78] [Visualingam89].

At the third place, it is noticeable that, in the above informational and knowledge perspective, it is self-evident that software, hardware, communication, operating system and data standards all contribute directly to enhancing the compatibility and interoperability of computer-based information handling systems. However, the reported experiences, such as [Zuallkernan86] [Vadera91] [Clark93], tell us that these fundamental components of 'open systems' computing are so well covered in the It literature, and are so generic in nature, that they almost do not warrant extensive treatment in the context of water resources GIS. On the other hand, consideration of an appropriate data model for a GIS ranges across a spectrum of relevant issues from the mundane to the philosophical. It is felt that such matters are bound to generate controversy as new approaches struggle for recognition.

## Conceptual issues to be addressed

We may have felt by now that tool driven technologies have indeed created many opportunities to widen computer applications, but in order to support decision-making in WRM, we need to have a more powerful approach to encapsulating tools into the relevant application-domain.

'Object' is one of the most important concepts dealing with just about any kind of tool-encapsulation, because it is at the heart of both software-application-analysis and software programming<sup>+</sup> [Stroustrup88].

<sup>+</sup> For simplifying our discussions, we shall not use the term 'software-life-cycle', which is generally meant by a repeated validation between software-requirement analysis, -specification, -design and -implementation.

For example, in a GIS development, arguments on object-oriented methods often take place from both users' and software constructional point of view. Many researchers have argued cogently that an object-oriented method is conceptually and technically complex and (more significantly in the long-term) that it constrains the user's flexibility in defining or dynamically creating feature relationships, see [Dangermond89] for example. In complete contradistinction, many others, such as Piazza, Pessaro and Hargis, identify that an object-oriented structure is a powerful vehicle by which we are able to bring a higher level of cognitive reasoning to GIS-based analysis, something which they regard as a necessary next step in GIS development [Piazza90] [Hargis92].

### **What is the object and where is it?**

Whether or not the above viewpoints are acceptable in any particular context, it seems that the literature of object-orientation clearly indicates that the following two conditions should be satisfied simultaneously:

- what the object-oriented method is taken for structuring data conditions, what a GIS *can do*, and how effectively it *can do so*,
- what the object-oriented method is chosen for fulfilling the users' demand conditions, what GIS *is really employed for*, and how effectively it *is performing so*.

To better understand these two conditions as well as their importance, we make distinction between so called implementation-driven objects and domain-driven objects.

#### *An implementation-driven object.*

In the case that the concept of an object emerges as a programming construct, the object is an outgrowth of abstract data types or an encapsulation of data and processes. The essence of this is that a program is viewed as a composition of several independent objects that communicate via messages.

In short, if objects emerge as programming concepts, the object-oriented methods associated with the objects are driven by the considerations of software implementations similar to structured programming [Jackson75] and object-oriented design (programming) [Birtwistle73] [Goldberg83].

#### *A domain-driven object.*

An object can be regarded as a construct for modelling a problem-domain. This approach is in line with recent views of software engineering [Lee91] [Chu92] [Soukup94] [Gamma94], and it still seems to be continually evolve [Coplien95]:

away from a coding oriented view - how to implement a solution on the computer,  
toward a knowledge-oriented view - how to capture the problem domain.

In other words, it is proposed that the significance of an object extends beyond the improvement of data and control transfer in programs, and the portability (or reusability) of program components. Rather, objects gain their importance because they reflect a 'natural' view of the world we are modelling (abstracting) in our software, e.g., [Chudoba95] [Hupfer95] [Mackie95] [Salvaneschi95]. This is the case that, if we view

programs as models of some reality, the use of objects amounts to adapting programming (or database design) constructs that are direct mappings of 'real' concepts.

Considering the external environments of DSE to be introduced in the next section, we should recognise that we need to manage these domain-driven objects in respect to different perspectives of the applications. This allows us to develop the *pattern directed* approach which will be mentioned shortly after.

## **The external environments of DSE**

Systematically, we shall distinguish three kinds of external environments.

### *Objective-optimal environment.*

A decision process begins by someone who has a particular reason to observe a system - he is interested in perceiving the need to alter the course of a system with which *he* is concerned. This situation is then diagnosed [Easton73], he feels the need to state general statements of overall objectives. In this context, it is perhaps more appropriate to call him *objective-modeler*. The objective-modeler, together with the models and instruments concerned constitute an initial environment, i.e., objective-optimal environment.

### *Physics-descriptive environment.*

For a set of objectives to be well defined, meaningful comprehensive analysis of the pertinent aspects of that system has to be undertaken. Facilitating this analysis then becomes the primary task. However, the only hope to fulfil this task is to try to understand the problem better. The initial environment is enlarged; similarly, we think there is another category of environment becoming involved, i.e., physics-descriptive environment. In this context, we call the person *observer*.

### *Politics-rational environment.*

As noted by Rigby, the term 'decision-maker' is difficult to define precisely [Rigby64]. Churchman refers to the decision-maker as "the person who has the ability to change the system, i.e., has the responsibility and authority for such a change" [Churchman68]. More specifically, and more appropriate to the context with which we shall be concerned, we shall take the decision-maker to be a person or a group of persons, who furnish(es) the *final* value judgement that may be used to rank available alternatives for the 'best' choice to be identified. Then, there is a politics-rational environment, where the implicit in a decision has to be made, i.e., whenever final value judgements need to be generated, they are often concerning the 'goodness' or 'badness' of a given choice.

## **Pattern-directed object-orientation**

In the external environments, the concept of object is 'some thing' that is rather dynamic, i.e., for one situation, some attributes are aggregated as an object to be studied; for other situations, a different collection of physical properties have to be considered in one class.

Then, the notion 'pattern' means metaphors of decision-making primitives for the relevant object-orientations to be reflected and accommodated. This theme has a place in AI. For example, Horswill found that AI designers often improve the performance of

their AI systems by specialising a task and an environment [Horswill95]. Hence, the adaptability of DSE does not mean the ability of entirely 'knowing' an external environment, at least not in the way that the knowledge is traditionally understood under the 'strong AI' approach.

Generally, the AI approach underlines a class of software designed to deal with the same types of problems which 'expert systems' are encountering, because of the seemingly similar goals of decision support and expert systems software. Expert systems emerge from the fields in AI, where a general goal is to embody a computer with human like intelligence. Early AI approaches concentrated on the general process of reasoning. By the 1970's the limits to this approach led to the hypothesis that the knowledge a person has, rather than the reasoning he uses, is the key to mimicking human intelligence [Feigenbaum77] [Hayes-Roth83]. This hypothesis is the essence of the current expert system approach - the programmer concentrates on *representing* the knowledge an expert uses to answer a specific question, instead of trying to computerise general reasoning principles that can be used for any problem. The critical bottle-necks in developing expert systems include knowledge acquisition, knowledge representations and integration of conceptual schemas. These subjects will be treated technically, and we shall introduce them in the next section.

Also, approaching to an adaptation of an object-orientation, we need to integrate the domain- and implementation-driven objects. Thereby the notion 'pattern' means an ensemble of information primitives described by a set of conceptual schemas representing not only the objects, but also the intrinsic aggregations among the objects, such as salience, overlap, fuzziness, etc.

All together, such an organisation is characterised as *pattern-directed*.

## Technical developments to be reported

We shall develop the following techniques in order to entail the pattern-directed approach.

### *A Domain-driven data model.*

The model has three dimensions, namely, spatiality, attribute and eventuality (eventuality will not be discussed at length). Then we make an order of a sequence of the dimensions for representing a context where such a model is to be used.

### *Storing and retrieving data-entity.*

When the main purpose is to *split* a large number of elements into a number of 'homogeneous' groups on a basis of multivariate observations, a type of data entity is usually formed by a set of samples. Since the entity has to be dynamically stored in a database, i.e., retrieval of the entity is dependent on a context where the entity is to be used, we need a multiple-index function for a data entity to access (or to be accessed by) relational databases.

### *Knowledge acquisition.*

Knowledge acquisition is a task of systematically ferreting out the informal and semiformal knowledge held by the expert [Hart89]. The breadth and sheer volume of an expert's knowledge is what makes his analysis general and powerful; yet, obtaining that

knowledge, which he often does not realise he is using, is a painstaking and inexact process.

Since the knowledge has to be stored in a knowledge-based system and may involve a variety of physical applications and theoretical analysis, the experts' opinions often prove to be unhelpful at a computing level. A repertory grid method managing the objects is established. In a repertory grid, a set of elements across the top of the grid is listed to represent the objects to be classified. Various combinations of objects can then be presented to a network of computing. This method seems to significantly improve the time spent for eliciting expertise from domains' experts.

#### *Effective knowledge representation.*

Having acquired the knowledge in its 'human' form, we must create program and informational structures for various kinds of knowledge to 'work together'. We need to represent the structures conveniently and efficiently for machine processing. This representation is called knowledge representation. When the knowledge exists at many different levels of abstraction and aggregation, one has a major design problem. Here, knowledge representation is suggested to be designed by a set of *effective conceptual schemas*.

Conceptual schemas are written to describe many of the characteristics of objects [Nijssen89]. Though they have been extensively studied by Bruner, Goodnow, and Austin [Bruner56], it is only recently that we have computer software concepts relevant to the well-known notions, such as scheme [Bartlett32] [Bobrow75], frame [Minsky75], and script [Schank77].

An effective conceptual schema is a model that is not only capable of describing the knowledge structures, but also of containing programming details that are not part of the knowledge structures, but which are necessary to make the model run with the available technology.

#### *Implementation.*

We shall also consider some of the implementation issues. We shall design several programs with AML.

AML programs organise a sequence of ARC/INFO commands into geo-processing operations; therefore, they are at the heart of any kind of ARC/INFO system (GIS). The programs intend to construct a) pattern-directed data-paths, and b) dynamic menus, i.e., when a certain category of data is available, an item in a menu is capable of loading data. Such techniques are beneficial to the user. For example, GIS requires to import data-files directly, and the import facility is usually not provided. Hence the user is required to produce the necessary code within the development language system of the GIS.

## **Structure of the dissertation**

The dissertation consists of three parts:

*Part I - On the externals of decision support.*

*Part II - Object-oriented templates vs. the environments.*

*Part III - The applications in water resources management.*

## **On the externals of decision support**

This part has three chapters.

Chapter 1 addresses one of the most fundamental problems, i.e., a lack of *software effectiveness in decision support*, or simply, *the software-effectiveness*. This problem could have two causes: requirements inappropriately collected, or non-adaptiveness to changes in the requirements. It shows that the complexity of applying decision support technologies is attributable not only to a way of a computing system-design, but also to the application domains; It reveals why DSE is needed, and why the software effectiveness is important.

Very similar arguments will be held in chapter 3, where the topics of AI are discussed. In part II & III, we intend to enhance the software-effectiveness.

Chapter 2 argues that we may ease the concerns for the use of DSE in the new contexts, i.e., a) a decision making environment is an instance of virtual existence, and b) the communications in DSE are aiming to facilitate the interactions between decision making environments.

Chapter 3 mainly has two groups with sections, and there is a tenor between them.

In the first group, we shall have a very general frame of 'cooperative and distributed expert systems' (CDES) including a centralised control system (i.e., a blackboard system and front-end processors).

The critical bottlenecks in the applications CDES are supported by knowledge acquisition and effective knowledge representations. These two topics will be discussed in chapter 6 & 7 respectively, since we need to have appropriate connections to the fields belonging to AI.

In the second group, we shall elaborate on the meaning of 'intelligence', 'object' as well as 'pattern'.

## **Object-oriented techniques vs. the externals**

This part has four chapters.

Chapter 4 begins with the statements of the adopted principles in object-oriented paradigm, and the differences between our methods characterised as domain-driven object-modelling and the methods which have been already well established in software engineering. Then, it introduces an *object-model* based on GIS for decision support.

Chapter 5 reveals the contexts where these kinds of data entities are normally used; It formulates a function for solving the problems of data entity-identifiers.

Chapter 6 establishes a repertory grid method for knowledge acquisition.

Chapter 7 introduces a method for constructing the effective conceptual schemas in chapter 8. SP is adopted. SP stands for 'simple and powerful' [Wolff91] and represents a new concept of software supported by a runnable pattern-based computing language also called SP which was originally developed by Wolff.

## **Applications in water resources management**

This part has two chapters.

Chapter 8 constructs effective conceptual schemas to support nonroutine- vs. routine-processes. Routine-processes require comparatively regular categories of

information; While nonroutine-processes are oriented with respect to parameters which are shared by, or at least should be known to, several participants.

Chapter 9 discusses the applicable interfaces implemented with AML in ARC/INFO systems.

## **Note**

This work has been carried out jointly with two other ongoing-programs, namely, 'model integration', see [Haagsma94a] [Haagsma94b] [Haagsma94c], and 'human organisational management', see [Johanns94] [Johanns95].





## **Part I**

### **On the externals of decision supports**



## A CRISIS OF SOFTWARE-EFFECTIVENESS IN DECISION SUPPORT

---

In this chapter, our main purpose is to address one of the most fundamental problems, i.e., a lack of software effectiveness in decision support for WRM, or simply, the *software effectiveness*. This problem could have two causes: requirements inappropriately collected, or non-adaptiveness to changes in the requirements.

To get an impression of this kind of problem, we shall investigate the existing decision support approaches, and argue that the complexity of applying decision support technologies is attributable not only to a way of a computing system-design, but also to the relevant application domain, i.e., WRM.

A case study is carried out in the field of water policy analysis. Based on this study, we shall show a few examples of digitised policy-scenarios, and argue why DSE is still needed, why the software effectiveness is important, and finally, highlight what should be improved.

### 1.1 The diverse approaches to decision support

The original ideas of building a computer system to support decision making were introduced by Gorry and Scott-Morton under the banner - DSS (decision support system) [Gorry71]. They developed a framework for constructing conventional software programs for supporting executive decision making instead of computing solutions to everyday operational problems in business management. Since then, on one hand, the development of DSS is rapidly expanding, while, on the other hand, the image of DSS is becoming more and more heterogeneous, and thus, unclear.

For example, Parker and Al-Utabi had already drawn a list of characteristics of DSS, after reviewing about 350 papers related to the subject [Parker86] (lately, a similar report is found in [El-Najdawi93]). Let us now examine the list by the following categories:

- the descriptions from users' perspective,
- the descriptions from functional perspective,
- the descriptions from design perspective,

and furthermore, consider the inter-relations between them as Fig. 1.1 illustrates.

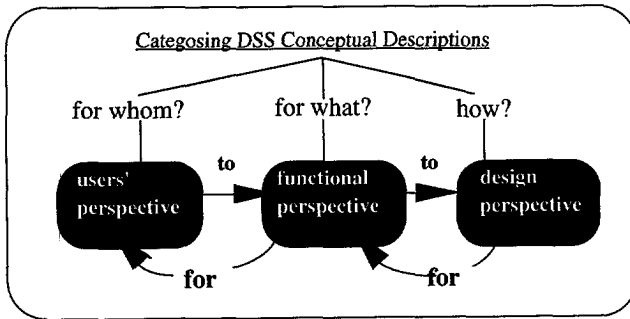


Fig. 1.1: an illustration of the relations between different categories of DSS conceptual descriptions

*The concept of DSS from users' perspective:*

- u1: assisting decision makers (managers) in their decision making processes for unstructured or semi-structured tasks;
- u2: supporting and enhance rather than replace managerial judgement.

*The concept of DSS from functional perspective:*

- f1: evaluating decision proposed by users;
- f2: generating decisions satisfying some user defined conditions;
- f3: improving the effectiveness of decision making rather than the efficiency, putting more weight on a correct decision than on the time it takes to make that decision;
- f4: enabling an intuitive approach towards a solution;
- f5: including trial-and-error procedures;
- f6: allowing the introduction of subjective judgements;
- f7: helping in tentative procedures, as they could be supported by fast prototyping environments;
- f8: performing data management functions;
- f9: combining the use of models or analytical techniques with data access functions;
- f10: emphasising flexibility and adaptability to respect changes in the context of the decision process;
- f11: focusing on features which make them easy to be used interactively by non-experienced users.

The design approaches may also be generally classified by three groups, they are *artificial intelligent*, *software environmental*, and the *mixed* approach (a mixture of both) which may respond to the two groups of functions we have just listed.

### 1.1.1 The artificial intelligent approach

The artificial intelligent (AI) approach underlines a class of software designed to deal with the same types of problems which *expert systems* are encountering, because of the *seemingly* similar goals of decision support and expert systems software (see the functional descriptions of f1, f2 and f3). Under this approach, one normally begins with an application of expert system technology evolved into the functions for which a type of DSS is proposed.

The general goal of an expert system is to embody a computer with human-like intelligence. Early approaches to artificial intelligence concentrated on the general process of reasoning. By the 1970s the limits to this approach led to the hypothesis that the knowledge a person has, rather than the patterns of reasoning he uses, is the key to mimicking human intelligence [Feigenbaum77]. This hypothesis is the essence of the current expert system approach - the programmer concentrates on *representing* the knowledge an expert uses to answer a specific question, instead of trying to computerise general reasoning principles that can be used for any problem.

Knowledge representations together with a way of organising them into a computer system become the most crucial impacts of building a DSS under the AI approach. There are many means existed for knowledge representations, such as rule-based, semantic nets, frame, etc. There are also many engineering branches existed for organising the representations, such as knowledge-based systems.

Fig. 1.2 shows a typical architecture of DSS under the AI approach.

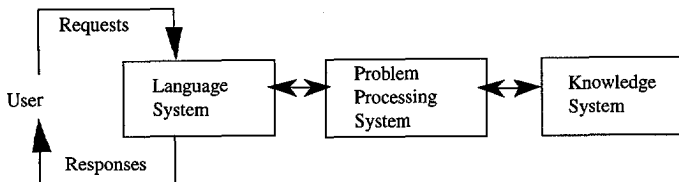


Fig. 1.2: DSS with respect to AI approach design [Bonczek81]

### 1.1.2 The software environmental approach

The software environmental (SE) approach underlines a class of software designed in regarding DSS as being more or less a collection of tools and data that are used to solve problems. The user can play with the tools and experiment to find a solution. The decision process is controlled by the user when he or she decides how to apply the different types of tools and data that are available. The DSS are not computerised experts; they are computerised assistants for users.

The software-tool driven technologies are supposed to lead DSS to fulfil the management of information in a type of decision making environment (see the descriptions of DSS functions - f8, f9, f10 and f11). The development of DSS is influenced by the

intersection of the two evolutionary trends from the field of data processing and the field of business modelling (management sciences) [Sprague87].

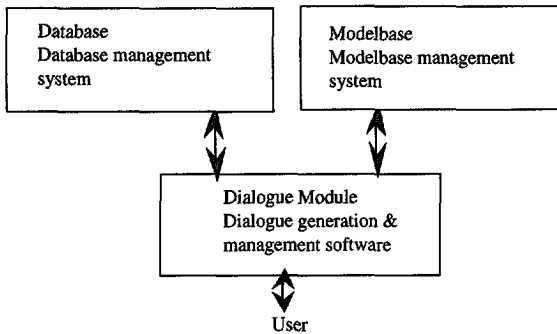


Fig. 1.3: DSS with respect to software environment approach

Therefore, it is commonly assumed that DSS lies in a common software environment via a common interface between databases and models. Hence, integration of data, models, and a sophisticated user interface is the most central characteristic of this type of DSS.

Fig. 1.3 shows a typical architecture of DSS under the SE approach.

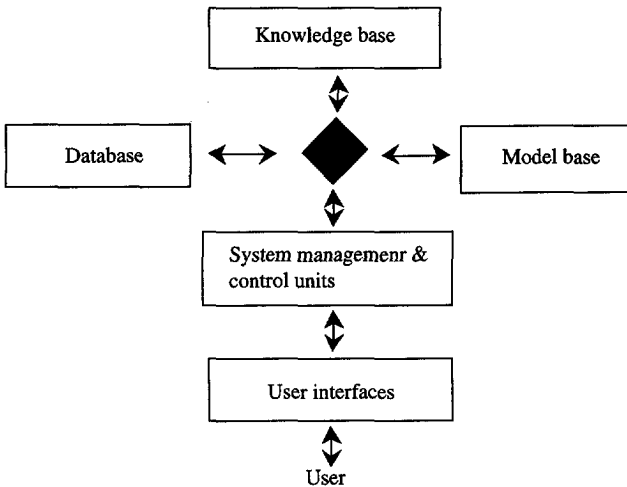


Fig. 1.4: DSS with respect to the mixed approaches

### 1.1.3 The mixed approach

The mixed approach underlines a class of software designed with respect to the definition of Spague and Carlson [Spague82] and with split components provided by Bonczek [Bonczek81]. These mixed approaches normally explicitly include a model-base, where requirements of a DSS contain procedural models, and thus, they are clearly distinct from DSS on the AI approach. Furthermore, the functions of the DSS designed on the mixed approach are also normally claimed to be more 'flexible' and 'integrated' comprising of two other kinds of approach (see the DSS functional descriptions of f4, f5, f6 and f7).

Fig. 1.4 shows a typical architecture of DSS under the mixed approach.

Table 1.1: a summary of the details of the differences between DSS on AI and SE approaches

	<b>Dimension</b>	<b>DSS Soft. Envir. Appr.</b>	<b>DSS Expert Syst. Appr.</b>
<b>User</b>	Mission	Support decision maker	Capture, transfer and make available expertise; making decisions
<b>Design perspective</b>	Means to achieve mission	Provide interactive access to data and models	Provide models of the expert's thought processes
	Focus	Decision focused with emphasis on flexibility and adaptability to user	Knowledge representation and reasoning with emphasis on justification of actions
	Application	Semistructured; broad problems	Problem domain dependent; Unstructured problems; Narrow domain problems
	Manipulation techniques	Numerical; quantitative; What-if-analysis	Symbolic; qualitative
	Knowledge	Complete; deterministic facts	Could be fuzzy; uncertain; incomplete facts
	Procedures	Mathematical, quantitative models and algorithms	Quantitative and qualitative inference procedures and heuristics
	Development life cycle	Adaptive design process	Problem definition; knowledge extraction/formalization; prototyping; fielding
<b>Functional perspective</b>	Behavioral concerns	User's cognitive style	Expert's reasoning process
	Development tools	Third- and fourth-generation languages	Fifth-generation languages and Expert syst. shells
	Learning capability	None	Potentially promising
	Feedback to user	Satisfactory or optimal solution	Alternative expert solutions; Reasoning path

## 1.2 Concluding remarks on the approaches to decision support

The approaches to build DSS can certainly be classified differently by many other reasons. Table 1.1 only summaries some of the details of the differences between DSS on AI and SE approach. A further comparison and evaluation of these might fill pages and pages, and we may still need to know how to follow these approaches to *conduct DSS applications*.

For instance, Sol stated 6 kinds of paradox existed in DSS developments, one of which is the paradox of input and output as follows:

"The model cycle behind a great many contributions in the DSS-field can be characterised as one, trying to integrate scientific, ethical and (a)esthetic modes of thought in a synthetic, interdisciplinary way. The availability of data and appropriateness of decision making in organisations is not much questioned. Or, to put it in another way, the modelling paradigm is mostly the one of thinking in terms of relationships between variables. Most models encountered in DSS are of the 'equation' types: a set of definition and behaviour; equations, possibly together with a goal function. Once the behavioural equations are estimated and empirically validated, one starts playing with the model, e.g., in an optimisation mode, or in a 'what-if' or 'goal-seeking' mode. However, these models are only applicable under the assumption that the equations can be filled in and that they give a valid description of reality, ..., A way-out can be found in the application of models of the 'process' type, ..." [Sol85].

After then, it seems that, in contrast to the 'equation-type' of DSS, the developments are overwhelmed by software tools. Again, Sol points out:

"Systems (DSS) which were said to enhance management effectiveness were automatically labelled by 'decision support system'. The market value of DSS label was instantly recognised by those in the field of operations research and the behavioural sciences. Phrases like 'information centre' and 'prototyping' were linked to DSS too readily. Consequently, distinguishing between useful concepts and short-lived buzzwords seems to have become a user responsibility" [Sol91].

## 1.3 An imposition of system engineering disciplines

At this stage, it is advisable to introduce some system engineering disciplines enabling us to identify the problems which should be treated first.

An employment of system methodology is not new. As Fig. 1.5 illustrates, the terminology *system* has been used extensively in various kinds of scientific fields: from the concern for classic engineering analysis; such as flow of matter and/or energy, to the concern for modern-day controlling analysis, such as information. Hence, for a research strategy, it forces one to look at a problem in its entirety. For software development in particular, it interrelates a large range of requirements which appear to be derived from different domains on one hand, but on the other, have to be constricted under different



disciplines for the same objectives, and be managed within one frame - a real or abstract simplification of the situation appropriate to the problems at hand.

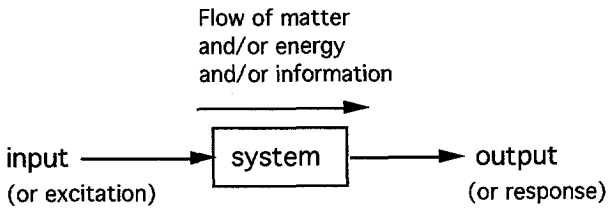


Fig. 1.5: the notion system

### 1.3.1 System

A system is a collection of components, also called parts, either physical or non-physical in nature, that a) exhibit a set of interrelations among themselves and interact together towards one or more goals; b) exhibit properties processed differently from the collection of properties processed by the individual parts.

A common objection to explore the concept of system is underlined by a phrase heard frequently, i.e., 'everything is a system even according to this definition, so what is its use then?'

Certainly, in any case, it is hard to imagine anything that could not be regarded as a system. But, this might be an important point: it is not the question whether something is a system or not. The question is, do we regard, or look at 'something' (X) as a system? If we do, the concept of system forces us to

- a) focus our interest on the properties that X (the 'something') has as a whole, and
- b) distinguish if another thing is the same as X.

### 1.3.2 Engineering

Let us now bound the considerations of systems in a sense of *engineering*. In other words, a system is now considered as an object of engineering. Thus, considerations of systems are encompassed by the most important objects or classes of objects that the engineering effort targets, including, for example,

- a) a product that is to be delivered to users after completion of the necessary engineering processes;
- b) the engineering processes that have to shape the product itself, such as, towards goals and establishments of requirements' elicitation.

Engineering is human effort to change a kind of environment in order to make this environment more suitable or responsive to perceived human needs and wants. The engineering effort can have various different kinds of physical output. For example, it may define, develop, or maintain a system. Many actors take part in an engineering effort. One group of actors are the engineers. Others are managers, still others may be people that create physical artefacts according to descriptions or models of these artefacts.

There are always discrepancies between engineering and methodical interests. In order to improve a system's functions as a *technical* realisation of *practical intent*, engineers view artefacts objectively (see, e.g., [Hofstede92]). Methodologists, with the same aim or intent, may be powerful enough to begin a method or technique, but find that, as the investigations proceed, the methodologies took on a life of their own [Shannon75].

### 1.3.3 System's observer

Based on a study of systems (their general properties and attributes), a *system engineering approach* derives a systematic way of tackling engineering problems.

"Such thinking (system thinking) starts with an observer/describer of the world outside ourselves who for some reason of his own wishes to describe it 'holistically', that is to say in terms of whole entities linked in hierarchies with other wholes" [Checkland81].

From this standpoint, it can be thought that a problem is always a problem of a person or a group of persons, and a problem has no existence independent of the people perceiving that specific problem.

## 1.4 The fundamental problems

Let us consider the following two impacts with the help of the notions we have introduced in the previous section:

a) The general properties:

DSS is a collection of software components which are man-made information substance. To this extent, DSS does not differ much from other computer based systems extensively studied in [Thome93]. But this standpoint does not seem to ease the next concern, i.e.,

b) The general identity:

essentially, what makes DSS different from other computer based systems, such as a database system or an information system?

### 1.4.1 Complexity attributable to the externals

Now we may come to the point, i.e., the existence of DSS has to be dependent on an observer or an observation pattern. Or, in other words, the identity of DSS is supposedly

dependent on the environment it is part of, or even vice versa, if the environment is a strongly constraining one.

A search is needed to pursue the following:

- a) to identify those stable features through which inputs of a DSS can be delivered to the targets, i.e., the outputs, and further,
- b) to develop inputs that target these features and yield an output that an individual prefers, i.e., in essence, destroying or disabling the object without affecting the whole DSS intolerably.

To this extent, the author agrees with other researchers, such as Tully, Kronlof, and Berry, that software engineering disciplines are matured enough to provide much support required to build a computer-based system, as, it is now possible to handle abstraction, modelling, formalism, as well as technologies to handle electronic information, etc. But, for coping with the complexity in which a computer system is involved, more adequate methods are still missing [Tully91] [Kronlof93] [Berry93].

In general, mathematics and physical sciences have come by a reduction of complexity; the selection of relevant elements and relationships is determined by the point of view of the analyst; and therefore, a different perspective means a different selection. But for a computer-based system to be built, ignoring the complexity can lead to a system that cannot meet the requirements it was designed for.

#### **1.4.2 A lack of software effectiveness**

The so called 'software crisis' is, in the author's opinion, an 'effectiveness crisis' rather than, or besides, an 'efficiency crisis' [Gane90].

Let us take an illustrative example. The U.S. Government Accounting Office reports the following breakdown of results [ACM85]: 47% of the software paid for was never delivered, 29% was delivered but never used; 19% was abandoned or re-worked; 3% was used after change; and only 2% of the spending was used in software as delivered. Thus, the effectiveness was 5% at best, 2% strictly speaking. Such a situation seems to be worsened by the rapidly growing informational technologies in the recent years [Brynjolfsson93]. It is a belief that this is not an illustrative example of an 'efficiency' problem to be tackled by reusability, as it was proposed by [Cox86] [Cox90], either by the tools such as CASE [Araujo92] [Vessey95]. It is a problem of a lack of effectiveness, which could have two causes: requirements wrongly collected, or non-adaptiveness to changes in the requirements [Lehman91] [Whytock93] [Liu94a] [Liu94c]. Although the efficiency could still possibly be a problem, the effectiveness is the real one. Methodological effectiveness is a necessary condition, programming and testing effectiveness is a desirable one.

Meta-software systems, such as CASE, code generations, etc. etc., speed software development up; they increase the development efficiency and decrease maintenance costs. Some packages are already compatible (e.g., from dBASE IV to LOTUS), and transferring information can be done successfully from one kind of presentation to another, e.g., in [Lam91], Lam has reported the experiments on integrating databases, spreadsheet, graphics, GIS, statistics, and simulation models.

The developments of tools must be a software business, but building a computer-based system is a human business, in particular, one for the use of decision-support. The latter development starts with thinking and acting in a pragmatic-teleological way, as Churchman (1971) would say. Hence, the high importance of approaching the problem can be from a systematic philosophy, theory and/or methodology. These three branches have shown to be very meaningful in building so many successful engineering works for our society, why not for building a computer-based system?

The answer might be that traditional engineering areas are more efficiency oriented rather than effectiveness oriented; a system approach to computer applications should therefore have the potential to not only tackle the complexity-dimension of a tool-construction, but also reveal the features of dynamic and uncertainty of environment which a tool is constructed for. As an example, the environment of a car assembly line is not that dynamic and uncertain, while the cooperative environment in managing water resources is the one which is dynamic and uncertain but not the assembly line's environment.

For these reasons and others mentioned in [Liu94b] [Liu94d], we shall focus on DSE (decision support environment). However, this does not make DSS less important. In particular, as we have mentioned in section 1.1.1 & 1.1.2, two design approaches are very beneficial, i.e., AI and SE.

## **1.5 Viewing water resources management as the application domain**

So far, we have argued that there can be little doubt that decision-making concerning the complex issues facing society today stands to benefit from the enormous software tools' capability - if they are properly used. 'Proper use' requires an adequate understanding of not only the tools, but also the systems and their processes of which the world is composed.

In this section, we closely investigate our application-domain, i.e., WRM.

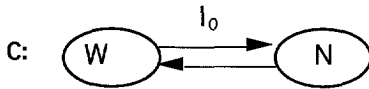
WRM involves several systems and the interaction between them. The relevant details of the definitions of the systems can be found in [Domenico72].

### **1.5.1 The knowledge domains in terms of systems**

The management involves a system called relatively closed hydrological system (C).

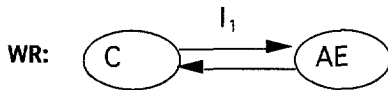
Hydrology is defined as the science that deals with processes governing the depletion and replenishment of water resources of the land areas of the earth. The long journey of events marking the process of a particle of water from the atmosphere to the land masses and oceans and its return to the atmosphere is termed the hydrologic cycle. If we only consider the cycle is solar powered, energy but not the matter (water) is exchanged with the outside environment, the cycle is called relatively closed hydrological cycle. On the other hand, if the cycle is considered neither energy nor matter being exchanged with the outside environment, it is called closed hydrological cycle.

If we define geologic formation of water storage as W and the set of the natural environmental elements and its effects as a system N, the complexity of the cycle can be, at least, denoted by the interaction  $I_0$  between W and N as follows:



The management involves a system called water resource system (WR).

The natural setting of the cycle is not that simple. It is a result of interaction between man and the hydrologic environment: few river systems are unregulated by surface-storage facilities; atmospheric scientists are actively pursuing the goal of augmenting and controlling precipitation, etc. In short, only considering the natural hydrologic phenomena is no longer suitable. We must take account of interaction with the hydrological artificial environmental elements. If we define the set of the artificial environmental elements and its effects as a system AE, the complexity of water resource system can be denoted, at least, by the interaction  $I_1$  between C and AE as follows:



From the information technology point of view, WRM concerns the problems of discrepancy which occur when the information of the major components of the system compare with established criteria to predict and understand the impacts of any action taken to control, manage and use water. We can then define the interaction ( $I_2$ ) between WR, and the criteria system (CR). CR consists its subsystems: the system which deals with the human demands for water for our survival and the effects to social systems; the system which deals with environmental demands, such as living plants and living organisms needing water, and effects to environmental systems; the system which deals with the fast developing economic needs, such as industry and agriculture, demanding water and the effects to economic systems, etc. Now we have the following:



The most important point of all is that the different levels of the interactions cause the different levels of knowledge scopes which can be defined as follows:

- C level ( $I_0$ ) : the knowledge which relates to result of interaction between geologic formation of water storage (W) system and the set of natural environmental elements (N).
- WR level ( $I_1$ ) : the knowledge which relates to result of interaction between C system and a set of artificial environmental elements (AE).
- WM level ( $I_2$ ) : the knowledge which relates to result of interaction between WR system and CR system.

Devising WRM through different levels of knowledge scopes enables us to explain that actions or decisions taken to address the interests of employing water will have to deal with

the knowledge on level  $l_2$ . Because the interests could not be isolated from the interests of knowledge on level  $l_1$ , they have to deal with  $l_2 + l_1$ , and possibly  $l_2 + l_1 + l_0$  as well, see Fig.1.6 illustrates.

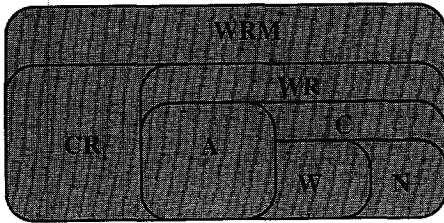


Fig. 1.6: a setting of WRM

### 1.5.2 A case of water policy analysis in the Netherlands and the geographical presentation

In analysing multi-stage problems of WRM that we have so far studied as our application domain, a decision can be made by a determination of the choice of *policy*. This topic will be studied in chapter 2 conceptually, and in chapter 8 technically.

But for now, we review a case of water policy analysis in the Netherlands and show a few relevant geographical visualisations. Through this, we shall note that this kind of visualisation would be more supportive to decision making, if it is sensitive to the context of data.

Natural water with good quality has become a scarce commodity in the Netherlands. Water level management and groundwater withdrawals have lowered the water level and reduced the reserves of fresh water. The water is being contaminated by point discharges and distributed pollution emanating from agriculture and industry, among others. The available quantity and quality of the water no longer meet the requirements of the various users, according to [GMN92] - a report written by groups of experts in policy analysis.

Indeed, the experts who participated in that study are from the four national provinces, namely, Noord-Holland, Flevoland, Utrecht, and Gelderland. They were aiming to find an efficient approach to provide solutions for the damage to nature caused by a lack of water. They were also challenged by the national government that, by the year of 2000, the degree of the nature-damages should be reduced by 25%. Reportedly, this task will not be easily fulfilled, because almost all the other objectives of water-uses have to be taken care of.

Examples of expertise are seven policy scenarios which are constructed in the five categories with regard to the five components, namely, industrial-use, natural environmental protection, drinking-water, surfacewater-level, and groundwater-use. In all the policy scenarios studied, the industrial-use of the north and south Veluwe are being reduced with 10 millions  $m^3$ . Fig. 1.7 shows the areas. The scored values of the scenarios are listed in Table 1.2, and these scenarios are visualised (see Appendix 1.1, 1.2 & 1.3).







The area studied consists of three main geohydrological scopes in regions visualised in the map, see Fig. 1.8.

Table 1.2: Data of policy scenarios

The characteristics of the nature in the studied areas are considered under the following categories:

- A: wide-range nature value available with high potentials.
- B: midum-range value available to be enlarged.
- C: midum-range to small range nature value in a larger area which have been paid by an attention.
- D: very small range nature-value available in a large area.

Alternative 1

reducing old land 19 millions  $m^3$  per year; extension in Flevoland with 39 millions  $m^3$  per year; surface and bank-groundwater 12 millions  $m^3$  per year; there is still a shortage of 6 millions  $m^3$  drinking-water per year.

Alternative 2a:

reducing old land 24 millions  $m^3$  per year; extension in Flevoland with 24 millions  $m^3$  per year; surface and bank-groundwater 35 millions  $m^3$  per year; there is still a shortage of 3 millions  $m^3$  drinking-water per year.

Alternative 2b:

reducing old land 24 millions  $m^3$  per year; extension in Flevoland with 24 millions  $m^3$  per year; surfacewater Markermeer 35 millions  $m^3$  per year; there is still a shortage of 3 millions  $m^3$  drinking-water per year.

Alternative 3a:

reducing old land 44 millions  $m^3$  per year; extension in Flevoland with 24 millions  $m^3$  per year; surface and bank-groundwater 54 millions  $m^3$  per year; there is still a shortage of 4 millions  $m^3$  drinking-water per year.

Alternative 3b:

reducing old land 44 millions  $m^3$  per year; extension in Flevoland with 24 millions  $m^3$  per year; surface Markermeer 54 millions  $m^3$  per year; there is still a shortage of 4 millions  $m^3$  drinking-water per year.

Alternative 4

comfort 2a, however, with the flooding concerns of the parts of Horstermeer polder in the Vechtplassen area; there is still a shortage of 3 millions  $m^3$  drinking-water per year.

Alternatives 5

reducing old land 4 millions  $m^3$  per year; extension in Flevoland with 29 millions  $m^3$  per year; surfacewater 9 millions  $m^3$  per year; there is still a shortage of 4 millions  $m^3$  drinking-water per year.

We may now concentrate on an aspect of the system schematisation in the case described in the previous section. The hydrological system in the area studied is simplified into a number of subsystems connected by hydrological processes which can be described mathematically. The schematisation takes account of the problems posed and the characteristics of the area studied. The model was devised using the STATRECT2 software program which was developed by the former National Institute for Water Supply in Leidschendam (now the National Institute of Public Health and Environmental Protection). A parameter to be mentioned here is called *transmissivity* which is identified at this stage as water-flow from one kind of geo-object to another measured by unit is  $\text{m}^2\text{day}^{-1}$  [Witmer89].

According to [Witmer89], it is important to let the transmissivity parameter be known. He describes that the high transmissivity and low adsorption capacity of the sediment of which the Gooi is composed and the absence of a covering layer make the groundwater resources vulnerable to pollution. The Gooi has a high population density and a large number of factories and companies are now located there or have been in the past. Concentration of population and industrial activities entail risks for the quality of the soil and groundwater. Both the drinking water, industrial water supply and the vegetation require groundwater of a high quality.

## 1.6 Remarks on the applications of decision support technologies

Many recent symposia and conferences which address themselves to WRM have adopted computer-based decision support as having an important impact of water resources research, e.g., [IAHS180] [IAHS211] [Hydrosoft94]. Appendix 2 lists the works having been surveyed, where three categories of the applications for the following groups can be reviewed,

- a) surface water systems managers;
- b) ground water systems managers;
- c) general water resources managers.

Through the survey, it is found that the large effort on the research side does not seem to generate a comparable diffusion on the application side, because the impacts of criteria-analysis for applying computers in the areas of decision processes have been either ignored completely, or not mentioned overtly. In particular, to make the presentations sensitive to applications is still the bottleneck that a software tool seems to be powerless to break. This problem is nevertheless getting more and more notable, e.g., in the article titled: 'Hydro-informatics - are we repeating past errors?', Ball points out a need for appropriate interfaces that allow alternative numerical models, information systems, and control systems to be used. Typically, not all the data items stored in a GIS (geographical information system) would be pertinent to current techniques for the relevant analysis [Ball94].

It is arguable that visual representations of concepts and objects play a major role in human thoughts. In particular, some basic concepts and natural categories can be learnt through interaction with the environment [Fodor87].

But visual representations are well known to be clustered in accordance with their similarity to a central prototype representing something like a most typical member with a *fixed* core, while variation is only allowed at the periphery [Brake191].

It follows that we are in need of better graphical tools and programs if a procedure, such as Agenda 21, the Rio Declaration, is to be more than a pious wish-list. But for the applications in WRM, the graphical representations should be 'sensitive' to a numerical analysis, dynamic information, and a context of a decision process, because

- data about a water-related system and its process are often representing *measured* phenomena.
- information can be characterised by subjective or collective values, judgements, preferences, perceptions and expectations.
- decision process can be carried out by social-economic driving forces and impacts, and social-political affections.

The above arguments will be further discussed in the next chapter.

## 1.7 Coping with the crisis

In coping with the crisis, the approach being developed here is aiming to adapt a situation at hand. For a design and an implementation of a specific system, the study intends to be a kind of 'demand-pull-evolution'. For absorbing diverse methods, tools and techniques, however, the study intends to be a kind of 'technological-push evolution'.

We shall follow the spirit of so called descriptive theories, specially the 'disjointed incrementalism' [Braybrooke70], which encourage an empirical-inductive construction of an approach. The term 'evolution' refers to a research manner wherein a computer-based system's planning is expected to respond to input-execution, or vice versa. That is, execution provides information for re-planning, or, 'analysis', 'learning', implementing, and 'analysis' are parallel to the synthesis of the desired software processes. Again, these points might be seen more explicitly as our discussion progresses.



## STUDIES OF DECISION MAKING PRIMITIVES

---

We have discussed that a software tool may be *efficient* in making a software method user-interactive, but may not be in any strong sense effective to support decision making. We have revealed that there are different levels of the knowledge domains (see section 1.5). Aiming to improve the software effectiveness, we have briefly mentioned the contexts of decision processes which DSE should adapt (see section 1.6). In this chapter, we shall consider these contexts further.

We shall argue that a model of a knowledge domain is not sufficient enough to represent such a context. There are three kinds of settings noteworthy, i.e., physics-descriptive, objective-optimal and politics-rational environment. They represent the relationships between 'intentionality and orientation of decision process', which, together, will become a basis for an employment of the notion - 'intelligence' and of the notion - 'object under study' in the next chapter.

We begin with an instance of a need for data.

### 2.1 An instance of a need for data

Data measurement can play an important role. For example, irrigation is one of the major impacts in water-policy assessment. The relevant data should be carefully stored, manipulated, transformed, interpreted or summarised.

#### **Example 2.1**

The strategy of the irrigation is easily revealed by a diagram. Alfalfa, a perennial crop, is maintained regardless of the flow of the river, where acreage of other crops is adjusted in accordance with flow. It is important to note here that planting decisions for crops other than alfalfa must be arrived at and executed at least 1 month in advance of the runoff season.

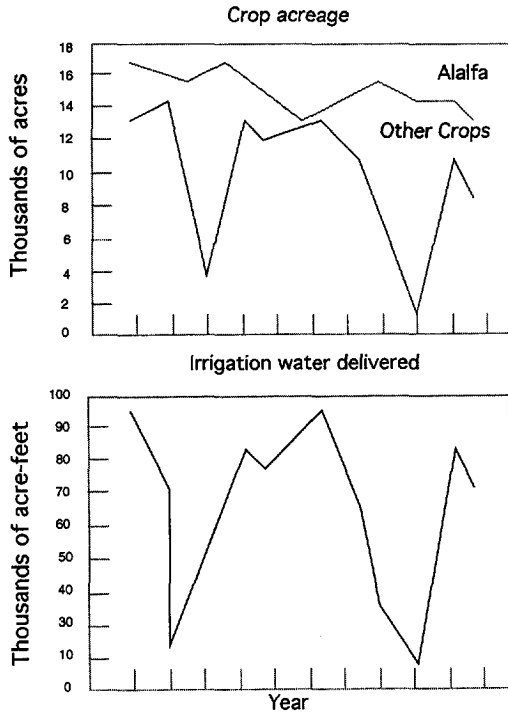


Fig. 2.1: relation between acres irrigated, crops grown, and surface-water deliveries, based on [Domenico72]

## 2.2 Remarks on some concepts in treating decision problems

Example 2.1 is taken from [Domenico72]. Fig. 2.1 illustrates that certain irrigation decisions are annually made on the basis of a 'transmission of information' on probable annual water supply. It is difficult to explain this example more explicitly at this stage. Our intentions will become clearer after we introduce some concepts.

### 2.2.1 Under a model-driven approach

A typical motivation under a model-driven approach may be best indicated by the following [Lindley85]:

"... there is essentially only one way to reach a decision sensibly. First, the uncertainties present in the situation must be quantified in terms of values called probabilities. Second, the various consequences of the courses of action must be similarly described in terms of utilities. Third that decision must be taken which is

expected - on the basis of the calculated probabilities - to give the greatest utility. The force of 'must', used in three places there, is simply that any deviation from the precepts is liable to lead the decision maker into procedures which are demonstrably absurd..."

Accordingly, decision problems are represented in the form of a *decision table*. The idea underlying this is that the consequence of any action is determined not just by the action itself, but also by a number of external factors. These external factors are both beyond the control of the decision maker and unknown to decision maker at the time of the decision. Take example 2.1 for instance: if one is making irrigation decisions, the external factors are crops grown, surface-water deliveries, etc.

A *state of nature*, *state of the world*, or simply, *state* is meant by a *complete* description of these external factors. E.g., it is often assumed that

if the decision maker knew the state of nature which would actually hold, i.e., if he (or she) knew the true values of the state of nature which would actually hold, and,  
if he also knew the true values of the external factors,  
he could then predict the consequence of any action with certainty; the state that actually holds is normally called the *true state*.

Another interesting case is often found under this approach, i.e., although the decision maker does not know the true state of nature, he does know what states are possible. Thus, in this case, the consequence of any action could be predicted with a degree of uncertainty. Many concepts are then derived. What follows names a set of most basic ones (for detail see Appendix 3.1),

decision table (basic representation of a decision problem),  
value and decision (interpretation of a consequence),  
certainty and decision (interpretation of a state),  
decision tree (basic representation of a multi-stage decision problem).

In analysing multi-stage problems, it should be noted that the aim is to determine the choice of policy. A *policy* is a complete contingency plan. It describes what choice the decision maker should make at each decision point that may be reached under the policy.

Table 2.1: Decision table representation of an irrigation problem

Consequence(Utilities)		State of nature		
		S <sub>1</sub>	S <sub>2</sub>	S <sub>3</sub>
	a <sub>1</sub>	1.0	0.4	0.0
Action(Policy)	a <sub>2</sub>	0.5	0.8	0.0
	a <sub>3</sub>	0.1	0.1	0.1

The policy indicates the decision to be made at decision points. When the concepts of a policy are established, it is possible to make the conversion into a decision table. Table 2.1 shows an example, where the items may mean:

- $a_1$  : authorise pumping water; if successful, allow a larger scale of irrigation;
- $a_2$  : authorise pumping water; if successful, allow the same scale of irrigation;
- $a_3$  : abandon pumping water;
- $S_1$  : agricultural development succeeds and water demand is high;
- $S_2$  : agricultural development succeeds and water demand is low;
- $S_3$  : agricultural development fails.

Groups arrive at decisions by some form of voting. Suppose there are two or more people who are jointly responsible for choosing an action from a given set of possible actions. Decision scientific developments are interested in how an action can be chosen from a group as a whole. The simplest context of group decision is that, they all truly agree with one decision table. Then, the preferences of a group of individuals can be represented by an order of a set of actions.

If we consider that a group of individuals may disagree with each other upon interpretations of states and the values of consequences, or, simply, they are choosing an action together by different decision tables, the decision problems there seem to be much more complicated. Therefore, there would be too many kinds of formalisms to be introduced here.

### **2.2.2 Under a data-driven approach**

Although a model-driven approach claims to be concerned with problems in general, for many decisions, the relevant knowledge is imprecise and often wholly qualitative. Effective procedures are often crucial as well as taking decisions.

#### *Problem solving and decision making.*

Both problem solving and decision making take place in the context of some problem. The problem context includes all the constraints that define our current situation, including factors such as the rules we are required to follow, the budget and other resources available to us, and the cast of characters with which we must deal.

It is important to remember that problem solving and decision making are always relative to the problem context. Given the problem context, some action might be required in order to achieve our desired objectives or goals.

If there is only one way to achieve the objectives, then we are engaged in problem solving. Problem solving is important; however, most real problems do not have single solutions. When there is more than one alternative for attempting to obtain our objectives, then decision making is involved, since we must select a course of action from a set of possible actions.



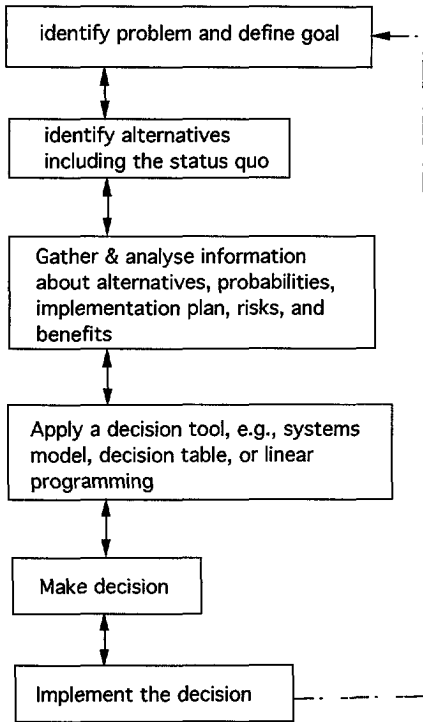


Fig. 2.2: general steps of decision processes

### *Decision process.*

After a decision has been made, there is a chance (we hope good chance) that our original problem will, in fact, be solved. There is also a chance that we will not have solved our problem. While everyone wants the result of a decision to be a good outcome, there are no guarantees at the time the decision is made.

If the alternative selected is from a thorough set of alternatives and it is efficacious, implement-able, ethical, and 'optimal', then we have done all. By efficacious we mean that the alternative must, in fact, be effective as a solution to the original decision problem.

Moreover, the alternative must be capable of implementation and it should not violate any norms of ethical conduct. An ethical alternative would not only consider the rights and best interests of the party making the decision, but also give consideration to the rights and interests of those *affected* by the decision, but not *explicitly involved* with the decision process.

Even if the selected alternative is efficacious, implement-able, and ethical, it still may not be the best choice. If there is another alternative that is also efficacious, implement-able, ethical, and that alternative is preferable to the first alternative, then we have not selected the best available alternative. Consequently, the selected alternative must be an optimal choice

(according to some set of criteria) from among the efficacious, implement-able, and ethical alternatives.

Another way to understand a good decision process is to look at the ideal processing steps involved in decision making and management. An ideal process is illustrated in Fig. 2.2.

## 2.3 Introducing system engineering disciplines

Decision making could be such a vital human activity that we cannot ignore the complex interdependencies that occur in nature. Human decisions are at the core of most actions affecting the natural environment around us.

Decision making has to be studied multi-disciplinarily, not only because it is at the heart of many problems which have to be governed by measurement, but also because it is at the core of solutions to those problems that have to be processed at first.

If we do not care to question the wisdom of reasoning or the decision process, and to exercise in the decision process on a similar problem, we do not need a system approach. Luck may have been the key, but next time a flawed process could cost dearly.

For example, automotive engineers do not design automobiles in order to pollute the air. They design cars to satisfy many criteria; the final product reflects many trade-offs. They have to quantify pollutant, though whether to allow producing those vehicles that pollute the air is a result of a series of industrial decision processes, in which the impact on environmental quality is no longer unknown, ignored, or trivialised. Note, in this example, the origin of decision problems is the engineering of a car, but the core of a solution is itself a problem also, i.e., how to include the industry response to regulatory standards. The imposed standard must become a design constraint fed-back to the engineers, see Fig 2.3. shows a simplified version of this problem.

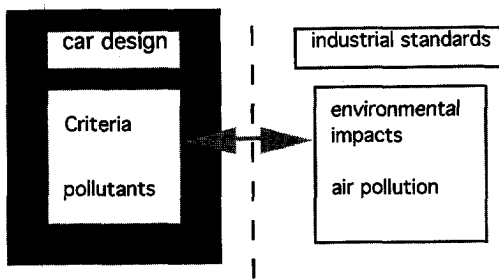


Fig. 2.3: an example of a car design problem enlarged to a decision problem

We have no reason to allow the decision processes to be repeated, if and only if

- a) as long as the standard was satisfied, the car was not causing a pollution problem from an engineers' point of view;
- b) without incentives for doing better in the elimination of polluting emissions, therefore, there is no expectation for improvements to be suggested.

Consequently, an approach has been introduced in the automotive industry for determining, to a large extent, the degree to which the car products remain a source of pollution. The enlargement of car design problems, because of an (natural) environmental concern, is governed by this approach. It allows people outside the decision sciences to focus on the product or result of a decision as a way to evaluate the quality of the decision; we are then able to emphasise the decision process as well as the decision product - the best chance for a good decision product depends on a good decision process. By now, we have already invited a system approach into our discussions.

The system approach begins and ends under the agreement with the empirical primary purposes, i.e., decision making is problem solving, including, all the processes necessary to finally solve the problems. But, the system approach agrees with a scientific manner to treat the problems by exploring system thinking to the study of decision making.

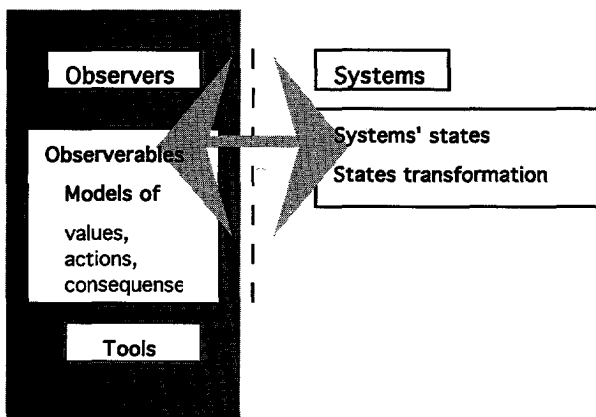


Fig. 2.4: a general frame of this system approach.

Informally speaking, system approach treats

- nature by system concepts;
- state of nature by specification of observables belonging to the system;
- action by action made by systems' observers to change a characterisation of a manner of a system, in which the observables are linked;

- consequence by a value to observers who are transforming one state of a system to another.

The left side of Fig. 2.4 is what we intend to formalise in terms of 'decision making environment'.

Now, to explain the decision problems in example 2.1, we can follow this primary system approach to have some ideas about a decision making environment.

### 2.3.1 The system of nature

Suppose that a piece of farm land is in a typical snow-melt area, where the accumulated winter snow-pack represents a large percent of the annual flow and is generally depleted during the months of a year, e.g., April, May, June, and part of July.

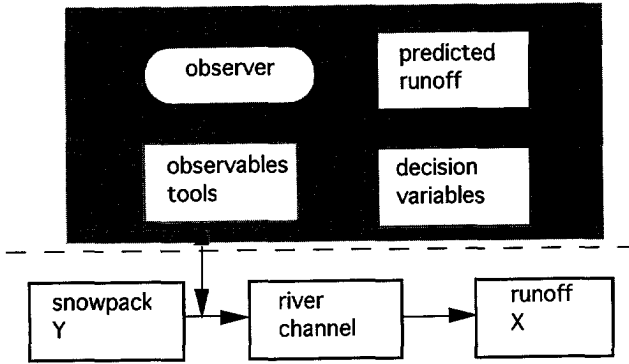


Fig. 2.5: an example of system approach to treat decision problems

As the ranchers appear to be making decisions from information received, one may make inquiries about nature. Here, if we explore system approach, nature is observed by means of system analysis. In this case, a federal agency usually conducts snow-pack measurements, which are converted to runoff, and makes these estimates known at various times previous to the runoff season. There are two systems to be observed: Snow-pack (Y) and Runoff (X). Fig. 2.5 is a representation of this aspect of nature.

### 2.3.2 The states of the system

Essentially, there are at least two distinguished states of the system. One is a state of the system before any decisions have been made, and the other is the system after a decision has been made. How to transform between these two states is therefore the whole reason for the decision processes being carried out. The observers, assisted by any necessary tools, may observe Y and X subsequently or simultaneously.

By observing, we mean, for example, measuring. The measurement may be conducted by several different methodological manners. In this case, system Y is observable, but system X is not. X is not directly accessible until after the fact, that is, until all planting decisions are executed. Therefore, based on information obtained by observing Y, a decision is made concerning the state variable, in this case, acreage of crops grown other than the perennial stand of alfalfa (see Fig. 2.1). If the system is 'noiseless', there is a one-to-one relation between X and Y. That is, there is no discrepancy between predicted and actual flows, and the problem is characterised by a complete lack of uncertainty.

'Noise', however, may enter for several reasons. An unseasonable rainfall, for example, may remove much of the snow-pack prior to the irrigation season. For this reason, measurements and reports are conducted at various times throughout the winter season. Another source of noise is the conversion of some part of the snow-pack into evaporation and groundwater inflow, which may vary from year to year.

Table 2.2: states  
representing uncertainty  
from experience

$Y_1$	$Y_2$	X
1	2	1
3	3	3
1	2	2
2	3	2
2	3	3
3	3	3
1	1	1
1	1	1
2	2	2
3	3	3
2	2	2
3	2	2

### 2.3.3 The states of certainty

The system analysis which has been given, so far, enables us to have a very useful manner to treat the uncertainty of decisions to be made. That is, an observable system Y could be presumed capable of transmitting certain information about X. How well it does so depends on how closely the systems are related. At one extreme, if X and Y are independent, it is not at all possible to remove any uncertainty. At the other extreme, given a one-to-one correspondence between X and Y, all uncertainty vanishes from this problem, i.e., we are

no longer interested in speculation about X's behaviour, because the observations of Y tell us all we wish to know about X.

Under this frame, we could review historic records by a simplified version of this problem treated in the past, see, e.g., Table 2.2, a table of the concerns for decisions made in the past, where numbers 1, 2, 3 represent estimated states or flows (X) that were available for irrigation use.

## 2.4 A decision making environment

What follows intends to describe the existence of three kinds of decision-making environments, they are: physics-descriptive, objective-optimal and politics-rational environment. The environments are regarded as settings of the 'external world' where DSE are aggregated.

Structurally, a decision-making environment could be seen as a constitution of the following:

- observer, who is observing a system,
- models, which are representing that system,
- instruments, which are enabling observer to have observables to 'see' that system.

But, because one observes a system intentionally, and one's intention may differ from the others during an entire decision process, treating the decision-making environment as a whole is no longer satisfactory.

### 2.4.1 Objective modeler

The concept of the decision-making environment emerges, because a decision process begins by someone who has a particular purpose to observe a system - he is interested to perceive the need to alter the course of a system with which *he* is concerned. This situation is then diagnosed [Easton73], he feels the need to state general statements of overall objectives. In this context, it is perhaps more appropriate to call him *objective-modeler* instead of observer. Objective-modeler, together with the models and instruments concerned constitute an initial environment, i.e., *objective-optimal environment* (O-O for short).

### 2.4.2 Observer

As the decision process goes on, the variety of intentions for observing that system is expanding<sup>+</sup>.

For a set of objectives to be well defined, meaningful comprehensive analysis of the pertinent aspects of that system has to be undertaken. Facilitating this analysis then becomes the primary task. However, the only hope of fulfilling this task is to try to understand the problems better. The initial environment is enlarged; similarly, we think that there is another

---

<sup>+</sup> we hope that we are still talking about the same system!

category of environment becoming involved, i.e., *physics-descriptive environment* (P-D for short). In this context, we still call the person *observer*.

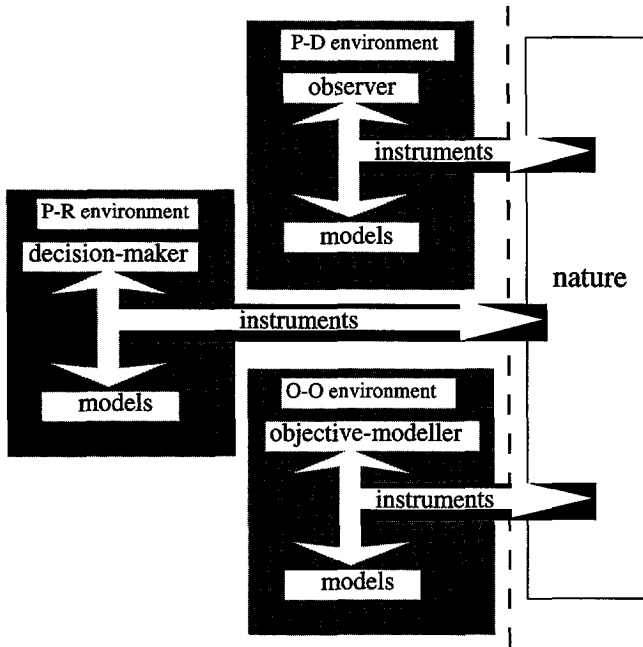


Fig. 2.6: a setting of the externals of nature

### 2.4.3 Decision maker

As noted by Rigby, the term 'decision-maker' is difficult to define precisely [Rigby64]. Churchman refers to the decision-maker as "the person who has the ability to change the system, i.e., has the responsibility and authority for such a change" [Churchman68]. More specifically, and more appropriate to the context with which we shall be concerned, we shall take the decision-maker to be an individual or a group of individuals, who furnish(es) the final value judgement that may be used to rank available alternatives for the 'best' choice to be identified. We think there is a *politics-rational environment* (P-R for short), where final value judgements need to be made, which are often concerning the 'goodness' or 'badness' of a given choice.

It is worthwhile to mention several classes of features as follows. For simplicity of exposition, we shall begin with the discussion of the P-D, and then assume that the concepts therein are applicable to the O-O, such as models of a system, as well as variables, parameters, inputs and outputs of a system. All together suggest that it may be very beneficial to treat the externals as Fig. 2.6 shows.

## 2.5 About P-D (physics-descriptive environment)

The concept of a model is a generation of the concept of a subsystem of a real natural system  $N$ . In particular, the essential feature of the modelling relation is the exploration of the idea that there is a set of circumstances under which the model describes the original system to a prescribed degree of accuracy.

In other words, a particular facet of a system's state could be identified by replacement of the original system by a proper subsystem. Take the explanation of example 2.1 for instance, runoff exists regardless of whether we measure it or not. We may have, on the other hand, many models representing runoff, each of which are simultaneously part of the entire interpretations of runoff and the subsystems of the original system, i.e., runoff here.

Consider a particular subset  $S$  of the observable world, and assume that  $S$  can exist in a set of physically distinct states  $W = \{S_1, S_2, \dots\}$ . Note that there must be at least one observer probing the behaviour of  $S$ , who may or may not be able to determine whether  $S$  is state  $S_i$ , or  $S_j$ ,  $i \neq j$ . It then all depends upon the resolution of the measuring apparatus at his disposal, i.e., the observables, so to speak. Also note that the set  $W$  may be finite or infinite. We call  $W$  the set of abstract *states* of  $S$ .

Here it is important to emphasise that what counts as a 'physically distinct state' is not an intrinsic property of the system, but depends crucially upon: a) the observer and the ways he has for probing the system, and b) distinguishing one state from another. For details see Appendix 3.2.

## 2.6 About O-O (objective-optimal environment)

The O-O obeys the same philosophy of modelling (or formulating) as the P-D does, though the means of modelling is quite different.

The problems are now viewed as *objective* realities, no longer as 'images'. Objective-modeler focuses on the facts as objectively existing realities. A problem is always selected in a pragmatic way, and remains objective.

The models are normative models. In other words, the models are used for a *subjective* presentation conceived by the modeler confronted with a reality which he or she sees and perceives to be unsatisfactory or satisfactory. If a subjective dimension is recognised, a problem becomes someone's problem, e.g., an optimistic viewpoint can also be closely correlated with the observations in the P-D, see [Mays89] for example.

Here, we shall focus the O-O on the orientation of multi-objective problems.

### 2.6.1 Multi-objective and attribute

An understanding of the meaning, structure, and properties of the terms - attributes and objectives - is crucial, if one hopes to capture the essence of a complex decision problem. Keeney and Raiffa give an excellent account of this topic, with numerous illustrative examples [Keeney76]. Nevertheless, a few basic concepts need to be reviewed here.



A multi-objective decision problem must also have objectives; otherwise, it would be vacuous or poorly defined. Essentially, an objective is a statement about the desired state of the system under consideration. Thus, in a multi-objective decision problem, there are several statements expressing one's desired state of a system. Objectives are statements of 'wants' and thus may or may not be achievable. Nevertheless, they are the goals toward which the system should be proceeding and they are standards against which the quality or performance of a given alternative may be evaluated.

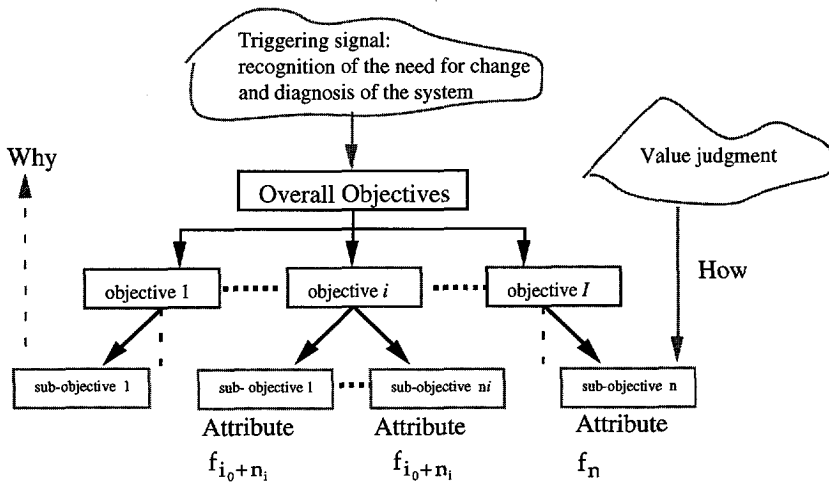


Fig. 2.6: Typical hierarchical structure of objectives

A well-defined set of objectives often exhibits a hierarchical structure, as illustrated in Fig. 2.6. The highest level of this structure generally represents the broad overall objectives that are instrumental in initiating the multi-objective decision problem in the first place. These objectives are, however, often vaguely stated and, hence, unoperational. As we go down the hierarchical level, objectives at the lower level are more specific and more operational than those in the higher level and are perceived as means to achieving higher ends represented by objectives in the higher level. Thus, objectives at the lowest level of the hierarchy are 'most specific' and 'most operational'.

An objective is operational, if there is a practical way to assess the level of achieving such an objective. To facilitate this kind of practical method, a set of attributes has to be assigned to each objective in the lowest level.

An attribute is a measurable quantity whose (measured) value reflects the degree of achievement for a particular objective (to which the attribute is ascribed).

### 2.6.2 Proxy attribute

In many instances, the value of an attribute (or attributes) will give an obvious and direct indication of the degree of achieving an associated objective. For example, the attribute 'net profit measured in terms of dollars' is a direct measurement of the degree of achieving the objective 'maximising profit'. For some problems, it may be possible to formulate accurately the multi-objective decision problem in such a way that objectives and attributes are related only by direct relationships. This type of direct relationship between objectives and attributes is, indeed, what we would like to have. The idea of articulating objectives into hierarchical levels is, in fact, a way of achieving this goal. For each objective in the lowest level, there should ideally exist an attribute (or a set of attributes) whose value is a direct measurement of the level of achieving that objective.

In some cases, however, there may be an objective (particularly if associated with ethics or values) for which there is no obvious attribute or set of attributes to measure the level of achievement directly. An attribute (or set of attributes) that is more conveniently measured and that indirectly reflects the level of achieving the objective may exist. Such an attribute is referred to by Keeney and Raiffa as a *proxy attribute* [Keeney76].

Whenever an objective is indirectly measured by a proxy attribute, an indirect relationship is implied implicitly between such an objective and the corresponding proxy attribute. By indirect relationship we mean that an additional activity involving value judgement by the decision maker is needed to complete the final evaluation of the level of achievement for the objective in terms of the value of the proxy attribute. For example, one of the objectives of water-resources planning for a river basin given earlier reads, 'to enhance recreation opportunities in the basin area'. There is no obvious attribute (or set of attributes) for this objective. A common proxy attribute used for this objective is perhaps the number of *recreation-days*, where one recreation-day is defined as one visit by one individual to a recreational development or area for recreation purposes during a reasonable portion or all of a 24-hr period. The implicit value judgement implied here is that the higher the number of recreation-days, the better the recreation opportunity in the basin.

### 2.6.3 The properties of an attribute

To assign an attribute (or set of attributes) to a given objective, two properties should be satisfied: comprehensiveness and measurability.

An attribute is comprehensive if its value is sufficiently indicative of the degree to which the objective is met. It is measurable if it is reasonably practical to assign a value in some scale to the attribute for a given alternative. Although both requirements are intuitively clear, the scale of measurement has in some quarters received less attention than it deserves.

### 2.6.4 The properties of a set of attributes

Just as an attribute assigned to a given objective should be comprehensive and measurable, a set of attributes representing the entire multi-objective decision problem should also possess some desirable properties. Keeney and Raiffa list five such properties: a set of attributes must be complete, operational, decomposeable, non-redundant, and minimal [Keeney76].

A set of attributes for a given multi-objective decision problem is *complete*, if all pertinent aspects of the decision problem are represented by the attributes; It is *operational*, if it can be utilised in some meaningful manner in the ensuing analysis; It is *decomposable*, if simplification of the evaluation process is possible by disaggregating the decision problem into parts; It is non-redundant, if no aspect of the decision problem is accounted for (by the attributes) more than once; Finally, it is minimal, if there is no other complete set of attributes representing the same multi-objective decision problem with a smaller number of elements.

### 2.6.5 The decision rule

Trying to choose the 'best' available alternative implies ranking all available alternatives according to their performance or quality measured in terms of the values of all attributes selected for the multi-objective decision problem.

A set of rules that facilitate a complete ranking of alternatives will be referred to as the *decision rule*. Part of the decision rule is often implied in the statements of objectives. The statement of objectives may even completely specify the decision rule. For example, in the classical theory of the firm, the decision problem consists of the single objective, 'to maximise profit', and the attribute 'net profit measured in terms of dollars' is used to measure the performance of a given alternative. The decision rule clearly implied in this case is, 'Choose an alternative having the maximum possible profit'.

For some other decision problems, the decision rule needs to be explicitly stated apart from the statements of objectives. For example, in a water-quality model for a river, the single objective may read, 'to improve the water quality of the river', and the attribute 'a level of biochemical-oxygen- demand (BOD) measured in mg/litre' is used to measure the performance of a given alternative. In this case, it is not clear from the statement of the objective itself how alternatives should be ranked. A decision rule that may be used for this purpose may read, 'Choose any alternative with BOD level below 5 mg/litre'.

## 2.7 About P-R (politics-rational environment)

The P-D and O-O are the environments where the process of measuring preference is merely a means of modelling the structure of decision-maker's preference, so that observer and objective-modeler will therefore be able to substitute some pre-specified decision rule for the action of the decision-maker.

Unfortunately, several issues suggest that a decision-maker has limited control over the decision process.

First, the means-end (attribute-objective) relationships are in the simplest form in the sense that the attributes typically serve as both decision variables and objectives.

Second, the set of alternatives should be made explicit. That is, once the set of constraints on possible values of the attributes is given, the value of each attribute can be picked from this set without raising concerns about unfeasibility.

Third, the final outcome (the 'best compromise' solution) is assumed to be obtained if a reasonably accurate preference measurement could ever be made.

By virtue of the third characteristic, the primary purpose for treating decision problems is neither for understanding the nature of the problem, nor the process of the problem. Rather, the purpose is to treat decision making as an act. These views lead to several methods which can be described as product oriented or outcome oriented [Starr77] rather than process oriented.

Therefore, it is important to take the P-R into account.

### 2.7.1 Quantitative or non-quantitative models?

Perhaps here is a place to distinguish between quantitative and non-quantitative models. In the O-O or P-D, a model represents a set of assumptions about a real-life phenomenon, from which conclusions are deduced. The deduction uses either empirical generalisation such as 'laws of nature'; or logic in mathematics. The proceeding sample of the model is a quantitative model. Numerical assumptions are made and numerical conclusions deduced.

It is true that quantitative models are less appropriate in the social sciences, where many phenomena are essentially non-quantitative, than in natural sciences.

### 2.7.2 Rationality

Rationality has had a variety of meanings. We shall address ourselves to the idea of 'rational behaviour', meaning 'optimising behaviour'. That is, we shall say that a decision  $d_i$  is rational, if there are a number of alternative decisions  $d_1, \dots, d_m$  with foreseeable results  $r_1, \dots, r_m$ , none of which may be preferred by the decision-maker to the result  $r_i$  of the decision  $d_i$ . But, that decision-maker still chooses the decision that yields him the most preferred result.

Thus, rationality may have nothing to do with what the decision-maker 'ought' to do in a moral sense. It has to do with satisfying his preferences, which morally may be good, bad, or indifferent.

Therefore, it would seem that rationality is a 'good thing' provided only that the decision-maker is good (has good preferences). This view underlines utilitarian ethics as expounded by Russell, who says explicitly, "Men's actions are harmful either from ignorance or from bad desires" [Russell67].

For this reason, a rational decision-maker is always concerned with getting as much relevant information as possible before making a decision - trusting and believing. There is little doubt that, if one had all the relevant information and there would exist a way of handling it, one could then choose alternatives rationally.

The distinction between 'subjective' rationality (based on the decision-maker's possibly erroneous information) and 'objective' rationality (based on the actual state of affairs) is thus of considerable importance. Regarding the latter category of rationality, there are some rationales that can be modelled mathematically, e.g., conflicts and modelling conflicts.

### 2.7.3 Types of conflicts

A *conflict* is a situation in which there is a 'condition of opposition' [Funk74], and parties with opposing goals affect one another. In other words, conflict is virtually inevitable whenever humans interact, either individually or in groups.

We may have military conflicts, business conflicts, international-affair conflicts, etc. The observer and objective-modeler may rarely be invited to analyse these kinds of conflicts. But, to analyse some cases of conflicts, ignoring observer and objective-modeler is not very wise.

For instance, conflict analysis is of significance to engineers, because of the increasing importance of social and political influences in engineering decision-making. A large scale project is oriented by engineering, which must be indeed feasible physically. But, lately, most of those projects must be required to be environmentally, financially, economically, as well as socially and politically acceptable, for example, the problem of pollution in the auto industry illustrated at the beginning of section 2.3.

The widespread use of natural resources putting a severe strain on the natural environment is another case. For example, 'acid rain' that results from a kind of pollution is killing life in the lakes and rivers of affected regions. The problem of acid rain is quite complicated itself, since pollution spreads from the source across local and national boundaries. Then, conflicts start at least between the regions that do not directly benefit from a certain kind of production and the regions that do, because both are adversely affected under the industrial by-products.

### 2.7.4 Modelling conflicts

If we suppose a human organisation to bear full or partial responsibility for a certain set of decisions, those who are in a higher position in the hierarchy of that organisation are mostly involved in real conflict-situation. Following this, a hierarchy of an organisation, regulations, policies and laws then become subjectives of conflicts.

Conflict can be modelled as a *game*, where the groups who are in dispute over some issue or resources are called *players*. The possible courses of action available to the players in the game are referred to as *options*. A set of options that can be taken by a particular player is called a *strategy*. When each player has selected a strategy, the result is referred to as an *outcome*. When it is logically impossible or highly unlikely for an outcome to occur, the outcome is *unfeasible*. After the unfeasible outcomes have been removed from the game, the *feasible* outcomes can ordinarily be ranked from most to least preferable for a given player in order to determine that player's preference *vector*. Conflict models consist of players, their options, and their preference vectors.

The *stability analysis* of the game is executed by determining the stability of each feasible outcome for every player. If an outcome is *stable* for a given player, the avoided outcome is *unstable*. An outcome that is stable for all the players in the game model is an *equilibrium* and constitutes a possible resolution to the conflict.

As the stability analysis stage is used to predict the possible equilibrium, it is sometimes referred to as the *prediction* or *forecasting* stage.

The importance of conflicts modelling has been emphasised in [Radford80] [Bennett82] [Hipel83] [Coppeters90] [Fang93] for example.

## 2.8 Conclusions

It is agreed that, for study of interaction between a particular biologically complex system - the human mind and the physical and social world, a central problem comes in determining how human beings proceed to interpret the world [Chomsky71] [Ringle79] [Gibson66] [Simon69] [Newell72].

We have conceptualised the problems of decision making systematically, so that most of the complexities of constructing decision support are associated with the objects in physics-descriptive environment, as well as objective-optimal and/or political-rational environment.

As a few cases have implied and more will be shown in the following chapters, kinds of multi-disciplinary decision processes virtually exist. Furthermore, communications among observers, objective-modelers and decision-makers are extremely crucial. However, the meanings of communications never seem to lie between data, but the knowledge beyond.

*Multi-disciplinary.* The system description required for the study of the problematique must be able to cover a multitude of scientific fields in an integrative manner while still being able to use highly developed language and methodology of the individual discipline.

*Dynamics.* Changes of the state variables, positive and negative feedbacks, delays, nonlinearities, stochastics and decisions lead to complex dynamic behaviour which can no longer be intuitively grasped and comprehended in the problematiques and systems under study. In model construction, the difficulty arises of correctly including the major influences on the dynamics of the system.

*Decision units.* An important characteristic of societal systems is the large number of decision units which are able to influence - independently or in interaction - the behaviour of a group. Each unit belongs to a more or less developed information processing apparatus with the corresponding programs of perception, assessment, problem solving, learning and response and the corresponding information storage. Each unit orients its behaviour with respect to its own norms, values, goals, and evaluations of the future. Conflicts between decision units are a necessary consequence; these conflicts are resolved through direct or indirect interaction (e.g., via a common environment vs. an exertion of decision-power).

*Openness.* The stochastic of occurrences and of the boundary conditions of the system, together with the partially determined behaviour of the individual decision units, result in a steadily growing bandwidth of the possible future development of individual state variables. The spectrum of possible futures quickly becomes exceedingly broad, and a full simulation of all possibilities may be completely out of the question.

## CONNECTIONS TO COOPERATIVE AND DISTRIBUTED EXPERT SYSTEMS

---

So far, we have mainly dealt with the first question raised in the beginning of this dissertation, i.e., in which sense do software tools support decision making?

In this chapter, we shall make connections to the AI approach which was introduced in section 1.1.1. There will be two groups with sections and a tenor between them.

In the first group, we shall get a very general frame of 'cooperative and distributed expert systems' (CDES).

The purpose to have the frame is to show that CDES is indeed essential, but there should be a strong managerial support from the domain area regarding the large commitment of CDES. To have the support, there is a need for a non-conventional approach which enables us to specifically manage *object-orientations*, *knowledge acquisition* and *knowledge representations*. We shall, however, treat these issues in part II, since we should first of all find a base in AI.

In the second group, we shall elaborate on the meaning of 'intelligence', 'object' and 'pattern'.

### 3.1 Being cooperative and distributed

The purpose of an expert system (ES) is to solve problems of significance in the 'real world' which are usually thought of as requiring human expertise. Most existing expert systems focus upon simulating the problem solving methods of a single human expert and do not simulate cooperation between human experts. By observing the problem solving approaches studied in chapter 2, it is clear that such cooperation is often essential.

In the last few years, researchers have built a number of distributed artificial intelligence systems [Decker87]. The term 'distributed' has, however, different emphases in different systems. Some stress the geographical or functional distribution of the sub-components of the system [Corkill83] [Smith79]; some stress parallel executions of processes to enhance efficiency [Yang85]; and still others stress the mechanisms and functions required if distributed artificial intelligence is to be contemplated [Durfee87] [Durfee91] [Velthuisen92]. Design frameworks for integrating distributed knowledge bases and other generic information systems can be also seen, e.g., [Bell85] [Bell90].

Cooperative modes of problem-solving among human experts can be classified into four predominant kinds according to their inter-dependence relationships.

*Horizontal cooperation.*

Horizontal cooperation is where each expert in the cooperative group can get solutions to problems without depending on their experts, but if the experts cooperate, possibly using different expertise and data, they can increase confidence in their solutions. Example 2.1 shows more or less such a kind of characteristics.

*Tree cooperation.*

Tree cooperation is where a senior expert depends on lower level experts in order to get solutions to problems. For example, a chief engineer's decisions often depend on the work of junior engineers.

*Recursive cooperation.*

Recursive cooperation is where different experts depend on each other in order to get solutions to problems. For example, in order to interpret geological data, geological experts, geophysical experts, and geochemical experts often depend on each other in a recursive way. That is, there is a mutual dependence in which a geophysicist may ask a geologist to perform a subtask which in turn requires the performance by a geophysicist of some sub-subtask.

*Hybrid cooperation.*

Hybrid cooperation is where different experts use horizontal cooperation at some level in an overall tree or recursive cooperation. On the other hand, they could equally validly use tree or recursive cooperation at some point in an overall horizontal cooperation. An example of the former is where several opinions are obtained in order to optimise the quality of the final result coming from the engineers at a given level of seniority.

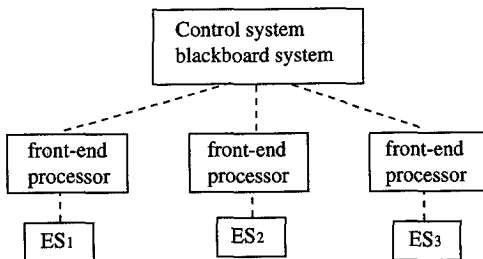


Fig. 3.1: an illustration of centralised-control architecture

### 3.2 An architecture based on centralised control

Centralised control with a Meta-ES (which means a control system and a blackboard system) permits easy extension of the system through the addition of other ESs, because when an ES is to be connected to a network, all that is needed is to modify only the Meta-ES and connected ESs.

As an illustration of the central control, Fig. 3.1 proposes that Meta-ES knows what problems can be solved by each individual ES, but it does not know how to solve a



specific problem. Any individual ES can solve some specific problems. The users normally access the distributed expert system by interacting with an individual ES. If the ES is unable to solve a given subtask, it applies to the Meta-ES for assistance. The Meta-ES checks for deadlock, etc. and then carries out transformations (e.g., on the inexact reasoning models used), selects another ES to which to entrust the subtask, and carries out all the communication activities required. Each site has a local subsystem which permits a new ES to join the global system. The architecture also should impose the four kinds of cooperation among ESs.

In Fig. 3.1, a dashed line indicates program linkage. The ESs, the control system, and the blackboard system can be considered as nodes in a network of computing facilities connected by telecommunications' links. More than one node can be located on one computing facility (or 'site'). Each ES in Fig. 3.1 processes domain-specific knowledge, and has the ability to plan and solve problems independently, and the ability to apply for assistance from and offer assistance to other ESs (via the control system). Each front-end processor in Fig. 3.1 is an interface between an ES and the control node and it is responsible for the management of cooperation and communication related to the ES with which it is directly connected. The blackboard system stores structured information for the control system and shared data for the ESs. The control system is responsible for the management of cooperation and communication between all ESs, and also for the run-time management of ESs' network. To reduce the cost of communication between the control system and the blackboard system, these two systems are put on the same site and are considered to be a single node because the blackboard system is essentially the information subsystem for the control node.

Normally three kinds of functional modules are necessarily needed in the subsystem of ES; they are: the management module, the communication module, and the man-machine interface module. As we have mentioned, a control system and a blackboard system being a single node share the same interface module and communication module. All modules are designed using knowledge-based principles, so each module is associated with a meta-knowledge base. This means that, if seeking a domain-independent kernel tool-based environment, we can change the domains by changing the contents of meta-knowledge bases only.

The scheduler module in the control system is the heart of ESs' network. There are four modules: the efficiency enhancer, deadlock treatment, transformer, and synthesiser, depending on the contents of the meta-knowledge base (which stores all the control information for all the ESs). Later we shall mention 'efficiency enhancer', and omit the other three. Nevertheless, they all together schedule the whole ESs in networking and manages the cooperation and communication between the ESs in order to perform addressed tasks.

The blackboard manager module in the blackboard system must do all the operations on the blackboard (such as search, read, and write) needed by other nodes.

The communication module provides the important means of cooperation between ESs. The communication between ESs must be through the communication model in the control node.

A front-end processor is an interface between one expert system and the control node and it manages a single expert system. We can link a pre-existing expert system to ESs network by simply changing the content of the meta-knowledge base and modifying the related template bases, e.g., the one for the communication module in the control node and the corresponding one in the front-end.

A user can submit a task (job) to any node in the network via the interface module. Each ES solves only the subtasks within its capability. When it meets a subtask it cannot do (there may be many), it applies to the scheduler, requesting for this sub-task to be solved.

The scheduler is a Meta-ES which is supposed to know (via the blackboard) all about the local ESs. When the scheduler accepts an application requesting the solution of a subtask from an ES or from a user, two steps are executed: a) it entrusts this sub-task to an appropriate ES, and b) after transforming the sub-task in the light of heterogeneity; furthermore, it makes ESs be able to distinguish their inference mechanisms and knowledge representation methods. Once reports are received from step b) about the result of this sub-task, the scheduler will inform the original agent on step a) of the result after transformation if necessary.

Centralised control has several advantages:

- a) It is convenient to store control information. We can avoid the repetitive storage of control information, such as information about all the cooperators (ESs) and running states, by storing it just once, i.e., in the control node.
- b) It is relatively easy to schedule tasks and sub-tasks for solving problems for efficient execution. The control node has a global view of the system to enable it to choose how and when to schedule subtasks on individual expert systems. The information needed by the scheduler included the priorities among sub-tasks, the load on each ES, and the current states of the ESs.
- c) The run-time management of the system is simplified, especially the treatments of deadlocks and the management of transformation of heterogeneity.
- d) It is easy to add a new ES to a network because we need to modify only the information in the control node and in one front-end processor.

There are, of course, disadvantages of centralised control in distributed systems such as information bottlenecks and reduced reliability. However, these shortcomings are not very severe for most operations that are localised, i.e., most ESs will independently perform most subtasks involved in solving a problem, so the frequency of communication is low.

### **3.3 Approaches to cooperation between expert systems**

Cooperation between ESs involves three essential elements:

- a) the cooperators;
- b) the management of cooperation; and
- c) communication.

The cooperators are ESs and they provide the problem solving capability of the entire network to be proposed. All domain-specific tasks are solved by ESs. Management of cooperation, the subject of this section, involves two parts: one is a front-end processor (see Fig. 3.1) in the control node which is responsible for the management of all the ESs with front-end processors, and for run-time system management.

Communication modules are responsible for the management of communication between the control node and ESs (we shall discuss this later).

### 3.3.1 Approaches to cooperation in the front-end processors

For any kind of cooperation three elementary steps are involved as follows: applying for aid, using aid, and offering aid. Each step includes three parts: strategy, function, and content.

#### Applying for aid

When an ES meets a sub-task which it cannot solve, it will apply to the scheduler for aid. When we use the expression 'a sub-task which it cannot solve', we mean that this sub-task cannot be solved under the *environment* of this ES, and this ES cannot ask the user for the result of the sub-task.

##### *The strategies.*

The strategies of applying for aid can be classified into three kinds, namely, natural application, precipitated application, and deferred application.

Natural application is exceedingly simple. When an ES is solving a task, it applies for aid as soon as it meets a subtask which it cannot solve.

Precipitated application is a little more complex. After an ES resolves a task into sub-tasks, it first finds some sub-tasks which it cannot solve and apply for aid to solve them, then it solves those sub-tasks which it can solve. This increases parallelism but wastes (computing) resource if the task needs only some of the results of sub-tasks.

Deferred application is directly opposite to precipitated application. After an ES resolves a task, it solves the sub-tasks which it can solve at once. Only then and only if the task has no result, the ES does apply for aid for the remaining subtasks. The advantage of deferred application is reduction of the waste of resources, and the disadvantage is reduction of the degree of parallelism of ES execution. If an ES can select the strategy of application flexibility, taking account of the manner of resolution of a task and the cost of solving sub-tasks, then waste of resources can be minimised and the degree of parallelism of ES execution can be maximised.

##### *The function.*

The function of applying for aid is to send an application sentence to the scheduler and subsequent action varies with the different strategies of application. For natural application, the ES continues to solve the unsolved (local) sub-tasks after it applies for a sub-task. It does not check the results of the sub-tasks applied for until no unsolved subtasks exist. When these results are obtained the ES uses them, otherwise the ES enters an interrupted state to wait for the result. For precipitated application, an ES begins to solve unsolved sub-tasks after it applies for all the subtasks it cannot solve. For deferred application, an ES enters an interrupted state to wait after it applies for all the sub-tasks it cannot solve.

##### *The content.*

The content of an application for aid is a subtask to be solved.

### **Using aid**

When an ES receives the results of sub-tasks it cannot solve locally, it uses them to continue its reasoning process.

#### *The strategies.*

There are three strategies to consider, corresponding to the three strategies of applying for aid, namely, natural use, precipitated use, and deferred use.

Natural use means that, when an ES meets a non-local sub-task (i.e., it cannot solve it), it immediately uses the result of the sub-task on the blackboard, when available.

Precipitated use means that the ES uses the results of its sub-tasks as they appear on the blackboard at once.

Deferred use means that the ES solves all the sub-tasks it can. Only if the task has no results, it then uses the results of other sub-tasks solved by other ES, when available.

#### *The function.*

The function of using aid is mainly to check the format of results of sub-tasks.

#### *The content.*

The content is the result of a sub-task.

### **Offering aid**

As well as applying to other ESs for aid, an ES can offer aid to other ESs.

#### *The strategies.*

The strategies for offering aid can be classified into three kinds, namely precedence aid, sequence aid, and initiative aid.

Precedence aid means that, when the ES accepts the sub-task entrusted to it by the scheduler, it solves this sub-task at once after interrupting the sub-task it is solving. When it has accomplished the new sub-task, it resumes the interrupted sub-task.

Sequence aid means that the ES does not solve the sub-tasks entrusted by the scheduler unless it has nothing to do.

Initiative aid means that the ES informs the scheduler when it has nothing to do and offers to do something.

#### *The function.*

The function of offering aid is to check the format of the entrusted sub-task first, then solve it, and finally report the result of the subtask to the blackboard manager.

#### *The content.*

The content in this case is limited to the data level. An ES aids other ESs only by returning results of sub-tasks. It cannot aid another ES by giving it knowledge or inferencing results of sub-tasks. It cannot aid another ES by giving it knowledge or inferencing methods. This would require transformation of heterogeneous knowledge and this is very difficult to do.

These cooperative approaches are realised by the front-end processors and the local ESs.

### 3.3.2 Efficiency enhancer

The efficiency enhancer improves the efficiency of ESs' network mainly by scheduling. Such scheduling is based on the fact that a given ES can solve several specific sub-tasks and some sub-tasks may be solvable by several ES's. There are many characteristics affecting this schedule. A very important consideration is to exploit parallel execution of ESs where possible. The choice of scheduling method is critical because we must take account of two facts (one sub-task related and one ES related):

- a) there are only a few sub-tasks which can be solved in parallel at a given time, and
- b) most sub-tasks can be solved only by one ES.

When a sub-task enters the READY state, the static scheduler is involved. It is governed by two principles:

- a) solve the sub-task with highest priority first;
- b) if a subtask can be solved by more than one ES, assign it to the ES with lightest load.

We can see from these two principles that the priority of a sub-task is a key element. It strongly affects the degree of parallelism in the execution of a job. The priority of a sub-task is determined by several factors: how many sub-tasks were interrupted by it. The priority is obtained by weighing and combining these factors. Principle a) embodies a method of improving the intrinsic parallelism of sub-tasks. Principle b) governs preparation for the dynamic scheduler and it embodies a method of improving the parallelism of ESs.

Normally, static scheduling which realises these two principles can be described by the following algorithms:

Algorithm STAT-SCHD

BEGIN

    Calculate the priority of each sub-task which has entered the READY state.

    Find all the ESs which can solve these READY subtasks and insert each into the waiting queue of each appropriate ES according to priority.

    For each sub-task

        Calculate the load on each ES which can solve it (e.g., based on the length of its queue).

        Select the ES whose load is lightest and delete this subtask from the queues of other ESs.

END

### 3.3.3 Dynamic scheduler

When an ES enters the INTERRUPT state, the dynamic scheduler is invoked. Note that the state INTERRUPT of an ES means, that no sub-task to be solved by the ES can be solved before the ES gets some information from the other nodes, i.e., the ES is available now to solve another new subtask from the scheduler. The dynamic scheduler then executes an algorithm such as follows.

**Algorithm DYN-SCHED****BEGIN**

If the queue of an ES is not empty,  
then the scheduler allows the sub-task at the front of the queue to seize this ES;  
If the queue is empty,  
then the scheduler searches the other queues for all sub-tasks which can be solved by this ES  
and allows the one with highest priority to seize this ES;  
Otherwise, the ES has to wait.

**END**

### **3.4 Some critical bottlenecks in the applications of cooperative and distributed expert systems**

With one of the fastest developments of technologies - communication networks, specialists now envision the possibility of digital connections not only among most telephones world-wide, but also among available hardware components having been applied to scientific fields, such as remote sensors, and so forth. Eventually, a network of approximately 50 billion components, all consisting of more or less intelligent agents, might be in operation [Page90] [Radley90]. Yet, it seems that very few have a full idea of how a reasonable usage of all this available information can be achieved. It is certain, however, that, with the right kind of data access and exchange and with an adequate array of methods from the many methodological frameworks, such as data analysis, pattern recognition, and statistics [Gaul88] [Bunke87] [Schader90], meaningful insight from these networks can be obtained in a sensible, socially acceptable way.

As Quine remarkably observed and his points are still significant today [Quine60], "the things in sharpest focus are the things that are public enough to be talked of publicly, ..., and near enough to sense to be quickly identified and learned by name or label; moreover, a common-sense talk of physical things often goes forward without benefit of explanations in more intimately experimental terms". "If we improve our understanding of ordinary talk of physical things; it will not be by reducing that talk to a more familiar idiom; there is none", Quine continued, "it will be by clarifying the connections, causal or otherwise, between ordinary talk of physical things and various further matters which in turn we grasp with help of ordinary talk of physical things".

Then, it is crucial to consider how to understand operators and operands that circulate communications. To this extent, effective knowledge representations, and automation of knowledge acquisition are indeed to be critical bottlenecks in applications of CDES. The following explains this based on [Liu93a] [Veer93].

The expert has built up expertise over a long period of task performance. Thus, the expert has the amount of experience necessary to be able to develop the insights into the area that result in heuristics.

The use of expert knowledge, judgement and experience is the key element in the performance of the domain-task being considered. The expert system development has, as its primary objective, the acquisition and implementation in a computer program of a portion of the knowledge, judgement and experience that an expert uses in the performance of a domain-task.

Experts exist in the domain. The domain must be established to the extent that there are recognised experts that solve the problem. The domain experts must be provably

better than amateurs in performing the task. If an area is too new or too quickly changing, there may no real experts.

The inputs needed to perform the task should be available to the system. If an expert uses inputs that are not documented or not 'computerised', such as local knowledge or informational knowledge, then ways must be identified to allow the expert system to have access to such knowledge.

The required outputs for the domain-task should be produced in the manner desired and delivered to the location desired. If the output is required to be in a form that may be difficult or very expensive for the expert system to produce or to deliver to the appropriate location, then this problem must be considered when evaluating the domain.

In short, there should be a strong managerial support from the domain area regarding the large commitment of CDES. To have the support, there is a need for a non-conventional approach. Consider using the technologies of ES when conventional programming (algorithmic) approaches to the task are not satisfactory. If a conventional approach will work well, then there is usually less technical risk to using it rather than an expert system approach. Note, however, that CDES may offer some additional advantages over conventional techniques, such as the opportunities expected ease of updating and maintaining a knowledge base and the ability to explain results from a variety of knowledge and information sources.

We shall come back to these topics in the following chapters. But for now, we need a basis on which a non-conventional approach can be found in a certain degree of depth in AI.

### 3.5 Strong AI or weak AI?

Searle published an article [Searle80] which created a considerable stir. In the article, he argues for a difference between what he called 'strong' and 'weak' AI. Searle characterises the claim of strong AI as follows:

"According to strong AI, the computer is not merely a tool in the study of the mind; rather, the appropriately programmed computer really is a mind, in the sense that computer given the right programs can literally be said to understand and have other cognitive states, ..., the programs are not mere tools that enable us to test psychological explanation; rather, the programs are themselves the explanations".

Searle dismisses strong AI as impossible on the grounds that the computer exhibits no 'understanding' or any other cognition state; hence, neither the computer nor the program we feed it have anything to do with human-intentionality. Searle makes this change extensively to all forms of Turing machine instantiation. What is more, he points out, is behind all work on strong AI as abiding dualism: i.e., the distinction between the program and its realisation in the hardware seems to be parallel operations between the level of the brain-operations. Intentionality was defined as mental states said to be directed at *objects* and *events* in the *external* world [Julie82] [Julie83]. It is this directedness that permits a distinction between, say, intentions. On such a view, intentionality encompasses diverse phenomena as wants, learning, remembering, knowledge, thinking, action, etc.

Nevertheless, it is the dualism just mentioned that is crucial to many methodological considerations of AI as a framework for cognitive science. Unfortunately, Searle does not tell us the specific nature of the weak AI program he suggests. He limits his observations to the following:

"According to the weak AI, the principle value of the computer in the study of the mind is that it gives us a very powerful tool" [Searle80].

However, these are clearly different events and the intentionalist makes no effort to tell us what makes them different. Should we assume that the procedural specificity of a programmer will provide the blueprint for drawing up the pertinent distinctions ?

But of course, the programmer must begin somewhere, in practice, a large pool of philosophical discourse or else introspection. Dennett characterises AI and philosophy as procedurally similar on the following grounds [Dennett81]:

- a) the same broad generations;
- b) the same blithe in difference to the hard-won data of the experimentalist;
- c) the same appeal to the deliverance of casual introspection and conceptual analysis;
- d) the reasonings about what is impossible in principle or what must be the case in psychology.

This is of course in sharp contrast to the experimentalist's view, characterised as 'bottom-up' - where complex phenomena are approached on the basis of reasonable well-established units discovered under simulation.

The top-down strategy is familiar to anyone acquainted with the procedures of computer science. Dennett illustrates it thus:

"The AI researcher starts with an intentionally characterised problem (e.g., how can I get the computer to understand questions in English?), breaks it down into sub-problems that are also intentionally characterised (e.g., how do I get the computer to recognise questions, distinguish subjects from predicates, ignore irrelevant parsings?)... until finally he reaches problem or task descriptions that are obviously mechanistic" [Dennett81].

Again, it is the task of the programmer to get the computer to 'recognise', 'ignore', 'distinguish', and so on; but it is the programmer also who solves the software-problems in advance as well.

It seems that the strong AI holds sway and continues to speak of learning and thinking as 'abstract symbol manipulation', perception as 'information processing', and so on. In Dennett's words, the programmer frankly views the computer anthropomorphically. And, yet, ironically, it is this anthropomorphism that accounts for the character of much AI work.

Some research trends have been seemingly evolute though. They do pay more and more attention to AI *applications*. For example, Marik and Peutrec independently approach to *integrative* distributed expert systems by adding many interactive notions to their autonomies for knowledge representation and acquisition [Marik92] [Peutrec94]; Also, software-production-management has been introduced into knowledge engineering, see [SEKE94].



But, does AI have anything to do with the concepts of decision making environments that we were talking about in the previous chapter? The author thinks it does.

Firstly, we have discussed nothing more than intentionality, distinction, distribution and integration of decision processes. And there is a kind of decision process which has to be carried out scientifically.

Secondly, intentionality seems to be the driven force to make any distinctions. For example, the observer, decision-maker and objective-modeler can view the systems which are similar to each other, but they can also view them from a quite different standpoint. Therefore, they have different concepts, methods, instruments and languages to interact with their external world.

Thirdly, computer seems to be the only substance that is helpful in playing a part *between them* for distribution and integration of data, information and knowledge. The topics of data, information and knowledge will be dealt in part II. But for now, it can be argued that the 'intelligent supports' should be featured by referential computing system which consists in switching the self-reference of such a system over to another reference. That is, of course, nothing but self-reference in disguise. This is indeed the very paradox that AI is called upon to take care of. Before we do so, it is advisable to keep two arguments in mind through the following two sections:

- as we have implied, P-D, P-R and O-O are referenced to each other;
- there can be different groups of people intending to interact with their own external environments; furthermore, P-D and P-R hold stronger 'reality' than the O-O.

Frame 3.1 below gives a short illustration. More explanations will be conducted later.

### Frame 3.1

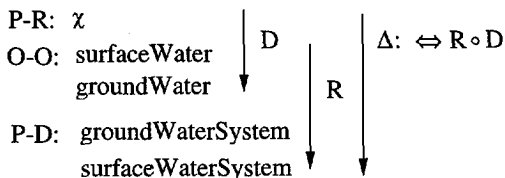
In P-D, the objects under study can be a system (or some components of a system), e.g.,

```
surfaceWaterSystem {reservoir, well}
groundWaterSystem {aquifer}
```

We may also have the following attributes:

```
surfaceWater {reservoir-capacity, pumpage-capacity, conduit-diameter}
groundWater {level-of-storage, pumping-storage, groundwater-salinity}
```

Furthermore, it can be assumed that surfaceWater and groundWater are the main impacts effecting all the kinds of water uses in O-O. Then, in general, we can have the following:



where,  $\chi$  is control vector;  $D$  is a naming relation;  $R$  is the relation of reference which gives information about what things a given construct is talking about at all. The inter-relationship in engineering practise is from an operation to the physical level, called denotation, arising from the composition of naming and reference:  $\Delta: R \circ D$ .

The logical structure induced by these three levels and their interconnections has, of course, correlated in the basic assumptions of both symbolic and hypothetical realism, as well as in common sense reasoning, which we have described in chapter 2.

### 3.6 The notion 'intelligence'

The author believes that Baecker, as a sociologist, has remarkably re-opened a debate on the subjects relevant to understanding the notion intelligence. His article - Intelligence and Referential System [Baecker94], seems to lighten the AI applications on one hand, but, on the other touches the nerve-end of AI foundation.

One of peculiar aspects of general and sociological systems research is that there seems to be no need to talk about intelligence. At the very moment of the introduction of self-reference into system analysis, the search for intelligence was suspended. To continue to speak of intelligence, however, is the hall-mark of any system analysis that avoids introducing self-reference. Biological and sociological systems theory, e.g. [Maturana80] [Luhmann90], seems to resist intelligence as insistently as artificial systems theory, e.g., [Newell72]. The former appears to say: life, thinking, and communication rely on unfoldment of self-reference, that is, paradox, whereas the latter keeps repeating: there is no way to run any computer software program as long as it risks hitting paradox.

There is, however, an occurrence of intelligence in sociological systems theory when intelligence is defined as an indication that it is impossible to observe how a self-referential system consulting itself chooses one and not another solution for a problem. Similarly, there keep popping up kinds of paradox in AI even whenever one has to admit a computer program somewhere that will only be displaced, but not removed, by debugging. And sometimes it happens that AI is taken as the design itself.

There are, of course, some ideas concerning intelligence which might prove to be helpful in showing us a way to develop a self-referential concept of intelligence. For instance,

- Ashby defines intelligence as "power of appropriate *selection*", and adds that it is not the sticking to a goal but the ability to set and change goals [Ashby81];
- Winograd and Flores understand that intelligence appears to consist in the avoidance of breakdown as well as in the handling of breakdown as soon as it takes place [Winograd86].

Here, in particular, the author considers 'appropriate selection' to be a very handy definition of intelligence that covers the setting of goals as well as the handling of breakdown, and try to adopt this definition in the philosophical notion of intelligence that attaches importance to the reflective capabilities of intelligence - intelligence appears to consist in a refraction of, and refraining from, much immediate access to whatever 'reality'.

There is some kind of hesitation involved in appropriate selection that we are only able to understand when we take reality to be not the reality but a reality watched by an observer (objective-modeler or decision-maker) who almost always checks on the reality he or she takes to be the reality.

Okuyama has added many useful systematic details in treating the notion 'reality'. In one of her latest articles [Okuyama94], it can be seen that, from the biological study on

the neurone system of organism, it has become clear that the neurone system is informationally closed and 'reality' is only staying in an individual. Moreover, 'reality testing' can be said 'reality consensus'. In other words, we all hold the field of not 'objective reality', but 'inter-subjective reality'.

However, she continues to show that this does not make 'reality' less important. She points out, in our daily life, 'reality' has a so important role that we never live in the society without 'inter-subjective reality'. She argues:

- a) Without the confidence of existence of object, we cannot communicate and cannot manage the daily life. Thus, changing the self-concept is through an ongoing process of communication between or among interaction of 'realities'.
- b) With regard to 'reality' in general, some 'reality' is very concrete and very strong, but some 'reality' is vague and not clear. For example, the present existence of stone is a 'strong reality', but the old history may be a 'weak reality'. Strong concrete reality seems to be the reality which gets almost all people's consensus. On the other hand, weak reality seems to be the reality which gets not many people's consensus.
- c) Strength of reality is the perception of individual. It can be said that one can create the confidence of reality through our experience. The confidence of reality is the continuum with two poles: very strong and very weak.

What we have learned here is

- psychotherapy is the effort to change the self-concept through on- going process of communication between or among therapist and clients. In other words, in psychotherapy, client and therapist co-operate to create the 'new reality' of self for the client, because clients have been exhausted in understanding his/her own 'reality' according to his/her own explanatory system.
- the sense of the communication is that the therapist interprets the text, which has been expressed by the client, into the therapist's explanatory system. Even when the response is just nodding, it is the expression of the 'therapist' s explanatory system.

### 3.7 Intelligent function and intelligent service

Back to our main issues which might lay some philosophical foundation for somewhat of intelligence to be built.

A distinguishing is made between an intelligence function and intelligent service. *Intelligent function* handles the exclusion problem in general. *Intelligent service* is specifically engaged in selecting excluded issues to be included. The intelligent function works on the problem of exclusion, whereas the intelligent service works on the problem of inclusion. They are like two sides of the same coin.

Nevertheless, there is no working on exclusion without including it, and no working on inclusion without considering it as exclusion in the first place. The difference between the two sides lies in the *selection* of a specificity that may itself be handled by the function which becomes, thereby, a service.

The main question seems to be how to understand operators and operands that circulate communications. To consider this, we can indeed take any operation to be an indication that relies on a distinction that distinguishes the indicated side of the distinction for everything else. Any distinction thus has two sides, a marked state and unmarked state. It is both sides we have to take together which constitutes the form of the distinction.

There might be only three ways to handle a distinction:

- a) one may call something, and thus confirm the distinction;
- b) one may cross from the marked state of the distinction to the unmarked state, and thus, cancel the distinction;
- c) one may re-entree the distinction into the indicated state, and thus, be able to observe (i.e., distinguishing) the form of the distinction - its two-sidedness.

There is not one intelligent operation. It takes at least two distinctions to re-enter one of them into the realm of the distinguished in order to be able to observe its form. That puts a limit to any attempt at implementing intelligence on a local and isolated level, as for example, in a computer.

### **3.8 Cases of the needs in altering knowledge-domains pragmatically**

In the previous section, we may have so broadly invited the notion 'intelligence' into our approach by stating that the basic intelligent operation - a pair of intelligent function and intelligent service performs as an inference of a marked state from an unmarked state, or a position of knowledge from a position of ignorance.

This section entails Frame 3.1. It shows that there are diverse knowledge-domains warranted to the decision processes in question. Then it is crucial to consider in which way that the knowledge should be adopted. Surely a line has to be drawn somewhere between the two extremes, i.e., "knowing 'everything' about 'nothing'" and "'nothing' about 'everything'", as Costanza and Sklar talk about [Costanza85]. The first refers to the use of all the observations by one relation to obtain a high accuracy and certainty, while the latter refers to the use of all observations by as many relations as possible. A decision maker cannot be studied on basis of the properties of all the cells of the body, nor can the function of a river be found through studies of the every single water-molecule. This will provoke the so called domain-driven object modelling to be introduced in chapter 4.

But for now, let us look at the following two simplified instances. The first instance shows that the process depends on the objects in P-D; and the second instance shows that the process also depends on the objects in P-R. More details can be reviewed in Appendix 4.

#### **3.8.1 Constraints of spatial properties**

Consider these criteria where groundwater is either the only source of water or the major component of a regional water supply. Further, we adopt an integrated groundwater-surfacewater system, or if an aquifer as a separate unit independent of a regional water

supply. This approach requires operating rules such that the system is operated in an optimal manner. These are generally determined through some economic or social objective associated with uses to which the water is put maximum yield, safe yield, or yield required to meet objectives.

### 3.8.2 Constraints of broad objectives

Very often, the objectives cited in the constraints of spatial properties and the manner in which they may be achieved are not in any apparent conflict with the broad objectives of social welfare. However, when reduction in natural discharge affects prior rights established on surfacewater resources or on spring flow, some conflicting factors arise.

Other conflicts may arise when two groups desire to use a common resource in different ways. This is exemplified by the inevitable conflict between holders of rights to pumpage and operators of a basin for storage and reservoir purposes. Thomas suggested a corporation policy for removal of this conflict [Thomas57]. Additional conflicts may occur when one group considers a certain management practice advantageous, whereas another group considers the same practice harmful. An example is the granting of additional rights to pump in a basin that is already over-appropriated. Control of groundwater storage may be succeeded by economic intervention, such as altering pumping tax, but the relatively large amounts of water in storage cannot be simply ignored. A precaution must be established to predict annual replenishment as the limiting factor in the potential growth of an area.

## 3.9 The notion 'object'

'Object-orientation' often assumes that 'the object' is itself something rather obvious, with well-known and well understood properties, see the surveys reported by [Brock90]. 'Object-orientation' continues to sweep through all area of knowledge encapsulation such as hydroinformatics [Abbott91].

However, in the article titled - 'The question concerning ethics, or: The metamorphosis of the object', Abbott argues:

"Under other disciplines, such as anthropology and psychology, the object is seen as multifaceted and multifarious, appearing in iridescent variety. In particular, in metaphysics, the relative stability in the view of the object that was established by Aristotle may be seen, in retrospect, to have crumbled already with the works of Descartes and Spinoza, to have been thoroughly cast into doubt by such as Locke, Berkeley and Hume, and to have been revised in principle already by Kant" [Abbott94b].

Furthermore, Abbott extensively questions the dimensions of knowledge, the intentionality of the objects and the particular need for a change of paradigm in the area of management of support systems.

It is argued here, that object-oriented *paradigm* enters in not only software engineering but also the field of knowledge representation [Marik92]; An object-oriented schema is about "knowledge concerning knowledge structure" [Klir91]; But, should whatever come out from these assumptions, it is often expected to be "chunks of

reasoning, language, memory and perception that ought to be larger and more structured, and their factual and procedural contents must be more intimately connected in order to explain the apparent power and speed of mental activities" [Minsky75].

While, in respect to system research, it is widely agreed that it is useful to take a starting point of a definition - 'object of study'. This has led many methods covering most areas depending on indicating material objects in the properties of which one is interested; in sociology one studies 'institutions' and their properties [Koppelaar76]; in other disciplines one studies 'populations', or 'points and lines' [Zeleny92] [Barrow91].

In particular, if we review frame 3.1, what more interesting is that a decision process never start with nothing or everything. Rather, an object under study is often dynamic, i.e., for one situation, some attributes are aggregated as an object to be studied; for other situations, a different collection of physical properties have to be considered in one class.

As an example, engineers cannot be experts in health and welfare, which follows from the contents of engineering curricula; but if we suppose engineers maintain or design sewage systems, certainly this requires that health be taken into account throughout the respective process. And if it is taken into account in an adequate manner, an engineer must know about 'health' in this context. His knowledge, however, is adapted from the sources that are outside the field of engineering.

In short, objects seem to emerge at the moment they are needed from the iteration of a large number of much simpler and familiar elements. The elements are or have to be working in concert with one another; they are creating environments where they are trying to be interpreted.

Nevertheless, it follows that, for a large application, object may be better understood by object-contexts, and object-contexts may be more usefully distinguished by an object that is grouped. A few instances are given below.

#### *Salience.*

Certain categories of objects or other perceptual entities are widely used by people, because they are salient, or natural groupings. The identities of these objects can be identified by their coherent clusters of features which appear in a variety of contexts. For example, it may not be necessary to define what physical object is, but simply to accept groupings of 'plant', 'water', and 'land'. Further employment of the groupings is in preference to arbitrary or bizarre categories such as 'Alfalfa, - a kind of crop - needs water there' may make a perfect sense.

#### *Hierarchy.*

One concept or class may include another, e.g., 'plant', 'crop', and 'Alfalfa'.

#### *Overlap.*

A given entity may be assigned to two or more conceptual classes which are not hierarchically related, e.g., 'river', 'water-authority'.

#### *Fuzziness of conceptual boundaries.*

The boundaries of conceptual categories are not usually sharply defined. In other words, the confidence with which objects may be assigned to a given category varies. For example, 'lake' is a category which shades into 'surfacewater-system' or 'water-storage'. It may be difficult to decide in which of these three categories a given stream belongs.

#### *Weighting attributes.*

Intuitively, some attributes are more significant in the development of classes than others. For example, 'water-flow' and 'wet' are weak determiners of the concept 'flood', while 'water-outpouring' and 'water-irruption' are relatively strong.

### **3.10 Pattern as a general representation of conceptual synthesis**

Norbert Wiener has stated on several occasions, e.g., [Wiener85], that one of the most interesting aspects of the world is that it can be considered to be made up of patterns. A pattern is essentially an arrangement. It is characterised by the order of the elements of which it is made rather than by the intrinsic nature of these elements.

It is useful to distinguish between man-made patterns and those created by nature. Man-made patterns often serve a purpose. Such a pattern might be connected with pattern syntactics, e.g., spoken or written language is such a case in the easiest understandable form.

The patterns formed by nature can be further distinguished between patterns formed by the inanimate world and those formed by biological systems.

As we have mentioned briefly at the beginning, we need to deal with the object of our investigation at two different levels.

At the first level, it is most natural to start with decision-processes which seem to be more easily describable. This leads us to start with patterns formed by the inanimate world, i.e., decision-making environments.

At the second level, we try to understand the creation of new qualities of a system by the cooperation of subsystems, where a human action (decision) is of a parameter of the quality. To explain this and to establish a link with pattern recognition, a simple example is given below.

Writing down the letters

a, a, h, i, i, m, n, s, s, t

does not make any sense. However, when we rearrange the letters into the sentence

"this is a man"

we immediately get the meaning of this sentence.

Thus, by the proper arrangement or, in other words, by corporation of the letters (subsystems), the total sentence acquires a new quality. The meaning of the sentence, i.e., the new quality, depends on the arrangement of subsystems, for instance a rearrangement into "is this a man" has changed the total meaning. On the other hand under certain conditions there are perturbations which do not destroy the original meaning. Thus, the meaning caused by the sequence of the individual letters determines the sentence in much the same way as the individual action.

Normally, *pattern recognition* deals with the problem of how certain objects, for instance letters, or objects in the landscape such as house, rivers etc. can be identified by machines.

However, according to the nature of the patterns to be recognised, we may divide the acts of recognition into two major types: the recognition of concrete items and recognition of abstract items.

We recognise characters, pictures, music, and the objects around us. This may be referred to as sensory recognition, which includes visual and aural pattern recognition. On the other hand, we can recognise an old argument, or a solution to a problem, with our eyes and ears closed. This process involves the recognition of abstract items and can be termed conceptual recognition, in contrast to visual pattern recognition. Obviously, we deal here with the second type of recognition.

We abuse pattern as a general representation to design machine intelligence in a pragmatic way. The ambition is a modest one but it attacks the very hard problems in the representations required, i.e., again, most of the methods delineates the soft, e.g., weakly structured or unstructured knowledge, or, in the composite, ignores the hard, e.g., data measurement or model constraint.

Since we are not really interested in object and object-context themselves, but the concerned phenomena to be represented, it does not seem to be necessary for us to make an exact division between objects in order to draw a conceptual schema merely.

However, being 'blind' to the 'true' inter-dependencies between objects and object-contexts does not seem to make object-oriented methodology less attractive, just like nobody knows whether unemployment is caused by inflation, or the other way around, or perhaps neither, yet there are many theories. But, a search for a higher level of systematic control over groups of objects is certainly necessary. We therefore abuse the ideas of 'patterns' to gain the descriptive power that we need.

The first is an interpretation of usual relations among actions (decision-making), actors (observer, objective-modeler and decision maker), objects (systems) and concepts (alternatives, values and attributes). Pattern is hidden (or partly hidden) relation among the information primitives. The other end of approach was the connectivist one, throwing away all hypothetically biased symbolic representations - as far as they could do this.

The second is that a pattern can also be a result of an experienced coherence. Learning coherence could be continued by learning coherence of events, i.e., different patterns. According to this view, there is a continuous development of pattern contexts to more complex patterns of pattern relations.

All methods of uncertainty calculations might be usefully treated as metapatterns, i.e., the statistics and event algebra-based probability, the set-based fuzzy, the overlapping hypothesis-based methods or any other. The model behind the method is just the pattern relations associated with one or the other typical experience. We name these patterns of pattern relations among metapatterns. The view is not only a nice attempt at some generation and naming, but also is an important pragmatic means for understanding and representing phenomena. If we take the methods resulting from the developments of pattern relations, i.e., metapatterns, we can have a much more liberal, pragmatic apprehension of problems in knowledge representation. Instead of dogmatic adherence to certain methods, we can have a choice related to subject of representation and an improvement of available tools.

The other advantage of the metapattern view is a free look for other pattern relations or combinations closer to the subject treated. An important caution to all these: it should not be mistaken by the considerations under the misconception of an advocate against well developed, professional methods of mathematics and logic in any case where they can do the job perfectly. But what essentially challenges here is that,



- a) the dissimilarities of mental patterns to visual ones are hardly supported by a software tool due to its 'non-algebraic' metric at the first place, and yet,
- b) more logic-based software methods hinder the further software developments being required, and thus provide little help for the software effectiveness.

The pattern-view is close to the fuzzy concept from the philosophical point of view, and to the connectivist methods from the viewpoint of representation. Furthermore, the inference philosophy is akin to case-based reasoning [Vamos91].

The main difficulty is hidden metric features in the pattern space.

A visual pattern may have a well-defined metrical in the space of geometrical dimensions in colour or grey scale.

While, patterns of general knowledge representations have an unclear structure. They have distances of explicit similarities and dissimilarities, but have a closeness to contexts of objects, because the contexts can be treated as variables to be augmented by things like viewpoints which are mostly time-dependent. The poverty of structure is related to the inside of individual patterns and to all combinatorial distances of patterns. However, this is again dealing with the weakness of conceptual structure, i.e., a definition of what is included and excluded by a given concept, while a refinement of a concept closer to the instantaneous status of knowledge, the objectives of processes and subjective interests is almost ignored.

Another usefulness of pattern-representation is dealing with the scaling problem by kinds of uncertainty estimations, memberships of fuzzy sets, beliefs, expectations, etc., where the scale is not that well fixed. For instance, time scale is another delicate issue. Most complex processes - e.g., economic, biological, meteorological - have not well-fixed time constants but the speed of similar transients is dependent on several hidden circumstances. These multiple confusions are burdened by the usual non-existence of sufficient amount of data for reliable statistics; estimations on distributions are weak; the same refers to the non-stationary behaviour of data and their statistical characteristics.

We shall show some applicable cases of patterns in chapter 8.



## **Part II**

### **Object-oriented techniques vs. the externals**



## A DOMAIN-DRIVEN OBJECT MODEL BASED ON GIS TECHNOLOGIES

---

In part II, we shall deal with the second question raised in the beginning of this dissertation, i.e., how to encapsulate existing and newly appearing tools into the relevant application-domain? Based on the connections we have made to the AI approach, we shall attempt to enhance the SE approach in order to improve the software effectiveness<sup>+</sup>.

As we have mentioned, the fast progress in computer technology has promoted GIS in the developments of spatial data base systems, of object-related map manipulation routines, and of high resolution visualisation techniques. In turn, these emerging facilities require object-oriented methods to evolve towards managing the diverse knowledge domains (see section 1.5 & 3.8, and frame 3.1), while still remaining advantageous to object-oriented software constructions.

In this chapter, our main purpose is to develop an *object-model* based on GIS technologies for decision support. The model consists of three dimensions, namely, spatiality, attribute and eventuality (the eventuality will not be discussed at length). We intend to characterise the use of such a model by an *order* of a sequence of the dimensions.

But before we embark on these topics, it is advisable to state the adopted principles in the object-oriented paradigm, and the differences between our methods characterised as domain-driven object-modelling and the methods which have been already well established in software engineering.

### 4.1 The adopted principles

The branch of philosophy of science that deals with modelling the existence of things in the world is ontology (or metaphysics). We turn, therefore, to ontology [Bunge77] [Pascoe86] to seek a base for the notion of object having been rather 'emotively' used without a sensible definition. The principles are:

---

<sup>+</sup> see section 1.1.2 for the term 'SE approach', and section 1.4.2 for the term 'software effectiveness'.

- The world is composed of things. Consequently, the sciences of reality (natural or social) study things, their properties and changes.
- Forms are properties of things. Firstly, we study and modify properties by examining things and forcing them to change; Secondly, properties are represented by predicates (e.g., functions) defined on domains that are, at least in part, sets of concrete objects.
- Things are grouped into systems or aggregates of interacting components. There is no thing that fails to be a part of at least one system. There are no independent things: the borders we trace between entities are often imaginary, e.g., the ones between physical, chemical, living and social system.
- Every thing changes.

A few terms which are importantly relevant to object-orientation are explained below.

#### *Attributes and properties.*

The world is viewed as composed of things of two kinds: concrete things that are called *entities* or substantial individuals, and *conceptual things*. The attributes of entities are observables. A distinction can be made between attributes and properties. An entity may have a property that is unknown to us. In contrast, an attribute is a feature assigned by us through a set of observables. Some of the attributes of an entity will reflect their properties. Indeed, we recognise properties only through attributes. Conceptual things are composite of the attributes of the entities.

#### *Changes and events.*

The knowledge of a thing requires information about how the states of the thing can change. When a thing undergoes a change, and if at least one property will have to change in value, *a change of a thing is manifested as a change of state*. It follows that, for a change to be possible, the thing has to have more than one state. A change of the state will be termed an *event*. An event can be described as  $a_1.a_2$  where  $a_1$  and  $a_2$  are the states before and after the change, respectively.

#### *Objects and states.*

Objects describe things through their attributes. The value of an attribute at a certain time is called a state variable. The set of state variables defines the state of the object.

#### *Action histories.*

An internal state of an object can be described by the following

- 1) An input domain,  $\phi$ , thought of as messages that can act upon the object.
- 2) An output domain,  $\psi$ , thought of as the responses of the object, which are either replies to messages or other messages sent.
- 3) An input/output relation,  $R$ , from domain  $\Phi$  to  $\psi$ , where  $\Phi$  is the set of elements of  $\phi$ .

Elements of relation  $R$  have the form  $(H, Y)$  with  $H = \dots a_n \cdot a_{n-1} \cdot a_{n-2} \dots a_2 \cdot a_1 \cdot a_0$ , where  $a_0$  is the current input,  $a_1$  is the previous input,  $a_2$  is the input before that, etc.; the dot represents concatenation, and  $Y$  is the output that the specifier considers correct for the input sequence  $\dots a_n \cdot a_{n-1} \cdot a_{n-2} \dots a_2 \cdot a_1 \cdot a_0$ . Such sequences will be referred to as *action histories*.

#### *Classes.*

A class defines the *behaviour* of similar objects by specifying the variables they contain and the methods available for responding to messages sent to them. However, to concentrate exclusively on the external behaviour of the objects, the specification of an object must abstract away from the variables the objects of a class contain. To satisfy this demand, the action history is often expected to be flexibly extended. For example,

- 1) The input domain  $\phi$  of the specification is taken to be the set of messages that the class defines. Hence, domain  $\Phi$  is the set of sequences of messages.
- 2) The output domain  $\psi$  of the specification is taken to be the set of responses that objects of the class can return method-inocations. The output of an object can be a *set of* messages to be sent to several other objects simultaneously; to deal with such occurrences, a theme is needed to define compound messages in a subsequent section.
- 3) The relations  $R$  of the specification is taken to define the link between the incoming messages and outgoing responses of the class being specified.

## 4.2 Domain driven object-oriented modelling

In order to grasp the essence of the modelling-driven approach, we intend to compare several well known technical themes to the ontological principles stated in the previous section. In chapter 7, we shall discuss a 'trade-off' between these two extremes.

There are five main categories of concepts that are attributed to the notion - object:

Data abstraction or encapsulation,  
Independence and persistence,  
Message passing,  
Inheritance,  
Homogeneity.

#### *Data abstraction.*

The behaviour of an object is encapsulated in its methods. Methods are mechanisms that have access to and can change the state of an object. Thus, an object type is described in terms of the form of its instances and the operations (methods) applicable to its instance variables. The instance variables, together with the methods, are called the properties of the object.

In the domain-driven approach, things must change and changes cannot be described separately from things. In this respect, encapsulation is a fundamental principle. However, the notion of methods or any other mechanism for changing the state of an object does not exist. Instead, the concept of a law defines the lawful or

'allowed' states (i.e., combinations of state variables). Laws are viewed as properties of things.

#### *Independence.*

The notion of independence incorporates two characteristics of an object. One is related to the state of the object and the other to its existence. Objects also have control over their own state. Once created, an object will continue to exist (or persist) even if its creator dies. Independence implies that the only way an object can change its state is through the actions of its own methods, namely, through its dynamic characteristics. Therefore, encapsulation is a necessary condition for independence.

In the domain-driven approach, the possible states of a thing are decided by the laws of the thing itself. There are no explicit mechanisms for the creation or destruction of things.

#### *Message passing.*

Objects can communicate only through message passing. A message can cause an object to 'behave' by invoking a method of an object. Thus, objects may affect each other through message passing and, since objects are independent, this is the only way they can do so. Indeed, methods can be viewed as definitions of responses to possible messages.

In the domain-driven approach, every thing is acted upon by other things. The effect of things on each other is manifested through the history of the things.

#### *Inheritance.*

Objects can be grouped together into classes. A class is a definition of an object type. All objects in a class have the same instance variables and methods and respond to the same messages. Objects can be categorised into subtypes or subclasses through specification. Specification forms an 'IS-A' or 'PART-OF' hierarchy. Objects in a subclass inherit all the instance variables and the methods. Multiple inheritance can exist; therefore, class hierarchy can be viewed as a directed acyclic graph. It is important to note that, in the object-oriented literature, class hierarchy is viewed as a mechanism to provide reusability of code and simplicity of programming.

In domain-driven approach, classes are defined on the notion of the scope of properties. The concept of a class is allocated to a kind by considering the combined scope of a set of properties, and further to a natural kind by considering the scope of a set of rules. Thus, both 'static' properties and 'dynamic' are used in the definition of a kind of a hierarchy.

#### *Homogeneity.*

Homogeneity implies that everything is an object. In particular, for complete homogeneity, messages and properties (instance variables) should be viewed as objects. Note that in both cases, this approach leads to circularity. For if messages are viewed as objects, then they should send messages to communicate with objects, and so on. Similarly, if attributes are objects, they should have their own attributes, which are objects, etc. Therefore, one has to stop arbitrarily at a certain level in order to break the circularity.

The domain-driven approach makes a clear distinction between things and their properties. In this sense, there is no complete homogeneity. Moreover, properties cannot



exist independent of things; hence, things are allowed to be composed of other things. This approach also makes a specific assumption that things exist that should not be further decomposed. Thus, the existence of a basic level is a principle, and not an arbitrary decision.

### 4.3 Introducing object-dimensions and a causative dimension

Here, it is assumed that an object has three dimensions: spatiality, attribute and eventuality. For the sake of simplicity, we mean *spatiality* to be a variable for accommodating different kinds of physical states; and mean *attribute* to be a variable for accommodating sets of attributes. Since both spatiality and attribute do not exist in a vacuum, they participate in relations which we have called states and events. Dimension 'eventuality' means a variable for accommodating a relationship between spatiality/attribute and state (event).

The orientations are certainly important at least in respect to the different environments we have. Many researchers suggest adding more dimensions to a construct of an object. For example, Dangermond found that the three dimensions seem to have inevitable ambiguity in applications [Dangermond83]. Another example is Langran's suggestion, i.e., a consideration of X and Y coordinates (and sometimes Z), together with time, as the axes of a three- (or four-) dimensional space [Langran92] (note, these suggestions are more or less bounded by a context of P-D).

To represent external variabilities of an object, we intend to use the following orderings,

spatiality-ordered	:	<spatiality, {attribute, eventuality}>
attribute-ordered	:	<attribute, {spatiality, eventuality}>
eventuality-ordered	:	<eventuality, {spatiality, attribute}>

What follows gives an example of forming two instances of this kind of object. For the simplicity of explanation, let us use the graphical symbols for describing spatiality-ordered objects, see Fig. 4.1 (note, similar signs are also applied to attribute-oriented objects).

According to [Bonvoisin93], the discharge of waste into a river network is normally restricted by a number of national, or perhaps even international laws. Observance of these laws is principally monitored by the Rivers Authority. Permission to discharge effluent to a water course or into *groundwater* is given in the form of a *discharge consent*.

Here, we consider a consent document. A piece of paper listing the conditions with which a discharge must comply. The top sheet of this document might contain the names and addresses of the consenter (discharger) and consenter (e.g., a river authority), a textual description of the consent and a reference identifier, see Fig. 4.2. In this context, the discharge itself and its characteristics are not of interest. What is of interest is that each discharge is monitored by sampling conditions that relate to it. The relations are of, in fact, an extent to the dimension of spatiality represented by Fig. 4.3.

More explanatory details are given shortly after.

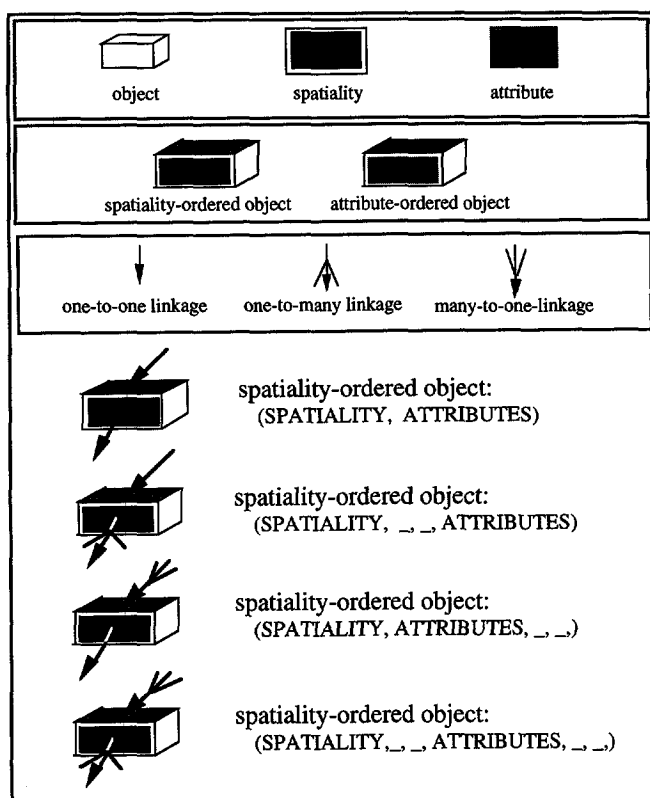


Fig. 4.1: graphical symbols illustrate spatiality-ordered objects

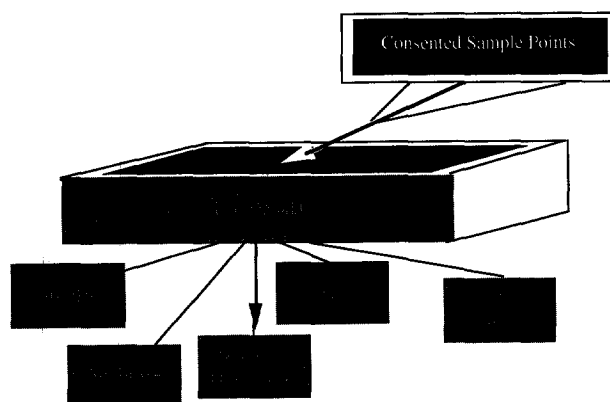


Fig. 4.2: an instance of attribute-ordered object

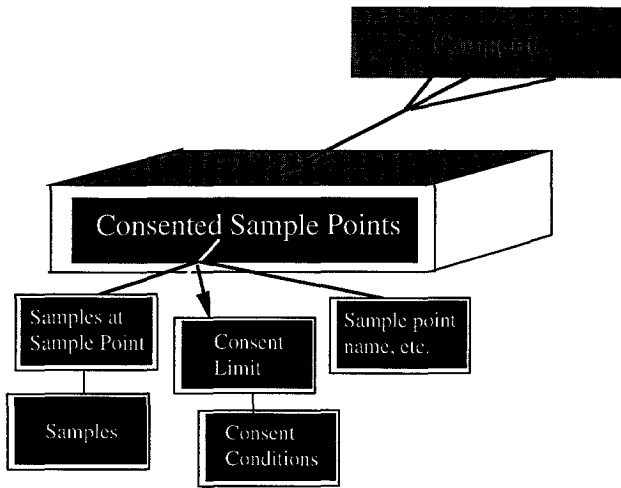


Fig. 4.3: an instance of spatiality-ordered object

#### Explanations of Fig. 4.2 & 4.3

A set of conditions defining what may be discharged *where*, *when* and in *what* amounts. Before granting a consent, the river authority first assesses whether the requested discharge will have a significant detrimental effect on the river network downstream of the discharge point. The combined effect of existing and proposed discharges is evaluated so as to assess its likely impact on the ecology of the water course and downstream abstractions. Potential breaches of legal restrictions on water quantity and quality, and implications for achieving water-related objectives, are also checked.

Having decided on the level of discharge which it will allow, the river authority issues a consent. A single consent document may cover many discharges, and will specify the quantity and quality conditions with which each of those discharges must comply. These conditions may be expressed descriptively, or numerically in terms of a specific measured variable, i.e., in the object-descriptive terms, *attribute*.

With regard to the relations between attribute and spatiality, it is worthwhile to mention that, relative conditions can be also used, with the attribute value for discharge at a downstream point called the reference site) not being permitted to differ from the value measured at an upstream point by more than a permitted amount, e.g., water temperature rise in a river.

A similar process of *authorisation* is followed when assessing a request to take water from a water course. In this case, an abstraction licence is issued, this time taking account of the reduced quantity of water downstream (or within an aquifer) available for the dilution of pollutants, for fisheries and for other downstream users. Having granted a discharge consent or an abstraction licence, the river authority is subsequently responsible for testing compliance with the conditions.

## 4.4 Spatial dimension of data model and digitised maps

A map generation is a process of transforming the real world into a map that is considered the cartographical conceptualisation and visualisation of reality. It is also known as cartographical abstraction. *It is the physical reality of our world compressed in a symbolic way.* In an artistic sense, the abstraction of reality is like a caricature in which certain features are emphasised and others are not in an attempt to present some particular aspect of geographical environment [Muehrcke78]. There are tools fully available to facilitate contour maps, colour coded maps, area statistics and map calculation.

Contour maps contain very comprehensive information about 2D variables. While the lines facilitate easy reading of the displayed variable's value at any point, also the direction and magnitude of its gradient can be seen. Two similar variables (e.g., two tables) can easily be compared by displaying their contour lines simultaneously on one map.

Colour coded maps can very efficiently convey information about 2D phenomenon. With an adequate selection of the colour palette, assignment of colours to values of the variable and the number of colours, interesting effects can be achieved. A gradually varying palette with many colours and a linear assignment to the values is well suited for displaying topography, geologic layers, etc. Note, the information is similar to contour maps. A palette of two colours, e.g., green and signal red can intuitively signal areas where certain constraints are not satisfied or violated.

Area statistics is a standard feature of any GIS. Arithmetic and Boolean operations on maps are well known characteristics of GIS. These operations require complex geometric intersection in vector based GIS such as ARC/INFO [Warwick94].

Generally, spatial knowledge can be classified in four classes: topological knowledge, metrical knowledge, vector knowledge or Euclidean knowledge. It appears that most of the properties used by the experts are topological properties, sometimes metrical ones while the initial description requires a Euclidean structure (because of the coordinates).

Nevertheless, in order to greatly stabilise a basis for establishing several experts systems that are collaborative (see section 3.3 & 3.4), it seems reasonable to accept that all the items (or entities) represented in a map are spatiality-ordered objects. In [Richardson93], Richardson significantly presents a model for spatial and thematic digital generation. By this work, it can be confirmed that a spatiality-ordered object in question is structurally static in nature; While an attribute-ordered object is dynamically virtual. As we have studied in chapter 2 and philosophically argued about the 'strong reality' and 'weak reality' in chapter 3, a water-body can be modelled as a water system whose spatial-characteristics seem to be the most interesting thing to us; it holds a 'strong reality'; thus we intend to label such a kind of entity by spatiality-ordered object. In contrast, however, the relations of concepts (or abstract entities) holds a 'weak reality'; in most situations, they can be only understood in individual context; we therefore label such a kind of entity by attribute-ordered objects.

Of course, there are relations between attribute-ordered object and spatiality-ordered object somewhere, otherwise, we would not have the management problems in the first place (the explanatory image has been given in terms of P-D, O-O and P-R). But, how should they be inter-related in a software system ?

## 4.5 Data models

The spatial data model is used as a conceptual tool for the representation of geographic information. This model featured the integration between classic abstraction primitives and spatial primitives; moreover, it has most of the features common to all object-oriented data models, such as object identity, subparts sharing, inheritance, and methods.

### 4.5.1 Spatial data model on the spatiality-dimension

#### Definition 4.1

A **class** is a general template to generate instances, that is, it describes the structural part (attributes) and the behavioural part (operations) for all the instances of the class itself. A class is specified with:

$\langle \text{class name} \rangle = \langle \text{superclass name} \rangle, \langle \text{attribute list} \rangle, \langle \text{operation list} \rangle,$   
 where  $\langle \text{superclass name} \rangle$  is the parent class which the class inherits from.  $\langle \text{attribute list} \rangle$  is  
 $(\langle \text{attribute} \rangle \{ , \langle \text{attribute} \rangle \})$ , and  $\langle \text{operation list} \rangle$  is  $(\langle \text{operation} \rangle \{ , \langle \text{operation} \rangle \})$ .

#### Definition 4.2

An **instance** is specified with an instance identifier, the class it belongs to, and an attribute value list:

$\langle \text{instance identifier} \rangle = \langle \text{its class} \rangle, \langle \text{attribute values list} \rangle,$  where  
 $\langle \text{attribute values list} \rangle$  is  $(\langle \text{attribute value} \rangle, \{ \langle \text{attribute value} \rangle \})$ .

The instance identifier uniquely identifies an object. We make distinction between standard classes and application-dependent classes.

#### Definition 4.3

**Application-dependent class** are objects defined by the schema designer and have a strong structure. Also, when a user browses over a map all the objects that are grouped are of application-dependent classes. Application-dependent class is therefore scalar valued.

Application-dependent class is formed by three basic classes of geographical type: Point, Line and Area ( Polygon ).

#### Definition 4.4

**Standard classes** are aggregates of instances of application-dependent classes and sets of value - integer, real, string, Boolean, e.g.,

$\{ ( \text{Country: Holland} ), ( \text{City: Delft} ), ( \text{Canal: } \_ ), 15, \text{Clean} \}.$

Standard classes are provided by a model. Their instances are scalar valued at first, then set-valued. However, we can consider the class Set or Sub-class to be scalar-valued for building a homogeneous aggregate, e.g., Set ( City ).

**Examples**

Classes: State, Country, City, Lake, Sea, River, Road, etc.

Representations:

```
State =      geographicArea,
            ( name :      String,
              region :    Area,
              population : Integer,
              capital :   City,
              counties :  Set ( Counties ) );
```

.....

```
waterResources = typesOfWaterBody,
                 ( name :      string,
                   river :     line,
                   lake :      area,
                   ...
```

**4.5.2 Topological operations on the attribute-dimension****Definition 4.5**

**Attributes** make up the structural part of an object. An attribute of a class has a name and a domain from which it takes its attribute values; such a domain is the extension (i.e., all the instances) of a class.

< attribute name > : < attribute domain >

An attribute value is an instance of the class specified in the attribute definition. Inside instances, attributes are assigned a value by:

< attribute name > = < attribute value >

where < attribute value > is an instance identifier. The construct to refer to an attribute value of an instance is ( < instance identifier >, < attribute name > ).

**Definitions 4.6**

**Operations** associated to an object makes up its behavioural part. An operation is implemented as a method associated to a class: the method can be applied to all the instances of that class and return another instance of a certain class. Methods have a method name, and optionally can have parameters, that are other instances used during the computation. Therefore an operation is defined by:

< method > : < result >, where < result > is a class name, and < method > is defined by:  
 < method name > [ ( parameter > : < class name > { , parameter > : < class name > } ) ]

The topological aspects of spatial objects represent important knowledge that users of GIS expect to retrieve from a spatial databases. Topological knowledge can be easily modelled in terms of spatial relationships, e.g., 'crosses', 'is inside', 'is adjacent'.

**Definition 4.7**

**Topological relationships** are characterised by the property of being preserved under topological transformations, such as translation, rotation, and scaling.

Topological relationships can be grouped together into basic positions, such as touched, in, crossed, overlapped and disjointed. These basic positions are quite suitable

for end-users as part of a spatial-query strategy. A simple set-up for this is to provide relationship-names that reasonably have an intuitive meaning for users of spatial applications. Such a name-driven informational entry is clean, because this approach described so far restricts any concept that cannot be 'seen' on a map. Therefore, users can directly browse over a map and get a straightforward answer (not for everything, but for spatial information).

#### **Example**

< I1, touch, I2 >, where I1 and I2 indicate two instances of application-dependent classes. It is meant that there are two geometric elements that touch each other, if the only thing they have in common is contained in the union of their boundaries.

### **4.5.3 Spatial-operations on the attribute-dimension**

#### **Line**

IF applied to a connected area with no holes,  
THEN it returns the circular line representing the boundary of the area.

#### **f: Point, t: Point**

IF applied to a non-circular line,  
THEN they return the two separate end-points making up the boundary of the line (f and t stand for 'from' and 'to' respectively).

#### **Distance(point1: Point, point2: Point): Real**

Applied to a network of lines representing rivers, channels, roads, railways, etc., in order to find the distance between two fixed points in the network, by following the shortest connections.

#### **After(point1: Point): Point**

This operation deals with the concept of direction. It can be applied to a line with points lying on it and ordered along the line itself. For example, if there is an exit of a stream flowing in a certain direction, the next exit with respect to the given one is:

( stream ( #, \_ ), after ( exit ( #, \_ ) ), where '#,\_' stands for a missing number.

## **4.6 A simple object-calculus**

The object calculus is a formal query language for object-oriented databases. It is based on the spatial data model and it allows the formulation of spatial queries against a geographical database. The result of a query is an aggregate of instances. The formulation of a query is carried out by giving properties that a group of instances of the database must satisfy; we adopt the general notation  $\{ i \mid f(i) \}$ , where  $i$  is the *target variable* and  $f(i)$  is a well-formed formula (or short formula) of the object calculus.

Formulae are made up of atoms connected by logical operators *and* ( $\wedge$ ), *or* ( $\vee$ ).

### **4.6.1 Basic notations**

Formulae depend on the notion of variable. Variables of the calculus are only instances (indicated by  $i, i1, i2, \dots$ ) and they may be bounded or unbounded: the only unbounded variable in a formula is the target variable, while other variables are supposed to be

existentially quantified. Constants of the calculus may be instances (indicated by I, I1, I2, ...). A, A1, A2, ... denote aggregates, r denotes any relationship between instances. Parameters in a method may be either variables or constants. The notation  $m(\alpha)$  denotes a method with parameters. The symbol m denotes either a method without parameters or an attribute of an instance. We use the notation  $\varphi$ , which has the following recursive definition:

$$\varphi = \begin{cases} I \\ i \\ (\varphi, m) \\ (\varphi, m(\varphi, \alpha)) \end{cases}$$

In other words,  $\varphi$  may be either a constant, a variable, or a construct of the spatial data model (attribute or operation, for examples, see section 4.4), returning an instance and possibly containing variables.

Also, consider a set of relational operators:

$$\Theta = \{ >, <, \geq, \leq, =, == \}.$$

The '==' may be used for testing object identity equality. The remaining operators apply only to scalar valued instances.

#### 4.6.2 Atoms

Atoms allowed in the calculus are defined as follows, where  $\vartheta$  represents any operator belonging to  $\Theta$ .

- 1)  $C(i)$
- 2)  $\varphi \in A$
- 3)  $< \varphi_1, r, \varphi_2 >$
- 4)  $\varphi_1 \vartheta \varphi_2$

Atoms of the kind 1) are called *range atoms* and they impose that the variable  $i$  ranges over instances of the class  $C$ .

Atoms of the kind 2) are called *set inclusion atoms* and they affirm an instance belongs to an aggregate.

Atoms of the kind 3) are called *relationship atoms* and they involve various kinds of relationships.

Atoms of the kind 4) are called *comparison atoms* and they express comparisons among instances.



### 4.6.3 Well-formed formulas (wff)

The following recursive definition holds:

- 1) an atom is a wff;
- 2) if  $f_1, f_2$  are wff, then  $f_1 \vee f_2$  is a wff;
- 3) if  $f_1, f_2$  are wff, then  $f_1 \wedge f_2$  is a wff;
- 4) if  $f_1$  is a wff, then  $(f_1)$  is a wff.

In the following, we examine various cases of aggregation to exemplify several kinds of formulae.

#### *Range atoms.*

An aggregate of the type  $\{i \mid C(i)\}$  represent the set of all the instances of a class C. For example,  $\{i \mid \text{State}(i)\}$  is the set of all states present in the extension of the class State. To specify a homogeneous aggregate, only one range atom must exist for the target variable  $\{i \mid C(i) \wedge f_1\}$ . If the aggregate contains instances of two different classes  $C_1, C_2$ , we write  $\{i \mid C_1(i) \vee C_2(i) \wedge f_1\}$ .

#### *Set inclusion atoms.*

It is useful to build aggregates starting from already defined aggregates;

If  $A_1 = \{i \mid f_1\}$ , then  $A_2$  can be defined by reusing  $A_1$ , i.e., for example,  $A_2 = \{i \mid i \in A_1 \vee f_2\}$ .

#### *Relationships atoms.*

They allow us to formulate queries involving topological relationships. We may propose several examples of such queries in order to illustrate the spatial application of the query language.

#### *Comparison atoms.*

They are used whenever a query imposes conditions on the attributes of an object or on the result of operations. An example is the aggregate concerning states with a population greater than 10.000.000 inhabitants:

$\{i \mid \text{State}(i) \wedge ((i, \text{population}) > 10.000.000)\}$ .

All the cities at a distance of 100 miles or less from Amsterdam can be expressed as:

$\{i \mid \text{City}(i) \wedge ((\text{intercities}, \text{distance}(i, \text{Amsterdam})) \leq 100)\}$ .

## 4.7 Concluding remarks

- 1) Most software tools used in constructing GIS enable us to flexibly represent the basic objects described above. However, when employed for the same task, conventional means of knowledge representations can become very static.

**Example**

Consider the following representation:

three different classes QUADRANGLE, RECTANGLE and SQUARE, abbreviated as Q, R, and S, respectively. Clearly, S is-a R and R is-a Q. A quadrangle could be spatially represented by four points, say, `upright_point ( ur_X , ur_Y )`, `up_left_point ( ul_X , ul_Y )`, `downright_point ( dr_X , dr_Y )`, and `downleft_point ( dl_X , dl_Y )`; then, a rectangle can be spatially represented by two points, say, `up_right_point ( ur_X , ur_Y )` and `downleft_point ( dl_X , dl_Y )`; and a square can be spatially represented by `( dl_X , ur_Y )` and `( dl_X , dl_Y )` plus the X coordinate of the `upright_point ( ur_X )`. What about a `downleft_point` and length?!

- 2) We need to appropriately represent those objects that hold 'strong reality'. Here, we intend to associate operations on spatiality-ordered object with a domain of topology. The usefulness can be implied by the following example.

**Example**

A fire station may be represented in a map; but there would be no need for us to know its physical properties rather than the amount of water available in a certain period of time (e.g., a year). In this case, the functions of the entity are the main concern (attribute-ordered). While on the other hand, if a new big fire station is to be built, its allocation may have to be carefully considered. In this context, the entity (fire station) is spatially important not only because of itself (i.e., spatiality-ordered), but also because of other spatial-objects involved (e.g., a piece of land).

- 3) In addition to the basic tools described in the previous chapter, GIS is capable of coupling with expert systems for decision support. For example, polygons can be a simple method of regionalisation of point related data. This method is reportedly popular for regionalisation of precipitation measurements. An estimate of available groundwater volume can easily be computed from a map of groundwater heads, storage coefficient and saturated thickness. Another example is rule-based combination of maps. GRASS has an expert system module that can evaluate a set of rules with raster maps as objects and generate a new raster map with the conclusion.

## DATA ENTITY AND A FUNCTION OF MULTIPLY INDEXING IDENTIFIERS

---

On several occasions<sup>+</sup>, we have stated the importance of data management and stressed that tools should be sensitive to data models.

Fortunately, we note that, on a methodological level, much attention has been paid lately to couple distributed heterogeneous databases with model-based systems, e.g., [Blanning91] [Krishman91]. In particular, Krishman (1991) intends to develop so called ASCED (Advanced System for Computations in Engineering Design) to identify the operations, such as refinement and integration, to provide a collection of operators that reduce the effort required by modelers to perform them. They show that, analogously, database management and model management share very similar characteristics. Conceptual schema integration is a problem which has been studied extensively by the database community and has direct corollaries for model integration [Peckham88] [Lockemann91]. Although 'models as data' is old hat by now, as Dolk would say, database theory (at least relational theory) still seems powerful enough to provide a convenient metaphor for data management [Dolk93].

However, in the circumstances, illustrated by example 2.1, Fig. 4.2 and Fig. 4.3, the 'object under study'<sup>++</sup> is often to *split* a large number of elements into a number of 'homogeneous' groups on the basis of multivariate observations. Methods and algorithms can be found in any up-to-date literature relevant to describing these kinds of processes, e.g., [Shahin95] [Jakeman93a] [Jakeman93b], to name just a few. Therefore, we only introduce the concept of 'data-entity' in order to describe the contexts of using the entity.

Accordingly, an object that describes the features of a set of observations cannot be formed until a data entity is available. Therefore, the entity has to be dynamically stored in a database, i.e., retrieval of the entity is dependent on a context where the entity is to be used. We then need a multiple-index function for a data entity to access (or to be accessed by) relational databases.

### 5.1 The use of data entity

Data entity is a subjective term: it emerges from use of data. There are several reasons for this (for details, see [Liu94d]).

---

<sup>+</sup> see section 4.3, 2.3.1, 2.1, 1.5.2 & 1.6.

<sup>++</sup> see section 3.9 for the term 'object under study'.

First, the decision-maker, observer, or objective-modeler uses data to serve their desires to understand, utilise, explain or accept a function of a water-resource system. So, data entity is subjective to data validity (being available for what).

Second, a data entity is subject to its presentation. A set of data may appear to be significant to an observer, but utterly nonessential to a decision-maker or objective-modeler.

Third, as an extension of the second standpoint, a data entity is subject to a knowledge domain. One may ignore data in the belief that they are totally irrelevant, but this might be because that person is lacking in the relevant knowledge, or deliberately doing so in order to obtain the answer that they prefer.

What follows is to classify data entities in pragmatic terms, i.e., the contexts where a data entity shows its significance. Here we consider *support data* and *property data*. Usual contexts of using these two data categories can be reviewed in Appendix 5.

#### *Support data.*

The access type for support data is mostly pattern-matching. A typical access involves finding a particular configuration point and the particular instrument associated with it. Proximity search is sometimes needed. For example, if it is highly arguable whether a certain water-resource system should be continually abused for a particular use, the configuration data may be consulted to find another water-resource system in a neighbouring location.

The data modelling requirements are fairly conventional, i.e., modelling of entities that have hierarchical or network relationships. Generalisation is an important modelling tool. For example, a conventional modelling requirement is geographical modelling of configuration data. In many cases, the surface is quite regular and could probably be modelled with simple types (points, lines, arcs, etc.). However, geometric shapes may be complex enough to require special modelling techniques similar to those required in engineering databases.

Another major requirement is for the support of historical data.

Data changes continuously over time, and the history of changes has to be recorded. In addition, operational scenarios, such as when a new policy has been imposed, how the effected system changes at the time, etc., need to be recorded. The history of configuration data changes also needs to be recorded, but not as often as instrumentation data, because they usually occur only between experiments.

#### *Property data.*

Since property data is a summary of many experiments and contain general knowledge of a subject area, it contains a summary of information extracted from external interactions. Different access types are needed. Pattern-matching once again should drive a specific entry of a location of data.

A range query could be used to find a desirable subset of the data. A proximity search will locate entries with properties as close as possible to the specified parameters. In the cases studied, it is not possible to issue ad-hoc queries or to browse the databases for information since the databases are not available on-line. An on-line version should provide such facilities.

The databases are organised logically as entities with mostly hierarchical relationships between them. There are some network relationships (permitting a many-to-many association between the entities).

Property data contain graphs and test which complicate their management a great deal. They also have to have a representation for special symbols that are unique to the scientific field.

In conclusion, the main difficulties in managing property data stem from the diversity of data. This is probably the reason that only special purpose database systems have been developed for the different disciplines. Once conventional data management systems do combinations of numeric, character, text, and graphs data, most data lose their contexts for further use.

## 5.2 An example of allocating a data entity in a window

Assume that a batch of data from two variables has been obtained, thus, a two dimensional plot of the points can be drawn with respect to time; each point indicating the values of each of the measured variables at a specific time instant. A complete plot of the batch of data points then shows the possible existence of clusters of points.

Clusters of points in a data plot arise when a number of points lie closely together and under such circumstances, the aspects denoted by the variables under consideration can be regarded as most likely to be operating under very similar conditions for all points within the cluster. This can be broadened to realise that points within a cluster indicate the same type of features that might be found, thus allowing the centre of the cluster to be a good overall measure of those aspects. Intuitively, for cases where more than one cluster is found, each cluster may represent another class of aspects.

The following algorithm itself works by placing a search window (at equal sample intervals in the data space) of pre-defined size,

$$[x_k - a_{x_k}, x_k + a_{x_k}]$$

for  $k = 1 \dots$  number of dimensions

where  $x_k$  is the centre of the window on the  $x_k$ -axis

$a_{x_k}$  is the pre-defined offset for  $x_k$ -axis.

The centre of gravity of this window can be then calculated by carrying out a test for each data strip  $d_i$  in the data set to see whether or not it lies within the boundaries of the window range specified. The procedure is as follows:

```

...
c = 0;
sumk = 0;
for i = 1 ... n
{
    if ( di,k ∈ [ xk - axk, xk + axk ] ∀ k )
    {
        sumk = sumk + di,k ;
        c = c + 1
    }
}

```

where  $n$  is the number of samples in the data set  
 $c$  is the number of data points found within the boundaries of the window.

The window is then recanted accordingly, thus  $x_k = \text{sum}_k/c$ .

Such a procedure is repeated until the centre of gravity converges to a constant and the window no longer moves or until a pre-defined iteration limit is reached. At this point, the window has reached a local maximum, having moved along increasing points of density gradients.

### 5.3 Identifiers of data-entities accessing databases

Identifiers in scientific databases are typically multi-dimensional. An important issue is the efficient storage and access of identifiers of data which are affected by the regularity, density and time-variation properties.

Identifiers whose dimensions have a regular structure are quite common. The main reason is that simpler algorithms can be developed for them, and that data can be organised in an orderly fashion. The simplest case exists when the configuration of the experiment or simulation forms a multi-dimensional mesh. In such a case, there is no need to store the identifiers of the data because the position of each data point can be calculated using the 'array linearization'. Indeed, the array capabilities of programming languages have been used extensively by scientific applications. This suggests that an array linearization access method would be most desirable in a scientific data management system. The advantage of such an access method is that it requires no storage for the identifiers and provides a very efficient random access to the data points.

At first glance, it seems that identifiers that consist of irregular dimensions, such as the numbers identifying soil types in a region, have to be explicitly stored with each corresponding data value. Such an approach wastes space since each dimension value has to be repeatedly stored with the data. Rather, the irregular dimensions can be enumerated and stored only once. Thereafter, the identifiers can be calculated using array linearization over the enumeration.

Irregular dimensions are most common in statistical databases, where the enumeration of each dimension and array linearization over them is a most effective method.

For our applications, pattern-matching leading to proximity search of data is important. Pattern-matchings derive the need to access specific data points randomly. To accommodate such a requirement, some kind of indexing or hashing technique of the multi-dimension of the data is required. The mapping of multi-dimensional space to linear space discussed above (e.g., array linearization) provides only a key-to-address mapping that is equivalent to hashing.

Thus, it is expected that the patterns could support proximity search of data. This is still hopeful, because it is possible to abuse a pattern to preserve logical locality in physical storage when points are logically close to each other in the multi-dimensional space. This suggests the organisation of physical storage into cells along the dimensions of the identifier. The data points within a cell will satisfy the proximity requirement. For elements on the borders of cells it is necessary to access adjacent cells, and therefore the placement of cells in physical storage is also important. The mapping of multi-dimensional space to linear space mentioned above may work significantly with such a cellular organisation, because it does not seem to disturb the logical proximity of the data points.

An arbitrary hash mapping would place data points into cells which would not necessarily preserve logical proximity.

Range queries over experiment data do not seem to be as important in supporting external interactions. Nevertheless, the cell organisation should benefit range access because the data are organised according to the dimensions for which ranges are specified.

Another important point is made relative to sequences of queries. It is often the case that the identifier(s) of the current query are close to the identifier(s) of the previous query.

## 5.4 Multiply indexing an identifier of a data entity

As we have indicated in the above section, there are many ways of characterising what constitutes scientific databases. However, one property that seems to be common to much scientific data is the frequent use of subscripted identifiers, or names.

Throughout this section, we use names or identifiers as synonymous. In a sense, this is fortuitous because one of the more difficult problems in implementing persistent databases is the simple shortage of appropriate names for identifying multi-dimensional data. Subscription is a technique for extending a name space. In traditional usage, the name describes the nature of entity being identified and the subscript distinguishes between different instances of this kind of entity. Most often we use non-negative integers as subscripts as in

simulation<sub>1</sub>, simulation<sub>2</sub>, ..., simulation<sub>k</sub>,

where the subscripts are used to denote data associated with the first simulation, the second simulation, and the k-th such simulation.

Positive integers are common subscripts, but other subscript domains are possible. Mathematicians frequently subscript with alphabetic characters as in

simulation<sub>14/12/88</sub>, simulation<sub>3/1/89</sub>, ..., simulation<sub>3/12/89</sub>

Here the subscript is used to distinguish simulation instances in a non-consecutive manner.

All algebraic programming languages support subscription - it is essential to support array manipulation. But there is usually a subtle semantic addition. A name such as 'x[3,4]' does denote one instance of an x-like entity, the x<sub>3,4</sub>-th. It also serves to denote the storage location of the instance relative to a base location denoted by the name x itself. This leads to extremely efficient data accessing mechanisms which require only a simple mathematical calculation to derive the location of the desired element by giving the location of the base identifier.

However, efficient use of subscripting to denote storage location also requires the relatively static placement of the data on a single storage device. But, if we continue to consider managing data to support optimal process, a parallel computing environment is the most advantageous.

In a parallel processing environment, where spreading various pieces of an array across distinct storage devices for use by different processors is needed, these elements

may have to be migrated locally. The traditional implementation of subscript notion is no longer efficient.

## 5.5 A pattern-based identifier access

We shall use the subscript domain

natural-nbrs = {0, 1, 2, 3, ...}

which is always available for subscripting. A collection of subscripted instances is declared by indicating the valid subscript domain(s), as in

**simulation[natural\_nbrs] instantiates\_a simulation**

This declares that the simulation[2] will denote a simulation, as well as simulation[9], or simulation[123000]. Note that no upper bound was specified in the declaration above. The name simulation together with any positive integer subscript will denote a SIMULATION. A fairly conventional, but unimaginative, implementation of this syntax is shown in Fig. 5.1. Here, the base name, simulation, is first found in a dictionary of persistent names. Then, subscript is used to access the correct storage pointer in the subscript index associated with entry.

Note that since we are examining the representation of persistent scientific data in a databases environment, all pointers must be interpreted in the storage medium. In particular, they are not memory locations. They may be disk block addresses in a static storage environment, or symbolic pointers in a dynamic storage environment. It is difficult to avoid using storage pointers in all but the simplest sequential file representations. But extensive use will seriously degrade data access performance.

A straightforward sequential file of blocks of simulation items would require no storage pointers. The name simulation could be used to identify the file. But, random access to such a sequential file requires computing the length of each simulation-data-set (in bytes) to calculate the corresponding offset. Further, it precludes distribution of the data set as might be described in a parallel database environment.

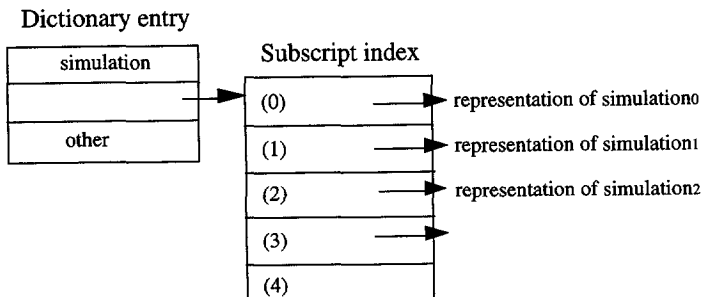


Fig. 5.1: conventional implementation of singly subscripted data



The structure of Fig. 5.1 is a more flexible kind of dope vector representation. But even this fundamental kind of storage structure has problems. Since no upper bound has been specified in the simulation declaration, we must assume that the subscript index is a linearly linked sequence of data blocks.

If we are storing data on thousands of simulations, access to simulation[11403] can be quite tedious. Moreover, if non-consecutive subscripts are needed, as in a sparse vector, then a heavy storage and access overhead can be incurred.

What both of these traditional approaches implement is a single valued location function,  $\text{loc}(\langle \text{identifier} \rangle, \langle \text{subscript} \rangle)$ , of two variables - a base identifier and an integer subscript. If we focus on this functional view, and ignore the array interpretation, any number of such accessing loc functions come to mind; for example, the two preceding implementations, or tree access, or hashing. The important criteria are that this loc function should be efficient, and flexible.

### 5.5.1 Multiple subscripts

In this sub-section, we look at doubly subscripted names, such as the elements of a matrix,  $x$ . We might declare this by

```
x[natural_nbrs, natural_nbrs]
instantiates _a REAL_MATRIX_ELEMENT
```

Employment of either a row or column major storage representation converts the pair of subscripts into a single integer offset; But to do so, they require upper bounds for all but one of the subscript domains. To implement variable dimensioning, and thus avoid the need for a priori specification of upper (and lower) bounds on the subscript domains, we may extend the dope vector representation of Fig. 5.1, so that it has a hierarchical index containing pointers, because this generalised dope vector, which is quite effective for accessing memory resident data, employs many many pointers. Their use with large, persistent, multiply subscripted disk resident data sets is relatively inefficient. Our goal, instead, will be to develop a function which converts a pair of subscripts into a single unique integer that can then be used as argument to the loc function.

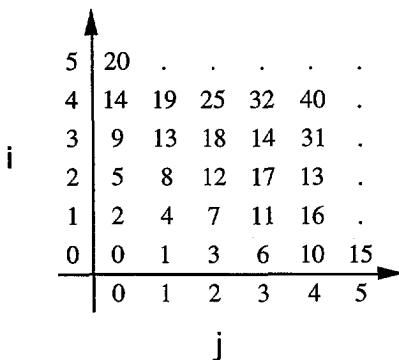


Fig. 5.2: diagonalization of  $N \times N$

Let  $N$  denote the natural numbers. There are several well known diagonalization processes, e.g., map the Cartesian product  $N \times N$  in a one-to-one fashion onto  $N$ . Such an one is illustrated in Fig. 5.2, which includes just a small portion of the entire unbounded lattice.

A general expression for this diagonalization function is

$$n = \text{diag}(i, j) = i + \sum_{r=0}^{i+j} r = i + \frac{(i+j)(i+j+1)}{2} \quad \text{Eq. 5.2}$$

Even if the subscripts are known to be bounded, use of this diagonalization formula to provide an offset into a block of contiguous storage would be disastrous. The offset in Fig. 5.2 of  $x_{2,3}$  would be 16, the offset of  $x_{4,4}$  would be 40. Nearly 50% of the contiguous storage will remain unused. Such an overhead is intolerable.

Instead, we let  $n$  derived from formula Eq. 5.2 serve as the integer argument to an appropriate location function,  $\text{loc}(\langle \text{base}, \text{identifier} \rangle, n)$  as described in the preceding sub-section.

In Fig. 5.3 we graphically illustrate this procedure with a triangle to symbolise the  $\text{loc}$  function, because in Fig. 5.1 we have been closer to an implementation of the  $\text{loc}$  function with a tree. Note that the dictionary entry for the  $\langle \text{base}, \text{identifier} \rangle$  points to the tree;  $n$  acts as the search key within the tree.

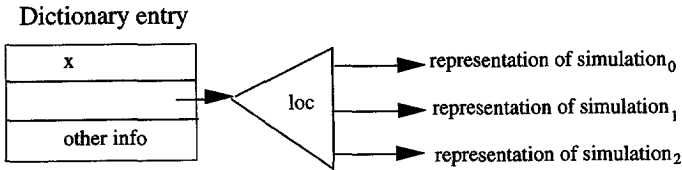


Fig. 5.3: the implementation of subscripted names

With this implementation, only subscript combinations that have actually been used generate entities in the tree and subsequently correspond to representations in the data space.

Now, when an identifier is encountered, such as  $x[2,3]$ , database system looks  $x$  up in the dictionary. If the name is declared to be subscripted, it uses all the  $k$  actual subscript values in the appropriate formula to obtain  $n$ . The  $n$  is used as the integer argument to the subscript function  $\text{loc}$ , to obtain a pointer to the current representation of that element. Note that using this representation, sparsely subscripted name spaces are accessed with the same efficiency as dense ones.

### 5.5.2 Keys and sequences of binary numbers

What we have described so far is an exploration of functional subscripting. It depends on a functional  $\text{loc}$  operation, symbolised by the triangle in Fig 5.3, which gave a single integer value for a unique storage pointer to be returned.

Readily, if functional subscripting is to be successful, a dynamic loc function that is both time and space efficient can be used. There are many ways of implementing dynamic functions. The most effective methods, to date, have employed either trees or hashing. So we could abuse the triangle in Fig. 5.3 to denote a B-tree similar to the one in [Bassiouni85].

Another alternative might be to let the triangle denote a hashing operator, which hashes the integer into pointer storage. Extendible hashing [Fagin79] [Larson88] will handle arbitrarily large sets. But our need is not only for retaining keys in the target space to resolve collisions that reduce the space efficiency, but also the need for a logically consecutive target space that may reduce its value in parallel databases implementations.

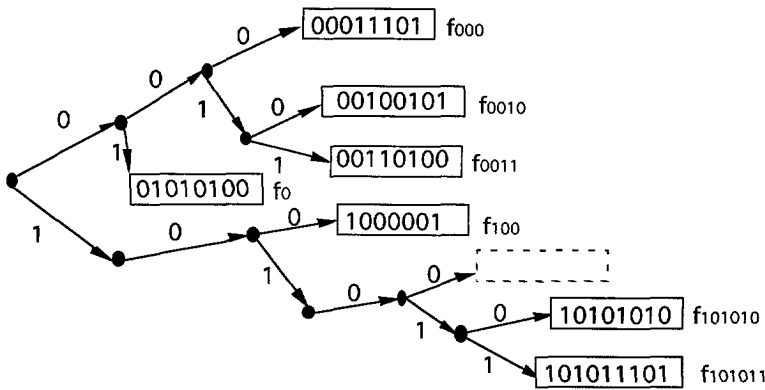


Fig. 5.4: modified B-tree

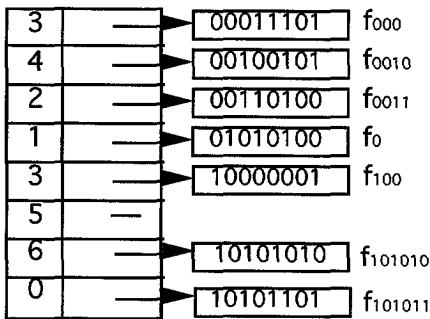


Fig. 5.5: the compact representation of the modified B-tree of Fig. 5.4

A further implementation of loc operator could be done by using a modified B-tree structure developed in [Orlandic89]. This B-tree behaves much like the 'better' B-tree except that it has the following remarkable property. Each key of arbitrary type and length (up to 254 bits, or 31 bytes) in the index is replaced by a single 8 bit type. Here 7 functional values,  $f_{key}$ , have been associated with 7 binary keys in such a way that these

values constitute the leaves whose access paths are prefixes of their respective keys, see Fig. 5.4.

Then, it can be shown that in any pre-order traversal of modified B-tree, every leaf (except the last) must be followed by a 1-node. In the compact representation, the key of each leaf is denoted by the depth of the 1-node that follows it in the pre-order sequences as Fig. 5.5 shows.

Surprisingly, we are able to use this compact representation for search and retrieval. Given a key  $k$ , we begin any search by creating an integer vector  $B[]$  that indicates the position of all 1 bits in  $k$ . For example, if  $k = 00110100$ ,  $B_k[]$  would be  $\langle 3, 4, 6, 9 \rangle$ . To ensure termination of the search algorithm, a final integer which exceeds the length of any key in the key space, in this case 9, must be appended to sequence  $B_k[]$ . The following single algorithm compares the sequence  $B_k[]$  of one bits in the key against the sequence  $D[]$  of 1-node depths in the index block to determine which entry points to the leaf in which  $k$  will be found - if it exists in the tree at all.

```

procedure search(B, D, bindex)
  integer B[], D[], bindex;
  { B is an array denoting the position of 1 bits in k }
  { D is the sequence of 1-node depths in the index block }

  var dindex: integer;
  begin
    dindex <- 1;
    while B[Bindex] <= D[dindex] do
      begin
        if B[Bindex] = D[dindex]
          then bindex <- bindex + 1;
          dindex <- dindex + 1;
        end;
      end;
    return dindex;
  end.

```

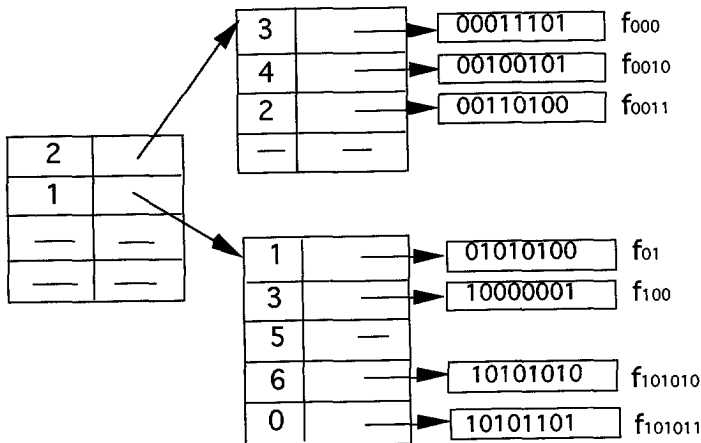


Fig 5.6: compact representation in a parallel manner

The argument  $B_{index}$  denotes which 1 bit in  $B_k[]$  is to be used to begin the comparisons in the block. Initially  $B_{index}$  will be 1. Note that search through an index block using this procedure is sequential, as are many B-tree search procedures, except that the comparison at each entry is always between short 8 bit entries, regardless of the initial key type, or length.

Fig. 5.6 shows the compact representation in Fig. 5.5 in a parallel manner.

### 5.5.3 Subscripted attributes

As in many databases systems, actual values here could be represented as attributes associated with individual entities. Thus, for example, if condition and velocity are attributes, the following can be declared:

```
STRING_ATTRIBUTE is a ATTRIBUTE
with image STRING
REAL_ATTRIBUTE is a ATTRIBUTE with image REAL
condition instantiates_a STRING_ATTRIBUTE
velocity instantiates_a REAL_ATTRIBUTE
```

STRING\_ATTRIBUTE and REAL\_ATTRIBUTE are the names of generic classes of single valued attribute functions. Conditions and velocity are specific instances respectively of these two kinds of attributes. Given a database entity  $x$ , on which the attributes condition and velocity are defined, then  $x$  condition and  $x$  velocity will denote the string and real values that represent the current condition and velocity of the entity  $x$ .

One way of defining a matrix is to declare it to be a set of doubly subscripted entities,  $x[1,1]$ ,  $x[1,2]$ , ...,  $x[4,4]$  on each of which is defined an attribute  $val$  that denotes the current value of that matrix element. So, for example, we might let the following expression denote the sum of the values along the main diagonal of a  $4 \times 4$  matrix  $x$ :

$$\text{trace} = x[1,1].val + x[2,2].val + x[3,3].val + x[4,4].val$$

The support will be given by creating a structure, such as Fig. 5.3 to represent the multiply subscripted entity,  $x$ . It will also implement the attribute  $val$  with a function index. Fig. 5.7 illustrates the implementation of such a subscripted velocity attribute function. In an expression, such as  $x.velocity[k]$ , the unique id associated with the entity  $x$  is provided to the correct function index velocity obtained from the subscript index that represents the subscripted dictionary entity velocity.

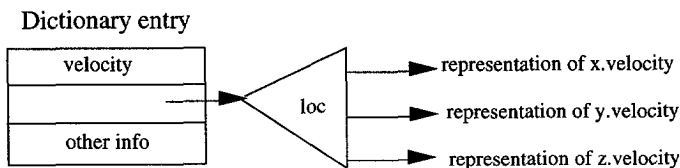


Fig. 5.7: a representation of a subscripted function, *velocity*

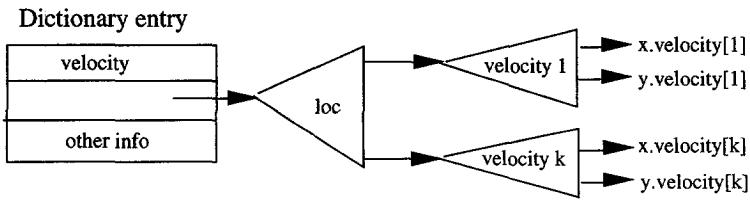


Fig. 5.8: representation of a doubly subscripted function,  $V$

Then, to evaluate an expression, such as  $x[2,2].val$ , the unique id of the element  $x[2,2]$  obtained from its subscript index is provided as the key to the  $val$  function index.

There are advantages, however, to define the matrix  $x$  to be a single entity with a set of doubly subscripted attributes, say,  $v[1,1]$ ,  $v[1,2]$ , ...,  $v[4,4]$  declared over it. Then the expression above would be written as

$$trace = x.v[1,1] + x.v[2,2] + x.v[3,3] + x.v[4,4]$$

This allows cleaner class declarations as in the following:

```
v[natural_nbr, natural_nbr] instantiates_a REAL_ATTRIBUTE
REAL_4×4 is a class
  having elements = {v[1..4, 1..4]}
x instantiates_a REAL_4×4_MATRIX
y instantiates_a REAL_4×4_MATRIX
```

The complete matrix  $x$  (or  $y$ ) is denoted by a single identifier not a collection of subscripted element identifiers. It is implemented by the structure shown in Fig. 5.8.

## KNOWLEDGE ACQUISITION FOR EMBEDDED MEANINGS OF OBJECTS

In this chapter, we shall discuss knowledge acquisition which was introduced as a critical bottleneck in building expert systems (see section 3.4).

In the past few years, many researchers have focused on the development of methodologies to improve knowledge acquisition processes by actively eliciting knowledge from people and then generating knowledge bases usable by computing systems, e.g., [Walz93].

We shall suggest a method of repertory grid, by which a set of elements across the top of the grid is listed for representing the objects that need to be classified. Various combinations of the elements can then be presented to a network of computing.

### 6.1 A repertory grid-method

The knowledge acquisition being reported here employs the repertory grid test originally developed by Kelly (1955), and the work ETS [Boose85], KNACK [Diederich87], NICOD [Ford91a], ICONKAT [Ford91b], and KSSO [Gaines87]. These systems significantly reduce the time spent in eliciting expertise from domain-experts; moreover errors in constructing knowledge bases are also reduced.

Table 6.1: rating scale

- 1 The object = {spatially ordered, attribute ordered} is very likely to have the opposite characteristic of the trait (i.e., its order is likely to be switched);
- 2 The object may have the opposite characteristics of the trait;
- 3 Unknown;
- 4 The object may have the trait;
- 5 The object is very likely to have the trait.

In a repertory grid, the trait is established by multi-users' different viewpoints of calculating procedures. The trait is listed on the left hand side of the grid, and its opposite is listed on the right hand side. Each pair of a trait and their opposite are called a *construct*. After the set of constructs is ready, the grid is filled with ratings. A 5-scale rating mechanism is usual in filling the grid, i.e., each rating is an integer ranging from 1 to 5 as Table 6.1 illustrates.

The whole concept of repertory grid can be exemplified as follows:

- Step 1: Elicit all of *elements* related to the object from the users, e.g.,  $E_1, E_2, E_3, E_4$  and  $E_5$  and place them across the top of a grid.
- Step 2: Elicit constructs (traits and their opposites) from the users. Each time three elements are chosen to allow users to manipulate them to enable a construct to distinguish one element from the other two;
- Step 3: Rate all of the <element, construct> entries of the grid. The value range from 5 to 1 whose meanings are illustrated in Table 6.2. An example is given by Table 6.3.
- Step 4: Generate rules from the repertory grid.

Table 6.2: value range

- 1 the element is very likely to have the trait  $C_i$
- 2 the element may have the trait  $C_i$
- 3 unknown;
- 4 the element may have the trait  $C_j$
- 5 the element is very likely to have the trait  $C_j$

Table 6.3: an example of value range

	$E_1$	$E_2$	$E_3$	$E_4$	$E_5$	
$C_1$	5	1	5	1	1	$C'_1$
$C_2$	4	4	4	1	4	$C'_2$
$C_3$	1	4	5	1	4	$C'_3$
$C_4$	1	1	1	5	1	$C'_4$

## 6.2 Embedded meanings and uncertainty

Embedded meanings represent the information that domain experts take for granted but are implicit to those who are not familiar with the application domain.

Embedded meanings are likely to be ignored during the processes of knowledge acquisition. This is the reason why most expert systems may fail to reach a conclusion during a decision process if the premise parts of available information are only partially matched.

### 6.2.1 Attribute ordering table (AOT)

Attribute Ordering Table (AOT) indicates the degree of importance each attribute has to each element.

The value of each AOT <attribute, object> entry may be labelled by 'X', 'D' or an integer number. 'X' means that the attribute has no relationship with the object. 'D' means that the attribute dominates the object, i.e., if the attribute is not equal to the entry value, it is impossible for the object to be implied. For those entries which are not labelled 'X' or 'D', integer numbers are used to represent the relative degrees of importance attributes have to the object of each column. An integer number means that the attribute does not dominate the object but is of some degree of importance relative to



other attributes. A larger integer number implies the attribute is more important to the object.

Consider the multi-type repertory grid shown in Table 6.4. Each entry of Table 6.4 consists of two parts: a value and a rating of certainty. The value part may be an integer, a real, a symbol, a set of values (e.g., { 3 , 5 , 8 } represents 'the value is equal to 3, 5, or 8'), a range of values (e.g., ( 7 , 16 ] means '7 < value ≤ 16'), or 'X' to represent 'UNKNOWN'. The rating part of the entry may be 1 or 2 (1: may be, 2: is most likely to be). If the rating of an <attribute, object> value is not depicted explicitly, the default value is 2. Table 6.5 is an example of an AOT derived from Table 6.4.

Table 6.4: an acquisition-table

	Obj <sub>1</sub>	Obj <sub>2</sub>	Obj <sub>3</sub>	Obj <sub>4</sub>	Obj <sub>5</sub>
A <sub>1</sub>	{3,5,8},2	20,2	(7-16),2	17,2	3.2
A <sub>2</sub>	YES,1	NO,2	YES,1	YES,2	NO,2
A <sub>3</sub>	X	X	4.3,2	2.1,2	6.0,2

Table 6.5: an example of an AOT derived from Table 6.4

	Obj <sub>1</sub>	Obj <sub>2</sub>	Obj <sub>3</sub>	Obj <sub>4</sub>	Obj <sub>5</sub>
A <sub>1</sub>	D	D	2	1	D
A <sub>2</sub>	1	1	1	D	D
A <sub>3</sub>	X	X	D	1	D

The third column of Table 6.4 expresses the following meanings:

1. A<sub>3</sub> dominates Obj<sub>3</sub>: if A<sub>3</sub> is not equal to 4.3, it is impossible for Obj<sub>3</sub> to be implied.
2. A<sub>1</sub> does not dominate Obj<sub>3</sub>: if A<sub>1</sub> < 7 or A<sub>1</sub> > 16, it is still possible for Obj<sub>3</sub> to be implied.
3. A<sub>2</sub> does not dominate Obj<sub>3</sub>: if A<sub>2</sub> ≠ YES, it is still possible for Obj<sub>3</sub> to be implied.
4. A<sub>1</sub> is more important to Obj<sub>3</sub> than A<sub>2</sub>, since  
AOT[ Obj<sub>3</sub> , A<sub>1</sub> ] > AOT[ Obj<sub>3</sub> , A<sub>2</sub> ]: the possibility of implying Obj<sub>3</sub> by negating ( A<sub>2</sub> = YES ), since the negation of the former lessens the degree of certainty more than that of the latter.

### 6.2.2 Eliciting embedded meanings

The AOT table can be used to capture embedded meanings from the original rules. For each rule, the attribute of each predicate is checked. If the associative AOT entry is labelled 'D', the predicate remains the same. The predicates whose related AOT entries are integer numbers will be selected by any possible combination. The negation of each selected predicate causes a generation of a new rule for each combination. The generated

rules are called the embedded rules. For example, the embedded rules generated from RULES<sub>3</sub> may contain the negated forms of  $T < A_1 \leq 16$  or  $A_2 = \text{YES}$  or both; therefore, the following additional rules are generated:

$$\neg (7 < A_1 \leq 16) \wedge (A_2 = \text{YES}) \wedge (A_3 = 4.3) \\ \rightarrow \text{GOAL} = \text{Obj}_3$$

$$(7 < A_1 \leq 16) \wedge \neg (A_2 = \text{YES}) \wedge (A_3 = 4.3) \\ \rightarrow \text{GOAL} = \text{Obj}_3$$

$$\neg (7 < A_1 \leq 16) \wedge \neg (A_2 = \text{YES}) \wedge (A_3 = 4.3) \\ \rightarrow \text{GOAL} = \text{Obj}_3$$

Table 6.6: illustration of intra-ART and inter-ART,  
(a) the intra-attribute-relation table

	values of GRID[i, k]			
	'U'	'X'	value, 1	value, 2
$\text{AOT}_{i,k} = 'X'$	Q		Q	Q
$\text{AOT}_{i,k} = 'D'$	Q	Q		
$\text{AOT}_{i,k} = \text{'integer'}$	Q	Q		

(b) the inter-attribute-relation table

	$C_{i,k} > C_{j,k}$	$C_{i,k} < C_{j,k}$
$\text{AOT}_{i,k} > \text{AOT}_{j,k}$		Q
$\text{AOT}_{i,k} = \text{AOT}_{j,k}$		
$\text{AOT}_{i,k} < \text{AOT}_{j,k}$	Q	

### 6.3 Knowledge validation in eliciting embedded meanings

As each portion of the expertise is represented by an ordered pair of tables (a repertory grid and an AOT) instead of a single repertory grid only, we can question about the existence of inconsistent information between the pair of tables. Without loss of generality, each repertory grid with multiple trait types is represented as

$$\text{GRID}[i, k] = (V_{i,k}, C_{i,k}),$$

where  $V_{i,k}$  is the value of attribute  $i$  if object  $k$  is the goal and  $C_{i,k}$  is the degree of certainty (1: maybe; 2: most likely to be) of filling  $V_{i,k}$ . Moreover,  $\text{AOT}_{i,k}$  represents  $\text{AOT}[\text{Attribute } i, \text{Object } k]$  in the following discussions.

The relationships between the ratings of a repertory grid and the entry values of an AOT can be represented by an intra-attribute-relation table (intra-ART) and an inter-attribute-relation table (inter-ART) as Table 6.6 illustrates.

The intra-ART records all of the possible relationships between the values of repertory grid and the corresponding values of AOT for the same attribute. The inter-ART records the possible relationships between the attributes by comparing their corresponding values of repertory grid and AOT, respectively.

The Q's in intra-ART and inter-ART are used to indicate the situations where inconsistency may occur. Since the consistency of those unmarked situations is obvious, we are only concerned with the marked ones here, as follows.

**Case 1:** ( GRID[ i , k ]  $\neq$  'U' ) and ( GRID[ i , k ]  $\neq$  'X' ) and  
( AOT<sub>i,k</sub>  $\neq$  'X' ).

Let  $\rightarrow$  stand for the 'support' relation (e.g.,  $A \rightarrow B$  means that the premise A can be an evidence to support B to be the goal), a situation can be represented as

$$(A_i \rightarrow \text{Obj}_k) \wedge \neg(A_i \rightarrow \text{Obj}_k) = \phi.$$

which leads to a contradiction (by filling an 'X' into AOT). A similar situation happens if ( GRID[ i , k ] = 'X' ) and ( AOT<sub>i,k</sub> = integer ).

**Case 2:** ( GRID[ i , k ] = 'U' ) and ( AOT<sub>i,k</sub> is an integer ).  
This expression is equal to

$$((A_i \rightarrow \text{Obj}_k) \vee \neg(A_i \rightarrow \text{Obj}_k)) \wedge (A_i \rightarrow \text{Obj}_k) \\ = (A_i \rightarrow \text{Obj}_k).$$

In this situation, there is no logical contradiction. For example, the actual value of  $V_{i,k}$  may be unknown, but the attribute is of some degree of importance to support (or negate) the object.

**Case 3:** ( GRID[ i , k ] = 'U' ) and ( AOT<sub>i,k</sub> = 'X' ).  
The expression is equal to

$$((A_i \rightarrow \text{Obj}_k) \vee \neg(A_i \rightarrow \text{Obj}_k)) \wedge \neg(A_i \rightarrow \text{Obj}_k) \\ = \neg(A_i \rightarrow \text{Obj}_k).$$

It can be seen that no contradiction occurs. For example, the value of GRID[ i , k ] may be thought to be undecidable; therefore, the attribute is of no importance in classifying the object.

**Case 4:** ( GRID[ i , k ] = 'U' and ( AOT<sub>i,k</sub> = 'D' ).  
The expression is equal to

$$((A_i \rightarrow \text{Obj}_k) \vee \neg(A_i \rightarrow \text{Obj}_k)) \wedge (\neg A_i \rightarrow \neg \text{Obj}_k) \\ = (\neg A_i \rightarrow \neg \text{Obj}_k).$$

Logically, there is no contradiction; however, as GRID[ i , k ] is undecidable or unknown, it is impossible to know when the attribute negates the object. Therefore, in this case, it would be better to show a warning message to verify the values. Even

though the value is stated definitely to be correct, attribute  $i$  still cannot be used in generating the rule to imply object  $k$ .

Before we discuss the inter-attribute relations, it should be noted that the occurrences of the following combinations are not possible if no intra-attribute inconsistency (Case 1) exists:

$$(AOT_{i,k} = 'X' \text{ or } AOT_{j,k} = 'X') \wedge (C_{i,k} > C_{j,k}),$$

$$(AOT_{i,k} = 'X' \text{ or } AOT_{j,k} = 'X') \wedge (C_{i,k} = C_{j,k}),$$

$$(AOT_{i,k} = 'X' \text{ or } AOT_{j,k} = 'X') \wedge (C_{i,k} < C_{j,k}).$$

This is because the known values of  $C_{i,k}$  and  $C_{j,k}$  imply that both  $GRID[i, k]$  and  $GRID[j, k]$  are not 'X' or 'U'. Besides, the 'D' value in AOT can be treated as zero (more important than other integers); therefore, the following situations are discussed.

$$(AOT_{i,k} = 'D' \wedge AOT_{j,k} = \text{integer}) \wedge (C_{i,k} > C_{j,k}),$$

$$(AOT_{i,k} = 'D' \wedge AOT_{j,k} = \text{integer}) \wedge (C_{i,k} = C_{j,k}),$$

$$(AOT_{i,k} = 'D' \wedge AOT_{j,k} = \text{integer}) \wedge (C_{i,k} < C_{j,k}),$$

$$(AOT_{i,k} = \text{integer} \wedge AOT_{j,k} = 'D') \wedge (C_{i,k} > C_{j,k}),$$

$$(AOT_{i,k} = \text{integer} \wedge AOT_{j,k} = 'D') \wedge (C_{i,k} = C_{j,k}),$$

$$(AOT_{i,k} = \text{integer} \wedge AOT_{j,k} = 'D') \wedge (C_{i,k} < C_{j,k}).$$

**Case 5:**  $(C_{i,k} < C_{j,k})$  and  $(AOT_{i,k} > AOT_{j,k})$

Assume that  $i$  and  $j$  are two distinct attributes, then we have

$$\begin{aligned} & ((A_i \rightarrow Obj_k) \rightarrow (A_j \rightarrow Obj_k)) \wedge \neg((A_i \rightarrow Obj_k) \rightarrow (A_j \rightarrow Obj_k)) \\ & = \phi. \end{aligned}$$

It seems that a contradiction exists; however, if attribute  $i$  can be used to distinguish more objects from object  $k$  than attribute  $j$ , the situation may become acceptable:

$$\begin{aligned} & ((A_i \rightarrow Obj_k) \rightarrow (A_j \rightarrow Obj_k)) \wedge ((A_i \rightarrow Obj_k) \rightarrow (A_j \rightarrow Obj_k)) \\ & = ((A_i \rightarrow Obj_k) \rightarrow (A_j \rightarrow Obj_k)) \end{aligned}$$

Consider the repertory grid and the AOT given in Table 6.7; it seems that  $A_1$  provides more certain evidence to support  $Obj_1$  than  $A_2$ . However,  $A_1$  can be used to distinguish  $Obj_2$ ,  $Obj_3$  and  $Obj_5$  from  $Obj_1$ , while  $A_2$  can only distinguish  $Obj_3$  and  $Obj_5$  from  $Obj_1$ ; therefore,  $A_1$  may be more important to  $Obj_1$  than  $A_2$ , and that confutes the aforesaid hypothesis.

Table 6.7: an illustration of inter-attribute relation,  
(a) an example of acquisition table

	Obj <sub>1</sub>	Obj <sub>2</sub>	Obj <sub>3</sub>	Obj <sub>4</sub>	Obj <sub>5</sub>
A <sub>1</sub>	2	D	1	2	1
A <sub>2</sub>	1	1	X	1	2
A <sub>3</sub>	X	2	2	2	3

(b) the corresponding AOT

	Obj <sub>1</sub>	Obj <sub>2</sub>	Obj <sub>3</sub>	Obj <sub>4</sub>	Obj <sub>5</sub>
A <sub>1</sub>	A,1	B,1	B,1	A,1	B,2
A <sub>2</sub>	YES,2	YES,2	NO,1	YES,2	YES,2
A <sub>3</sub>	X	10,2	13,2	10,2	20,2

To sum up, there are two types of inconsistent condition that may exist between a repertory grid and its corresponding AOT:

TYPE-1:

(( GRID[ i , j ] ≠ ( 'U' or 'X' ) ) and ( AOT<sub>i,k</sub> = 'X' ) ) or  
(( GRID[ i , j ] = 'X' ) and ( AOT<sub>i,j</sub> = integer ) ).

TYPE-2:

( C<sub>i,k</sub> < C<sub>j,k</sub> ) and ( AOT<sub>i,k</sub> > AOT<sub>j,k</sub> )

(except that attribute i can be used to distinguish more objects from k than attribute j).

Moreover, it would be better to give some warning messages if Case 2 or Case 4 occurs. The next section will introduce some strategies of resolving the inconsistency during the process of knowledge acquisition.

## 6.4 A consistency test for knowledge validation

The checking and resolving of TYPE-1 inconsistency is straightforward, and hence its discussion is omitted here. To resolve TYPE-2 inconsistency, some measure for the amount of information provided by each attribute is needed.

For example, assume that C<sub>i,k</sub> < C<sub>j,k</sub> and AOT<sub>i,k</sub> < AOT<sub>j,k</sub>, we require a way to tell how many objects are classified from Obj<sub>k</sub> by A<sub>i</sub> and A<sub>j</sub> (and how sure the classification is).

The existence of inconsistency can then be confirmed. Therefore, a circumstance influential function which sums up the degree of certainty for each object classified from Obj<sub>k</sub> by A<sub>i</sub> is defined as follows:

$$\text{INFL}(i, k) = \sum_{v_{it} \neq v_{ik}} \alpha(C_{i,t}),$$

where  $\alpha$  is a mapping function which maps  $C_{i,t}$  to a degree of certainty.

One of the major problems of this approach is how to decide the mapping function. Some possible answers to the problem are shown as follows:

- 1) Let  $\alpha(C_{i,k}) = 1$  for every attribute  $i$  and object  $k$ . The degree of certainty is ignored here, and hence the influential function represents the number of objects distinguished from Object  $k$  by Attribute  $i$ .
- 2) Let  $\alpha(C_{i,k}) = \beta C_{i,k} + \gamma$  for every attribute  $i$  and object  $k$ , where  $\beta$  and  $\gamma$  are parameters depending on the application domain, e.g.,  $\beta = 0.2$  and  $\gamma = 0.6$  may be used to build a prototype. Such a function can be useful to transfer certainty into an expert system to be developed.
- 3) Let  $\alpha(C_{i,k})$  be other functions, e.g., exponential distribution function, normal distribution function, etc. These complex functions may be far away from physical meanings, and may confuse the expert in the refinement process.

Table 6.8: an illustration of using INFL function,  
(a) an acquisition table

	Obj <sub>1</sub>	Obj <sub>2</sub>	Obj <sub>3</sub>	Obj <sub>4</sub>	Obj <sub>5</sub>
A <sub>1</sub>	A,1	B,1	B,1	A,1	B,1
A <sub>2</sub>	YES,2	YES,1	NO,2	NO,2	YES,2
A <sub>3</sub>	X	10,2	13,2	10,2	20,2

(b) the corresponding AOT

	Obj <sub>1</sub>	Obj <sub>2</sub>	Obj <sub>3</sub>	Obj <sub>4</sub>	Obj <sub>5</sub>
A <sub>1</sub>	2	D	1	2	1
A <sub>2</sub>	1	1	X	1	2
A <sub>3</sub>	X	2	2	2	3

Unfortunately, no matter what mapping function is chosen, it is possible that the system still fails to check out the inconsistency. For the example given in Table 6.8, if  $\alpha(C_{i,k}) = 1$  is chosen, INFL( $i,k$ ) may tell that this situation is acceptable, since A<sub>1</sub> distinguishes only two (Obj<sub>3</sub> and Obj<sub>2</sub>). However, as the certainties of the objects distinguish Obj<sub>1</sub> from others, similar situations may occur if other mapping functions are used, since  $C_{i,j}$  values, 1 and 2, are only the approximated values in representing relative degrees of certainty. For the same row or column, an entry with  $C_{i,j} = 2$  gives stronger evidence than that with  $C_{i,j} = 1$ . But if the  $C_{i,j}$  values of both entries are the same, it is difficult to tell their relative relationships. Therefore, sums of the generated values may include errors which mislead the inconsistency checking.

A consistency-test can be carried out by comparing the relative influence of two attributes to the same object. The test guarantees that each passed case is consistent, and hence no inconsistent condition will be ignored. Before introducing the test, we must note that

$$C_{i_1,k_1} > C_{i_1,k_2}, C_{i_1,k_1} \equiv C_{i_2,k_1} \text{ and } C_{i_2,k_1} \equiv C_{i_3,k_1} \text{ imply } C_{i_2,k_1} > C_{i_1,k_2},$$

but do not guarantee  $C_{i_3,k_1} > C_{i_1,k_2}$ . It is because each equality relation is only an approximation that more than two transitions of them may result in an inequality relation. Moreover, the sum of  $C_{i,k}$ s and the product of  $C_{i,k}$ s may mean nothing.

In the following, we show how our consistency test works by applying it to the example given in Table 6.9:

1) Find the condition:  $C_{i,k} = 1$  and  $C_{j,k} = 2$  and  $AOT_{i,k} > AOT_{j,k}$ .

Since  $C_{1,1} < C_{2,1}$  and  $AOT_{1,1} > AOT_{2,1}$ , the following steps are executed to compare  $A_1$  and  $A_2$ .

2) Find the distinguishable set of each attribute to each object.

A distinguishable set of  $A_i$  to  $Obj_i$  is defined as  $DIST(i,k) = \{ (Obj_t, C_{i,t}) \mid Obj_t \text{ is an object distinguished from } Obj_k \text{ by } A_i \}$ .

Table 6.8: an illustration of consistency-test,  
(a) an acquisition table

	Obj <sub>1</sub>	Obj <sub>2</sub>	Obj <sub>3</sub>	Obj <sub>4</sub>	Obj <sub>5</sub>
A <sub>1</sub>	A,1	B,2	B,2	A,1	B,1
A <sub>2</sub>	YES,2	YES,1	NO,1	NO,1	YES,1
A <sub>3</sub>	X	10,2	13,2	10,2	20,2

(b) the corresponding AOT

	Obj <sub>1</sub>	Obj <sub>2</sub>	Obj <sub>3</sub>	Obj <sub>4</sub>	Obj <sub>5</sub>
A <sub>1</sub>	2	D	1	2	1
A <sub>2</sub>	1	1	X	1	2
A <sub>3</sub>	X	2	2	2	3

The objects distinguished by  $A_1$  and  $A_2$  from  $Obj_1$  are then represented as

$$DIST(1,1) = \{ (Obj_2, 2) (Obj_3, 2) (Obj_5, 1) \}$$

$$DIST(2,1) = \{ (Obj_3, 1) (Obj_4, 1) \}$$

- 3) Compare the two sets. For each  $(Obj_t, C_{i,t})$  in  $DIST(i, k)$  and  $(Obj_t, C_{j,t})$  in  $DIST(j, k)$ , if  $C_{i,t} > C_{j,t}$ , eliminate both elements from the two sets, respectively. In the example,  $(Obj_3, 2)$  and  $(Obj_3, 1)$  will be removed from both  $DIST(1, 1)$  and  $DIST(2, 1)$ , respectively.
- 4) For  $(Obj_{t1}, C_{i,t1})$  in  $DIST(i, k)$  and  $(Obj_{t2}, C_{j,t2})$  in  $DIST(j, k)$ , check the existence of the following conditions:

$$\begin{aligned} &\exists C_{i,t2} \text{ such that } C_{i,t1} > C_{i,t2} \geq C_{j,t2}, \\ &\exists C_{i,t2} \text{ such that } C_{i,t1} \geq C_{i,t2} > C_{j,t2}, \\ &\exists C_{j,t2} \text{ such that } C_{i,t1} > C_{j,t1} \geq C_{j,t2}, \text{ or} \\ &\exists C_{i,t2} \text{ such that } C_{i,t1} \geq C_{i,t2} \geq C_{j,t2}. \end{aligned}$$

If one of the inequalities holds, remove  $(Obj_{t1}, C_{i,t1})$  and  $(Obj_{t2}, C_{j,t2})$  from the sets. In the example,  $C_{1,2} > C_{1,4} \equiv C_{2,4}$ , and hence  $(Obj_{2,2})$  and  $(Obj_{4,1})$  are removed from sets.

- 5) If the  $DIST(i_2, j)$  is empty, then the situation is acceptable; otherwise, inconsistency may exist and a warning message is printed out. In the example, the situation is accepted, since the final  $DIST(2, 1)$  is empty. However, if  $C_{2,3} = 2$  or  $C_{2,4} = 2$ , a warning message will be printed out, since there may exist inconsistency in the case.



## KNOWLEDGE REPRESENTATIONS BASED ON EFFECTIVE CONCEPTUAL SCHEMAS

In this chapter, we shall discuss some methods for knowledge representations; specifically speaking, we shall focus on *effective conceptual schemas*.

An effective conceptual schema is a model which is not only capable of describing the knowledge structures, but also of containing programming details that are not part of the knowledge structures, but which are necessary to make the model run with the available technology. Such an approach is needed for having a trade-off between domain- and implementation-driven approach that we have mentioned in section 4.2. To achieve this, SP is adopted.

SP was originally developed by Wolff; It stands for 'simple and power' and represents a new concept of software supported by a runnable pattern-based computing language also called SP [Wolff91].

### 7.1 A 'trade-off' between domain- and implementation-driven modelling

There are three ways of modelling: syntactics, semantics, and pragmatics. For details, see [Morris55] [Zemanek66].

*Syntactics* studies the formal relations of signs to one another. It deals with the combination of signs without regard to their specific signification or their relations to the behaviour in which they occur. *Semantics* studies the relation of signs to the objects to which they are applicable. It deals with the signification of signs in all modes of signifying. *Pragmatics* studies the relation of signs to interpreters. It deals with the origin, use, and effect of signs within the behaviour in which they occur. It is evident that here we mostly use syntactics and pragmatics.

In the computational extensions of the theory, theories of mapping vary primarily according to what aspects of similarity are deemed crucial - syntactic and pragmatic. In particular, the theory that may be least dependent on semantics and therefore more amenable to our purpose is Gentner's structure mapping theory [Gentner83] with Holyoak's pragmatic theory [Holyoak89]. Gentner works within a network representational system containing objects (or concepts) and predicates that link them together. Important syntactic distinctions are made with respect to predicates: those which are similar, but are irrelevant to relations, are not mapped in an inter-domain situation. SP theory and language highly support these themes.

SP views the language as being 'universal' not in the sense of it being any kind of complete formal system, but rather in the sense of using the language to represent diverse

kinds of knowledge, including software, in an efficient way. The efficiency is regarded as a trade-off between the model which contains very little practical implications, and the casual conceptual model which often contains very little computing details. For achieving the efficiency, SP intends to manage redundancy in a knowledge structure, such as

- the detection of redundancy by *pattern matching*,
- the reduction of redundancy in the structure by *searching* for the greatest possible *unification* of patterns.

### **7.1.1 Effectiveness supported by efficiency plus redundancy**

Since the extraction of redundancy by *pattern matching* and *unification of patterns* can provide an unified view of several concepts in computing, SP theory has a very close tie with part of Shannon's information theory [Shannon49]. SP theory also takes advantage of the ideas of searching for economical structures which have figured in research on 'neural computing', see, e.g., [Rumelhart86].

*Redundancy means repetition of information.*

Redundancy in information always means the repeating of information. Repletion of information means redundancy when the repeating patterns are more frequent than other patterns of the same size.

*Redundancy and structure.*

The concept of redundancy (in the Shannon sense) is closely related to the concept of *structure*. A body of information which has no redundancy is entirely random; it has no structure. The structure of a body of information may be equated with the patterns of redundancy in it. Thus, any discussion of the structure or organisation of a body of information may be mirrored by a corresponding discussion about the redundancy in the information.

*Redundancy and inductive reasoning.*

Whatever the particular reasons for storing and manipulating information (in computing systems, on paper, or in our heads), the underlying reason is always to make predictions. The principle of inductive reasoning - that the past is a guide to the future - is the basis of all kinds of natural or artificial cognition.

### **7.1.2 Software as a compressed representation of inputs and outputs**

Two inter-related propositions are discussed here:

- that any function, program or information system may be defined in terms of its inputs and its outputs.
- that 'well structured' software is a model of the domain which it serves.

*Defining a function in terms of inputs and outputs.*

Functions in computing and mathematics are often defined 'intentionally' in terms of rules or operation required to perform the function. However, functions are also

defined 'extensionally' by specifying one or more outputs for every input or combination of inputs [Sudkamp88]. This idea applies to functions of various kinds ('total', 'partial' and 'multi') and also to 'programs' and information systems in general.

Input1	Input2	Output
0	0	0
1	0	1
0	1	1
1	1	0

Fig. 7.1: definition of an function

For example, a function may be defined as in Fig. 7.1. A decision-rule may be defined in Fig. 7.2 during a process to improve the water quality of the river by control of the level of biochemical-oxygen- demand (BOD) to below 5 mg/litre. The BOD and the current state of water may be regarded as inputs, and the new state of water may be regarded as an output.

BOD-5	Current state	New Demanding state
high	bad	bad
high	good	bad
medium	good	good
medium	bad	bad
low	good	good
low	bad	good

Fig. 7.2: defining a water quality function

In this example, principles and practice coincide; it is quite realistic and convenient to define each of these two functions by a table, and yet, a table gives a comprehensive definition of a decision-rule.

#### *Software design as modelling.*

The ideas that, any function may be defined in terms of its inputs and its outputs, relate to another idea which has become prominent in the computation. That is, as we have discussed, the organisation of well designed software must reflect or 'model' the organisation of its external world.

In the early days of machine computing (when all programming was done in assembly language or worse), software design was heavily oriented towards the organisation of the computer. Instructions to load registers, rotate bits in a register or jump from one memory location to another were prominent features of software.

Since the invention of compiled and interpreted high level language, there has been a trend in software design to disguise the workings of the underlying machine. Many

essentially independent developments have arrived at the idea that the organisation of well designed software should reflect or model the external world with which the computing system interacts (see chapter 4 for example).

### 7.1.3 Schema plus correction

In a case of object-oriented programming, a class is a general framework whose attributes are 'inherited' by every sub-class or instance that it dominates. Essentially, conceptual schemas, see, e.g., [Sowa92], have great difficulties in describing two or more patterns of information which are similar, but not identical.

For inheritance of attributes in two or more contexts in an object-oriented hierarchy, the attributes of a given class may be regarded as 'corrections' to the schema; While, correction means information compression, as follows.

A schema can be corrected by information redundancy, i.e., the parts which are the same may be recorded once at the first place, whereas the other parts of the pattern, which vary from one instance to another, may be given as alternatives at appropriate places or omitted altogether. These corrections can be used to select alternatives in the schema or to fill in gaps.

```

object → ordered_AND_object | unordered_AND_object |
        OR_object | simple_object
ordered_AND_object → () | (body)
unordered_AND_object → [] | [body]
OR_object → { } | { body }
body → object body | object
simple_object → symbol | _
symbol → symbol symboch | symboch
symboch → A | ... | Z | a | ... | z | 0 | ... | 9 | % | & | # | ...

```

Fig. 7.3: SP syntax

## 7.2 SP language

The syntax of the current version of the SP language is shown in Fig. 7.3, and an instance of its parsing is shown by the program in Appendix 6. The language comprises five kinds of objects.

- An 'OrderedAND-object' (OAO).  
OAO represents a sequence. It is similar to a 'list' in language like Lisp, but without the internal head-tail structure.

- An 'UnorderedAND-object' (UAO).  
UAO represents a collection of objects in which order is not significant. It is essentially the same as a 'bag' in logic.
- An 'OR-object' (ORO).  
ORO is similar to a UAO, but its constituent objects represent alternatives in the given context.
- A 'Symbol' is a string of one or more characters (bounded by space or meta-symbols) treated by the computing system as being atomic.
- A 'Variable' ('\_') is a place marker for missing information, rather like an unnamed variable in Prolog.

The semantics of the SP language are provided by the core processes of pattern-matching, unification and search. In outline, the search process is shown in Fig. 7.4.

The search method unifies patterns in order of decreasing size (which means decreasing amounts of redundancy). The method is quite effective for finding 'good' sets of unifications and compressing the data, although it is not guaranteed to find the best possible answer.

```

WHILE no more extractable redundancy in the given information DO
BEGIN
  1 Search for redundancy in the given information by systematically
    comparing substrings in the information and identifying
    matching patterns.
  2 Select the pair of matching patterns which represents the
    greatest amount of redundancy.
  3 Merge or unify the selected pair of matching patterns.
END
    
```

Fig. 7.4: search process in SP language

Table 7.1: evocation, assessment, organisation and transformation between patterns ( $P_1, P_2, P_3$ )

processes	meaning	recommended SP representation
evocation		
$P_1 \rightarrow P_2$	$P_2$ used to follow $P_1$	$(P_1, P_2)$
$P_1 \sim P_2$	$P_1$ and $P_2$ used to appear together	$\{P_1, P_2\}$
assessment		
$P_1 < P_2 \xleftarrow{v} P_3$	$P_1$ strengthens $P_2$ from a $P_3$ viewpoint	$[P_2, [(P_1, P_2, \_), (P_1, P_3, \_)], \_]$
$P_1 > P_2 \xleftarrow{v} P_3$	$P_1$ weakens $P_2$ from a $P_3$ viewpoint	$[P_2, [(P_2, P_1, \_), (P_1, P_3, \_)], \_]$
organisations		
$P_1 \times P_2$	$P_1$ (re) organizes some structural properties of $P_2$	$[[P_1, P_2], \_]$
transformation		
$P_1 \sim P_2 \rightarrow P_2'$	the presence of $P_1$ transforms $P_2$ into $P_2'$	$\{P_1, (P_1, P_2), \_ \}$

We may use SP to organise a pattern-based system governed by the rules illustrated by Table 7.1. We may also use SP to manage object-oriented software constructions to be described in the next section.

### 7.3 Connections to object-oriented software constructions

Although many of the characteristics of objects have been extensively studied by Bruner, Goodnow and Austin [Bruner56], it is only recently that we have computer software concepts relevant to the well-known notions, such as scheme [Bartlett32] [Bobrow75], frame [Minsky75], and script [Schank77].

Like many other theoretical constructs, SP language enables us to view a generalised pattern as a kind of reflection of a common recurring set of objects. It allows slots in the framework where alternatives may be inserted at '\_', see Fig. 7.3. An alternative may be a function as default - the assumed filter for the slot when there is, of course, no contrary evidence. It is capable to describe that members of each disjunctive set may typically vary in their contextual probability, and further, the most probable one may be regarded as a default element. We shall briefly discuss this topic under other relevant subjects as follows.

```
[person [name, _]
      ((head ((eyes _)(nose _ _))
      (body _)(legs _ _))
      [eats _][sleeps _][breathes _] _
      [profession
      {[thinker _]
      [tailor _]
      ...
      ]}]
      [gender
      {[male _]
      [female _]
      }]]]
```

Fig. 7.5: the integration of class-inclusion relations, part-whole relations and inheritance of attributes.

#### 7.3.1 In class-inclusion relation, part-whole relation and inheritance of attributes

Fig. 7.5 shows how class-inclusion relations, part-whole relations and inheritance of attributes may be integrated. The sets of three dots ('\_') show where other information would go in a more fully developed description. This structure shows the class 'person' as having the attributes 'head', 'body', 'legs', 'eats', 'sleeps', etc. Note, these are only parts which compose the concept - 'person'; the parts may themselves have sub-parts on any unknown lower level.

'Persons' have sub-class 'thinker', 'tailor', etc. and they are also cross-classified as 'male' or 'female'. Any number of levels is possible in this kind of classification

scheme. Note, how part-whole relations equate with AND relations, while class inclusion relation equate the OR relations.

An 'instance' of a person may be represented as:

```
[person [name, Tom], _, [profession [thinker, _]] [gender [male, _]]]
```

or even more succinctly as:

```
[_, [_, Tom], _, [_, [thinker, _]] [_, [male, _]]]
```

Either pattern will unify with the class schema giving an instantiation, i.e., a full description of 'Tom' something like this:

```
[person [name, Tom]
  (head ((eyes _)(nose _))
  (body _)(legs _)
  (eats _)(sleeps _)(breathes _)_
  [profession [thinker _]]
  [gender [male _]]].
```

We may stand to gain another kind of benefit through this integration also.

### **7.3.2 Class, meta-class, instance and the evolution of classes**

Most object-oriented systems make a sharp distinction between 'classes' and 'instances'. Classes serve as templates for the creation of instances, but instances may also be templates for something.

The structure of classes is established by a designer before the system runs and remains fixed while the system runs. As it runs, it may create new instances of classes dynamically, but not new classes.

Since classes, in most systems, are regarded as objects, this means that they must be instances of something. To avoid classes being instances of classes (which would violate the sharp distinction between instances and classes), it is necessary to introduce the concept of 'meta-class'. As we know, classes may then be instances of meta-classes and so on [Ungar87].

In SP, the advantage of merging the concepts of 'instances', 'class' and 'meta-class' removes unnecessary rigidities in computing, i.e., any object may serve as a template for the creation of other objects and any object may be created dynamically as the system runs.

This potential might be particularly powerful for enriching the concepts of object treated in domain- and implementation-driven modelling all together, because it may be true that there are distinct classes, instances of classes, etc. in our daily life, but that does not mean that those kinds of distinctions are appropriate representations of the objects in engineering. There are classes of visual, auditory and tactile images, which would be something inappropriate to label as 'Mary' for example, because the images may be very individual and highly primitive precepts.

### 7.3.3 In logic and logical inference with uncertainty

SP has potential as a medium for expressing logical propositions and making logical inferences. This is perhaps not surprising, if SP gives the significance in the simple logical relations, AND and inclusive OR.

Here is a familiar and elementary example:

All men are mortal.  
Socrates is man.  
Therefore Socrates is mortal.

In first-order predicate calculus, this may be expressed as:

$$\forall x \in \text{man. mortal}(x) \wedge \text{Socrates} \in \text{man} \Rightarrow \text{mortal}(\text{Socrates})$$

In SP, the same premises and the inference may be expressed as follows:

```
[[mortal[man, _]]
[man, Socrates]]
→ [mortal [man, Socrates]]
```

'→' is not part of SP. The significance is that it is produced in the course of computing ! In the first line, the pattern [man, \_] matches any UAO which has 'man' as a constituent object. Thus, it may be read as 'all men', or 'any man'. The underscore symbol then removes the need for an universal quantifier in predicate calculus.

The conjunction of 'mortal' with [man, \_] gives the pattern [mortal [man, \_]]. This represents 'all men are mortal'. Likewise, [man, Socrates] will unify. In this way, the proposition 'Socrates is mortal' is validated.

Switching forms of representing logical inference like this gives us the freedom of representing uncertain-objects-contexts by inserting other symbols in replacing '\_'.

Some problems, such as distinguishing between 'good' and 'bad', would be better treated by 'finding matches' between patterns. We can use symbol '%' on items within ORO to represent weights on an object, so that the object is shown by the apparent strength of the case that the object should be considered.

## 7.4 Concluding remarks

Our main technical themes are directed by patterns.

The starting point of our considerations is an evolutionary hypothesis, i.e., all primitives and advanced forms of mind representations are close to the metaphor of patterns. Pattern means an ensemble of information primitives, e.g., utterance, pixels, etc.

Nevertheless, the research here does not aim at developing a mental model or replacing other non-connectivist methods of AI. The main objective is to fill a niche between the existing methods by exploring AI in a pragmatic way. The ambition is a modest one, but it tackles some very hard problems.



In spite of the aforementioned and limited approach, we have made a few valuable connections.

The first is an interpretation of usual relations among actions, objects and concepts. Pattern has been shown as a hidden (or partly hidden) relation among the information primitives. For representing the contexts of data being transferred in DSE however, neither natural nor conventional programming language are satisfactory. The former is too informal and ambiguous, while the latter is usually a too detailed and low-level-tool.

The second is that we are actually learning how we can represent, generate, and investigate a model based on the abstract images extracted from the *vision* of the problem. If a process in DSE attempts to find data, this process might be in a context of decreasing the dimension of the considered system by following an approach in a certain data-subject, or by sharing more than two data-subjects simultaneously [Albus91] [Botvinnik84] [Mesarovich70] [Simon80]. However, implementations based on the formal theories of planning meet hard efficiency problems [Chapman87] [Fikes71] [Nilsson80]. There is no general constructive approach to such implementations. Each new problem must be studied carefully.

The third is that the development of information technologies, such as GIS, should take the social, economic, political, and cultural environments into account, because the quality of decision making relies heavily on the result of interpretation of two pieces of information. One is about the broader environments the decision maker faces and the other is about behaviours of the organisation the decision maker belongs to. Many familiar elements need to be cast in a form which DSE adapts.



## **Part III**

### **The applications in water resources management**



## ORIENTORS vs. INDICATORS

---

In part III, we shall first of all establish a set of effective conceptual schemas which are used as *orientors*. Then we shall report some implementations with AML via ARC/INFO systems.

### 8.1 Orientors or indicators ?

It is perhaps easy to explain what orientors mean by getting an impression on *indicators* which have been well established and widely accepted in the existing literature.

An indicator indicates routine behaviour of information processing. Fig. 8.1 & 8.2 illustrate two sets of indicators. To avoid a misunderstanding of their work, no more comments are given.

Here the main point is that, since information processes may involve a fairly complex sequence of cognitive processes, indicators should have a *provisional* classification of the situation required for the guidance of the problem-solving process. We then need to consider how to generate suitable responses to the indicators; otherwise, the interactions between indicators would be in utter confusion.

We call these orienting parameters *orientors* in order to avoid some of the connotations commonly attached to the various types of normative parameters playing a role in the behaviour of humans, such as objectives, values, priorities and preferences which have been mentioned in chapter 2.

Orientors are meaningful to a computing system only if the state of the system itself, and that of the environment relevant to the system, can be assessed with respect to the orientors. This requires firstly that the system perceives an external state through a set of orientors, i.e., perceived state variables (the composition of this set may change from moment to moment); and secondly that the perceived system and external state can be mapped on the relevant orientor space. What follows explains these in detail.

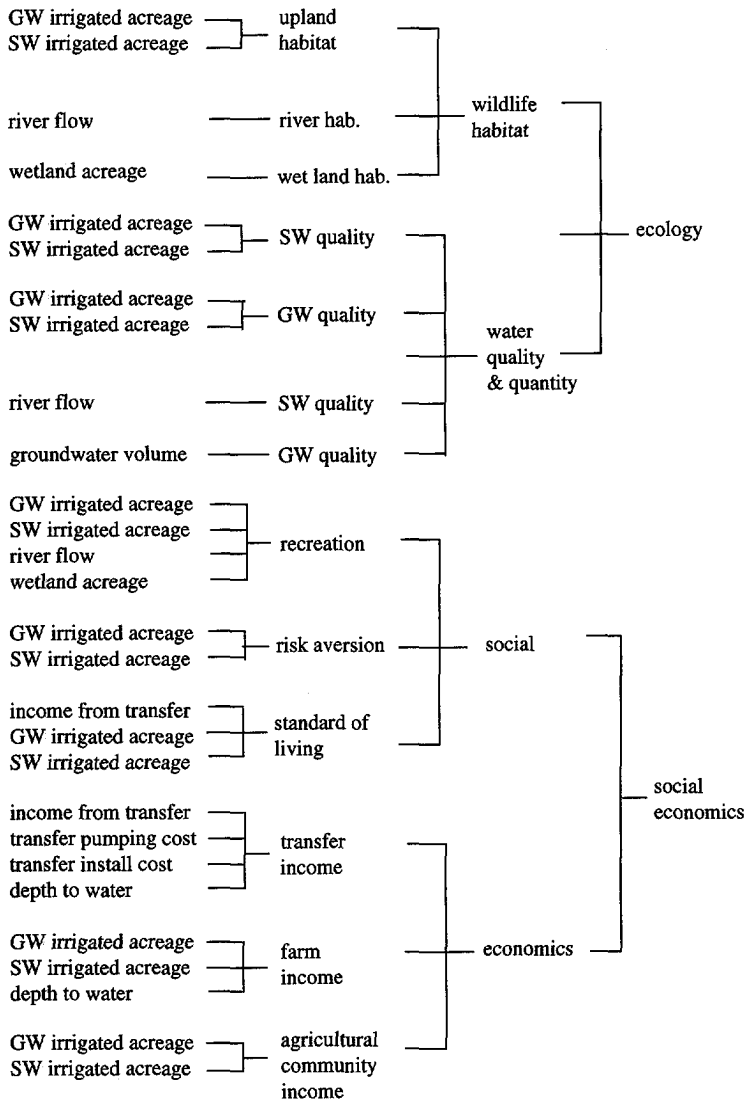


Fig. 8.1: aggregation of basic indicators into final trade-off indicators, after Stansbury (GW stands for groundwater, and SW for surfacewater [Stansbury91])

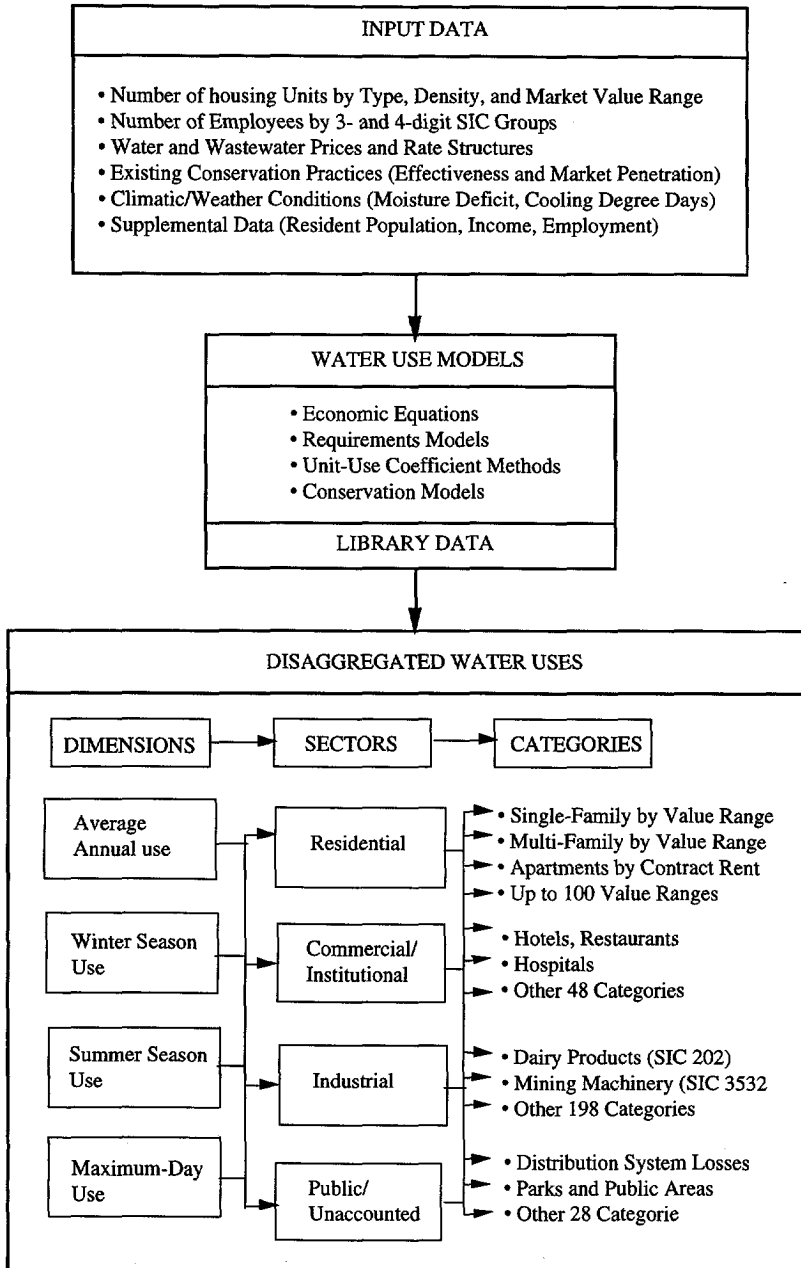


Fig. 8.2: IWR-MAIN procedure for estimating water use [Dziegielewski87]

## 8.2 Water-use and water-demanding

Water-use can be divided into two categories, *consumptive use*, in which water is an end to itself, and *non-consumptive use*, in which water is a means to an end. Examples of consumptive use include agriculture, industry and mining; examples of non-consumptive use include instream uses such as hydropower, transportation and recreation. All together, water-use has to refer to the amount of water applied to achieve various ends, so that it is a literally descriptive concept.

Water demanding is the scheduling of quantities that consumers use per unit of time for particular prices of water, which is an numerically analytical concept.

```
[ objectHouse, _[
  name { shared_objects }
  waterDemanding ( waterResources, waterUse, _ )

  /* ----- */
  waterResources ( SURFACE-WATER, GROUND-WATER, _ )
  /* ----- */
  SURFACE-WATER ( _, )
  /* ----- */
  GROUND-WATER ( _, )
  /* ----- */
  waterUse { consumptiveUse, nonConsumptiveUse, _ }
] ]
```

Fig. 8.3: a case of the shared objects between the P-R and P-D

### 8.2.1 Schemas

Fig. 8.3. shows the shared objects in **bold** type, and comments are given below.

- External interactions between the P-R and P-D.

We see that they require the management of the objects shared between a goal of policy assessment (**waterUse**) and the necessary measurement of water (**waterDemanding**).



- Re-ordering the objects.

A need for water measurement is imposed by re-ordering the non-terminals in the schema. The schema is oriented by waterUse.

- An Input and Output.

waterUse becomes a non-terminal, and later, filled with two terminals, i.e., consumptiveUse and nonConsumptiveUse. Therefore, waterUse could be regarded as an input. waterResources contains two non-terminals, and are still supposed to be filled by non-terminals. It could be therefore treated as an output.

- The missing places.

Some missing places will be filled in due course, while others have to wait for the external processes.

- The data for the players.

There is a shared object for the players, i.e., waterResources.

```
[ objectHouse, _ [
  name {policy-assessment}
  waterUse (nonConsumptiveUse, consumptiveUse, waterDemanding, _ )

  /* ----- */

  nonConsumptiveUse { drinking-water,
                      recreation,
                      natural-environmental-quality,
                      transportation,
                      hydropower, _ }

  /* ----- */

  consumptiveUse { agriculture,
                  industrial,
                  mining, _ }

  ]
]
```

Fig. 8.4: a goal of policy assessment

### 8.2.2 Other constructions for policy-assessment

The schema in Fig. 8.4 is to augment some attributes to the concepts of P-R suitably to policy-assessment.

#### *Classes of models.*

From an economic viewpoint, we are well able to model consumptive uses. Consumptive uses are modelled by using consumptive functions. Some impacts of non-consumptive uses are modelled using production functions, such as transportation; while others are modelled descriptively, such as recreation.

*Municipality of players.*

For water non-consumptive use, the national or regional authorities who owe or govern WRM have to be invited into P-R. Furthermore, since consumptive use as well as some impacts of non-consumptive use are leading to economical consequences, other kinds of authorities have to be invited into P-R as well, see Fig. 8.5. Policy assessment to balance water-uses is normally achieved by a project, see [Erlenkotter76] for the formal definitions, and [Barritt-Flatt91] for project management. Therefore, we are in need for software-tools enabling us to visualising the information so that the policy instruments can be coupled with the concerned conflict-cases more explicitly.

```
[ objectHouse, _ [
  name {players}
  waterAuthorities (governingScale, waterResources, policyInstruments)

  /* ----- */

  governingScale { national, regional, district, operating-board,
                  public-members, private-members, _ }

  /* ----- */

  waterResources { river, lake, reservoir, groundwater, _ }

  /* ----- */
  /
  policyInstruments { water-law, water-policy, water-regulation,
                    operating-rules, _ }
] ]
```

Fig. 8.5: a representation of the player and a context

### 8.2.3 The concept of the value in the non-consumptive uses

The existence of policy-alternatives is surely because of the different values which each policy-alternative owes.

By values, we shall mean general beliefs as well as basic attitudes in a case of the non-consumptive use. Thus, it might be profitable to view the policy-assessment not only as the determination of some courses in a plan, but also as an unification of objects' values in 'reality'. With the help of visualising the policies, this could be achieved by the tables coupled with each visualised policy-alternative.

The value of recreation could be measured by a table. The recreation is treated as an *entity*: An entity consists of classes of objects; A class of object is a number of recreation-spots in a region; An object contains two items: one is the item for a set of recreation-activities; and the other is for the value. The first item stores the information

about a set of activities available in a recreation-spot, and the second item stores a number of recreation-days, where a recreation-day is measured by a visit of one individual to a recreation-spot for the recreational activities registered in the first item.

An area is treated as an entity which consists of classes of objects. Then, a class of objects is a number of regions in an area; and the value of an area could be described by a table. For example, for a measurement of the value of recreation, the attribute-dimension of the object may contain two items: one is for a set of natural objects, such as types of trees, wildlife, etc., and the other is for a number of user-days, where one user-day is the participation of one individual in the uses of a wildlife-resource that do not reduce the supply, such as bird-watching or nature photography, during a reasonable portion of, or a whole 24-hr period.

Also note, since each of these values has a certain importance of relative durability (so to speak), 'value change' means a change in the importance of a value, e.g., a change in the preference ordering of all values. In considerations of the non-consumptive use, such changes occur slowly. This reminds us that it is very important to manage the information related to the consumptive-use and the information related to the non-consumptive-use separately.

### 8.3 The policies and spatial attributes

Here, we have at least two reasons for systematically treating the interactions between the P-R and the P-D.

First, the scenarios emerging from a policy-assessment would be better quantified, so that the sensitivity of the separate models could be systematically concerted into reliability limits of the value-scores.

Second, for the non-consumptive use, as we have mentioned, water is the means to an end. Some of the uses in this category are directly dependent upon an acceptable behaviour of a water-resource system, such as transportation and hydro-power generation; While others are dependent upon an interpretation given by a view to the properties, then that *view* could be considerably effective in assessing a policy. Examples can be once again selected from the report [GMN92] we have studied:

- a) planting dark pine trees instead of deciduous trees can positively influence the groundwater system in the middle of the Netherlands;
- b) in the Vechtplassengebied and in the Vallei channel (south of Veenendaal), 10% of the pine trees have been replaced by deciduous trees;
- c) in Utrecht, it shows a lack of groundwater supply in the very near further, but a lot of valuable nature areas have been changed by lowering the groundwater level.

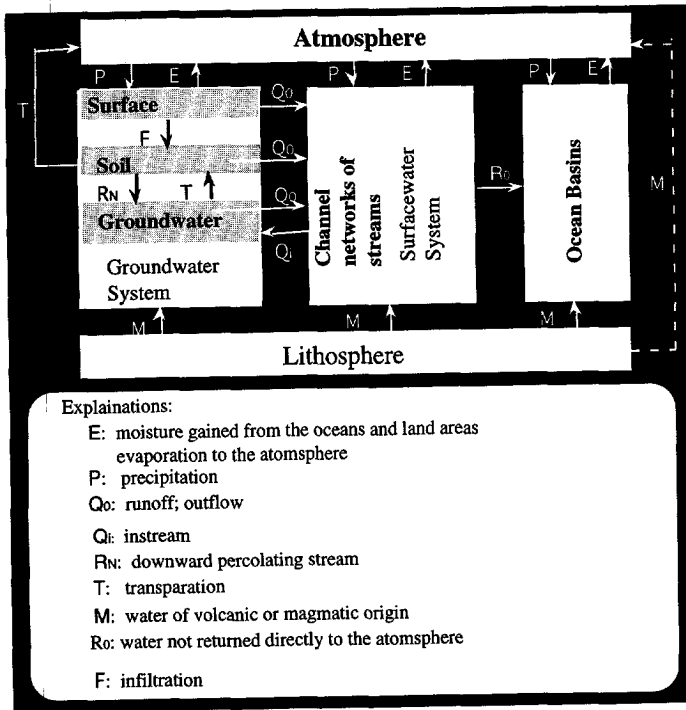


Fig. 8.6: examples of systems, variables and parameters selected from [Domenico72]

### 8.3.1 Representing some spatial properties

There is a large amount of literature to describing the basic knowledge to modelling a water-resource system, e.g., see [Singh88] for modelling a surfacewater system, and [Domineco72] for modelling a groundwater system.

There are also many software packages enabling us to calculate some of the frequently used models, if, of course, the data to be required by the parameters and variables are available.

In a system approach, a hydro-system seems to be always described in terms of 'black box', complete with inputs and outputs.

Input terminals represent the excitation variables, which influence system behaviour, and are identified by 'arrows' pointing towards the box. Output terminals represent the response variables, representing those aspects of system behaviour which are of interest, and are identified by arrows pointing away from the box. Intermediate variables, which are neither excitation nor response variables, are assumed to be embedded in the box, and are important because of their combined effect on the internal mechanism relating input.

However, it is believed that what constitutes a box and an arrow is duplicated through our views to the spatial properties, and importantly, the different views provoke the different means for explicating the knowledge. The following explains this standpoint.

```

[objectHouse, _ [
    name { spatialProperties }
    geoObjects { SURFACE-OBJECT, GROUND-OBJECT }
    SURFACE-OBJECT { _ }
    GROUND-OBJECT { _ }

/*-----*/

    waterFeature ( CHEMICAL-ELEMENTS,
                    flowUp, flowDown, flow )
    CHEMICAL-ELEMENTS { _ }
    timeScale { second, minute, hour, week, month, year }
    geoScale ( _ )
    physicalProperties { MOISTURE, PRECIPITATION,
                        EVAPORATION, RUNOFF,
                        INFILTRATION, TRANSPIRATION,
                        OUTFLOW, INTERFLOW,
                        DOWNWARD-FLOW, _ }

/*-----*/

    MOISTURE ( PRECIPITATION )

    PRECIPITATION { ( atomsphere, waterFeature, timeScale,
                     SURFACE-OBJECT, geoScale ),
                   ( waterFeature, timeScale,
                     GROUND-OBJECT, geoScale )
                  }

    EVAPORATION { ( MOISTURE,
                   ( SURFACE-OBJECT, geoScale,
                     waterFeature, timeScale, atomsphere ),
                   ( GROUND-OBJECT, geoScale,
                     waterFeature, timeScale, atomsphere )
                  }

    RUNOFF { ( PRECIPITATION, SUFACE-OBEJCT,
               waterFeature, timeScale, SUFACE-OBJECT, RUNOFF ),
            ( PRECIPITATION, SURFACE-OBJECT,
               waterFeature, timeScale, GROUND-OBJECT, RUNOFF )
          }

/*----- to be continued by Fig. 8.8-----*/

```

Fig. 8.7: a conceptual schema representing some physical properties

```

/* ----- to continue Fig. 8.7 ----- */

INFILTRATION (( SURFACE-OBJECT, geoScale,
                 waterFeature, timeScale,
                 SURFACEOBJECT, geoScale ),
              ( SURFACE-OBJECT, geoScale,
                waterFeature, timeScale,
                GROUND-OBJECT, geoScale ),
              ( GOUND-OBJECT, geoScale,
                waterFeature, timeScale,
                GROUND-OBJECT, geoScale )
            )

TRANSPIRATION ( GROUND-OBJECT, geoScale,
                 waterFeature, timeScale,
                 SURFACE-OBJECT, geoScale )

OUTFLOW (( SURFACE-OBJECT, geoScale,
            waterFeature, timeScale),
          ( GROUND-OBJECT, geoScale,
            waterFeature, timeScale )
        )

INFLOW (( SURFACE-OBJECT, geoScale,
           waterFeature, timeScale,
           GROUND-OBJECT, geoScale ),
        ( GROUND-OBJECT, geoScale,
          waterFeature, timeScale,
          SURFACE-OBJECT, geoScale )
      )

DOWNWARD-FLOW ( GROUND-OBJECT, geoScale,
                 waterFeatures, timeScale,
                 GROUND-OBJECT, geoScale )

spatialProperties ( physicalProperties, ( variable-name, meauseUnit),
                  (parameter-name, measurUnit)

]

```

Fig. 8.8: a continuation of Fig. 8.7

If we emphasise the properties which orientate the processes in the P-D to be understood truly, 'explicating knowledge' is for explaining the behaviour of a water-resource system, but restrict to an observer's concerns. To support decision processes, it may therefore be very difficult to draw a line between what in a software system is essential and what is not. At least, it is arguable if this kind of software system can do anything about the information-loss caused by a broken decision-process, e.g., a process *between* the P-R and P-D.

If, however, we emphasise the properties which are likely to be warranted to the processes involved in both the P-R and P-D, 'explicating knowledge' is for making the attributes of the shared-objects explicit to both parties. Then, we may gain two kinds of

benefits: the attributes revealed would support both a policy-assessment and models' calibration.

A conceptual schema representing some of the properties is presented in Fig. 8.6 continued by Fig. 8.7. Some explanations are given below.

### 8.3.2 Remarks on the objects in the schemas for representing spatial-properties

#### *The missing places.*

SURFACE-OBJECT, GROUND-OBJECT and CHEMICAL-ELEMENTS are non-terminals.

#### *geoScale.*

The geoScale emphasises a geographical allocation of a member of geoObject. It is always placed after the member, so that a mark can be made for registering the allocations of several members of geoObject which are involved with a physical-phenomena.

We regard the scale of a landuse-map as the unified scale; We allow other kinds of surface-objects such as vegetation and soil to have their own scales. This would enable an object to be studied separately at a level of detail on one hand, and possibly to be unified at a regional level on the other hand.

Now, three objects should be filled into the 'bag' - surfaceObject, and we therefore have

surfaceObject {landuse, vegetation, soil, \_}

At this stage, it is not required to fill geoScale, because there is no pattern to coordinate the relevant objects. A pattern will be explained soon.

#### *waterFeature.*

An order between waterFeature and an object could be used to identify the following:

- water flows from an object  
(\_, **object**, geoScale, waterFeature, timeScale, \_)
- water flows into an object  
(\_, waterFeature, timeScale, **object**, \_)

Other directions of water-flow may be identified when an object is coordinated.

#### *Patterns.*

physicalProperties is an object collecting some basic patterns. Each pattern intends to identify a category of the properties. Here, we are only going to explain this by an example. More relevant applications will be described in the coming section.

Suppose there is a process in which the phenomena 'evaporation' are studied, and there are objects belonging to Vegetation and Soil involved in observing an event of evaporation. The pattern *evaporation* enables us to mark the following contexts:

EVAPORATION (landuse, geoScale1, waterFeature, timeScale, atmosphere)  
EVAPORATION (vegetation, geoScale2, waterFeature, timeScale, atmosphere)  
EVAPORATION (soil, geoScale3, waterFeature, timeScale, atmosphere)

During the pattern-matching, we may have the following:

- geoScale1 is linked with geoScale2 and geoScale3, while the following is identified:
- water is flowing out of a surface object;
- selecting flowUp (a water-flow direction. Note, this is because the coordinated object is 'atmosphere');
- the phenomena is fallen into a category of either evaporation or moisture.



## IMPLEMENTATIONS

---

In the previous chapter, we have represented a set of orientors by effective conceptual schemas for allocating the necessary contexts where a decision process takes place, e.g., the constructions for water policy assessment. While, in chapter 1, we have mentioned that the menus (see Fig. 1.7) should correspond to data or information subjectively.

In order to implement the couplings between the menus and the subjective data, we shall, in this chapter, consider ARC/INFO [ARC92]. ARC/INFO is a set of tools for building GIS. These tools are very sophisticated, and therefore it is almost impossible here to describe all the functions that ARC/INFO can offer.

Nevertheless, we shall focus on using AML programming language [AML92]. AML programs organise a sequence of ARC/INFO commands into geoprocessing operations; therefore, it is at the heart of any ARC/INFO system.

We shall be mainly concerned with the following issues:

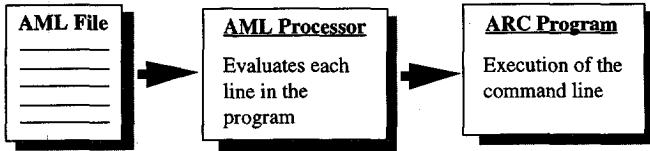
- how to generate a dynamic menu, i.e., when a certain category of data is available, an item in a menu is capable of loading data; and
- how to make such a pattern-based data path.

We begin with a conversion of a SP schema to a menu, then describe how to connect the menu to data paths.

### 9.1 AML, a menu and conceptual schema

AML is an interpreted language. AML enables us to write programs that contain a sequence or combination of ARC commands, host operating system commands, and AML directives, functions and variables. Every command line, either entered from the keyboard or an AML file, is interpreted by the AML processor before the command is executed by the current ARC program. Thus, the actual command, as executed in ARC, never sees directives, functions or variables; it sees only the results of their interpretation.

Each line of an AML program passes through the AML processor before it is executed by the current ARC program, after [AML92].



AML variables store dynamic information. We can think of such a variable as place-holder that contains a specific type of data. A variable can contain a character string such as the name of a coverage or the path-name to a workspace, or a numeric value representing the number of arcs in a coverage, or, in addition, it can contain a variable or a Boolean value that, for example, indicates whether a particular object exists. Before a command is executed, the AML processor evaluates the variable, performing variable substitution. Thus, by controlling the value stored in a variable, one can control the types of operations that are executed.

One of the most exciting and highly visual features of AML is the ability to create menu-based user interface. This kind of interface have been shown in Fig. 1.6 & 1.7. So, the question is how to enable the menus to meet some of our specific needs.

Normally, menus are ASCII text files that are created with the operating system's text editor. The format of the text in the file is important because it determines the choices that are displayed and the actions that are taken when a selection is made. The first line of a menu file identifies the menu type. This is a number from one to seven that represents the following types:

- 1 = Pulldown menu
- 2 = Slidebar menu
- 3 = Matrix menu
- 4 = Key menu
- 5 = Tablet menu
- 6 = Digitizer menu
- 7 = Form menu

```

1 Main pulldown for DSE
Analysis
'governing scale' &r analysis init scale
'water resources' &r analysis init resources
'policy instruments' &r analysis init instruments
'Tool box'
draw &r tool create draw &stripe 'Draw Tool' &position &right &display
query &r query init
selection &r select init
statistics &r stats init
buffer &r buffer init
'External connections'
'hydro simulation' &r simulation init
'objective optimisation' &r init optimisation
Quit & r quit
  
```

Fig. 9.1: an illustration between a AML program and a SP schema in Fig. 8.5

The remainder of the file consists of statements that define the menu choices. The syntax of these statements is dependent upon the type of menu being created.

Because of this format, it can be assumed that a SP conceptual schema can be easily transformed into such a menu-file; then, of course, the menu established can hold some properties that we have addressed in SP paradigm.

As an illustration, Fig. 9.1 shows an imposition on the relationships between the schema shown in Fig. 8.5 (a representation of the player and a context), and the main menu. In that file, as we have mentioned the first line is indicating the mode of the menu, i.e., '1' represents what follows is a pulldown menu. In addition, it should be noted that 'governing scale', 'water resources', policy instruments' are supposed to be directly linked with governingScale, waterResources and policyInstruments which are treated as three terminals in the schema in Fig. 8.5. Each item is followed by '&r' command ('&run'), then, the name of an executable AML program and a set of parameters to be passed.

```

7 draw.menu
/*
Subjects for Available information
% choice1
%x1 drinking water      %x6 industry
%x2 recreation          %x7 mining
%x3 nature quality      %x8 agriculture
%x4 transportation      %x9 %1
%x5 hydropower          %x10 %2
%button1 %button2
%choice1 CHOICE .CHOICE1 PAIRS KEEP ~
HELP 'draw or label available coverage feature' ~
RETURN '&r switchcheck' ~
'DRAW' 'd' 'Label' '1'
%x1 CHECKBOX .drinking-water KEEP ~
RETURN '&r draw drinking-water %.drinking-water%'
%x6 CHECKBOX .industry KEEP ~
RETURN '&r draw x'
%x2 CHECKBOX .recreation KEEP ~
RETURN '&r draw x'
%x7 CHECKBOX .mining INITIAL .TRUE.
%x3 CHECKBOX .nature-quality KEEP ~
RETURN '&r draw x'
%x8 CHECKBOX .agriculture KEEP ~
RETURN '&r draw x'
%x4 CHECKBOX .transportation INITIAL .TRUE.
%x9 CHECKBOX .other1 KEEP ~
RETURN '&r draw other1 %.other1%'
%1 DISPLAY .other1m 16 VALUE
%x5 CHECKBOX .hydropower KEEP ~
RETURN '&r draw x'
%x10 CHECKBOX .other2 KEEP ~
RETURN '&r draw other2 %.other2%'
%2 DISPLAY .other2m 16 VALUE
%button1 BUTTON KEEP 'Refresh' &r redraw
%button2 BUTTON KEEP 'Clear' &r clear
%FORMOPT SETVARIABLES RETURN
%FORMINIT &s .other1m Other ; &s .other2m Other ; &s .other3m Other

```

Fig. 9.2: an illustration of an augmentation of informational subjective to a SP schema.

Fig. 9.2 shows another kind of imposition, emphasising the relationships between the non-terminals in Fig. 8.5 and a menu subjective to available data/information.

In Fig. 9.2, we note the first line in that file begins with number '7'. Thus such a menu is a form-menu. From x1 to x8, the non-terminals in Fig. 8.5 are placed for representing the subjectives. More subjectives can also be introduced and accommodated in 'x9' and 'x10' whose correspondence to the relevant AML programs are conducted by the variables, namely, '%1' and '%2', respectively. Here, it is assumed that the non-terminals are known as subjects, but not that the relevant information has to be coupled. Therefore, types of information that should be coupled to each subject are determined separately by the menu.

To simplify our discussions, we take the coverage as an informational type. In other words, we assign the purpose of this menu to be 'drawing'. If a coverage is available to be drawn, we allow an AML program to fit that coverage into a particular subject. The subject is enabled by the command following that subject-item (i.e., drinking water). Obviously, such a set up provides a great opportunity for exploring the properties of SP. For example, a pattern-matching can be operated to assign what is available for a subject and what way the available data is to be displayed. More about a determination of informational types are discussed below.

## 9.2 Dynamic data-paths

Once the item 'governing scale' in Fig. 9.1 is enabled, operations on informational scales are displayed. These operations are to search the objects related to a general geographical region, while in the meantime, being dependent of the subjects that are available. A combination of scales can be done through the four buttons at the top. Each button is supposed to carry out a set of executions of string-matchings for finding out the maximum number of objects under the subject enabled at that time. Here we have three sets of variables named by 'choice' and distinguished by affixed numbers: 01 ~ 05, 10 ~ 15, and 20 ~ 25.

As a simple illustration of making informational types distinguished, let us again consider coverage as a type of information; and furthermore, we take an AML program as another possible path to gain information about an object. We then have Fig. 9.4, where the left side lists the available names of coverages on the objects having been selected at that time, and the right side lists the available AML programs supposed to be linked with SP programs.

Central to controlling data-path is the use of the function [EXISTS] in determining whether an object exists. The [EXISTS] function can be used to make this test, for example:

```
[EXISTS <object> {-FILE | -ADDRESS |
-ANNOTATIONS {subclass} | -ARC | -CLEAN | -COVER |
-DIRECTORY | -GRID | -INFO | -LAYER | -LIBRARY |
-LINE | -LINK | -NETWORK | -NODE | -POINT |
-POLYGON | -ROUTE {subclass} | -SECTION {subclass} |
-TAT {subclass} | -TIC | -TIN | -VAT | -WORKSPACE}]
```

The keywords in the usage indicate the object-data type whose existence can be tested. [EXISTS] returns either a value of .TRUE or .FALSE depending on whether the specified object exists.

Fig. 9.3 shows that, once the item 'governing scale' in Fig. 9.1 is enabled, operations on informational scales are displayed. These operations are to search the objects related to a general geo-graphical region, and in the meantime, being dependent of the subjects that are available. A combination of scales can be done through the four buttons at the top. Each button is supposed to carry out a set of executions of string-matchings for finding out the maximum number of objects under the subject enabled at that time. This can be done by an execution of the file in Fig. 9.4. Specifically, there are three sets of variables named by 'choice' and distinguished by affixed numbers: 01~05, 10~15, and 20~25.

```

7 governing-scale.menu
/*

%choice1

%1          %2          %3
%datalist1  %datalist2  %datalist3


%button1    %button2
%choice1 CHOICE OP SINGLE ~
  QUERY '&full &pop ; &sys arc help %op% ; &full &off' ~
  'union' 'orientation' 'intersection' 'separation'
%1 DISPLAY 'governing scale' 20 VALUE
%2 DISPLAY 'water resources' 20 VALUE
%3 DISPLAY 'instruments' 20 VALUE
%datalist1 INPUT item1 20 TYPEIN NO SCROLL YES ROWS 4 CHOICE ~
  choice01 choice02 choice03 choice04 choice05
%datalist2 INPUT item2 20 TYPEIN NO SCROLL YES ROWS 4 CHOICE ~
  choice11 choice12 choice13 choice14 choice05
%datalist3 INPUT item3 20 TYPEIN NO SCROLL YES ROWS 4 CHOICE ~
  choice21 choice22 choice23 choice24 choice05
%button1 BUTTON KEEP 'OK' &r analysis %op% %item1% %item2%
%item3%
%button2 BUTTON KEEP RETURN 'Cancel' &return ; &thread &focus &on
&all

```

Fig. 9.3: an illustration of constructing a dynamic data-paths.

### 9.3 Naming variables for a pattern-based path

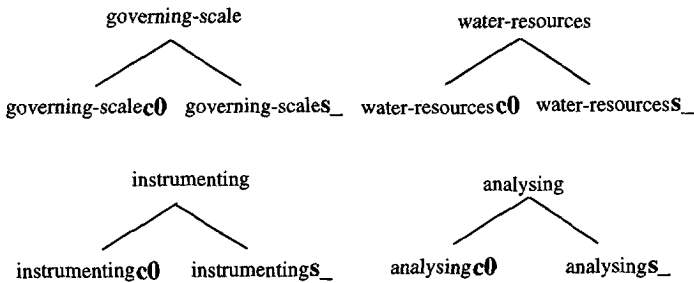
The following features of AML in naming and aggregating variables are particularly useful for generating a pattern-based path of data, information, object-scale, etc.

```

&sv ... /* a command to set a variable
%      /* representing a variable, e.g., %max%
%<variable1>%&<variable2>% /* variable1 concatenated with variable2
.<variable> /* prefixing a variable by one dot is to name that variable globally

```

Patterns can be formed by affixing a sequence of numbers or characters to a name of a variable. For example, in order to organise the following tree:



we can simply represent it by the following program

```

&do i &list governing-scale water-resources instrumenting analysing
  &s .%i% = <value>
  &s .%i%c = <value>
  &s .%i%s = <value>
  &do j := 0 &to 9
    &s .%i%c%j% = 0
    &s .%i%s%j%s = _
  &end
&end

```

where, the first loop is affixing 'c' and 's' to all the identifiers listed; the second loop is to affix '0' to those identifiers that have affixed 'c', and to affix '\_' (unknown) to those that have already affixed by 's'.

## 9.4 Other examples for organising AML programs

In this section, all the AML syntax related to the given sub-routines or programs are referred to [AML92].

### 9.4.1 Main program

```

/* A DSE: Decision Support Environment
/* An Integrated Information Platform
/*
/* dse.aml - This aml starts the demonstration of the dse. It allows
/* the user to select subjects, display coverages and inquiry of information.
/* The software system is managed by so called informational patterns;
/* it facilitates overlay, buffer, statistics, query, and selection by
/* a statistical means; it has learning capabilities.
/*
/* Directory structure: This AML assumes that the directory containing the
/*                      AMLs and Menus is named AML and is parallel to the
/* workspace          DATA directory in the users workspace. &amlpath
/*                      and &menupath are set using this directory structure.
/*
/*      workspace      |
/*      |               |
/*      |               |
/*      AML    DATA
/*
/* Arguments
/* station - the station file.
/*
/* Variables
/*
/* calls to - main.menu, draw.menu, zoom.menu
/*
/* written March 1995, Ying LIU, TU Delft

&args station
/* do some error checking.
&if [null %station%] &then
    &return Usage: &RUN INTRO <station_file>

&if [show program] ne ARCPLOT &then arcplot

/* set AML environment.
&amlpath [subst [locase [dir [path *]]] data aml]
&menupath [show &amlpath]
&station %station%

/* set ARCPLOT environment.
shadeset color
clipmapex off

/* -----
/* initialize variables, each of which is taken as a subject -
/* an object-label in SP, or a label of a virtual class which
/* collects those objects that are available to be drawn for
/* that subject
/* -----

&do i &list governing-scale water-resources instrumenting analysing
    &s.%i% = 0
    &s.%i%c = 0
    &s.%i%s = 0
    &do j := 0 &to 9
        &s.%i%c%j% = 0
        &s.%i%s%j%s = _
    &end
&end
&do i &list drinking-water recreation nature-quality ~
    transportation hydropower ~

```

```

        industry mining agriculture ~
        surface-water ground-water
    &s.%i% = .false.
    &do j &list d l
        &s.%i%%j% = .false.
    &end
    &do j &list c s
        &s.%i%%j% = %i%
    &end
    &s.other1m = Other
    &s.other2m = Other
    &s.other3m = Other
    /* &r select initchoices
    &s.whattodo d
    &s.zoom$origmapex = [show mapex]

/* Size and position menus on the screen
&r tool create main &stripe 'A DSE: Decsion Support Environment'~
    &position &above &display &pinaction '&r quit'
&r tool create draw &stripe 'Draw & Informational Tools' &position ~
    &right &display
&s.width = [extract 1 [show &thread &size draw]]
&r tool create zoom &position 1500 1500
&s.zoomheight = [extract 2 [show &thread &size zoom]]
&thread &delete zoom
&r tool create zoom &stripe 'Pan/Zoom Tool' &size %.width% %zoomheight% ~
    &position &below &thread draw
&s.zoomposition = [show &thread &position zoom]
&s.zoomsize = [show &thread &size zoom]
&s.selectposition = [extract 1 %.zoomposition%],~
    [calc [extract 2 %.zoomposition%] + [extract 2 %.zoomsize%]]
&thread &delete &self

```

#### 9.4.2 A program of thread management

```

/* DSE - a Decision Support Environment
/* TOOL.AML - Thread management tool used to create, focus and delete
/*      threads.
/*
/* arguments
/* tool - the name of the thread to perform some action.
/* action - indicates the action to perform on the named thread.
/* modifier - a modifier for the action
/*
/* written March 1995, Ying LIU, TU Delft

&args action tool modifier:rest
&call %action%
&return

/*-----
&routine create
&if ^ [show &thread &exists %tool%] &then &do
    &thread &create %tool% &menu %tool% [unquote %modifier%]
&end
&return

/*-----
&routine delete
&if [show &thread &exists %tool%] &then

```



```
&thread &delete %tool%
&return
```

### 9.4.3 A program for managing drawings

```
/* A DSE: Decision Support Environment
/*
/* draw.aml - turns such a coverage on or off that is available
/*      for drawing, labeling and retrieving information
/*
/* arguments
/* object - the object to draw. This will be one of the subjects,
/*      such as drinking-water recreation, other1, other2, other3.
/* onoff - either .true. or .false. as checked or unchecked in draw.menu.
/*
/* Variables - The basic variable structure used to store information
/* as to the current drawing state of a given coverage is described
/* below. Three variables are used for each individual coverage.
/*
/* .subject - the variable associated with the checkbox in draw.menu
/* .subjected - boolean indicating whether the cover is drawn
/* .subjectl - boolean indicating whether the cover is labeled.
/* .subjectc - the coverage name (for consistency with other covers)
/* .subjects - how to symbolize the coverage. Values can be: landuse,
/*      roads, soils, streams, sewers.
/* There are similar variables for each coverage: soils, streams, etc.
/*
/* "other" variable definitions (1 can be replaced with 2 and 3 as well)
/* .other1 - boolean representing checkbox in draw.menu
/* .other1d - boolean for coverage drawing.
/* .other1l - boolean for coverage labeling.
/* .other1c - the name of the coverage for drawing or labeling.
/* .other1s - how to symbolize the coverage. Values can be:
/*      landuse, road, stream, sewer, soil, default.
/* .other1m - holds the word "Other" or the name of the coverage for
/*      display in draw.menu
/* .whattodo - d or l. Set in draw.menu indicating that coverage
/*      feature or labels are to be drawn.
/*
/*
/* called by - draw.menu, drawother.aml
/* calls to - drawcover.aml, redraw.aml
/*
/* written November 1994, Ying LIU, TU Delft
```

```
&args object onoff
```

```
&if [keyword %object% drinking-water recreation nature-quality ~
      transportation hydropower industry ~
      mining agriculture surface-water ground-water ~
      other1 other2 other3] < 0 &then
&return &inform Invalid OBJECT %object% (draw.aml).
```

```
/* set the draw indicator to .true. or .false.
/* (e.g., &s .landused = .true.)
&sv %object% .whattodo% = %onoff%
```

```
&if ^ %onoff% &then
&r redraw
&else
&r drawcover %object%
&return
```

### 9.4.4 A sub-routine of analysis

```

/* A DSE: Decision Support Environment
/* an integrated information platform
/* analysis.aml - initializes the analysis menu and performs some
/* operations to construct informational patterns
/*
/* arguments
/* op - the routine to call
/* menu-node - the tree-node generating the relevant information
/* menu-item - the tree-leaf allowing information to be merged
/*
/* called by - main.menu
/* calls to - analysis.menu, msworking.aml msinform.aml
/*
/* written April 1993, by Ying LIU, TU Delft

&args op node
&call %op%
&return

/*-----
&routine init
&if %node% = scale &then &call scale-fix
&else &do
    &if %node% = resources &then &call resources-types
    &else &if %node% = instruments &then &call instruments-types
    &end
&s .display11 = %.nextnode01%
&s .display12 = %.nextnode02%
&r tool create %node% &stripe 'Analysis Identification' &position ~
    &below &display &pinaction '&thread &focus &on &all'
&r focus one %node%
&return

&routine relatedinfo
&r tool create %.nextnode01% &stripe 'Available Objects' &position ~
    &below &display &pinaction '&thread &focus &on &all'
&r focus one %.nextnode01%
&return

/*-----
&routine scale-fix
&s .choice01 = international
&s .choice02 = national
&s .choice03 = regional
&s .choice04 = district
&s .choice05 = 'operation board'
&s .choice06 = 'public members'
&s .choice07 = 'private members'
&s .nextnode01 = water
&s .nextnode02 = policy
&return

/*-----
&routine resources-types
%node% = resources &then
    &do
        &s .nextnode01 = scale
        &s .nextnode02 = policy

```

```

&end

/*-----
&routine instruments-types
&else &do
    &s .nextnode01 = water
    &s .nextnode02 = scale
&end
&end

/*-----
&routine generic
/* generic overlay operations
&s cover2 [extract 1 [unquote %more%]] /* the overlay coverage
&s outcover [substr %op% 1 2]~
[substr [entryname %cover1%] 1 3][substr [entryname %cover2%] 1 3]
&if [exists %outcover% -cover] &then
    &call exists
&run msworking init Creating [locase %outcover%],~
the [locase %op%] of [entryname %cover1%] and [entryname %cover2%].
&if [locase %op%] = union &then
    &s covtype =
&else &do
    &if [exists %cover1% -poly] &then
        &s covtype = poly
    &else
        &s covtype = line
&end /* else do
&sys arc %op% %cover1% %cover2% %outcover% %covtype%
&run msworking exit
&return

/*-----
&routine exists
&r msworking init Preparing workspace for overlay operation.
&sys arc kill %outcover%
&r msworking exit
&return

```

## 9.4.5 Zoom menu

```

7 zoom.menu
/*
%1 %2 %3 %4 %6
%1 BUTTON ~
ICON corners32.icon ~
HELP ' set corners' ~
'SET CORNERS' &r adi_zoom setcorners
%2 BUTTON ~
ICON zoomin32.icon ~
HELP ' zoom in' ~
'ZOOM IN' &r adi_zoom zoom_in
%3 BUTTON ~
ICON zoomout32.icon ~
HELP ' zoom out' ~
'ZOOM OUT' &r adi_zoom zoom_out
%4 BUTTON ~
ICON pan32.icon ~
HELP ' pan' ~
'PAN' &r adi_zoom pan
%6 BUTTON ~

```

```
ICON orig32.icon ~
HELP ' back to original' ~
'RESET' &r adi_zoom reset
```

### 9.4.6 Quit

```
/* A DSE: Decision Support Environment
/* Quit.aml - aml to terminate the integrated information platform.
/*
/* Called by - Main.menu
/*
/* Written November 1994, Ying LIU
/*
&do i &list draw select zoom
    &r tool delete %i%
&end
&dv .*
quit
&thread &delete main
```

## SUMMARY

---

Functionally, decision-support environment (DSE) is a set of software tools where users are enabled to share the knowledge, information and/or data in order to carry out a decision-process or make a decision in a collaborative manner.

Water resources management concerns the problems of discrepancy which occur when we compare the information of the major components of water systems with established criteria to predict and understand the impacts of any action taken to control, manage and use water. In this field, collaborative relations have been, for many years, actively stimulated between groups, such as managers, system analysts, policy-makers and decision-making bodies.

For the applications of DSE in water resources management, it is clear that DSE has to do with the knowledge of water as standing reserves on one hand, and on the other, to impart the knowledge that the information conveys. Accordingly, we have to be concerned with the issues underlined by the following two questions:

- in which sense do tools support the decision-making in water resources management ?
- how to encapsulate existing and newly appearing tools into the relevant application-domains?

The relevant studies are undertaken by two strategies.

Firstly, it is noticeable that problem solving and decision making take place simultaneously in the context where some human action might be required in order to achieve our desired objectives or goals. Under system thinking, these contexts are treated as the externals of DSE labelled by a) P-D: physics-descriptive environment, b) O-O: objective-optimal environment, and c) P-R: politics-rational environment.

Then, the notion 'object' is understood by 'some thing' that is rather 'dynamic', i.e., for one situation, some attributes are aggregated as an object to be studied; for other situations, a different collection of physical properties have to be considered in one class. The notion 'pattern' means metaphors of decision-making primitives for the relevant object-orientations to be reflected and accommodated. This theme has a place in AI (artificial intelligence); but the adaptation is quite different from the ability to 'know' an environment, at least not in the way that the knowledge is traditionally understood by 'strong' AI.

Secondly, it is highly desirable to expanding the potentials of *geographical information systems* (GIS) for building DSE. Then we need to have an open platform of GIS which allows us to flexibly store information and stabilise GIS for the information. To develop such a platform, the following computer-based impacts are indeed crucial:

- software tool,
- conceptual schema,
- data model.

However, it is found that each impact mentioned above has functional limitations on itself; but all applied together can address most types of problems and better than either can alone.

Then, the notion 'pattern' means an ensemble of *information* primitives which are originated not only by an object, but also by the intrinsic aggregations among the objects, such as salience, overlap, fuzziness, etc. Together with the first strategy, such an organisation is characterised as *pattern-directed*.

To entail this pattern-directed approach by technical means, considerations are given to the issues, including *domain-driven data model*, *storing and retrieving data* and *programming*.

*Domain-driven data model.* The model has three dimensions, namely, spatiality, attribute and eventuality (eventuality is not discussed at length). Cases are shown that such a model can be characterised by a sequence of the dimensions, and an order of the dimensions. This idea is in line with recent views of software engineering: away from a computer-oriented view - how to implement a solution on the computer, toward a problem (or knowledge-oriented view) - how to capture the problems.

*Storing and retrieving data-entity.* In many situations, the main purpose is to *split* a large number of objects into a number of 'homogeneous' groups on the basis of multivariate observations. A type of data entity is then formed by a set of samples; moreover, that entity has to be dynamically stored in a database, because a retrieval of the entity is dependent of a context where the entity is to be used. A function of multi-index is developed for a data entity to access (or be accessed by) relational databases.

*Knowledge acquisition.* Since the knowledge has to be stored in a knowledge-based system, which may involve a variety of physical applications and theoretical analysis, a repertory grid method is established for listing a set of elements across the top of the grid. Such a list represents the objects to be classified. Various combinations of objects can be presented to a network of computing. This method seems to significantly improve the time spent in eliciting expertise from domains' experts.

*Effective knowledge representation.* An effective conceptual schema is a model which is not only capable of describing the knowledge structures, but also of containing programming details that are not part of the knowledge structures, but are necessary to make the model run with the available technology.

*Implementations.* Programs are written with the help of ARC/INFO that is a set of software tools for building GIS. AML is a language for organising a sequence of ARC/INFO commands into geoprocessing operations. We use AML for constructing pattern-directed data-paths, and dynamic menu, i.e., when a certain category of data is available, an item in a menu is capable of loading data.

## EVALUATION

---

DSE is a kind of software environment where users are enabled to share the knowledge, information and/or data in order to carry out a decision-process or make a decision in a collaborative manner.

Our study shows that there is a genuine need for developing DSE as an engineering object.

## Conclusions

Recent advances in computer technology have opened many possibilities enabling us to manage the knowledge, information and data for improving the quality and efficiency of decision making.

We view WRM as an application to DSE, and have found that both inevitably stand in stark contrast in object-oriented disciplines. That is, the management does require relative stability in the view of the object, but the object-orientation is seen to be multifaceted and multifarious, because, in WRM, a decision process is often carried out with sharing multi-knowledge domains. While the DSE, on the other hand, assumes that 'the object' is itself something rather obvious, with well-known and well understood properties. Thus, it is essential to search, revise and mechanise pertinent object-oriented concepts, such as class-and-object, within a problem domain and a system's responsibility, because the object is rarely 'ready somewhere just for picking and using'.

Domain-driven object oriented concept is emerged to enable us to allocate the contexts of data or information. A pattern-directed approach is established to manage those domain-driven objects. It seems to be a potentially rich, i.e., it directs software systems to be constructed in serving between object-oriented programming and domain-driven data model; it is suitable to parallel processing and interactive user-interfaces; and moreover, it has the following simplicities.

First, it does not provide a type of complicated calculation, mathematical formulation or method which is not generally used.

Second, it provides a suitable representation for weakly and unstructured knowledge both from epistemic and manipulation points of views.

Third, it offers a bridge to methods of the pattern-string representations, widely used in coding, for extraction of knowledge contexts.

Fourth, it leads to the pattern-view that is very open-ended for knowledge acquisition and dialogue methods.

## Limitations and recommendations to the future work

The notion 'intelligence' means - the power of appropriate selection for not only sticking to a goal, but also the ability at the avoidance of breakdown as well as at the handling of breakdown as soon as it takes place.

Our work is limited by categorising the decision making primitives which have been viewed as a set of factors that cause an enlargement of the gaps between the levels of decision processes. Further considerations should be given to the applications of artificial intelligence (AI for short), such as

- a) decision making environments can be taken as DSE *references*.
- b) intelligent operations can be maintained inside a medium of perturbations by changing a distinctive reference from one kind of decision-making environment to the other.
- c) the groups who are involved in a decision process can be led to *prefer* knowledge over ignorance with, again, switching the references between groups' functional space. Therefore, an intelligent DSE can be regarded as a computing station plus computational couplings with the references.
- d) such a network takes *intelligence* to reverse that preference, to know the arbitrariness of the preference, and to assume itself to be in a position of ignorance.

The selectivity of a decision-support system can be formed and be balanced by the corresponding processing effort required to produce it. This will provide us with a variable selectivity approach to the processing of non-routine or non-numerically interpreted data. Thus, further developments should be based on not only a hierarchical structure of concepts such as a set of indicators, but also utilising linguistic and pattern recognition ideas and techniques, and especially the weighted distance of strings.

An event of pattern-matching should be represented by a string. Such a string can be evaluated as a code by using the methods of decoding or further-matching in searching for differences in the string, looking for similar objects, or objects combinations. We can use evaluation function, e.g., linear-combinations of scaling values weighted by relevance and reliability data; and we can use methods of multi-variant analysis, if sufficient data are available for some significant statistical reasoning.

In respect to re-use the allocated expertise for the policy-assessment in the future, we expect the links between the data communicated and the data to be communicated, where the data should have something to do with policy-utilities. The following theme is recommended:

- Iconing municipality of water-authorities on a map.
- Making it possible to browse a set of visible objects related to a policy-scenario.
- Ordering the items in those tables that are associated with the selected objects.
- Allowing the communications to occur between the icons by
  - a) requesting the data which are missing in an item;
  - b) requesting the data which are to replace the data in an item;
  - c) adding a new item to a table, so that
  - d) a new category of data can be requested either by a) or b).

By these ways, DSE is perhaps capable of collecting information about measured features of an object that is being understudied at a specific decision mode; that all the information used by a local recognition algorithm at that specific decision node is collected as a segment; that the database in multistage pattern recognition includes a meta-data and a leaning set, i.e., the meta-data represents the decision tree, global and



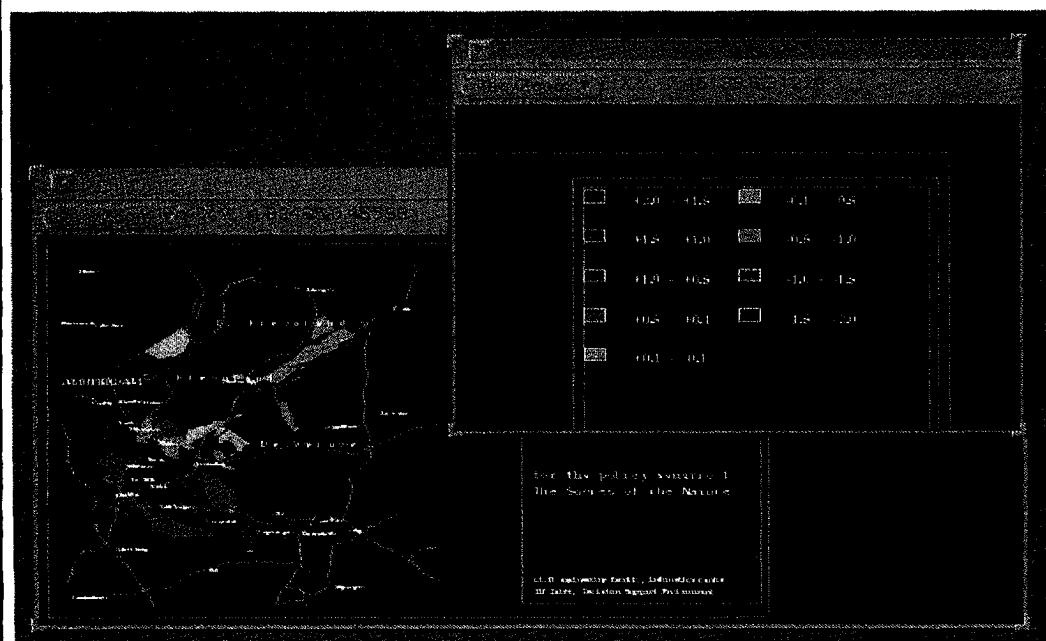
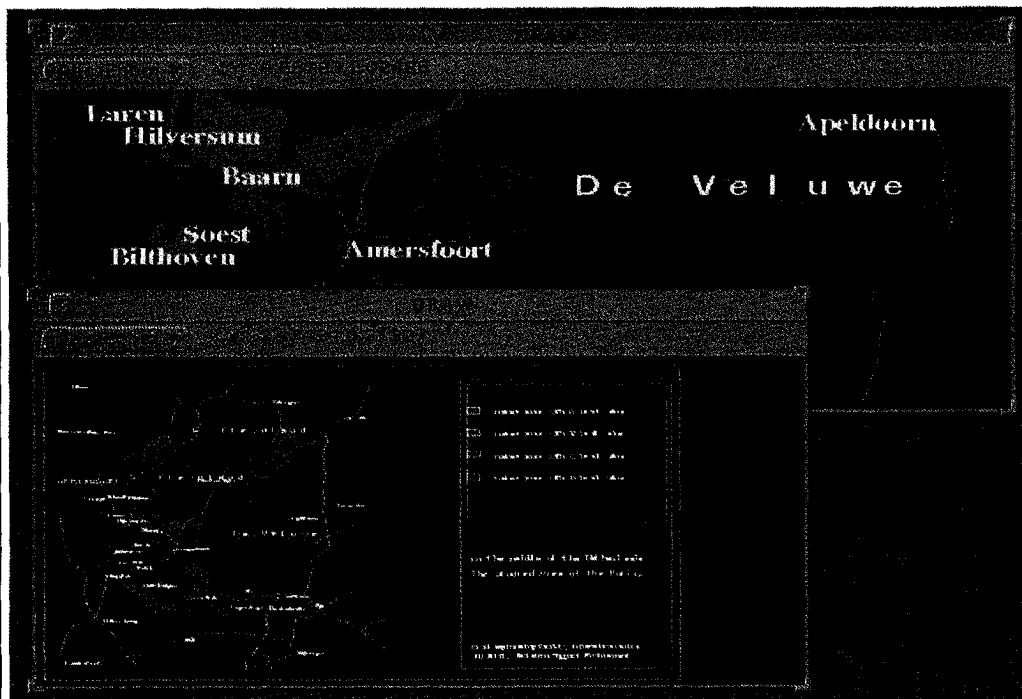
local data views and relationships in decision-making, and finally, that the meta-data and learning sets are constructed in a decision-process performed.

In a consideration of integrating models, optimisation models in particular, the existence of a strong theoretical foundation nurtured and strengthened the discipline of database management significantly. Model management research is now at about the equivalent stage of evolution with respect to languages. However, the following questions remain:

- Can a calculus or algebra be devised for particular classes of models which exhibit the property of transitive closure as in the relational model ?
- What constitutes a set of primitives for a model manipulation language ?
- Is there a set of necessary and sufficient abstract data model and inheritance hierarchies for modelling ?
- Will model manipulation language be built in a constructive mode or will they evolve from an appropriate theoretical foundation ?

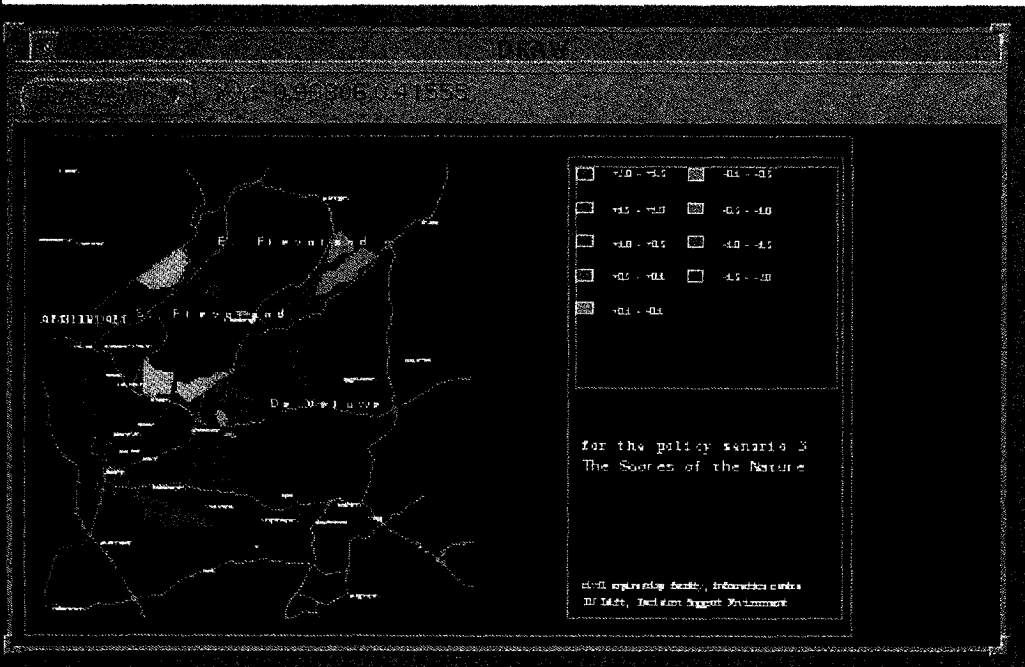
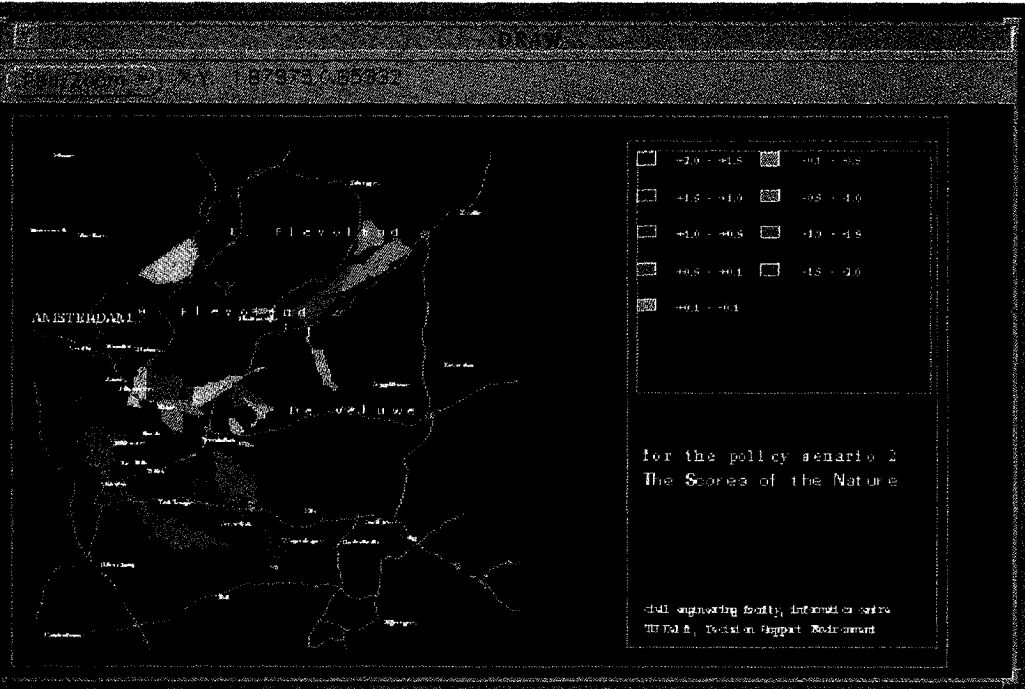


## Visualised Scenario - alternative 2a/b

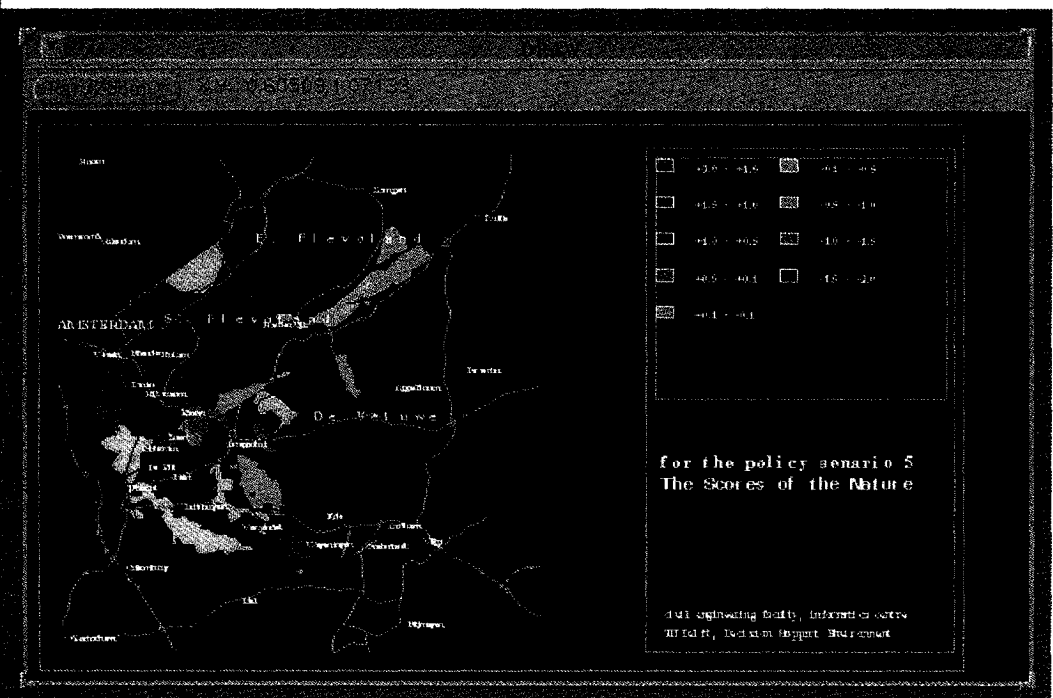
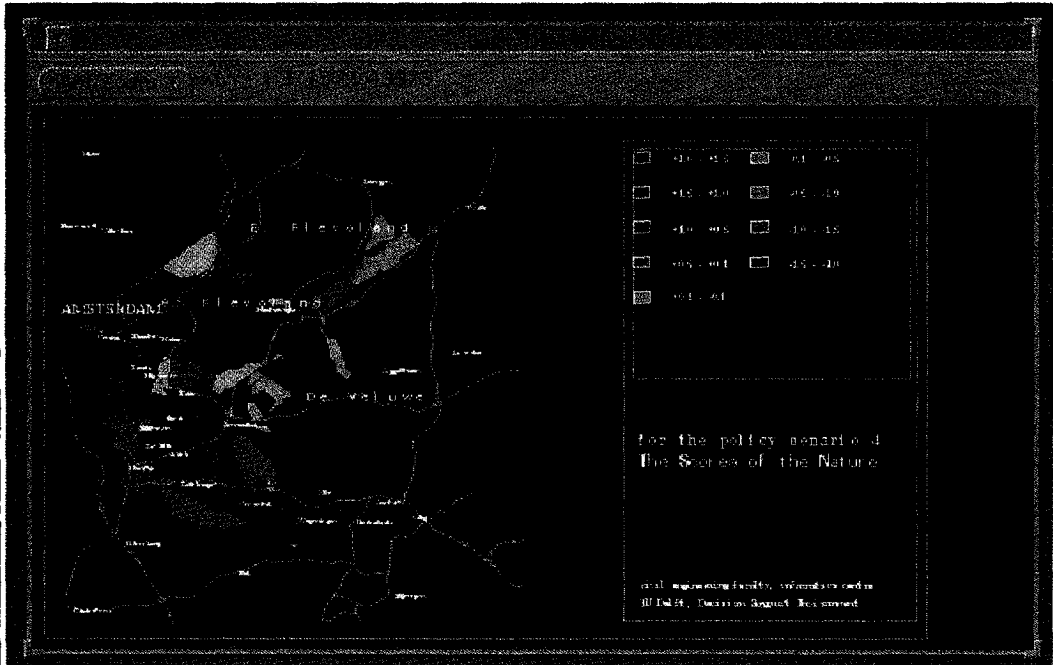




Visualised Scenario - alternative 3a/b



Visualised Scenario - alternative 4/5



## APPENDIX 2

### Tables of Survey

S : a software tool or software environment;

E : an expert system;

M: an intelligent system (expert system) plus software tools;

For surface water managers

DSS Developments for Surface-water Managers					
functional requirement		design approaches		references	status
		types	design methods		
Engineering and managing basins of river, lake, reservoir, etc.	Water source operational control	S	Dynamic simulation, Model-based	[Walsh89]	model
	River flows operational management	S	Numerical modelling packages, model-based	[Moll89]	model
	Drout management	E	Natural language interface for expert syst.	[Palmer89]	model
	River basin management	M	Integrating graphical tools with model and rule based	[Meijerink93]	architecture
	Enviornmental hydraulics engineering	S	Object-oriented, graphical tools, databases	[Simonovic93]	architecture
Controlling flood	Urban flood control	M	Graphical tools, knowledge and data bases inference engine, knowledge	[Ruland93]	architecture
		E	Rule and model based, qualitative simulation	[Cuena89]	theoretical
Protecting natural environment	waste water treatment	M	Graphical tools, databases and knoweldge bases	[Frysinger93]	proposals

For groundwater managers

DSS Developments for Groundwater Managers		
functional requirement	design approaches	
	types	design methods
Managing water pumping for domestic, industrial and agricultural supply; engineering hydraulic construction; managing influential regional water cycle.	S	Integrating graphical tools, spatial databases, and spatial data analysis
Modelling ground water contaminance	S	Model-based, integrating interfaces implemented by graphical tools, object-oriented and hypertexts
		[Furst93]
		[Newell90]

For water managers in large-scale

DSS Developments for Water Resources Managers in Large-scale				
functional requirement		design approaches		references
		types	design methods	
conjunctive ground water/surface water modle and impact trade-off analysis for evaluating water tranfer		M	Graphical tools integrated with model-based or other knowledge-based methods	[Stansbury91]
Management of ecohydrological environment of catchment area		M	Graphical tools integrated with fuzzy expert models	[Droesen93]
Natural environment protection		S	Graphical tools, information management	[Frysinger93]
Environmental hydraulics management		S	Graphical tools, object-oriented and data manipulation	[Ruland93]
General water management	water management	S	Model-based, numerical simulation	[Kadar89]
	analysing water resources policy	S	Model-based, numerical simulation	[Mei89]
	regional planning	S	Model-based and data management	[Linden89]
				case study
				model
				case study proposals
				architecture case study
				models
				case study and models
				case study, architecture



## APPENDIX 3.1

### Selected Concepts of Decision Problems

#### Decision table

Generally, one of the basic assumptions held is that only a finite number of mutually exclusive states are possible; that only a finite number of actions are available; and, that one and only one of these actions must be chosen. Thus, we label these states as  $s_1, s_2, \dots, s_n$ , and, similarly, actions as  $a_1, a_2, \dots, a_m$ . Letting  $x_{ij}$  be the consequence of taking action  $a_i$  when  $s_j$  is the true state, we have a decision table, see Table A.3.1. The symbol  $x_{ij}$  stands for a complete, holistic description of the possible consequence. If the decision problem involves monetary outcomes, the  $x_{ij}$  may be single numbers; but, in general, they are not so.

Table A.3.1: a general decision table

Consequences	States of nature			
	$s_1$	$s_2$	...	$s_n$
$a_1$	$x_{11}$	$x_{12}$	...	$x_{1n}$
$a_2$	$x_{21}$	$x_{22}$	...	$x_{2n}$
Actions	.	.	...	.
	.	.	...	.
	.	.	...	.
	.	.	...	.
$a_m$	$x_{m1}$	$x_{m2}$	...	$x_{mn}$

#### Interpretation of consequence - value and decision

However, although the  $x_{ij}$  are not necessarily numbers themselves, we can assume that the decision maker can value them numerically, i.e., the decision maker can measure the value of  $x_{ij}$  to him through some real-valued function  $V(\cdot)$ . By "measurable value", we mean that  $V(x_{ij}) > V(x_{kt})$ , if and only if the decision maker would prefer the consequence  $x_{ij}$  to the consequence  $x_{kt}$ . Letting  $V_{ij} = V(x_{ij})$ , the general form of a decision table becomes the one shown in Table A.3.2.

Table A.3.2: a general decision table with the consequences interpreted by their values

Consequence(Value)	States of nature			
	$s_1$	$s_2$	...	$s_n$
$a_1$	$V_{11}$	$V_{12}$	...	$V_{1n}$
$a_2$	$V_{21}$	$V_{22}$	...	$V_{2n}$
Actions	.	.	...	.
	.	.	...	.
	.	.	...	.
	.	.	...	.
$a_m$	$V_{m1}$	$V_{m2}$	...	$V_{mn}$

Note, consequence, as an item of the decision table, has been not just replaced by new labels, but interpreted as *measurable* preferences of a decision maker.

Indeed, our main concern is for such a kind of interpretation suitable to measurement.

On one hand, some decision problems are absolutely crucial to human kind, and measurement is truly essential; it would be wise to explore this model-driven approach, rather than completely ignoring it.

But, on the other hand, it is impossible to wait for a complete interpretation to have a "super" decision table before a decision problem at hand can be tackled. We shall discuss this concern more explicitly shortly after reviewing a data-driven approach.

## Interpretation of state - certainty and decision

*Decision under certainty.* Here it is assumed that the true state is known to the decision maker before he has to make his choice; i.e., he can predict consequences of his actions with certainty. In effect, this is equivalent to assuming that  $n=1$ . Thus, for him the decision table has the trivial form shown in Table A.3.3.

Because the  $V_{ij}$  increase with increasing value of the consequences to the decision maker, the optimal choice is, of course, to pick an action with the highest numerical value of  $V_{ij}$ . Decision problems under certainty are, therefore, very straightforward: how should the  $V_{ij}$  be measured to fairly and truthfully represent the decision maker's preferences?

Table A.3.3: a decision under certainty represented by a state

Values	State	
	$s_1$	
	$a_1$	$v_{11}$
	$a_2$	$v_{21}$
Actions	.	.
	.	.
	.	.
	$a_m$	$v_{m1}$

Table A.3.4: decision under risks

Consequences	States of nature			
	$s_1$	$s_2$	...	$s_n$
$a_1$	$x_{11}$	$x_{12}$	...	$x_{1n}$
$a_2$	$x_{21}$	$x_{22}$	...	$x_{2n}$
Actions	.	.	...	.
	.	.	...	.
	.	.	...	.
$a_m$	$x_{m1}$	$x_{m2}$	...	$x_{mn}$
Probabilities of states				
	$P(s_1)$	$P(s_2)$	...	$P(s_n)$

*Decisions with risk.* Although the decision maker does not know the true state of nature for certain, he can quantify his uncertainty through a probability distribution:  $(P(s_1), P(s_1), \dots, P(s_1))$ . We have Table A.3.4. If the  $V_{ij}$  are measured in a certain way and if the decision maker is prepared to accept a certain definition of rationality, he should choose  $a_j$  to maximise the sum:

$$\sum_{j=1}^n P(S_j) \cdot V_{ij}$$

This sum is known as the expected utility of  $a_i$ .

## Decision tree - a representation of a multi-stage decision problem

The decision table representation of a decision problem is static. It pretends that there is only one point of choice. The decision maker has to decide between actions  $a_1, a_2, \dots, a_m$ . One and only one must be chosen; and, when chosen, the decision is made, the problem "resolved".

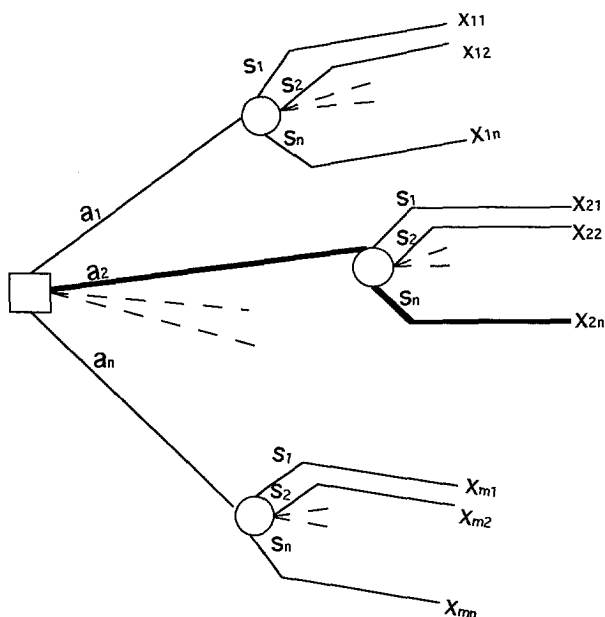


Fig. A.3.1: a general decision tree of Table A.21.1 (decision table)

Consider dynamic decision problems formally: one decision problem leads to another, and that to another, and so on. Moreover, the problems are interrelated: the possible actions and the possible states in later problems may depend upon which actions were chosen in earlier problems. We call these problems *multi-stage decision problems*. Fig A.3.1 shows a general decision tree representation of the same problem represented by the decision table (Table 2.1). The square, or decision point or node, at the left of the tree, represents the decision maker's choice between  $a_1, a_2, \dots, a_m$ ; each branch of the tree stemming from this point represents an action. At the end of each action's branch is a chance point or node, represented by a circle, at which each branch subdivides into  $n$  further branches, one for each possible state. At the end of these are the ultimate consequences,  $x_{ij}$ . Thus, the branch in the tree shown as a thicker line represents the choice of action  $a_2$  with the subsequent occurrence of  $s_n$ , which leads to the consequence  $x_{2n}$ .

Given the decision tree format, it is a simple matter to represent subsequent decisions by introducing further decision points later in the tree. The possibility that certain choices may only be relevant if certain states occur is easily represented by introducing decision points only into the relevant branches.



## APPENDIX 3.2

### Model and Modelling - a Philosophical Perspective

#### The concept of model

In order to "see" the complete system  $S$ , we would need an infinite number of *observables*  $f_{\alpha}: \Omega \rightarrow \mathfrak{R}$ , where  $\alpha$  ranges over some possibly uncountable index set, and  $\mathfrak{R}$  is a complete set of real numbers.

Thus, we may assume that the complete system  $S$  is described by  $W$  and the entire set of observables  $F = \{f_{\alpha}\}$ . But, obviously, this is a typical assumption, because, for modelling purposes, it is often inconvenient to work with such a large set of observables. So we boldly just throw most of them away and focus our attention on a proper subset  $A$  of  $F$ . We call  $A$  an abstraction of  $S$ , since the view we have of  $S$  using the observables of  $A$  is necessarily a *partial* view formed by abstracting, i.e., throwing away all the information contained in the observables  $F - A$ .

It is an amusing aside to wonder why self-styled practically-oriented people reserve their greatest scorn for "useless abstractions", when the very essence of an abstraction is to reduce the description of a system to a simpler, and presumably more tractable form.

We are now in a position to put forth the notion of a *natural system*  $N$ . For us,  $N$  consists of an abstract state-space  $W$ , together with a finite set of observable  $f_i: \Omega \rightarrow \mathfrak{R}$ ,  $i = 1, 2, \dots, n$ . Symbolically,

$$N = \{\Omega, f_1, f_2, \dots, f_n\}$$

We employ the term "natural" to distinguish  $N$  from the idea of a "formal" mathematical system to be briefly discussed below. It should not be interpreted to mean restriction of  $N$  to the class of systems studied in the "natural" sciences (hydrology, physics, astronomy, etc.); it includes social, behavioural and living systems as well. In all that follows, we shall use the term *natural system* in this extended sense.

#### Causality and determinism in modelling

Often, the essential system nature of  $N$  is contained in the relationships linking the observables

$$f_1, f_2, \dots, f_n$$

We term such a set of relationships the *equation of state*, or the *description*, for  $N$ . Formally, the equations of state can be written as

$$\Phi_i(f_1, f_2, \dots, f_n) = 0, \quad i = 1, 2, \dots, m,$$

where the  $\Phi(\Sigma)$  are mathematical relationships expressing the dependency relations among the observables. We can write this more compactly as

$$\Phi(f) = 0$$

#### Example

Let  $N$  consist of one mole of an ideal gas contained in a closed vessel. Take  $s$  to be the positions and velocities of molecules making up the gas, and define the three observables

$P(s)$  = pressure of the gas when in state  $s$ ;

$V(s)$  = volume of the gas when in state  $s$ ;

$T(s)$  = temperature of the gas when in state  $s$ .

Then the ideal gas law asserts the single equation of state

$$\Phi(P, V, T) = 0, \quad \text{where} \quad \Phi(x, y, z) = xy - z$$

The example, simple as it is, serves to illustrate the distinction between causality and determinism. An equation of state  $F(f)=0$ , necessarily establishes a deterministic relationship between the observables. But it contains no information whatsoever about any possible causal implications among the elements of the set  $\{f_i\}$ . Thus, the ideal gas law asserts that once we know any two of the three observables,  $P$ ,  $V$  and  $T$ , the remaining observable is determined by the other two, but not *caused* by them.

It is relatively easy to have theoretical and/or empirical equations of states, but hard to have much knowledge of how to separate observables into "subsets". Of course, the same obstacle may stand in the way of making effective use of a system-theoretic technique in many sub-domains of problems. The difficulty in assigning causal ordering to the set of observables is one of the principle difficulties in understanding water issues.

Now, let us have the last group of notions to be introduced in the P-D, which are often used in water-resources management also.

## Parameters, inputs and outputs

Imagine that there are  $r$  observables that remain constant for every state  $s \in W$ . For simplicity of exposition, assume there are the first  $r$  observables. Such quantities are usually termed *parameters*, and arise as a matter of course in virtually all natural systems.

If we let  $f_i(s)=a_i$ ,  $s \in W$ ,  $a_i$  is a real number,  $i = 1, 2, \dots, r$ . We can write the equation of state as

$$\Phi_{\alpha_1, \alpha_2, \dots, \alpha_r}(f_{r+1}, f_{r+2}, \dots, f_n) = 0 \quad (\dagger)$$

which indicates explicitly the dependence of the descriptions upon the parameter values  $a_1, a_2, \dots, a_r$ . In other words, we have an  $r$ -parameter family of descriptions, and for each set of values of the set  $\{a_i\}$ .

We now introduce the additional assumption that the last  $m$  observables  $f_{n-m+1}, \dots, f_n$  are functions of the remaining observables  $f_{r+1}, \dots, f_{n-m}$ , i.e., we can find relations  $y_i(\bullet)$ ,  $i = 1, 2, \dots, m$ , such that

$$\begin{aligned} f_{n-m+1}(s) &= y_1(f_{r+1}(s), \dots, f_{n-m}(s)), \\ &\vdots \\ f_n(s) &= y_m(f_{r+1}(s), \dots, f_{n-m}(s)) \end{aligned}$$

If we introduce the notion

$a$  for  $(a_1, a_2, \dots, a_r)$ ,

$u$  for  $(f_{r+1}, f_{r+2}, \dots, f_{n-m})$ , and,

$Y$  for  $(f_{n-m+1}, f_{n-m+2}, \dots, f_n)$ ,

the equations of the state (†) become

$$F_a(u) = Y \quad (\dagger\dagger)$$

In (††), it is natural to think of the observables  $u$  as representing the *inputs* to the system, with the observables  $Y$  being the resulting *outputs*.

The vector  $a$  is, as before, the set of parameters. Note, an assumption is actually held, i.e., we can solve for the observables  $f_{n-m+1}, f_{n-m+2}, \dots, f_n$  in terms of the observables  $f_{r+1}, f_{r+2}, \dots, f_{n-m}$ . That is to say that causality must be introduced into (†).





## Explanations of Knowledge Domains in Terms of Systems

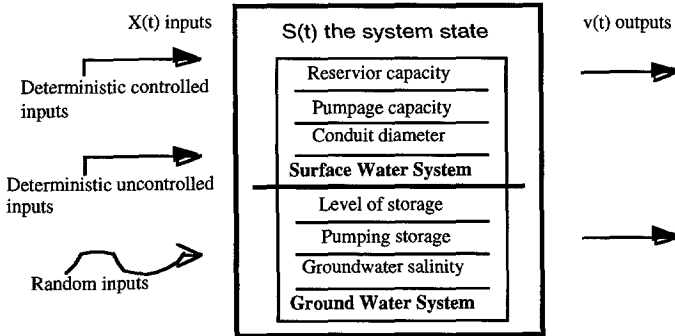


Fig. A.4.1: a state of the system considered

Let us consider these criteria where groundwater is either the only source of water or the major component of a regional water supply, and further, consider an integrated groundwater-surface water system, or simply, an aquifer as a separate unit independent of a regional water supply. This approach requires operating rules such that the system is operated in an optimal manner. These are generally determined through some economic or social objective associated with uses for which the water is put, e.g., maximum yield, safe yield, or yield required to meet objectives.

Fig. A.4.1. shows the relations between inputs, or resources needed for a production process, and outputs. The inputs are identified in terms of replenishment, both natural and artificial. Artificial replenishment is often deterministic and subject to control, whereas natural replenishment is not. Outputs include pumpage, a controllable variable, and outflows, such as spring discharge, which is generally not directly controllable.

If an operating rule is to be formulated in terms of the state of the system, the state must be known at all times. State variables are separated into those which describe the physical components of the system, such as pumping or conduit capacity, and those which describe the instantaneous level of operation, such as storage volume. Any transformation from state  $S(t)$  to state  $S(t + \Delta t)$  one time period later will depend on uncontrollable parameters, such as natural recharge; parameters which are determined by the system's state itself such as a reduction or increase in outflow or spring discharge; and parameters subject to controls, such as pumping and artificial recharge rates. For continuous time, these variables are related to each other by the continuity equation

$$\frac{dS}{dt} = R_N(t) + R_A(t) - X(t) - Q(S) \quad \text{Eq.A.4.1}$$

where  $t$  is time,  $S$  is storage in the aquifer,  $R_N$  is natural replenishment,  $R_A$  is artificial replenishment,  $X$  is pumpage, and  $Q$  is outflow or spring discharge, here considered as a function of storage.

In this formulation,  $S$  is a state variable,  $R_A$  and  $X$  are decision variables,  $Q$  is uncontrollable (directly) output, and  $R_N$  is uncontrollable input.

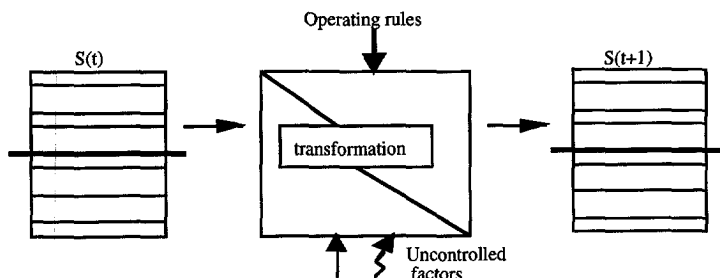


Fig. A.4.2: input and output relations for discrete time state transformation

For discrete time, the transformation from state  $S$  to the state  $S(t + \Delta t)$  on time period later depends on the control policy (pumping and artificial-recharge rates) to be executed over this period, as well as uncontrollable factors. Eq A.4.1 becomes

$$S_{t+\Delta t} - S_t = \Delta t [R_{Nt} + R_{At} - X_t - Q(S_t, S_{t+1})] \quad \text{Eq A.4.2}$$

where  $\Delta t$  is the length of a season, and the subscript  $t$  corresponds to the number of the season in a sequence of seasons. Fig. A.4.2 shows the relationships in Eq A.4.2.

The parameters subject to control in the above cited problem constitute the decision variables.

The objectives of water resource management must be classified in groups with considerable overlapping such as separating (and unifying) water-body and water-use-body. This could be expressed by

$$\max R_N = \max [(Q_t - Q_{t+\Delta t})\chi] \quad \text{Eq. A.4.3}$$

where  $R_N$  is captured, or effective, natural discharge from time  $t$  to  $t + \Delta t$ ;  $Q_t$  is natural discharge at time  $t$ ;  $Q_{t+\Delta t}$  is the difference in natural discharge over the time period later; and  $Q - Q_{t+\Delta t}$  is the difference in natural discharge over the time period as a function of the control vector  $\chi$ . It is the vector  $\chi$  to be adjusted in accordance with the objectives of Eq A.4.3.

The simple message of Eq A.4.3 is that the maximum possible value for recharge over a given time period is achieved by maximising the difference in natural discharge over this period; Or, in other words, a minimum change in storage is desirable from the viewpoint of stability of production, i.e., the control vector  $\chi$ .

Linear stability analysis at such a point reveals that, in general, only a few collective modes of the system become unstable and tend to grow exponentially whereas all the other modes remain stable. It can be shown that the behaviour of the total system at such instability points is dominated by those modes which tend to become unstable. They serve as so-called order parameters. The stable modes or short living systems are slaved by the order parameters. For this reason, it is sufficient to consider the behaviour of the order parameters alone; but this consideration does not allow the system and subsystems to be left alone, because, the processes are mostly associated with data and knowledge.

### Data-entities in Pragmatic Terms

#### Experimental data

Most data result from experiments and simulations.

Data from experiments are usually measurements of some water phenomena. Data from simulations typically result from complex computations derived by using values from the previous time interval. Both experiment and simulation data have similar characteristics, and therefore are considered jointly. In order to simplify the terminology used here, we refer to such data as *experiment data*, regardless of whether they are experiment or simulation data.

Experiment data can be classified according to two important characteristics: regularity and density. Regularity refers to the pattern of the points or coordinates for which values are measured or computed. For example, detectors may be placed in a specific configuration. If the configuration describes a regular grid or some other geometric structure, the experiment is said to have spatial regularity. Similarly, many simulations assume some regular grid for which values are computed, if values are measured or computed as another regular coordinate of the data.

In general, regularity implies that a mapping between the coordinates of measured values and the storage locations of these values can be made by means of a computation (such as many linearization), which is simply a mapping from multi-dimensional space to linear, similar to array mapping.

Density indicates whether all the potential data points have actual values associated with them. For example, simulation data of water-flow motion computed on a regular grid would have data values (directions, etc.) computed for each point of the grid, and therefore the data are considered dense. On the other hand, in many experiments a large number of measurements that are below a certain threshold are discarded and not recorded. In fact, the level of sparseness can be quite high, i.e., only a small fraction of the potential data points have recorded values.

#### Data in pragmatic terms

In addition to the experimental data discussed above, there exist data in support of the experiments, and data that are generated from the experiment data. Support data fall into two types which we call configuration data and instrumentation data. Similarly, generated data fall into three types: analysed data, summary data, and property data. These types are discussed below. To distinguish these additional data types from the experiment data, we refer to them collectively as *data in pragmatic terms*.

##### *Instrumentation data*

Instrumentation data consist of descriptions of the different instruments and substances used in an experiment, and their changes over time. These data are crucial for the correct analysis of the experiment data. They include information such as the pressure and temperature of a water-storage.

##### *Configuration data*

Configuration data are data that describe the initial structure of an experiment or simulation. For example, in simulating surfacewater-runoff in a given area for a particular season, soil types have to be described. Similarly, the configuration of an experiment describes the position of different devices and detectors. The configuration layout actually determines the regularity (or irregularity) of the experiment data mentioned above. Usually, it does not change in the course of the experiment or simulation. However, it can change between experiments or simulations.

### *Analysed data*

The previous two data types are essential in order to support the analysis of experiment data. The analysis process produces many databases that also need to be managed along with their relationships to the experiment data and to each other. The analysis process may require several steps. It is important to capture the analysis process, the input and output databases of each step, and the relationships between the steps.

### *Summary data*

Similar to statistical databases, which deal with statistical summaries (aggregations) of data sets, scientific databases are often aggregated [Michalewicz91]. For example, in experiments of run-off, the amount of water-loss and types of chemical elements carried thereby can be estimated over several points, summarised over entire area studied, or aggregated over days into a season. In turn, such a summary becomes attributes to some objects, e.g., water-quantity, water-quality, and soil-moisture, and becomes also the data for the relevant variables in an equation to be calculated. Another example is the generation of histograms from many experiments to determine the likelihood of a certain phenomenon. As in the cases of statistical databases, there is a need to organise, search and browse collections of summary data, and to preserve their relationships to lower level data from which they were derived.

### *Property data*

The summary of information learned over a period of time, e.g., a year, is useful to optimal process. There is a substantial amount of work devoted to the organisation and classification of water resources and other substances. For example, there are often several water-systems devoted to water-storage in supply of water. Many property databases cannot now be accessed on-line. The data are only available in periodically published reports, research-papers, or books, and may not be up-to-date. Property data are non-uniform: they contain numeric, text, as well as images and graphs. This is one of the reasons why special-purpose database-systems have been developed for each area. General purpose data management systems that can deal with such diversity of data types are very much required. In addition, because of the complex terminology involved with such data, sophisticated search and browsing capabilities are needed as well.

## **Context of data**

### *Analysed data*

The previous two data types are essential in order to support the analysis of experiment data. The analysis process produces many databases that also need to be managed along with their relationships to the experiment data and to each other. The analysis process may require several steps. It is important to capture the analysis process, the input and output databases of each step, and the relationships between the steps.

### *Summary data*

Similar to statistical databases, which deal with statistical summaries (aggregations) of data sets, scientific databases are often aggregated [Michalewicz91]. For example, in experiments of run-off, the amount of water-loss and types of chemical elements carried thereby can be estimated over several points, summarised over entire area studied, or aggregated over days into a season. In turn, such a summary becomes attributes to some objects, e.g., water-quantity, water-quality, and soil-moisture, and becomes also the data for the relevant variables in an equation to be calculated. Another example is the generation of histograms from many experiments to determine the likelihood of a certain phenomenon. As in the cases of statistical databases, there is a need to organise, search and browse collections of summary data, and to preserve their relationships to lower level data from which they were derived.

### *Property data*

The summary of information learned over a period of time, e.g., a year, is useful to optimal process. There is a substantial amount of work devoted to the organisation and classification of water resources and other substances. For example, there are often several water-systems devoted to water-storage in supply of water. Many property databases cannot now be accessed on-line. The data are only available in periodically published reports, research-papers, or books, and may not be up-to-date. Property data are non-uniform: they contain numeric, text, as well as images and graphs. This is one of the reasons that for each area special-purpose database-systems have been developed. General purpose data management systems that can deal with such diversity of data types is very much required. In addition, because of the complex terminology involved with such data, sophisticated search and browsing capabilities are needed as well.



## APPENDIX 6

### An Example of Parsing a SP Program

%token CHARACTER  
%start object

%%

```
object      : ordered_AND_object  
            | unordered_AND_object  
            | OR_object  
            | simple_object  
            ;  
  
ordered_AND_object : '(' /* empty */ '  
                    | '(' body ')'  
                    ;  
  
unordered_AND_object : '[' /*empty */ '  
                      | '[' body ']'  
                      ;  
  
OR_object : '{' /* empty */ '  
          | '{' body '}'  
          ;  
  
body : object body  
     | object  
     ;  
  
simple_object : symbol  
             | _  
             ;  
  
symbol : symbol symboch  
       | symboch  
       ;  
  
symboch : CHARACTER  
        ;
```





## BIBLIOGRAPHY

### A

- [Abbott91] Abbott, M. B.  
Hydroinformatics, information technology and the aquatic environment, (Avebury Technical) 1991.
- [Abbott93] Abbott, M. B.  
"The electronic encapsulation of knowledge in hydraulics, hydrology and water resources", *Advances in Water Resources* 16, 1993. pp. 21 - 39.
- [Abbott94a] Abbott, M. B.  
"Hydroinformatics: a copernican revolution in hydraulics", *J. of Hydraulic Research*, Extra Issue, Hydroinformatics, Vol. 32, No. 3, 1994. pp. 3 - 13.
- [Abbott94b] Abbott, M. B.  
"The question concerning ethics, or: the metamorphosis of the object". In: Hydroinformatics' 94, Verwey, A., Minns, A. W., Bahovic, V. & Maksimovic, C. (eds.), (A.A.Balkema/Rotterdam/Brookfield) 1994. pp. 3 - 9.
- [ACM85] \_.  
*ACM Sigsoft Software Engineering Notes*, Vol. 10, No. 5, October 1985.
- [Adler92] Adler, P. S. & Winograd, T.  
A. USABILITY: Turning technologies into tools, (Oxford University Press, New York) 1992.
- [Albus91] Albus, J.  
"Online for a theory of intelligence", *IEEE Trans. on Systems, Man, and Cyber.*, 3 March 1991. pp. 473 - 509.
- [Alexander74] Alexander, M. J.  
Information systems analysis, (SRA, Chicago, IL) 1974.
- [AML92] \_.  
AML user's guide, Environmental System Research Institute, Inc. USA, 1992.
- [Araujo92] Araujo, T. & Carapuça, R.  
Issues for a future CASE. In: Proc. of the Third Workshop on the Next Generation of CASE tools, UMIST, Manchester, UK. 1992. pp. 225 - 243.
- [ARC92] \_.  
Understanding GIS - the ARC/INFO method, Environmental System Research Institute, Inc. USA, 1992.
- [Aronoff89] Aronoff, S.  
Geographic information systems: a management perspective, (WDL Publications, Ottawa, Canada) 1989.
- [Ashby56] Ashby, W. R.  
An introduction to cybernetics, (Wiley, New York) 1956, reprinted in (Chapman and Hall) 1973.

- [Ashby81] Ashby, W. R.  
Mechanisms of intelligence, (Seaside, California) 1981.

## B

- [Baecker94] Baecker, D.  
"Intelligence of ignorance in self-referential systems". In: *Cybernetics and Systems*, Trappl, R. (eds.), Vol II. (World Scientific) 1994. pp. 1555 - 1562.
- [Ball94] Ball, I. E.  
"Hydro-informatics - are we repeating past errors?". In: *Hydroinformatics' 94*, Verwey, A., Minns, A. W., Bahovic, V. & Maksimovic, C. (eds.), (A.A.Balkema/ Rotterdam/ Brookfield) 1994. pp. 25 - 30.
- [Barritt-Flatt91] Barritt-Flatt, P. E.  
"Implementing a computer aided support system for water resources research and management". In: *Decision Support Systems*, Loucks, D. P. & Costa, J. R. de (eds.), (Springer-verlag) 1991. pp. 87 - 96.
- [Barrow91] Barrow, J. D.  
Theories of everything, (Vintage, London) 1991.
- [Bartlett32] Bartlett, F. C.  
Remembering: an experimental and social study, (Cambridge Uni. Press, London) 1932.
- [Bassiouni85] Bassiouni, M. A.  
"Data compression in scientific and statistical databases", *IEEE Trans. on Software Eng.* SE-11, 10, October 1985. pp. 1047 - 1058.
- [Bell85] Bell, D. A.  
"A framework for integrating data, knowledge, and information bases", *Informatics 8*, (University of Oxford, England) 1985.
- [Bell90] Bell, D. A. & Zhang, C.  
"Description and treatment of deadlocks in the HECODES distributed expert system", *IEEE Trans. Systems Man Cybernetics*, SE-20, 1990.
- [Bennett82] Bennett, P. D. & Huxham, C. S.  
"Hypergames and what they do", *J. of the Operational Research Society*, No. 33, 1982. pp. 41 - 50.
- [Berry92] Berry, D. M.  
Academic legitimacy of the software engineering discipline, Technical Report, Conegie Mellon University, CMU/SEI-92-TR-34, ESC-TR-92-034, November, USA, 1992.
- [Bertalanffy68] Bertalanffy, von L.  
General systems theory, (George Braziller, New York) 1968.
- [Bieber92] Bieber, M.  
"Automating hypermedia for decision support", *Hypermedia*, No. 2, 1992. pp. 83 - 110.
- [Bieber95] Bieber, M. & Kacmar, C.  
"Designing hypertext support for computational applications", *CACM*, Vol. 38, No. 8, August, 1995.

- [Birtwistle73] Birtwistle, G. M., Dahl, O. J., Myrhaug, B. & Nygaard, K.  
Simula begin, (Van Nostrand Reinhold) 1973.
- [Blanning91] Blanning, R. W.  
"Expert modelbase systems: research directions in recent developments in decision support systems". In: Recent Developments in Decision Support Systems, Holsapple, C. & Whinston, A. B. (eds.), (Springer-Verlag) 1991. pp. 311 - 333.
- [Bobrow75] Bobrow, D. G. & Norman, D. A.  
"Some principles of memory schemata". In: Representation and Understanding, Bobrow, D. G. & Collins, A. (eds.), (Academic Press, New York) 1975.
- [Bonczek81] Bonczek, M. L., Holsapple, C. W. & Whinston, A. B.  
Foundations of decision support systems, (Academic Press, Orlando) 1981.
- [Bonvoisin93] Bonvoisin, N. J. & Moore, R. V.  
"The use of GIS techniques to assess discharge consents and abstraction licences". In: Application of Geographic Information Systems in Hydrology and Water Resources Management, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993.
- [Boose85] Boose, J. H.  
"A knowledge acquisition program for expert systems based on personal construct psychology", *International Journal of Man Machine Studies*, No. 23, 1985.
- [Botvinnik84] Botvinnik, M. M.  
Computers in chess: solving inexact search problems, (Springer, New York) 1984.
- [Brakel91] Brakel, J.  
Meaning, prototypes and the future of cognitive science, minds and machines 1, (Kluwer Academic Publishers) 1991. pp. 233 - 257.
- [Braybrooke70] Braybrooke, D. & Lindblom, C. E.  
A strategy of decision: policy evaluation as social processes, (The Free Press, New York) 1970.
- [Brock90] Wirfs-Brock, R. J. & Johnson, R. E.  
"Surveying current research in object oriented design", *CACM*, Vol. 33, No. 9, 1990. pp. 104 - 124.
- [Brynjolfsson93] Brynjolfsson, E.  
The productivity paradox of information technology, *CACM*, Vol. 36, No. 12, 1993. pp. 67 - 77.
- [Bruner56] Bruner, J. S., Goodnow, J. J. & Austin, G. A.  
A study of thinking, (Wiley, New York) 1956.
- [Bunge77] Bunge, M.  
Ontology. Vol. 1. the furniture of the world, (Reidel, Dordrecht) 1977.
- [Bunke87] Bunke, H. & Sanfeliu, A. (eds.).  
"Syntactic and structural pattern recognition". In: Theory and Applications, series in computer science 7, (World Scientific) 1987.
- [Buras75] Buras, N.  
Scientific allocation of water resources, (American Elsevier) 1975.
- [Burrough86] Burrough, P.A.  
Principles of geographical information systems for land resources assessment (Oxford University Press, New York) 1986.

## C

- [Casti85] Casti, J.  
"System similarities and the existence of natural laws". In: *Differential Geometry, Topology, Geometry and Related Fields*, Rassias, G. (eds.), (Teubner, Leipzig) 1985. pp. 51 - 74.
- [Chapbell38] Campbell, N. R.  
"Symposium: measurement and its importance for philosophy", *Aristotelian Society*, Vol. 17, (Harrison, London) 1938.
- [Chapman87] Chapman, D.  
"Planning for conjunctive goals", *Artificial Intelligence*, No. 3, 1987.
- [Chechile91] Chechile, R. A.  
Environmental decision making: a multi-disciplinary perspective, (Van Nostrand Reinhold) New York, 1991.
- [Checkland81] Checkland, P. B.  
Systems thinking, systems practice, (John Wiley & Sons) 1981, reprinted with corrections in 1984.
- [Checkland90] Checkland, P. B. & Scholes, J.  
Soft systems methodology in action, (John Wiley & Sons) 1990.
- [Chomsky71] Chomsky, N.  
Problems of knowledge and freedom, (Pentheon Books, New York) 1971.
- [Chow64] Chow, V. T. (eds.).  
Handbook of applied hydrology, (McGraw-Hill, New York) 1964.
- [Chu92] Chu, W. W & Chen, Q.  
"A pattern based approach of integrating data and knowledge to support cooperative query answering". In: *Proc. of the First International Conference of Systems Integration*, (IEEE Press) 1992. pp. 615 - 623.
- [Chudoba95] Chudoba, R. & Bittnar, Z.  
"Explicit finite element computation: an object-oriented approach", In: *Computing in Civil and Building Engineering*, Pahl, P. J. & Werner, H. (eds.), (Balkema, Rotterdam) 1995. pp. 139 - 145.
- [Churchman68] Churchman, C. W.  
The systems approach, New York, 1968.
- [Clark93] Clark, M. J.  
"Data constructions on GIS application development for water resource management", In: *Application of Geographic Information Systems in Hydrology and Water Resources Management*, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993. pp. 451 - 463.
- [Cohen86] Cohen, B., Harwood, W. T. & Jackson, M. I.  
The Specification of complex systems, (Addison-Wesley Publishing Co.) 1986.
- [Coplien95] Coplien, J. O. & Schmidt, D. C. (eds.).  
Pattern languages of program design, (Addison-Wesley Publishing Company) 1995.

- [Coppieters90] Coppieters, D.  
An intelligent gaming environment, PhD Dissertation, Delft University of Technology, Delft, The Netherlands, 1990.
- [Corkill83] Corkill, D. D.  
A framework for organisational self-design in distributed problem solving networks, PhD dissertation, University of Massachusetts, Amherst, 1983.
- [Costanza85] Costanza, R. & Sklar, F. H.  
"Articulation, accuracy and effectiveness of mathematical models: a review of freshwater wetland applications", *Ecological Modelling*, No. 27, 1985. pp. 45 - 69.
- [Cox86] Cox, B. J.  
Object oriented programming, (Addison-Wesley Pub. Co. Reading, Massachusetts) 1986.
- [Cox90] Cox, B. J.  
"Planning the software industrial revolution", *IEEE Software*, September 1990.
- [Cuenca89] Cuenca, J.  
"Perspectives in artificial intelligence". In: Expert Systems: application and technical foundations, Cuenca, J (eds.) 1989.
- [Cunge95] Cunge, J. A.  
"Review of informatics applications in water engineering". In: Computing in Civil and Building Engineering, Pahl, P. J. & Werner, H.(eds.), (Balkema, Rotterdam) 1995. pp. 3 - 12.

## D

- [d'Abro50] d'Abro, A.  
The evolution of scientific thought, from Newton to Einstein, (Dover Publications, Inc. New York) 1950.
- [Dangermond83] Dangermond, J.  
"A classification of software components commonly used in geographic information systems". In: Design and Implementation of Computer Based Geographical Information Systems, Peuquet, D. J & O' Callaghan, J. (eds.), (IGU Commission on Geographical Data Sensing and Data Processing , Amherst, New York) 1983.
- [Dangermond89] Dangermond, J.  
GIS data structures: objects vs. layers. The 1989 GIS Year Book, (GIS World Inc.) 1989.
- [Decker87] Decker, K. S.  
"Distributed problem-solving techniques: a survey", *IEEE Trans. Systems Man Cybernetics*, SM-17, 1987. pp. 729 - 740.
- [Dennett81] Dennett, D.  
Brainstorms, (MIT Press) 1981.
- [Deutsch63] Deutsch, K. W.  
The nerves of government - models of political communication and control, (Free Press of Glencoe, New York) 1963.
- [Dewire94] Dewire, D. T.  
Application development for distributed environments, (McGraw-Hill, Inc) 1994.

- [Diederich87] Diederich, J. & May, M.  
"KRITON: a knowledge acquisition tool for expert systems", *International Journal of Man Machine Studies*, Vol. 26, 1987. pp. 29 - 40.
- [Dolk93] Dolk, D. R. & Kottemann, J. E.  
"Model integration and a theory of models", *Decision Support Systems*, No. 9, 1993. pp. 51 - 63.
- [Domenico72] Domenico, P. A.  
Concepts and models in groundwater hydrology, (McGraw-Hill, Inc.) 1972.
- [Droesen93] Droesen, W. & Geelen, L. H. W. T.  
"Application of fuzzy sets in ecohydrology expert modelling". In: Application of Geographic Information Systems in Hydrology and Water Resources Management, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993. pp. 3 - 13.
- [Durfee87] Durfee, E. H., Lesser, V. R & Corkill, D. D.  
"Cooperation through communication in a distributed problem solving network". In: Distributed Artificial Intelligence, Huhns, M. N. (eds.), (Morgan Kaufman, Los Altos, CA) 1987. pp. 29 - 58.
- [Durfee91] Durfee, E. H.  
"Special section on distributed artificial intelligence", *IEEE Trans. on Sys., Man, and Cybernetics*, Vol. 21, No. 6, 1991.
- [Dziegielewski87] Dziegielewski, B.  
"The IWR-MAIN disaggregate water use model". In: Proc. of the 1987 UCOWR Annual Meeting, (UCOR Executive Director' s Office, Carbondale, I11) August 1987.

## E

- [Easton73] Easton, A.  
Complex managerial decisions involving multiple objectives, (Wiley New York) 1973.
- [Eddington58] Eddington, A.  
The nature of the physical world, (University of Michigan Press, Ann Arbor) 1958.
- [El-Najdawi93] El-Najdawi, M. K. & Stylianou, A. C.  
"Expert technologies", *Comm. ACM*. Dec. 1993.
- [Erlenkotter76] Erlenkotter, D.  
"Co-ordinating scale and sequencing decisions for water resources projects". In: Economic Modelling for Water Policy Evaluation, Thrall, R. M., Heady, E., Schad, T., Schwartz, A. K. & Thompson, R. G. (eds.), (North-Holland Publishing Company) 1976. pp. 97 - 100.

## F

- [Fagin79] Fagin, R.  
"Extendible hashing - a fast access method for dynamic files", *ACM Trans. Database systems*, Vol. 4, No. 3, September 1979. pp. 315 - 344.

- [Fang93] Fang, L., Hipel, K. W. & Kilgow, D. M.  
Interactive decision making; the graph model for conflict resolution, (Wiley, New York) 1993.
- [Fedra90] Fedra, K. & Reitsma, R.  
"Decision support and geographical information systems", In: Geographical Information Systems for Urban and Regional Planning, Scholten, H. J. & Stilwell, J. C. H. (eds.), (Kluwer Academic Publishers) 1990. pp. 177 - 188.
- [Feigenbaum77] Feigenbaum, E. A.  
"The art of artificial intelligence: themes and case studies of knowledge engineering".  
In: Proc. of International Conference on Art. Intelligence, 1977. pp. 1014 - 1029.
- [Feuchtwanger92] Feuchtwanger, M.  
Geographical Semantic Database Modelling, PhD Dissertation, University of Massachusetts, Amherst, 1992.
- [Fikes71] Fikes, R. E. & Nilsson, N. J.  
"STRIPS - A new approach to the application of theorem proving in problem solving",  
*Artificial Intelligence*, No. 2, 1971. pp. 189 - 208.
- [Fodor87] Fodor, J.  
Psychosemantics, (The MIT Press) 1987. pp. 114 - 122.
- [Ford91a] Ford, K. M., Petry, F. E., Adams-Webber, J. R. & Change, P. J.  
"An approach to knowledge acquisition based on the personal construct systems",  
*IEEE Tran. on Knowledge and Data Eng.*, Vol. 3, No. 1, March 1991. pp. 78 - 88.
- [Ford91b] Ford, K. M., Canas, A., Jones, J., Stahl, H., Novak, J. & Adams-Webber, J. R.  
"ICONKAT: an integrated constructivist knowledge acquisition tool", *International Journal of Knowledge Acquisition*, Vol. 3, 1991. pp. 215 - 236.
- [Fox90] Fox, J., Clark, D. A., Glowinski, A. J. & O'Neil, M.  
"Using predicate logic to integrate qualitative reasoning and classical decision theory",  
*IEEE Trans on Systems Man and Cybernetics*, No. 20, 1990. 347 - 357.
- [French93] French, S.  
Decision theory, (Horwood, New York) 1993.
- [Frysinger93] Frysinger, S. P., Thomas, R. P. & Parsons, A. M.  
"Hydrological modelling and GIS: the Sandia Environmental Decision Support Systems", In: Application of Geographic Information Systems in Hydrology and Water Resources Management, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993.
- [Funk74] \_,  
Funk and Wagnells Standard Dictionary, (Fizhenry and Whiteside, Toronto, Canada) 1976.
- [Furst93] Furst, J., Girstmair, G. & Nachtnebel, H. P.  
"Application of GIS in decision support systems for groundwater management". In: Application of Geographic Information Systems in Hydrology and Water Resources Management, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993, pp. 13 - 21.

## G

- [Gaines87] Gaines, B. R.  
"An overview of knowledge acquisition and transfer", *International Journal of Man Machine Studies*, Vol. 26, 1987. pp. 453 - 472.
- [Gamma94] Gamma, E. et al.  
Design patterns: elements of reusable object-oriented software, (Addison-Wesley) 1994.
- [Gane90] Gane, C.  
Computer-aided software engineering: the methodologies, the products and the future, (Prentice-Hall, NJ) 1990.
- [Garrote95] Garrote, L. & Bras, R. L.  
"An integrated software environment for real-time use of a distributed hydrological model", *J. of Hydrology*, Vol. 167, 1995. pp. 307 - 326.
- [Gaul88] Gaul, W., & Schader, M. (eds.).  
Data, expert knowledge and decisions, (Springer - Verlag) 1988.
- [Gentner83] Gentner, G. D.  
"Structure-mapping: a theoretical framework for analogy", *Cognitive Science*, No. 7, 1983. pp. 155 - 170.
- [Gibson66] Gibson, J. J.  
The senses of considered as perceptual systems, (Houghton-Mifflin) 1966.
- [GMN92] \_  
Een nieuw evenwicht, Rapport van de stuurgroep GMN, Lelystad, 6 Nov. 1992 (in Dutch).
- [Goldberg83] Goldberg, A.  
Smalltalk-80: the language and its implementation, (Addison-Wesley) 1983.
- [Gorry71] Gorry, G. A. & Scott-Morton, M. S.  
"A framework for management information systems", *Sloan Management Review*, 1971. pp. 55 - 70.
- [Greniewski60] Greniewski, H.  
Cybernetics without mathematics, (Pergamon Press Ltd. Headington Hill, Oxford) 1960.

## H

- [Haagsma94a] Haagsma, IJ. G., & Johannis, R. D.  
"Decision support systems, and integrated and distributed approach". In: Proc. of Envirosoft, San Francisco, November 1994. pp. 205 - 212.
- [Haagsma94b] Haagsma, IJ. G., & Johannis, R. D.  
"The integration of computer models and data bases into a decision support system for water management". In: Proc. of IAHS Symposia and Workshops - Modelling and Management of Sustainable Basin-scale Water Resources Systems, Boulder, USA, 3-14 July, 1995.



- [Haagsma94c] Haagsma, IJ. G. & Johannis, R. D.  
 "The interaction of ground water and surface water studied by loosely coupled models". In: Proc. of Water Down Under 94, Adelaide, Australia, 21-25 November 1994.
- [Hall62] Hall, A. D.  
 A methodology for systems engineering, (Van Nostrand, Princeton) 1962.
- [Hargis92] Hargis, J. E.  
 "Object modelling techniques ease relational databases design", *GIS World* Vol. 5, No. 6, 1992. pp. 80 - 83.
- [Hart89] Hart, A.  
 Knowledge acquisition for expert systems, (Kogan Page, London) 1989.
- [Hayes-Roth83] Hayes-Roth, F., Waterman, D. A. & Lenat, D. B.  
 Building expert systems, (Addison-Wesley, Reading, Massachusetts) 1983.
- [Hipel83] Hipel, K. W. & Fraser, N. M.  
 "Conflict analysis in systems management", *Encyclopaedia of Systems and Control*, Singh, M. G. (eds.), (Pergmon Press, Oxford) 1983.
- [Holyoak89] Holyoak89, K. J. & Thagard, P.  
 "Analogy mapping by constraint satisfaction", *Cognitive Science*, No. 13, 1989. pp. 259 - 355.
- [Hofstede92] Hofstede, ter. A. H. M. & Weide, v. d. Th. P.  
 Formalisation of techniques: chopping down the methodology jungle, *Information and Software Technology*, Vol. 34, No. 1, January 1992. pp. 57 - 65.
- [Horswill95] Horswill, I.  
 "Analysis of adaptation and environment", *Artificial Intelligence*, Vol. 73, No. 1-2, 1995. pp. 1 - 30.
- [Hupfer95] Hupfer, P.  
 "Modelling of fuzziness". In: Computing in Civil and Building Engineering, Pahl, P. J. & Werner, H. (eds.), (Balkema, Rotterdam) 1995. pp. 241 - 244.
- [Hughes91] Hughes, J. G.  
 Object-oriented databases, (Prentice Hall, U. K. ) 1991.
- [Hull87] Hull, R. & King, R.  
 "Semantic database modelling: survey, applications, and research issues", *ACM Computing Surveys*, Vol. 19, No. 3, September 1987. pp. 201 - 260.
- [Hydrosoft94] \_\_,  
 Water resources and distribution: Vol. I, II, Blain, W. R. & Katsifarakis, K. L. (eds), (Computational Mechanics Publications) 1994.

## I - J

- [IAHS180] \_\_,  
 Systems analysis for water resources management: closing the gap between theory and practice, Loucks, D. P. (eds.), (IAHS Publication) No. 180, 1989.

[IAHS211] \_.

Application of geographic information systems in hydrology and water resources management, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993. pp. 3 - 61.

[IAHR94] \_.

Extra issue, hydroinformatics, *J. of Hydraulic Research*, Vol. 32, No. 3, 1994.

[Jackson75] Jackson, M. A.

Principles of program design, (Academic Press) 1975.

[Jakeman93a] Jakeman, A. J., Beck, M. B. & McAleer, M. J. (eds.).

Modelling changes in environmental systems, (John Wiley, Chichester) 1993.

[Jakeman93b] Jakeman, A. J. & Hornberger, G. M.

"How much complexity is warranted in a rainfall-runoff model?", *Water Resources Research*, Vol. 29, No. 8, 1993. pp. 2637 - 2649.

[Jenkins69] Jenkins, G. M.

"The system approach", *J. of System Engineering*, 1969.

[Johanns94] Johannis, R. D. & Haagsma, J. G.

"Integration of integrated water management". In: Proc. of the International UNESCO Symposium on Water Resources Planning in a Changing World, Karlsruhe, Germany, 28-30 June 1994. pp. III - 41 - 49.

[Johanns95] Johannis, R. D. & Haagsma, J. G.

"Integrated water resources management in the information age". In: Proc. UNESCO International Symposium - Integrated Water Management in Urban Areas, Lund, Sweden, 26-29 September 1995.

[Julie82] Julie, P.

"Understanding and understanding the listener", *American Psychological Association Meetings*, Washington, D. C., U. S. A. 1982.

[Julie83] Julie, P.

Explaining models in linguistics: a behavioural perspective, (Princeton University Press) 1983.

## K

[Kadar89] Kadar, Y. & Damelin, E.

"Simulation in operation of water supply systems". In: Application of Geographic Information Systems in Hydrology and Water Resources Management, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993. pp. 95 - 103.

[Keeney76] Keeney, R. L. & Raiffa, H.

Decisions with multiple objectives: preferences and value tradeoffs, (John Wiley & Sons, New York) 1976.

[Kelly55] Kelly, G. A.

The psychology of personal construct, (van Nostrand-Reinhold, New York) 1955.

[Klir69] Klir, G. J.

An approach to general systems theory, (van Nostrand-Reinhold, New York) 1969.

[Klir91] Klir, G. J.

Facets of Systems Science, (Plenum Press) 1991.

- [Klir92] Klir, G. J.  
 "Systems science: a guided tour", *J. Biological Systems*, No. 1, 1, 1992. pp. 27 - 59.
- [Koppelaar74] Koppelaar, H. & Kruijt, D.  
 "Regular careersystems", *Annals of Systems Research*, No. 4, 1974. pp. 131 - 139.
- [Koppelaar94] Koppelaar, H.  
 Massive databases in a spatial paradigm, Literature Survey, Knowledge-based System Foundation, Department of Informatica, TU Delft, the Netherlands, 1994.
- [Krishman91] Krishman, R., Piela, P. & Westerberg, A.  
 "Reusing mathematical models in ASCEND". In: Recent Developments in Decision Support Systems, Holsapple, C. & Whinston, A. B (eds.), (Springer-Verlag, 1991). pp. 275 - 295.
- [Kronlof93] Kronlof, K. (eds.).  
 Method Integration, (John Wiley & Sons) 1993.

## L

- [Lam91] Lam, D. C. L. & Swayne, D. A.  
 "Integration databases, spreadsheet, graphs, GIS, statistics, simulation models and expert systems: experiences with the RAISON system on microcomputers". In: Decision Support Systems, Loucks, P. & Costa, R. da (eds.), (Springer-Verlag) 1991. pp. 429 - 459.
- [Langran92] Langran, G.  
 Time in geographic information systems, (Taylor & Francis, London) 1992.
- [Larson88] Larson, P.  
 "Dynamic hashing tables", *J. ACM*, April 1988. pp. 446 - 457.
- [Lehman91] Lehman, M. M.  
 "Software engineering, the software progress and their support", *IEEE Software Engineering Journal*, September 1991.
- [Lee91] Lee, S. & Carver, D. L.  
 "Object-oriented analysis and specification: a knowledge base approach", *J. of Object-Oriented Programming*, Vol. 3, No. 5, January 1991.
- [Liebscher93] Liebscher, H. J.  
 "Hydrology for the water management of large river basins", *Hydrological Sciences - Journal des Sciences Hydrologiques*, Vol. 38, No. 1, 2. 1993.
- [Linden89] Linden, van der, J. Ph., Koudstaal, R. & Nyongsta, L. M.  
 "Water resources information systems for regional planning". In: Systems Analysis for Water Resources Management: Closing the Gap between Theory and Practice, Loucks, D. P. (eds.), (IAHS Publication) No. 180, 1989. pp. 85 - 95.
- [Lindley85] Lindley, D. V.  
 Making decision, (Wiley, London) 1985, reprinted in 1992.
- [Liu93a] Liu, Y.  
 "To approach an intelligent system in coping with an aspect of complexity of water management". In: Proceedings of the 1993 ACM Symposium on Applied Computing, Indianapolis, 1993.

- [Liu93b] Liu, Y. & Veer, v.d. P.  
 "An incentive to build an intelligent system to cope with complexity derived from systems' interaction". In: *Proceedings of IEEE International Conference on Developing and Managing Intelligent System Projects*, Washington, DC, (IEEE Press) 1993.
- [Liu93c] Liu, Y. & Veer, v.d. P.  
 "DISSWAM: a framework for delft intelligent system supporting system in water management". In: *Advanced Technologies*, M. R. Beheshti (eds.), (Elsevier, Amsterdam) 1993.
- [Liu94a] Liu, Y.  
 "Treating the interactions systematically", *IEE Software Engineering J*, March 1994.
- [Liu94b] Liu, Y.  
 "How much complexity is warranted to develop DSS for engineering use in large?". In: *Proceedings of 3rd International Conference of System Integration*, Ng. P. A, et (eds.), (IEEE press) 1994.
- [Liu94c] Liu, Y.  
 "Augmenting some attributes to the concepts of cyberspace". In: *Cybernetics and Systems Research*, Trappl, R. (eds.), (World Scientific) 1994.
- [Liu94d] Liu, Y.  
 "A model of DSE: decision support environment for engineering use". In: *Water Resources and Distribution*, Blain, W. R. & Katsifarakis, K. L. (eds.), (Computational Mechanics Publications) 1994.
- [Lockemann91] Lockemann, P. C., Nagel, H-H. & Walter, I. M.  
 "Databases for knowledge bases: empirical study of a knowledge base management for a semantic network", *Data & Knowledge Engineering*, No. 7, 1991. pp. 115 - 154.
- [Loucks85] Loucks, D. P., Kindler, J. & Fedra, K.  
 "Interactive water resources modelling and model use: an overview", *Water Resources Research*, Vol. 21, No. 2. 1985. pp. 95 - 112.
- [Loucks89] Loucks, D. P (eds.).  
 "Closing the gap between theory and practice", *Proceedings of International Conference - Systems Analysis for Water Resources Management*, (IAHS Publication) No. 180, 1989.
- [Loucks91] Loucks, D. P. & Costa, da J. R. (eds.).  
*Decision Support Systems*, (Springer-Verlag) 1991.
- [Luhmann90] Luhmann, N.  
*Essays on self-reference*, (Prentice Hall, New York) 1990.

## M

- [Mackie95] Mackie, R. I.  
 "Object-oriented methods - finite element programming and engineering software design". In: *Computing in Civil and Building Engineering*, Pahl, P. J. & Werner, H. (eds.), (Balkema, Rotterdam) 1995. pp. 133 - 139.

- [Marik92] Marik, V. & Vlcek, T.  
Knowledge engineering - an overview. In: Advanced Topics in AI, Trappl, R. (eds.), Lecture Notes in AI 617, 1992.
- [Maturana80] Maturana, H. & Veerela, F.  
Autopoiesis and Cognition, (Dordrecht) 1980.
- [Mays89] Mays, L. W.  
"Hydrosystems engineering simulation vs. optimisation: why not both?". In: Systems Analysis for Water Resources Management: Closing the Gap between Theory and Practice, Loucks, D. P (eds.), (IAHS Publication) No. 180, 1989. pp. 225 - 233.
- [Mei89] Mei, X., Rosso, R., Huang, G. L. & Nie, G. S.  
"Application of analytical hierarchy process to water resources policy and management in Beijing, China". In: Systems Analysis for Water Resources Management: Closing the Gap between Theory and Practice, Loucks, D. P. (eds.), (IAHS Publication) No. 180, 1989. pp. 73 - 85.
- [Meijerink93] Meijerink, A. M. J., Mannaerts, C. M., Brouwer, H. A. & Valenzuela, C. R.  
"Application of ILWIS to decision support in watershed management: case study of the Koering river basin, Indonesia". In: Application of Geographic Information Systems in Hydrology and Water Resources Management, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993. pp. 35 - 45.
- [Mesarovic91] Mesarovic, M. D. & Takahara, Y.  
Abstract systems theory, (Springer-Verlag) 1991.
- [Mesarovich70] Mesarovich, M. D., Macko, D. & Takahara, Y.  
Theory of hierarchical multilevel systems, (Academic Press) New York, 1970.
- [Meyer88] Meyer, B.  
Object-oriented software construction, (Prentice Hall, New York) 1988.
- [Minsky75] Minsky, M.  
"A framework for representing knowledge". In: The Psychology of Computer Vision, Winston, P. H. (eds.), (John Wiley & Sons, New York) 1975.
- [Moll89] Moll, J. R. & Crebas, J. I.  
"An operational management system for river flows". In: Systems Analysis for Water Resources Management: Closing the Gap between Theory and Practice, Loucks, D. P. (eds.), (IAHS Publication) No. 180, 1989. pp. 211 - 225.
- [Morris55] Morris, C.  
Signs, Language, and behaviour, (Braziller, New York) 1955.
- [Muehrcke78] Muehrcke, P.  
"Map use, Reading, analysis, and interpretation", (JP Publications, U.S.A) 1978.
- [Muller91] Muller, N., & Hahn, H. H.  
"The German water authorities needs for computer-aided support systems". In: Decision Support Systems, Loucks, D. P., & Costa, J. R. de (eds.), (Springer-verlag) 1991. pp. 259 - 305.

## N

- [Newell72] Newell, A. & Simon, H. A.  
Human problem solving, (Engelwood Cliffs, New Jersey, U.S.A) 1972.

- [Newell90] Newell, C. J., Haasbeek, J. F. & Bedient, P. B.  
 "OASIS: A graphical decision support system for ground-water contaminant modelling", *Ground Water*, Vol. 28, No. 2, April 1990.
- [Nijssen89] Nijssen, G. M. & Halpin, T. A.  
 Conceptual schema and relational database design: a fact oriented approach, (Prentice-Hall, Sydney, Australia) 1989.
- [Nilsson80] Nilsson, N. J.  
 Principles of artificial intelligence, (Tioga publishers, Palo Alto. CA) 1980.

## O - P

- [Okuyama94] Okuyama, M.  
 "Reality and communication in psychotherapy". In: *Cybernetics and Systems*, Trappl, R. (eds.), Vol. II. (World Scientific) 1994. pp. 1647 - 1654.
- [Orlandic89] Orlandic, R. & Pfaltz, J. L.  
 "Analysis of compact o-trees: a new access method to large databases". In: *Proc. 7th FCT Conference*, (Szeged, Hungary) August 1989.
- [Page90] Page, I. D.  
 "Information products in the 90s". In: *Proc. Hewlett-Packard European Scientific Symposium*, Vienna, 1990. pp. 95 - 110.
- [Palmer89] Palmer, R. N.  
 "A natural language interface for a drought management expert system". In: *Systems Analysis for Water Resources Management: Closing the Gap between Theory and Practice*, Loucks, D. P. (eds.), (IAHS Publication) No. 180, 1989. pp. 233 - 241.
- [Parsons51] Parsons, T.  
 The social system, (Routledge and Kegan Paul, London) 1951.
- [Pascoe86] Pascoe, G. A.  
 "Elements of object oriented programming", *BYTE*, August, 1986. pp. 139 - 144.
- [Peckham88] Peckham, J. & Maryanski, F.  
 "Semantic data models", *ACM Computing Survey*, Vol. 20, No. 3, 1988.
- [Peutrec94] Peutrec, S. L.  
 "Autonomy and knowledge organisation". In: *Cybernetics and Systems*, Trappl, R. (eds.), Vol. II, World Scientific, 1994. pp. 1547 - 1554.
- [Piazza90] Piazza, P. A. & Pessaro, F.  
 A cognitive model for a "smart" GIS. *The 1990 GIS Yearbook*, (GIS World Inc.) 1990. pp. 273 - 279.
- [Prieto91] Prieto-Diaz, R. & Arango, G.  
 Domain analysis and software systems modelling, (IEEE Computer Society Press) 1991.

## Q - R

- [Quine60] Quine, W. v. O.  
 Word and object, (MIT Press) 1960.

- [Radford80] Radford, K. J.  
Modern managerial decision making, (Reston Publishing Company) 1980.
- [Radley90] Radley, P.  
"The next decade in telecommunications". In: Proc. Hewlett Packard European Scientific Symposium, Vienna, 1990, pp. 13 -33.
- [Rapoport66] Rapoport, A.  
"Mathematical aspects of general systems theory", *General Systems*, Vol. 8, 1966. pp. 123 - 128.
- [Richardson93] Richardson, D. E.  
Automated spatial and thematic generalisation using a context transformation model, PhD dissertation, ITC (International Institute for Aerospace Survey and Earth Sciences), the Netherlands, 1993.
- [Rigby64] Rigby, F. D.  
"Heuristic analysis of decision situations". In: Human Judgements and Optimality, Shelly, M. W. & Bryan, G. L. (eds.), (John Wiley & Sons, New York) 1964.
- [Ringle79] Ringle, M.  
"Philosophy and artificial intelligence". In: Philosophical Perspectives in Artificial Intelligence, Ringle, M. (eds.), (Humanities Press INC, New Jersey, USA). 1979.
- [Roberts76] Roberts, M.  
"Institutions and objectives on the water, energy, environment interface". In: Economic Modelling for Water Policy Evaluation, Thrall, R. M., Heady, E., Schad, T., Schwartz, A. K. & Thompson, R. G. (eds.), (North-Holland Publishing Co.) 1976. pp. 57 - 71.
- [Ruland93] Ruland, P., Arnold, U. & Rouve, G.  
"An integrated information system for environmental hydraulics using Smallworld GIS". In: Application of Geographic Information Systems in Hydrology and Water Resources Management, Kovar, K. & Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993. pp. 51 - 60.
- [Rumelhart86] Rumelhart, D. E. & McClelland, J. L. (eds.).  
Parallel distributed processing, Vol. I & II, (Cambridge Mass. MIT Press) 1986.
- [Russell67] Russell, B.  
What I believe, reprinted in: The Basic Writings of Bertrand Russell, (Simon and Schuster, New York) 1967.

## S

- [Salvaneschi95] Salvaneschi, P. & Gambirasi, P.  
"Information systems for civil engineering: integration through object oriented technology". In: Computing in Civil and Building Engineering, Pahl, P. J. & Werner, H. (eds.), (Balkema, Rotterdam) 1995. pp. 365 - 369.
- [Schader90] Schader, M. & Gaul, W. (eds.).  
Knowledge, data and computer-assisted decisions, NATO ASI Series, (Springer-Verlag) 1990.

- [Schank77] Schank, R. C. & Abelson, R. P.  
Scripts, plans, goals, and understanding: an inquiry into human knowledge structures, (John Wiley & Sons, New York) 1977.
- [Schlickmann92] Schlickmann, T. W.  
Improving the effectiveness of decision-making using a distributed decision support system, PhD dissertation, Delft University of Technology, Delft, the Netherlands, 1992.
- [Schultz89] Schultz, G. A.  
"Ivory tower versus ghosts? --or-- the interdependency between system analysis and real-world decision makers in water management". In: Proceedings of International Conference - Systems Analysis for Water Resources Management, Loucks, D. P (eds.), (IAHS Publication) No. 180, 1989. pp. 23 - 24.
- [Searle80] Searle, J.  
"Minds, brains and programs", *Behavioural and Brain Sciences*, No. 3, 1980. pp. 417 - 424.
- [SEKE94] \_  
Special issue: life cycle of knowledge-based systems, *Software Engineering & Knowledge Engineering*, Vol. 3, No. 1, 1994.
- [Shahin93] Shahin, M., Oorschot, v. H. J. L. & Lange, de S. J.  
Statistical analysis in water resources engineering, (A.A.Balkema/ Rotterdam/ Brookfield) 1993.
- [Shannon49] Shannon, C. E. & Weaver, W.  
The mathematical theory of communication , (Urbana University of Illinois Press) 1949.
- [Shannon75] Shannon, R. E.  
Systems simulation: the art and science, (Prentice Hall, Englewood Cliffs) 1975
- [Simon69] Simon, H. A.  
"The operations in character recognition", *Perception and Psychophysics*, No. 2, 1969. pp. 43 - 53.
- [Simon80] Simon, H. A.  
The sciences of the artificial, Second edition, (MIT Press, Cambridge, MA) 1980.
- [Simonovic93] Simonovic, S. P.  
"Flood control management by integrating GIS with expert systems: Winnipeg City case study". In: Application of Geographic Information Systems in Hydrology and Water Resources Management, Kovar, K. and Nachtnebel, H. P. (eds.), (IAHS Publication) No. 211, 1993. pp. 61 - 72.
- [Singh88] Singh, V. P.  
Hydrologic systems, Vol. 1: rainfall-runoff modelling, (Prentice-Hall) 1988.
- [Smith79] Smith, R. G.  
"A framework for distributed problem solving". In: Proc. of 6th International Joint Conf. on Artificial Intelligence, 1979. pp. 836 - 841.
- [Sol85] Sol, H. G.  
"DSS: buzzword or OR challenge", *European J. of Operational Research*, 1985. No. 22, pp. 1 - 8.



- [Sol91] Sol, H. & Bots, P. W. G.  
 "Information systems to support decision processes: from decision support to networks of inquiry systems". In: *Recent Developments in Decision Support Systems*, Holsapple, C. & Whinston, A. B. (eds.), (Springer-Verlag) 1991. pp. 23 - 31.
- [Soukup94] Soukup, J.  
 Taming C++, pattern classes and persistence for large projects, (Addison-Wesley) 1994.
- [Sowa92] Sowa, J. F.  
 "Conceptual graphs as a universal knowledge representation", *Computers & Math. with Applications*, Vol. 23, No. 2-5, 1992. pp. 75 - 93.
- [Sprague87] Sprague, R. H.  
 "DSS in context", *Decision Support Systems*, No. 3, 1987. pp. 197 - 202.
- [Stansbury91] Stansbury, J., Woldt, W. & Bogardi, I.  
 "Decision support system for water transfer evaluation", *Water Resources Research*, Vol. 27, No. 4., April 1991. pp. 443 - 451.
- [Star90] Star, J. & Estes, J. E.  
 Geographic information systems: an introduction, (Prentice-Hall) 1990.
- [Starr77] Starr, M. K. & Zeleny, M.  
 "MCDM - state and future of the arts in multiple criteria decision making". In: *Studies in Management Sciences*, Starr, M. K. & Zeleny, M. (eds.), (North-Holland, Amsterdam) 1977. pp. 5 - 30.
- [Stevens51] Stevens, S. S.  
 "Mathematics, measurement, and psychophysics". In: *Handbook of Experimental Psychology*, Stevens, S. S. (eds.), (Wiley, New York) 1951.
- [Stroustrup88] Stroustrup, B.  
 What is object-oriented programming, *IEEE Software*, May 1988. pp. 10 - 20.
- [Sudkamp88] Sudkamp, T. A.  
 Language and machines, an introduction to the theory of computer science, (Addison-Wesley) 1988.

## T - V

- [Thome93] Thome, B (eds.).  
 System engineering, principles and practice of computer-based system engineering, (John Wiley & Sons) 1993.
- [Torgerson63] Torgerson, S. S.  
 Theory and methods of scaling, (Wiley, New York) 1963.
- [Tully91] Tully, C. J.  
 A view of systems engineering, IEE First Vacation School on System Engineering at the University of Reading, UK, April 7-12 1991.
- [Unger87] Unger, D. & Smith, R. B.  
 "Self: the power of simplicity". In: *Proc. of the Conference on Object Oriented Programming Systems and Languages (OOPSLA' 87)*, 1987. pp. 227 - 241.

- [Vadera91] Vadera, S. & Nechab, S.  
 "Are expert system shells and toolkits too general?". In: Proc. of Decision Support Systems and Qualitative Reasoning, Singh, M. G. & Massuyes, T. (eds.), 1991.
- [Vamos91] Vamos, T.  
 Computer epistemology, (World Scientific) 1991.
- [Veer93] Veer, van der P.  
 "Decision support for integrated water resources management". In: Multicriteria Decision Analysis in Water Resources Management, Bogardi, J. J & Nachtnebel, H-P (eds.), UNESCO, Paris, 1994. pp. 315 - 321.
- [Veer94] Veer, van der P.  
 Exact solutions for two-dimensional groundwater flow in a semiconfined aquifer, J. of Hydrology, No. 156, 1994. pp. 91 - 99.
- [Veldhorst82] Veldhorst, M.  
 An analysis of sparse matrix storage schemes, Report, Mathematical Centre Amsterdam, the Netherlands, 1982.
- [Velthuisen92] Velthuisen, H.  
 The nature and applicability of the blackboard architecture, PhD dissertation, Limburg University (Rijksuniversiteit Limburg), Maastricht, The Netherlands, 1992.
- [Vessey95] Vessey, I. & Sravanapudi, A. P.  
 CASE tools as collaborative support technologies, *CACM*, Vol. 38, No. 1, January 1995. pp. 83 - 95.
- [Visualingam89] Visualingam, M.  
 Cartography, maps and GIS in perspective, *Cartography J.* Vol. 26, No. 1, 1989. pp. 26 - 32.

## W

- [Walsh89] Walsh, P. D. & Walker, S.  
 "Decision support systems as an aid in the operational management of multiple water source systems, In: Systems Analysis for Water Resources Management: Closing the Gap between Theory and Practice", Loucks, D. P. (eds.), (IAHS Publication) No. 180, 1989.
- [Walz93] Walz, D. B., Elam, J. J. & Curtis, B.  
 "Inside a software design team: knowledge acquisition, sharing, and integration", *CACM*, No.4, April 1993.
- [Wang93] Wang, Y. & Parnas, D. L.  
 "Simulating the behaviour of software modules by trace rewriting". In: Proc. of the 15th International Conference on Software Engineering, Baltimore, MD, USA. May 1993.
- [Warren85] Warren, V. Jr. & Welty, C.  
 Water management, technology and institutions, (Harper & Row, Publications, Inc.) 1985.
- [Warwick94] Warwick, J. J. & Haness, S. J.  
 "Efficacy of ARC/INFO application to hydrologic modelling, *J. of Water Resources Planning and Management*, Vol. 120, No. 3, 1994. pp. 366 - 381.

- [Whytock93] Whytock, S.  
 "The development life-cycle". In: System Engineering, principles and practice of computer-based system engineering, Thome, B. (eds.), (John Wiley & Sons) 1993. pp. 81 - 89.
- [Winograd86] Winograd, T. & Flores, F.  
 Understanding computers and cognition, (Norwood, NJ) 1986.
- [Witmer89] Witmer, M. C. H.  
 Integral water management at regional level, Rijkswaterstaat Communication, Dissertation, University of Utrecht, the Netherlands, 1989.
- [Wolff91] Wolff, J. G.  
 Towards a theory of cognition and computing, (Ellis Horwood) 1991.

## Y - Z

- [Yang85] Yang, J. D. & Stephens, L. M.  
 "An architecture for control and communications in distributed artificial intelligence systems", *IEEE Trans. Systems Man Cybernetics*, SM-15, 1985. pp. 316 - 326.
- [Zadeh73] Zadeh, L. A.  
 "Outline of a new approach to the analysis of complex systems and decision processes", *IEEE Trans. on Systems, Man and Cyber.* Vol. SM-3, No. 1, January 1973. pp. 28 - 44.
- [Zeleny92] Zeleny, M. & Hufford, K. D.  
 "The application of autopoiesis in systems analysis: are autopoietic systems also social systems?", *International Journal of General Systems*, No. 21, 1992. pp. 145 - 163.
- [Zemanek66] Zemanek, H.  
 "Semiotics and programming languages", *CACM*, No. 9, 1966. pp. 139 - 143.
- [Zualkernan86] Zualkernan, I.  
 "Expert systems and software engineering: ready for marriage?" *IEEE Expert*, Vol. 1, No. 4, 1986.



## EXPLANATION OF CONVENTIONS

This section provides examples of the main layout and style generally adopted during the preparation of the dissertation.

### Heading

A heading has two line-spacing before and one line-spacing after itself.

### Sub-heading

A sub-heading has one line-spacing before and one line-spacing after it.

#### *Sub-sub-heading.*

Normally, a sub-sub-heading is printed in italic unless the section it leads contains sub-section. In the later case, sub-sub-headings are printed in bold, and subsequently, a sub-sub-sub heading is printed in italic.

#### **The Example, Case, Frame, and Definition**

When examples, cases, frames and definitions lead to separated sections, they are headed by their names printed by 10 pt in Helvetica. Their sections are printed by 10 pt in Times.

Titles of figures are printed in 10 pt Helvetica, and they have two line spacing before and one line spacing after them, for example:

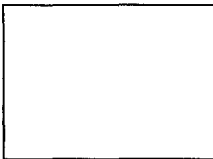


Fig. 1.1: an example

### Single, double quote and other use of italic

A single quote is used for an informal quotation mark; it serves as an indicator to the reader suggesting that the term being quoted is given by a particular or technical meaning which one may read as (the) so-called ..., or may consider to be intuitive, imprecise or inexact.

Double quote is used for quoting the words being copied from a given reference. It is also used in the section - Bibliography to be mentioned later.

Except for a sub-sub-heading, italic printing is used for emphasising a term (or terms), or introducing the term (or terms) which will be explained afterwards.

## References

In the chapters, references are printed by the format - [First-author' s-SurnameYear], e.g., [Wang93] or [Liu93a].

## Bibliography

In the bibliography, a reference is printed by the following format.

If it indicates a book, then

### Example:

[Abbott91] Abbott, M. B.

Hydroinformatics, information technology and the aquatic environment,  
(Avebury Technical) 1991.

where, the bracket closes the name and place of a publisher.

If it indicates an article in a book, then

### Example:

[Baecker94] Baecker, D.

"Intelligence of ignorance in self-referential systems". In: Cybernetics and  
Systems, Trappl, R. (eds.), Vol II. (World Scientific) 1994. pp. 1555 -  
1562.

If it indicates an article published by a journal, then

### Example:

[Jakeman93b] Jakeman, A. J. & Hornberger, G. M.

"How much complexity is warranted in a rainfall-runoff model?", *Water  
Resources Research*, Vol. 29, No. 8, 1993. pp. 2637 - 2649.

## SAMENVATTING

Van het proefschrift door Ying LIU, getiteld

### Object Georiënteerde Behandelingen in Beslissingsondersteunende Systemen voor Waterbeheer

Dit proefschrift, door Ying LIU, behandelt object georiënteerde disciplines en software couplings in 'decision support system' (DSS) voor waterbeheer.

Feitelijk is DSS software die gebruikers in staat stelt kennis informatie en data te gebruiken om zodoende een besluitvormings proces uit te voeren of collectief beslissingen te nemen. Onderzoek toont aan dat er behoefte bestaat aan verdere ontwikkeling van DSS als technisch object.

De in ons onderzoek onderzochte elementen zijn afkomstig van waterbeheer. In dit vakgebied geven bepaalde situaties grote moeilijkheden betreffende besluitvorming, om zodoende accuraat te blijven betreffende ontwikkeling en samenwerking in relatie tot technici, onderzoekers en beleidsmakers. Tevens vereist goed management een voldoende begrip van de natuur, een juiste inschatting van hoe waterschap-technische projecten en criteria relateren aan menselijke- en milieu aspecten. Inzicht in wat moet worden bevordert en wat niet belangrijk is.

Als we waterbeheer beschouwen als een toepassing voor DSS zien we dat beide ondubbelzinnig een sterk contrast vormen met object georiënteerde disciplines. Voorgaande wil zeggen dat goed management relatieve stabiliteit vraagt aangaande het object, maar object-orientatie is multivariabel; DSS gaat er van uit dat het object in zichzelf een duidelijk geheel vormt, met welbekende begrenzing en eigenschappen.

Zodoende is het noodzakelijk onderzoek te doen naar pertinent object georiënteerde concepten en deze als zodanig te automatiseren.

Het is uitermate wenselijk de mogelijkheden die GIS ( geografische informatie systemen) biedt te gebruiken bij besluitvormings ondersteuning, daar de betrokken kennis die dient te worden behandeld hoofdzakelijk ruimtelijk van aard is; Het is wenselijk een open platform aangaande GIS te realiseren om zodoende een flexibel gebruik van de informatie te bewerkstelligen en anderzins een stabiele werkomgeving voor de informatie te creëren. Software conceptuele schema en data modellen hebben een cruciale invloed op de ontwikkeling van een dergelijk platform.

De ervaring leert echter dat de invloed van de drie bovengenoemde bepalende factoren enorme beperkingen hebben als ze onafhankelijk van elkaar worden benaderd. Benadert men de drie factoren echter als een geheel dan kan aan de meest voorkomende problemen het hoofd worden geboden. Volgens bovengenoemde strategie worden 'domein-driven data modelling, storing and retrieving data-entity, knowledge acquisition, en effective knowledge representation' in het proefschrift behandeld.

In deze context verwijst de term 'pattern' naar samenvoeging van informatie niet alleen afkomstig van een specifiek object, maar tevens van het intrinsieke samengaan van verschillende objecten, elkaar overlappend, elkaar omsluitend of anderzins. Een

dergelijke organisatie als voorgaand omschreven wordt aangeduid met de term 'patroongestuurd'.

Voorgaande komt overeen met recente opvattingen betreffende software engineering en deze opvattingen vormen een blijvende verwijdering van een computer-georiënteerd uitgangspunt: hoe een oplossing te implementeren in de computer? Daar tegenover stelt dit proefschrift een probleemgeoriënteerde (of kennis georiënteerde) aanpak.

Het is opmerkelijk dat tegenwoordig nogal wat aandacht besteed wordt aan koppeling van heterogene databases met model-based systemen. Echter, er zijn situaties waarin beslissings-processen bestaan uit cluster-analyses waarbij de hoofdzaak is het grote aantal objecten te splitsen in een kleiner aantal homogene groepen op basis van multi-variabele observaties. Naar aanleiding van een aantal steekproeven wordt een datatype gerealiseerd; deze vorm van data moet 'dynamisch worden opgeslagen' in een database, daar het terugvinden van deze data afhankelijk is van de context waarin deze wordt toegepast. Voor dit soort data is in dit proefschrift een 'multiple-index' functie ontwikkeld, om zodoende toegang te verkrijgen tot verwante databases.

In een datatype van 'repertory grid', wordt aan een aantal losse elementen, die de gekwalificeerde objecten vertegenwoordigen, een plaats gegeven in het bovenste deel van het schema. Als zo danig kunnen diverse combinaties gerealiseerd worden en worden aangeboden via computer netwerken.

Met behulp van de expertise vermeld in het rapport van de stuurgroep GMN ('een nieuw evenwicht', Lelystad, 1992) is de data voor de beleids-taxatie ingedeeld in vijf groepen met inachtneming van de industriële toepassing, milieu bescherming en drink-, oppervlakte- en grondwater gebruik. De betrokken beleids scenarios zijn gevisualiseerd volgens deze indeling in het ARC/INFO systeem via Unix.

ARC/INFO is een software gereedschap voor het realiseren van GIS. AML programma's vormen een reeks van ARC/INFO commando's, voor 'geo-processing' calculaties; AML is het meest essentiële onderdeel van ARC/INFO systemen (GIS). AML wordt in dit proefschrift gebruikt voor patroon-bestuurd data-pad en dynamisch menu, d.w.z. indien een bepaalde categorie data beschikbaar is.



## **Curriculum Vitae**

The author was born in Beijing, China, on 2 August 1960. He graduated with BSc degree in computer science from Xi' an Institute of Technology, Xi' an, China in 1982. He then worked as a lecturer assistant until 1986. He studied in software engineering at computer science department, Queen' s University of Belfast, and employed as a research fellow in the field of parallel translation of programming languages at Computing Department, Nottingham Trent University in Britain.

In 1991, he was employed by applied computer science department, Delft University of Technology in The Netherlands, working for the project called DIAC (Delft Intelligent Assembly Cell). In 1992, he worked as a research assistant pursuing PhD degree at the Department of Civil Engineering Informatics for the project of decision support system for water resources management. He was supervised by Prof. dr. Henk Koppelaar and Prof. dr. ir. Peter van der Veer.