An Intelligent Clinical Information Management Support System for the Critical Care Medical Environment

Dimitris Kalogeropoulos

Thesis submitted for the degree of Doctor of Philosophy

Centre for Measurement and Information in Medicine, City University

February 1999

Table of Contents

List of Figures

- 2 -

- 3 -

- 4 -

- 5 -

List of Tables

List of Program Listings

Acknowledgements

I wish to express my appreciation to my research supervisor Professor Ewart Carson at City University and clinical advisor Doctor Paul Collinson at Mayday University Hospital, for entrusting me with the completion of this project. Special thanks are extended to Professor Ewart Carson and the Centre for Measurement and Information in Medicine which he directs, for financing my participation in international conferences.

I also wish to thank a number of people for their enthusiasm, guidance, and support. Special thanks are extended to Ewart Carson, Paul Collinson and Professor Derek Cramp at City University, and Ian Mason at West Middlesex University Hospital. Special thanks are also extended to a number of people at Rational Technology Ltd. I thank Mike Tudball for his assistance in the implementation of object-oriented blackboard systems, Jim Grant and John Lambert for their efforts to arrange support, and especially Grady Booch, chief scientist at Rational, for sending me a pre-print of his latest book on object oriented methods.

I would also like to thank Doctor Ron Summers at City University for correcting the manuscript, Professor Alan Rector at the University of Manchester, Medical Informatics Group of the Department of Computer Science for co-examining the thesis, and Doctor John Chelsom Junior, for the most inspiring work that he left behind to guide me into the abyss of knowledge based-systems.

Special thanks also to my parents, to whom this work is dedicated for their loving support, and Paul Zwillenberg, managing director of Associated Electronic Publishing, Associated Newspapers Ltd., for kindly offering the use of their equipment for processing and printing the first manuscript.

Declaration

I grant powers of discretion to the University Librarian to allow this thesis to be copied in whole or in part without further reference to me. This permission covers only single copies made for study purposes subject to normal conditions of acknowledgement.

Abstract

Significant advances have been achieved in the fields of medical informatics and artificial intelligence in medicine in the past three decades and, having demonstrated an ability to support clinical decisions, knowledge-based systems are becoming increasingly ubiquitous in various clinical settings. Nonetheless, few systems have so far been successful in entering routine use. On the one hand, primarily due to methodological difficulties and with very few exceptions, developers have failed to show that pertinent systems are effective in improving patient care. On the other hand, support systems have not been sufficiently well integrated into the routine information processing activity of the clinical users. As a consequence, their clinical utility is disputed and constructive assessment is further hindered.

This thesis describes the development of an intelligent clinical information management support system designed to overcome these obstacles through the adoption of an integrated approach, geared toward the solution of the problems encountered in the acquisition, organisation, review and interpretation of the clinical decision supporting information utilised in the process of monitoring intensive care unit patients with acid-base balance disorders. The system was developed to support this activity incrementally, using the methods of object-oriented analysis, design and implementation, with the active participation of a clinical advisor who assessed the functional and ergonomic compatibility of the system with the supported activity and the integration of a previously validated prototype knowledge-based data interpretation system, which could not evaluated in the clinical setting for the reasons described above.

- 1 -

Introduction

1.1. Background

Medical professionals in the current high-technology health care environment, are under increasing pressure to improve on the quality and cost-effectiveness of their patient management decisions, while the volume of clinical information they are forced to absorb in the process is constantly expanding. Thus, there is a pressing need to develop usable computer-based tools for the solution of the medical decision support problems encountered in the presently information-overloaded health care delivery system. This is particularly true of the critical care environment (Price and Mason, 1986; Wright et al, 1991; Carson et al, 1991; Hayes-Roth et al, 1989, 1992), especially in relation to clinical laboratory usage (Speicher and Smith, 1983; Bradshaw et al, 1984; Cramp and Baron, 1985; O'Moore, 1995), which can be regarded as an exemplar of what holds true for the wider health care system (Cramp and Carson, 1985; Jennett, 1986; Greenes and Shortliffe, 1990; Henkelman, 1995; Berleur et al, 1995; Saranummi, 1995; van der Werff, 1997).

Medical informatics researchers have for long maintained that the effective introduction of information technology (IT) to the task of supporting and facilitating clinical decisions, will help improve the quality of patient care, optimise the cost-benefit equation, and ultimately transform the traditional structure of health care provision (Schwartz, 1970; Shortliffe, 1984; Cramp and Carson, 1985; Greenes and Shortliffe, 1990; Wong and Abendroth, 1996; van der Werff, 1997). Substantial evidence has been produced to support these claims, the strongest case being de Dombal's wide-spread system for computer-assisted diagnosis of acute abdominal pain (de Dombal et al, 1972, 1974, 1975; 1984, 1991; Horrocks et al, 1972, 1975, 1976; Wilson et al, 1975; Gunn, 1976, 1991; Adams et al, 1986; Anderson et al, 1988; McAdam et al, 1990), and systems designed to provide IT-based solutions to a variety of medical decision support problems, are becoming increasingly ubiquitous in the clinical setting (Miller, 1994). Nonetheless, few systems have so far been accepted in routine clinical practice.

Medical decision support problems can be broadly defined as determining, how, when, and in what manner to provide information to health care professionals in order to increase the quality of their decisions with respect to individual patients or populations of patients (Aliferis and Miller, 1995). In developing IT-based solutions to such problems, a plethora of useful strategies and techniques have been explored to provide assistance with all aspects of the process of clinical decision making by physicians, nurses and other health care professionals, which range from the acquisition and review of patient data to the cognitive information processing activity underlying diagnosis, prognosis, and therapy (Shortliffe, 1987, 1989, 1990; Greenes and Shortliffe, 1990; Pryor, 1990; Wyatt, 1991; Carson et al, 1991; Groth and Collinson, 1993; Grémy and Bonnin, 1995; van der Loo, 1995). To this end, numerous prototype knowledge-based systems (KBS) have been developed (Cramp and Goodyear, 1989; Talmon and van der Loo, 1995), merging techniques from the fields of artificial intelligence (AI) and decision science (Szolovits, 1982; Clancey and Shortliffe, 1984; Kulikowski, 1988; Miller, 1988; Horvitz et al, 1988; Shortliffe, 1991, 1993; Cooper, 1993). Nonetheless, successful products have not yet emerged in the clinical setting and the underlying technologies have not diffused (Uckun, 1992; Shortliffe, 1993; Miller, 1994; Saranummi, 1995; O'Moore, 1995; van der Werff, 1977).

Excluding liability issues (Miller et al, 1985; Miller, 1989; Brahams and Wyatt, 1989; Brender, 1997), the evolutionary development and the final decision to introduce into routine practice any type of computer system claiming to improve the quality of clinical decisions, depends on the results of a three-phased process which parallels the evaluation of pharmaceuticals (Wyatt and Spiegelhalter, 1990; Wyatt, 1992). In the case of evaluating the development of medical KBS, and by contrast to other application areas, this is a lengthy process (Ohmann et al, 1995) which involves considerable methodological complexity. The difficulty faced is that due to the nature of the decisions they are designed to support, and the inherently imprecise and incomplete knowledge they employ in the process, all medical KBS are to some extent heuristic (Aliferis and Miller, 1995). That is, their knowledge base is the result of the subjective experience of individuals who are expert in managing the uncertainty

and complexity that results from reasoning with partial belief and incomplete information in a particular domain. And since opinion disagreements occur often, even among experts (Yu et al, 1979b; Miller, 1988b), there is neither an unambiguous standard against which to compare the accuracy of the generated decision-supporting information, nor a precise and absolute method for establishing the correctness of the process by which it is generated (Aliferis and Miller, 1995; Talmon and Smeets, 1995).

Hence, it is not surprising that out of a considerable number of assessments reported in the literature, a large proportion are concerned only with the first phase of the evaluation process (van der Loo, 1995; Talmon and van der Loo, 1995; Brender, 1997), which comprises an assessment of the validity of the knowledge contained in the system and an assessment of the accuracy of the results produced from its application to selected test cases (Engelbrecht et al, 1995; Talmon and Smeets, 1995). This means that most KBS have not reached the second and third phases, to be independently evaluated for performance in clinical environments (Uckun, 1994; Ohmann et al, 1995), and thus lack the evidence required to prove that they have a beneficial and cost-effective impact on the structure, process and outcome of patient care, measured in terms of respective parameters such as efficacy, efficiency and effectiveness, length of stay, complications, morbidity and mortality (van Gennip, 1995; Jørgensen, 1995; Nøhr, 1995; Consorti, 1995). The consequence is that by failing to observe and assess the systems concerned at work, developers have for long omitted many of the important criteria for effectiveness and acceptance in the clinical setting; the most important being the functional, cognitive and ergonomic compatibility of the support system in question with the real needs and activities of the clinical user (Shortliffe, 1982; Miller and Masarie, 1990; Uckun, 1992; Wielinga et al, 1992; O'Moore, 1995; Peterson et al, 1995; Brender and McNair, 1996; Beuscart-Zephir et al, 1997; van der Hofstede et al, 1997; Brender, 1997).

In order to provide the means to assess these problems and to develop effective and usable solutions, KBS and other clinical decision support systems must be integrated within the information processing activity of the clinical user, for the development of systems geared toward supporting the management of clinical information (Greens

and Shortliffe, 1990; Hunter et al, 1991; Uckun, 1992; Groth and Collinson, 1993; Shortliffe, 1993; Stefanelli, 1993; Wong and Abendroth, 1996; van der Werff, 1997). Early examples of the clinical efficacy of this approach include the MYCIN-derived EMYCIN-PUFF system (Aikins et al, 1983), the HELP-COMPAS-CORE hospital information system (HIS) integrated decision support modules (Pryor et al, 1983; Pryor, 1988; Sittig et al, 1989, 1990; Henderson et al, 1989; 1991), and QMR (Miller et al, 1986a,b; Masarie and Miller, 1987), which was originally designed to function as a standalone consultation system (Miller and Masarie, 1990) in the pioneering INTERNIST-1, II and CADUCEUS AI incarnations (Pople, 1977; Pople, 1982; Miller et al, 1982; Miller, 1984; Masarie et al, 1985), and which became successful in clinical practice (Bankowitz et al, 1989a,b; Parker and Miller, 1989) and medical education (Miller and Masarie, 1989; Bankowitz et al, 1989a), following its conversion into an integrated clinical information management tool. Similar results were also observed in the development of the ONCOCIN system, which operates as a therapy critiquing tool within an integrated patient record and monitoring system (Langlotz and Shortliffe, 1983; Shortliffe, 1986). In fact, the few KBS that have been accepted in clinical practice have emerged primarily in data-intensive environments where they fulfil a practical and perceived need to help control the information overload problem (O'Moore, 1995).

In contrast to the traditional involvement of the clinician during the initial stages of support system development and during the final stages of the evaluation process as a source and referee of knowledge and expert opinion, the recommended approach requires that a clinical advisor must be actively involved throughout the integration-development process, to assess the extent to which his or her needs are satisfied with respect to the above issues. This points to an iterative, incremental, and user-driven, life-cycle development methodology, in agreement with the concept of formative rather than summative assessment. In contrast to the methodology of summative evaluation, which aims to provide an empirical measurement of the performance and effectiveness of an already developed system (Engelbrecht et al, 1995; Brender, 1997), with such a development methodology, the assessment activities can be used constructively as a means for making decisions actively, on directions and corrections

to apply during the development of a clinical decision support application (Brender et al, 1995; Brender, 1997). Thus in the case of constructively assessing the integration and evolutionary development of clinical decision support tools within routine clinical practice, the clinical advisor is required to operate as a source of expert opinion both in the sense of being a source for the elicitation of medical know-how, as well as in the sense of being a source for the elicitation of know when and in what manner to generate the decision supporting information

1.2. Aims and Objectives

This thesis describes the incremental, assessment-driven development of an intelligent clinical information management support (ICIMS) system for the critical care environment. The underlying aim was to overcome the aforementioned obstacles encountered in the development and dissemination of clinical decision support KBS, by designing a system that supports and facilitates the solution of the problems associated with the observation, interpretation, monitoring and management of complex clinical events, in the context of dealing with the wider problem of information management in the modern intensive care unit (ICU). The chosen problem domain for the development of a prototype system was the management of clinical information on patients with abnormalities of acid-base balance. The application domain was chosen because the management of such patients requires having a clear clinical picture and taking prompt action (Collinson et al, 1990), while the interpretation of the constantly expanding set of related parameters (Siggaard-Andersen et al, 1990) is subject to considerable error rates (Schreck et al, 1986).

A prototype KBS had been developed within the research to provide tools for the acquisition, representation and manipulation of the domain knowledge-base required for interpretative decision-making in the domain of acid-base balance, and to thereby assess the performance of a singly connected hierarchical belief network in providing assistance with the interpretation of blood-gas laboratory analysis data (Chelsom, 1990). The validated prototype, which was implemented in a logic-based environment

(Prolog) to function as a standalone consultation system, was retrospectively evaluated with 60 challenging cases and was found to perform at the level of the expert who designed the knowledge-base. Overall, the study showed that there was considerable disagreement between the human decision making participants and that, as expected, the system could not distinguish between the members of the represented classes of complex disorders. Excluding such cases, the prototype was in agreement with either with the expert or senior clinician involved in the study in 83% of the cases, using three commonly measured parameters. This performance could have been improved by training the network, given data in sufficient quantity and quality (Jørgensen, 1995b). Furthermore, the ability to recognise complex disorders could be developed given access to a comprehensive patient database.

Nonetheless, as discussed above, these results are on their own insufficient as criteria for the decision whether to admit the system or reject it from further evaluation. It was therefore decided to use the knowledge contained in the prototype KBS, and its basic inference strategy for evidence propagation in the domain knowledge belief network, and to develop an integrated system geared toward the management of the clinical information generated in the process of monitoring the ICU patient with acid-base balance disorders. The objectives of the work involved in the integration of the KBS data interpretation prototype within the information processing activity of the clinical user and the development of the resultant ICIMS system architecture were:

1. To develop a system which would combine the computer-based clinical decision support tasks of the acquisition, organisation, storage, update and review of the information generated in the process of monitoring the ICU patient, as well as of the domain knowledge-base required for the contextual interpretation of the acquired clinical information, within a singular system architecture.

2. To use the clinical information management support system in order to develop and constructively assess the integration of the cognitive, clinical information processing tasks comprising the prototype KBS interpretative problem-solving task-domain into the ICIMS system, and consequentially into clinical practice,

in order to incorporate the computational intelligence necessary for the interpretation of the patient data acquired in the process being supported.

3. To provide the means to assess specific problems encountered in the integration process, and to develop effective and usable solutions, by employing an approach which would enable the active participation of a clinical advisor who would act as an assessor of the functional, cognitive and ergonomic effectiveness of the KBS integration process, and of the overall decision support provided by the ICIMS system during its development.

1.3. Thesis Outline

Chapter 2 discusses the fundamental problem of reasoning with partial belief and incomplete information that characterises clinical decision making, and reviews a number of formal and heuristic methods of inexact reasoning which were developed to replace probability for the acquisition, representation and manipulation of uncertain medical knowledge, due to misconceived limitations of the theory for the task. This analysis exposes that these methods promote errors in judgement and lead to interpretative decision making behaviours of poorer performance and accuracy, thereby justifying the decision to use the system described in chapter 3, as the integration prototype for the incorporation of the computational intelligence required in the development of the ICIMS system.

Chapter 4 describes the development of the core of the ICIMS system, that being an object-oriented clinical information management system designed to combine the tasks of acquisition, organisation, storage, update and review of the information generated in the process of monitoring the ICU patient, as well as of the domain knowledge base required for the interpretation of the acquired clinical information, within a single system architecture. Following that, the chapter describes the integration of the KBS prototype cognitive information processing task model, for the

incorporation of the computational intelligence required for the interpretation of the acquired clinical information.

Chapter 5 presents an overall view of the ICIMS system implementation, describing its ergonomic features and the decision support functions it provides, justifying it cognitive compatibility with the information processing activity of the clinical user in a critical care environment. Finally, Chapter 6 presents the conclusions drawn from the work described in this thesis, discusses the contribution of the work to the fields of medical informatics and clinical medicine and summarises the recommendations made in Chapter 5 for further development.

- 2 -

Medical Uncertainty Management in Knowledge-Based Systems

2.1. Introduction

As stated in the introduction to this thesis, the difficulty faced in developing and evaluating medical KBS is that the knowledge employed in the process of supporting and facilitating clinical decisions is inherently imprecise and incomplete. Therefore, the key element of medical KBS is the methods employed for the acquisition, representation, and manipulation of the interpretative uncertainty which results from the lack of information typically encountered when processing case evidence.

The present chapter provides an introduction to the complex task of reasoning with partial belief and incomplete information, and discusses a number of methods that have been explored for the management of uncertainty in medical KBS. The review starts with the formal probabilistic approach to interpretative decision-making, and proceeds to review and discuss a number of alternative, heuristic methods, which were devised to overcome the apparent limitations of the former. It turns out that albeit designed to have the opposite effect, these non-probabilistic approaches have been based on erroneous heuristic assumptions of inexact reasoning, which appear to promote errors in reasoning also made by human experts.

Having discussed these problems, the chapter discusses simulation reasoning and the use of causal models in supporting the management of uncertainty. Finally, the chapter describes belief networks as a formal probabilistic approach which, building on the knowledge representation methods explored in heuristic systems, supports and facilitates the explicit representation of domain knowledge modelling assumptions, thereby assisting the proper communication and management of uncertainty.

The chapter which follows describes the structure and performance evaluation of the prototype knowledge-based data interpretation consultation system, which was developed to reason with partial belief and incomplete information for the interpretation of laboratory investigation data on disorders of acid-base balance, using a singly connected hierarchical belief network.

2.2. Introduction to Reasoning with Partial Belief and Incomplete Information

There are a number of ways in which to characterise and categorise medical KBS, depending on whether their knowledge content is based on associative, behavioural, functional, or structural causality, the form of the selective representation of such dependencies among medical events, be that rules, frames, or other objects, and the purpose of their knowledge base, that is, automated diagnosis, explanation, tutoring, therapy management, critiquing, monitoring, etc. (Clancey, 1985, 1989; Uckun, 1992; Ramoni et al, 1992; Stefanelli, 1993). Regardless of any such categorisation, all medical KBS are designed to model and support the information processing activities or tasks underlying clinical cognition (Newell, 1982; Clancey, 1992).

From the viewpoint of medical professionals, their cognitive activity is essentially intuitive and thus cannot be modelled (Kassirer and Gorry, 1978; Blois, 1980). However, the repeated observation of certain prototypical classes of patients facilitates conceptual descriptions of these classes at multiple levels of a cognitive abstraction hierarchy (Blois, 1988), which can be used as knowledge-based inferential tools for the recognition and classification of patterns of evidence (Ledley and Lusted 1959; Feinstein, 1963; Duda and Shortliffe, 1983; Blois, 1983; Chandrasekaran, 1986; Clancey, 1985), purporting to place the patient in some diagnostic, prognostic and therapeutic context (Cramp and Baron, 1985; Schmidt et al, 1990), in which interpretative decisions must be made as to the degree of fit of the evidence to the case, in order to assess alternative hypotheses, actions and outcomes, request further information, and eventually select the case that best matches the available evidence (Pople, 1982; Clancey, 1985; Szolovits et al, 1988; Horvitz et al, 1988; Clancey, 1992).

The difficulty faced by the clinician (Tversky and Kahneman, 1974; Blois, 1988; Kassirer and Kopelman, 1989; Heckerman et al, 1992) and in developing medical KBS (Davis, 1982; Aliferis and Miller, 1995), is that due to the vast amount of medical knowledge (Pauker et al, 1976; Miller, 1984) and the resultant inferential complexity, the exact sequence of causal events underlying the pathogenesis of an observed clinical problem is often implied in the form of empirical or heuristic

associations between findings constituting evidence and diseases. Such knowledge is formed at the highest levels of the cognitive abstraction hierarchy and is by definition imprecise and incomplete, and furthermore, due the inherent variability of human physiology and anatomy, not generally applicable.

For interpretative problem domains with access to comparatively exact biological knowledge (Blois, 1988), allowing the causal simulation or synthesis of the behaviour, structure and function of the body organs involved in the pathogenesis of a problem being solved, it is possible to, at least in part, construct non-classificatory and thus non-heuristic systems (Patil et al, 1982; Patil, 1987; Szolovits et al, 1988; Miller and Fisher, 1988; Patil and Senyk, 1988; Clancey, 1989; Uckun, 1992, 1994), which once verified for their internal consistency should provide accurate solutions. However, for the majority of medical decision support problems, primarily due to resource limitations (Blois, 1988; Uckun, 1992; Hayes-Roth, 1989, 1992), clinicians working with KBS developers are forced to introduce heuristic simplifications at various phases of domain knowledge modelling, including elicitation, design or representation, and implementation (Aliferis and Miller, 1995). Consequentially, the accuracy of the classifications or contextual interpretations these systems perform, depends on the method used for the acquisition, representation, and manipulation of the uncertainty that results from the imprecision and incompleteness of their knowledge base, in order to reason with partial belief and incomplete information.

2.3. Classification Reasoning

2.3.1. Introduction

This section describes classification reasoning and a number of computer-based methods for the acquisition, representation and manipulation of the interpretative uncertainty with which patient classification decisions are made. The section starts with the simple-Bayes model of interpretative decision-making and proceeds to discuss a number of heuristic methods which were devised to overcome certain

misconceived limitations in the application of probability theory to the task. As mentioned above, these heuristic models of inexact reasoning were shown to have both theoretical as well as practical limitations, however, their development exposed key issues in the acquisition, representation and manipulation of medical knowledge, which led to the development of qualitative modelling for KBS and belief networks, combining the former with the formal calculus of probability theory for the proper communication and management of the interpretative uncertainty encountered in processing clinical information.

2.3.2. Classification using Probabilistic Inference for Interpretative Decision-Making

2.3.2.1. The Complete Model

Using the original version of Bayes' theorem for reversing the direction of probabilistic inference, clinical decision support KBS may solve the problem of contextual data interpretation in their respective domains, in the following manner (Heckerman et al, 1992).

Suppose the represented or modelled domain comprises m known diseases, or patient classifications, and n disease features which can be used as evidence for the recognition of the m diseases. Let $d_1, d_2, ..., d_m$ denote the m represented diseases, and D_k some instance of these diseases. If each disease d_j may be present in the patient or absent, then D_k denotes some assignment of present or absent to each of the diseases $d_1, d_2, ..., d_m$. Furthermore, let $f_1, f_2, ..., f_n$ denote the domain features, and let i_j denote the observed instance for the jth feature. Now, for simplicity suppose that the system has been supplied with observed instances for the first q out of the n features, and is required to generate the probability of each disease instance, given the observations $f_1i_1, f_2i_2, ..., f_qi_q$. This quantity is known as the a posteriori probability of D_k, i.e. that which is used for reasoning in the required diagnostic direction, or from observed effects to probable causes, and is denoted:

$$p(D_k|f_1i_1, f_2i_2, \ldots, f_qi_q, \xi). \tag{2.1}$$

Given an expert assessment of the probabilities that the set of observations $f_1i_1, f_2i_2, \ldots, f_qi_q$ will appear given a particular disease instance D_k, denoted:

$$p(f_1i_1, f_2i_2, \ldots, f_qi_q|D_k, \xi). \tag{2.2}$$

and the a priori probability of disease instances $p(D_k|\xi)$, i.e. that which is used for reasoning in the predictive direction, or from causes to effects, Bayes' theorem can be used to reverse the direction of inference and generate the desired a posteriori probability of each disease instance using the following formula, where the sum over D_l runs over all disease instances:

$$p(D_k|f_1i_1, f_2i_2, \ldots, f_qi_q, \xi) = \frac{p(f_1i_1, f_2i_2, \ldots, f_qi_q|D_k, \xi)p(D_k|\xi)}{\sum_{Dl} p(f_1i_1, f_2i_2, \ldots, f_qi_q|D_l, \xi)p(D_l|\xi)} \tag{2.3}$$

By far the biggest asset of the probabilistic model of interpretative decision-making for classification reasoning is that it is based on a well-defined formal theory of reasoning with partial belief and incomplete information. However, its application to non-trivial real-life classification problems results in a debilitating if not prohibitive knowledge acquisition effort. Taking into consideration that a patient may have more than one disease present, and the number of the possible sequences of the diagnostic tests generating the evidence, the number of probabilities of the form $p(f_1i_1, f_2i_2, \ldots, f_qi_q|D_k, \xi)$ which have to be acquired is exponential both in the number of diseases and in the number of features.

2.3.2.2. The Simplified Model

In very narrow domains the knowledge acquisition complexity which arises from the application of the complete model of probabilistic classification reasoning might not be a problem, however, to manage the complexity of the general case, researchers who built the first probabilistic KBS made the following two simplifying assumptions

(Szolovits and Pauker, 1978; Horvitz et al, 1988; Cooper, 1988; Heckerman et al, 1992).

The first assumption was that all findings were conditionally independent, given any disease instance. That is, if the true disease state of the patient was known, then the likelihood of making any observation $f_k i_k$ did not depend on observations made about any other features. Thus,

$$p(f_j i_j | D_k, f_1 i_1, \ldots, f_{j-1} i_{j-1}, f_{j+1} i_{j+1}, \ldots, f_q i_q, \xi) = p(f_j i_j | D_k, \xi) \tag{2.4}$$

Given this assumption it follows from the rules of probability that:

$$p(f_1 i_1, f_2 i_2, \ldots, f_q i_q | D_k, \xi) = p(f_1 i_1 | D_k, \xi)\, p(f_2 i_2 | D_k, \xi) \ldots p(f_q i_q | D_k, \xi) \tag{2.5}$$

The second assumption was that the represented diseases were mutually exclusive and collectively exhaustive. That is, each disease instance corresponded to a situation where only one disease was present. Given these two assumptions, the required a posteriori probabilities can be generated by:

$$p(d_k | f_1 i_1, f_2 i_2, \ldots, f_q i_q, \xi) = \frac{p(f_1 i_1 | d_k, \xi) p(f_2 i_2 | d_k, \xi) \ldots p(f_q i_q | d_k, \xi) p(d_k | \xi)}{\sum_{d_l} p(f_1 i_1 | d_l, \xi) p(f_2 i_2 | d_l, \xi) \ldots p(f_q i_q | d_l, \xi) p(d_l | \xi)} \tag{2.6}$$

where d_k represents the disease instance in which only disease d_k is present. Thus, only the mn conditional probabilities $p(f_j i_j | d_k, \xi)$ and the m-1 a priori probabilities $p(d_k | \xi)$ are required for the computation.

This simplified model of interpretative decision making for classification reasoning was proposed nearly forty years ago (Ledley and Lusted, 1959) and is termed the simple-Bayes model. Early applications include the domains of congenital heart disease (Warner et al, 1961; 1964; Gorry, 1973), disorders of thyroid function (Overall and Williams, 1963), primary bone tumours (Lodwick et al, 1963; Gorry, 1973), acute renal failure (Gorry and Barnett, 1968; Gorry et al, 1973), and the acute abdominal pain program cited in section 1.1.

2.3.2.3. Performance Evaluation of the Simple-Bayes Model

In the 1970s, probabilistic diagnostic systems came under severe criticism for their reliance on oversimplified and normative methods to dissect, model and support clinical decisions (Feinstein, 1977). However, despite the two simplifying assumptions, the systems that used the simple-Bayes model performed as well, and in some cases better than experienced physicians.

For example, in one of the many evaluations cited in section 1.1, the system developed by de Dombal and his colleagues averaged over 90% correct diagnoses of acute abdominal pain, where expert physicians were averaging 65-80% correct (de Dombal et al, 1974). In a British multi-centre evaluation performed in 1986 with the participation of 250 doctors and 16737 patients, the system resulted in a rise of diagnostic accuracy from 45.6% to 65.3%, and a fall in rate of perforation among patients with appendicitis and negative laparotomy, from 23.7% to 11.5%. The bad management error rate fell from 0.9% to 0.2%, and the observed mortality fell by 22%. The savings made were estimated as amounting to 278 laparotomies and 8516 bed nights during the trial period, or the equivalent of annual savings in resources throughout the British National Health Service worth over 20 million pounds sterling (Adams et al, 1986).

Nonetheless, researchers argued that albeit their competent performance, the errors due to the assumptions of conditional independence and mutual exclusivity, would become unacceptable as the domains of these systems were expanded (Gorry, 1973; Shortliffe, 1976), and pursued alternative theories and methods for reasoning with partial belief and incomplete information, mostly but not always as an adjunct to AI techniques for knowledge representation (Szolovits and Pauker, 1978; Spiegelhalter and Knill-Jones, 1984; Cheeseman, 1985; Lauritzen and Spiegelhalter, 1988; Horvitz et al, 1988; Heckerman et al, 1992). However, none of these schemes were successful in overcoming the apparent limitations of probability theory. In the contrary, albeit designed to avoid certain misconceived limitations of probability theory (Cheeseman, 1985), the alternative ad hoc methods described below have been shown to have a probabilistic basis and, moreover, to contain implicit and obscured assumptions of inexact reasoning, which lead to larger errors than the formal probabilistic model

(Horvitz et al, 1988; Heckerman and Shortliffe, 1992; Heckerman et al, 1992). Specifically, it has been shown that these schemes assumed evidence was conditionally independent, given each disease and the negation of each disease, and that when there are more than two mutually exclusive and exhaustive diseases in a domain, these assumptions are stronger than are the assumptions in the simple-Bayes model (Heckerman et al, 1992).

The sections that follow describe the most significant contributions, the lessons learned, and the reasons that forced the research community to return to the calculus of probability as a system of personal belief, justifying its use both from a theoretical and a pragmatic perspective, in providing a flexible and operational means of uncertainty assessment, representation and manipulation, which helps physicians avoid the errors promoted during knowledge elicitation, reduces errors in reasoning with partial belief and incomplete information, and generates decision supporting information of higher accuracy.

2.3.3. Rule-Based Systems

Rule-based KBS were based on the AI hypothesis that an expert's knowledge can be represented in a large number of independent situation-specific rules of deduction, and that computers can simulate the problem solving behaviour of an expert by stringing these rules together in chains of logical implication, matching the premise or antecedent of one applicable rule with the conclusion or consequent of another (Shortliffe, 1986; Schwartz et al, 1987; Rennels and Miller, 1988; Keravnou and Washbrook, 1989).

The MYCIN system described below was the first in a family of rule-based KBS prototypes to be applied to the field of clinical medicine (Shortliffe et al, 1973; Wraith et al, 1976; Shortliffe, 1976; Buchanan and Shortliffe, 1984).

2.3.3.1. The MYCIN system for heuristic classification

MYCIN was designed to classify severe infections, such as meningitis or septicaemia, and to recommend treatment. The system solved the problem of identifying the organism causing the infection using two types of rules: rules for simple classification and rules for heuristic classification (Clancey, 1985). Simple classification rules were applied where definitional or factual features were available. For example, an unknown organism could be classified directly into the hierarchy depicted in Figure 2-1, given the supplied definitional features of gram-stain, morphology, and aerobicity using the following rule:

if: 1. The stain of the organism is *gram-negative*, and
2. The morphology of the organism is *rod*, and
3. The aerobicity of the organism is *anaerobic*.
Then: There is suggestive evidence [0.6] that
the type of the organism is *bacteroids*.

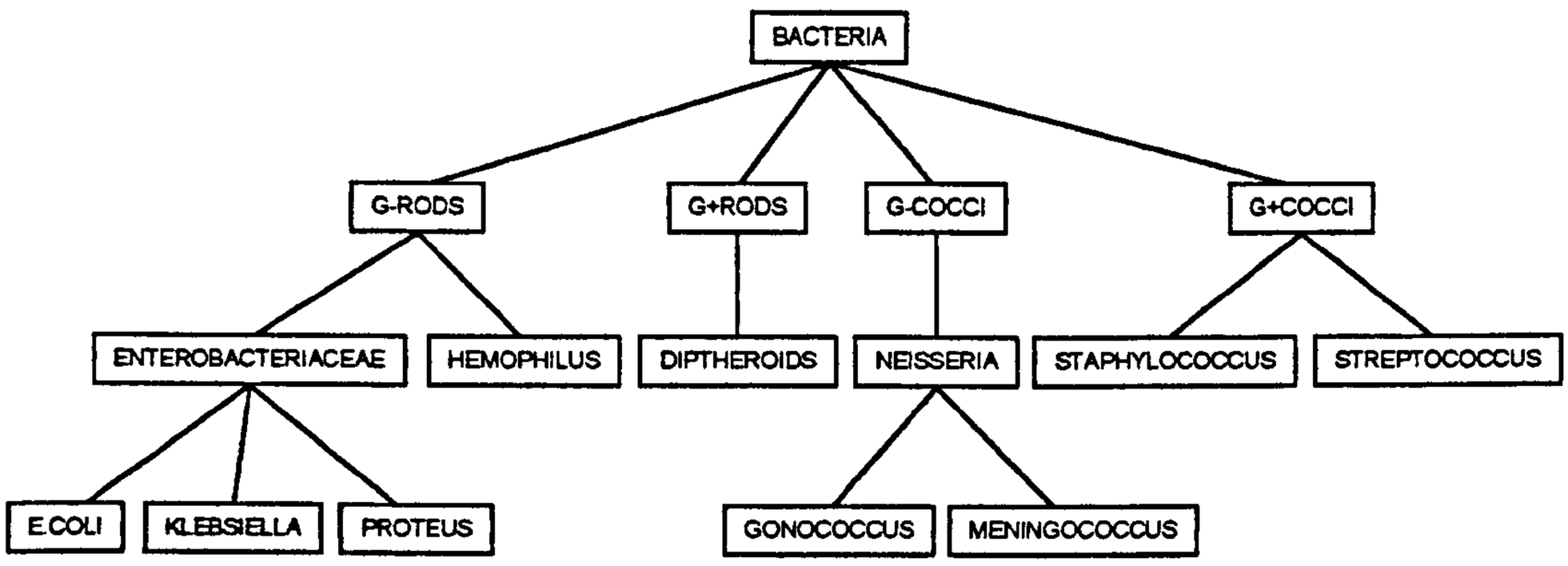


Figure 2-1. Hierarchical classification of bacteria in MYCIN (from Clancey, 1985).

Nonetheless, definitional associations such as the one represented by the above rule, offered evidence on general solution classes and the inferred solution class typically had to be refined, by acquiring heuristic information that enabled the discrimination of subtypes. In doing so, MYCIN heuristically and non-hierarchically related an abstract characterisation of the patient to the classification of organisms represented in its rule base. For example the system might refine the classification made by the above rule using the following heuristic information:

if: 1. The infection is *meningitis*.
2. The type of the infection is *bacterial*.
3. The patient has undergone *surgery*.
4. The patient has undergone *neuro-surgery*.
5. The *neuro-surgery-time* was 2-months ago.
6. The patient received a *ventricular-ureteral-shunt*.

Then: There is evidence that the organism which might be causing the infection is 1. *E.coli* [0.8]
2. *Klebsiella-Pneumoniae* [0.75].

Furthermore, in cases where solution features were not supplied as data, they were inferred using rules for data abstraction. There were three types of rules for this purpose: rules for definitional abstraction, qualitative abstraction, and generalisation in a subtype hierarchy. Figure 2-2 depicts the implicit inference structure which resulted upon execution by chaining together applicable MYCIN rules in a goal-driven manner.

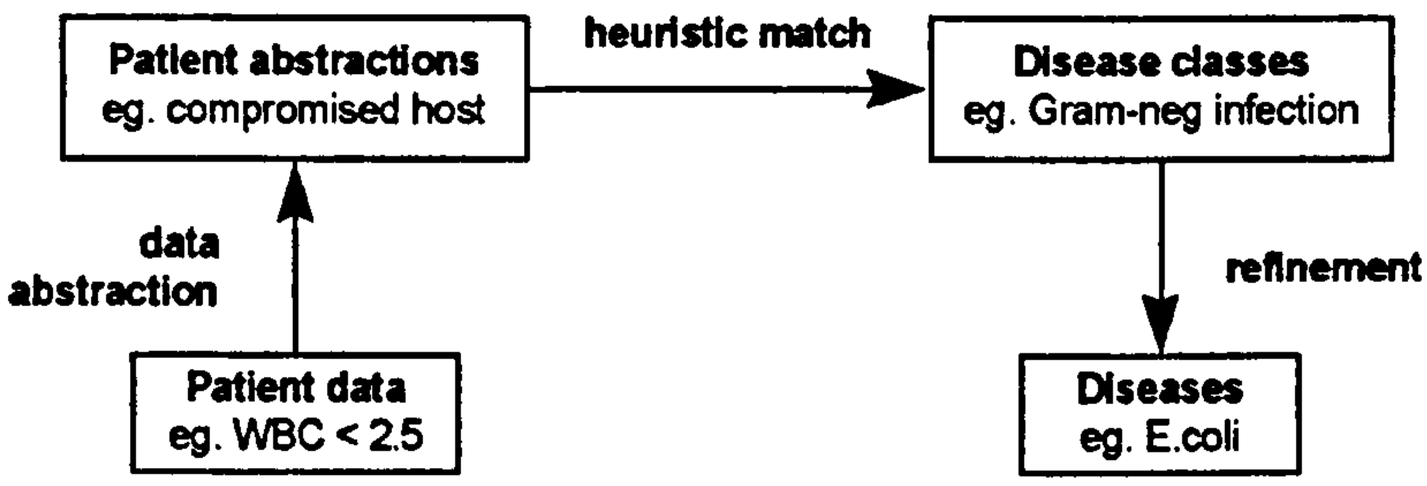


Figure 2-2. Implicit inference structure for simple and heuristic classification in the MYCIN rule-base (from Clancey, 1985).

Overall, basic observations about the patient were abstracted to patient categories which were heuristically linked to diseases and disease categories. For example, in the classification inference example shown in Figure 2-3, given a white blood cell count (WBC) of 2000, MYCIN might produce the following chain of rules, in reverse order. By qualitative abstraction, a WBC less than 2500 is a low WBC. Low WBC is by definitional abstraction leukopenia. Generalising, leukopenia is a form of immunosuppression, which by further generalisation is a form of compromised host. Finally, compromised hosts are heuristically or incidentally associated with Gram-negative infections, such as E.coli infection.

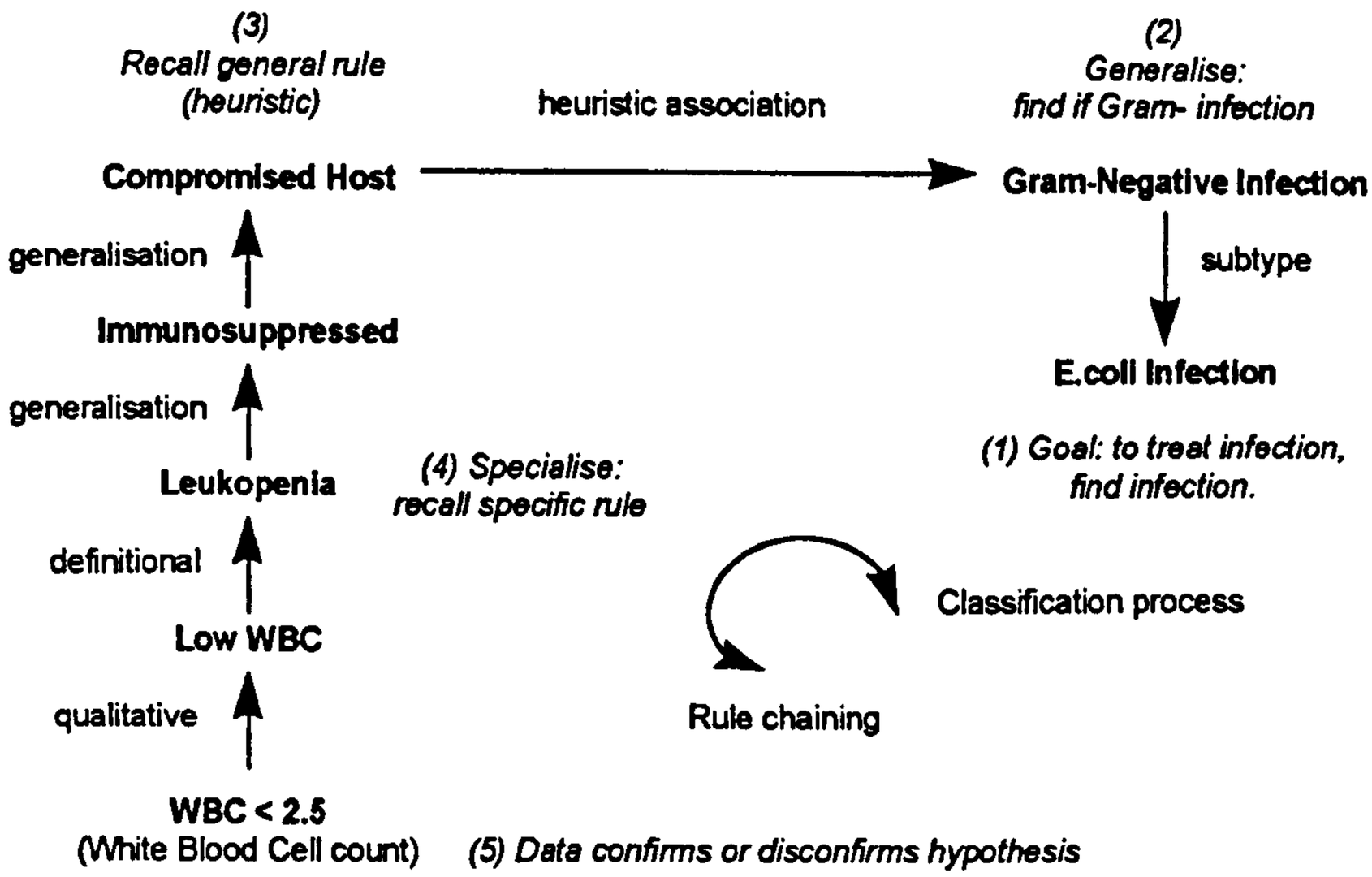


Figure 2-3. Example of the implicit inference structure which resulted from the application of the MYCIN rule-based (from Clancey, 1985).

Thus, the important link added in MYCIN was the heuristic association between general characterisations of the patient, for example "compromised host", and general categories of diseases, for example "Gram-negative infection", in the form of rules of inference (Clancey, 1985). However, as stated above, such associations are based on the assumption that there exists a chain of causal events between findings constituting evidence and diseases, and are therefore incomplete and uncertain. Consequentially there was the need to augment the rule-based knowledge representation formalism with a mechanism for the propagation of the uncertainty encoded in heuristic associations, expressed as the numeral in brackets in the example rules above, through the chain of logical implications.

The rule-based approach to domain knowledge acquisition, representation, manipulation and explanation adopted in the development of the MYCIN system, required a modular approach to uncertainty management, and the Bayesian model was found inadequate for the task. The certainty factor (CF) model was thus designed to function as a modular belief updating scheme, and to thereby overcome the limitations imposed on the simple-Bayes model by the assumptions of conditional independence between pieces of evidence and diseases (Shortliffe and Buchanan, 1975) However, as

discussed below, the property of modularity was shown to be invalid, resulting in hidden non-modular interactions and sudden degradation of system performance as a consequence of the expansion of the system's knowledge base. In fact this was a problem shared by all of the AI-based modular belief updating schemes described in section 2.3 (Heckerman et al, 1992; Aliferis and Miller, 1995).

2.3.3.2. The Certainty Factor Model

In using the CF model, the domain expert was asked to attach a certainty factor to each if-then rule in the knowledge base. The certainty factor was meant to represent the expert's change of belief in the consequent of the rule, given the antecedent. A CF between 0 and 1 meant that the expert's belief in a consequent increased if the antecedent was true, whereas a CF between -1 and 0 meant that the expert's belief decreased. In a rule base, the consequent of one rule might serve as the antecedent of another. In addition, two or more rules might share the same antecedent or consequent. As a result, the rule base formed an inference network: a directed graph in which an arc from proposition *A* to proposition *B* corresponded to the rule "if *A* then *B*". The CF model prescribed a method for propagating certainty factors through such a network. That is, given an observation of an antecedent in the network, CF functions were used to compute the effective certainty factor for any consequent in the network that was a descendant of that antecedent.

Although the CF model was designed for MYCIN, it found many applications in varied domains and became the most popular method for managing uncertainty in rule-based systems (Heckerman et al, 1992). As a result of its extensive use, there are many variations among the implementations of the CF model. One such implementation is described below (Heckerman and Shortliffe, 1992).

Suppose a rule base contains the following simplistic rules:

R_1 : IF acute abdominal pain THEN acute abdominal infection, $CF_1 = 0.5$
R_2 : IF diarrhoea THEN acute abdominal infection, $CF_2 = 0.7$
R_3: IF acute abdominal infection THEN appendicitis, $CF_3 = 0.5$

CF combination functions are applied to the CFs that lie between the evidence and the hypothesis in question in the following manner. Firstly, CF_1 and CF_2 are combined to give the CF for the new composition rule:

R_4 : IF acute abdominal pain AND diarrhoea
THEN acute abdominal infection, CF_4

using the following function:

$$\begin{aligned} CF_4 &= CF_1 + CF_2 - CF_1CF_2 && \text{if } CF_1, CF_2 \geq 0 \\ &= CF_1 + CF_2 + CF_1CF_2 && \text{if } CF_1, CF_2 < 0 \\ &= (CF_1 + CF_2) / (1 - MIN(|CF_1|, |CF_2|) && \text{otherwise} \end{aligned} \qquad (2.7)$$

Equation 2.7 is called the parallel-combination function, yielding $CF_4 = 0.5 + 0.7 - (0.5)(0.7) = 0.85$. Secondly, CF_3 and CF_4 are combined to give the CF for the new composite rule R_5:

R_4 : IF acute abdominal pain AND diarrhoea
THEN appendicitis, CF_5

The combination function for this case is:

$$\begin{aligned} CF_5 &= CF_3CF_4 && CF_3 > 0 \\ &= 0 && CF_3 \leq 0 \end{aligned} \qquad (2.8)$$

Equation 2.8 is called the serial-combination function, yielding $CF_5 = (0.85)(0.5) = 0.425$.

The serial and parallel combination functions are applicable where the evidence and hypotheses in a rule base are simple propositions, as in the case of R_1, R_2 and R_3. Since this is often not the case, the CF model also incorporated combination functions to accommodate rules that contained conjunctive and possibly disjunctive patterns of evidence. For example, suppose the following rule aids in the diagnosis of chest pain:

R_6 : IF chest pain AND shortness of breath
THEN heart attack, $CF_6 = 0.9$

Further, suppose that there are rules that reflect indirect evidence for chest pain and shortness of breath:

R_7 : IF patient grimaces THEN chest pain, CF_7= 0.7
R_8 : IF patient clutches throat THEN shortness of breath, CF_8= 0.9

CF_6, CF_7, and CF_8 can be combined to yield the CF for the new composite rule:

R_9 : IF patient grimaces AND patient clutches throat
THEN heart attack, CF_9

The combination function for this case is:

$$CF_9 = CF_6 \min(CF_7, CF_8) = (0.9)(0.7) = 0.63. \quad (2.9)$$

Finally, for evidence in a disjunction, combined CFs are computed using the maximum CF. That is, if the evidence in R_6 was disjunctive, the following combination function would be used:

$$CF_9 = CF_6 \max(CF_7, CF_8). \quad (2.10)$$

2.3.3.3. MYCIN's Performance Evaluation

Upon evaluation, the MYCIN system demonstrated an ability to perform at or near the level of expert physicians in performing the diagnosis of isolated bacteraemias and meningitis. More specifically, in the domain of bacteraemia, therapy recommendations made by the system met the Stanford experts' standards of acceptable practice 90.9 % of the time, with some variation noted both among individual experts and between Stanford experts and others (Yu et al, 1979a). In the domain of meningitis, eight independent evaluators with special expertise in the management of such infections compared MYCIN's choice of antimicrobials with the choices of nine human prescribers for ten test cases of meningitis. The system received an acceptability rating of 65% by the evaluators; the corresponding ratings for acceptability of the regimen prescribed by the nine faculty specialists ranged from 42.5% to 62.5%. MYCIN's

meningitis treatment advice was rated as acceptable by most blinded expert reviewers with the same frequency as was the actual delivered therapy (Yu et al, 1979b). Furthermore, the system never failed to cover a treatable pathogen while demonstrating efficiency in minimising the number of antimicrobials prescribed.

Despite this favourable evaluation, MYCIN never made it into routine clinical practice for the following reason. Rule-based KBS were designed to follow the tenets of AI most closely, in using symbolic reasoning and avoiding numerical assessments (Cohen, 1985; Shortliffe, 1986). However conceptually attractive it may seem, in medical domains deterministic modelling is inappropriate, since it violates the inherent uncertain nature of medical knowledge (Aliferis and Miller, 1995). Furthermore, despite the success of the CF model for managing the uncertainty of MYCIN's rule base, its developers warned that the model had been designed for a domain with unusual characteristics, and that the model's performance might be sensitive to the domain of application (Heckerman and Shortliffe, 1992). A sensitivity analysis of the CF model demonstrated that MYCIN's therapy recommendations were remarkably insensitive to perturbations in the CF values assigned to diagnostic rules in the system (Buchanan and Shortliffe, 1984). Since MYCIN was primarily a therapy advice system, and since antibiotics often cover for many pathogens, variations in diagnostic hypotheses often had minimal effect on the recommended therapy. MYCIN's diagnostic assessments, however, showed more rapid deterioration as CF values were altered.

MYCIN's developers noted that expanding the system beyond a certain level of complexity, led to unanticipated interactions between rules, and eventually sudden degradation of program performance (Clancey and Letsinger, 1981; Davis, 1982; Schwartz et al, 1987). The same observation was made in the development of PUFF, cited in section 1.1, the knowledge base of which was engineered using the EMYCIN (Essential-MYCIN) expert system shell (Aikins et al, 1983). PUFF is one of the few systems which were accepted for routine use, however, PUFF contained only a fraction of MYCIN's rule base. The original bacteraemia knowledge base of the MYCIN system comprised a combination of heuristic and factual associations. By contrast, the meningitis knowledge base was more complex as it could infer the organism class purely heuristically, from patient abstractions, without having a culture

result, and hence required a larger number of heuristic associations to eliminate uncertainty, encoded in a larger number of rules of inference (Clancey, 1985).

In order to overcome this problem, MYCIN's designers developed NEOMYCIN (Clancey and Letsinger, 1981), which attempted to explicate the types of knowledge which were implicit in MYCIN in order to improve its performance while maintaining the represented domain of infectious diseases. NEOMYCIN facilitated the explicit representation of three types of knowledge: structural, support and strategic knowledge. This marked the transition from domain-specific to task-specific KBS architectures (Clancey, 1992). However, the above observations forced its developers to re-examine the CF model, thereby identifying both theoretical and practical limitations to its general applicability.

2.3.3.4. Problems with the CF model

The rule-based approach to medical knowledge modelling had appeal as providing a general and flexible scheme for acquiring and representing expert knowledge in a declarative and modular form, conferring the ability to add or remove rules from a knowledge base without modifying other rules (Buchanan and Shortliffe, 1984). The modularity of rules in a logical production system is a consequence of the monotonicity of logic: once asserted, the truth of a proposition cannot be changed by other facts. This notion of rules as modular representation of knowledge in deterministic production systems was carried over to rule-based methods for uncertain reasoning. However, analysis of modularity has demonstrated that partial beliefs are intrinsically less modular than beliefs held with certainty, frequently making the rule-based calculi inefficient for reasoning with uncertainty (Heckerman and Horvitz, 1987; Horvitz et al, 1988; Heckerman and Shortliffe, 1992; Heckerman et al, 1992). In fact, researchers showed that the parallel and serial combination functions 2.7 and 2.8 given above, imposed assumptions of conditional independence on the propositions involved in the combinations. Moreover, the assumptions of the parallel-combination function were stronger than those of the simple-Bayes model, the same model whose limitations motivated in part the development of the CF model (Heckerman and Shortliffe, 1992).

In addition to the theoretical difficulties of updating beliefs within the CF model, the model contained some serious practical problems. Specifically, the CF model required that rules were encoded in the direction in which they were used, thereby promoting errors in assessment. Tversky and Kahneman (1982) have shown that people are most comfortable when they assess the strength of relationships in predictive rules, i.e. if cause then effect, as in the case of formal probabilistic systems, rather than in diagnostic rules, i.e. if effect then cause. The developers of the INTERNIST family of systems described below have made a similar observation (Miller et al, 1982). Indeed, the majority of medical literature describes predictive rules for a given disease, rather than diagnostic rules for a given finding.

2.3.4. Frame-Based Systems

Following the tradition AI methods for general problem-solving, goal-driven rule-based systems such as MYCIN, were designed to solve problems by generating potential solutions and testing them; that is by searching their entire rule base for applicable rules. However, by contrast to MYCIN and other systems which searched for matching solutions exhaustively, studies of clinical cognition performed at the time when the system was developed, indicated that the key element in expert performance was the ability to limit the number of active hypotheses under active consideration at any one time during the process of differential diagnosis (Kassirer and Gorry, 1978).

Overall, the exhaustive search for matching patterns of evidence, coupled with the imprecision and incompleteness of the evidence and the often occurring presence of multiple concurrent diseases with interacting mechanisms, caused the sudden proliferation of an unreasonable number of interpretative hypotheses. Two strategies were employed to control this problem in classification systems: triggering and the hierarchical organisation of hypotheses and aggregates (Patil, 1987; Szolovits et al, 1988). These aspects of problem solving competence were implicitly and poorly encoded in MYCIN's rules (Clancey, 1985) and were made explicit (Keravnou and Washbrook, 1989) in the development of the NEOMYCIN system cited above and briefly described in section 2.4.2. However, the frame-based Present Illness Program

(PIP) and the INTERNIST-1 system described below, were the first medical KBS to explore the concepts of hierarchical classification using general and explicit control heuristics.

Whereas simple classification methods were constrained to proceed hierarchically top-down or directly bottom-up from known data, in these systems triggers allowed search to non-exhaustively combine reasoning forwards from data and backwards from solutions. Thus, triggers made search opportunistic (Clancey, 1985). Data abstraction techniques were applied immediately as data became available, the abstractions triggered hypotheses, followed by a focused, hypothesis-directed search. If hypotheses were hierarchically organised, then aggregates were considered before the more specific hypotheses. New data might cause re-focusing and the cycle was repeated. Unlike MYCIN which requested information only when it was not available or could not be generated, most of the reasoning power represented in focused systems was in the information-gathering process they simulated.

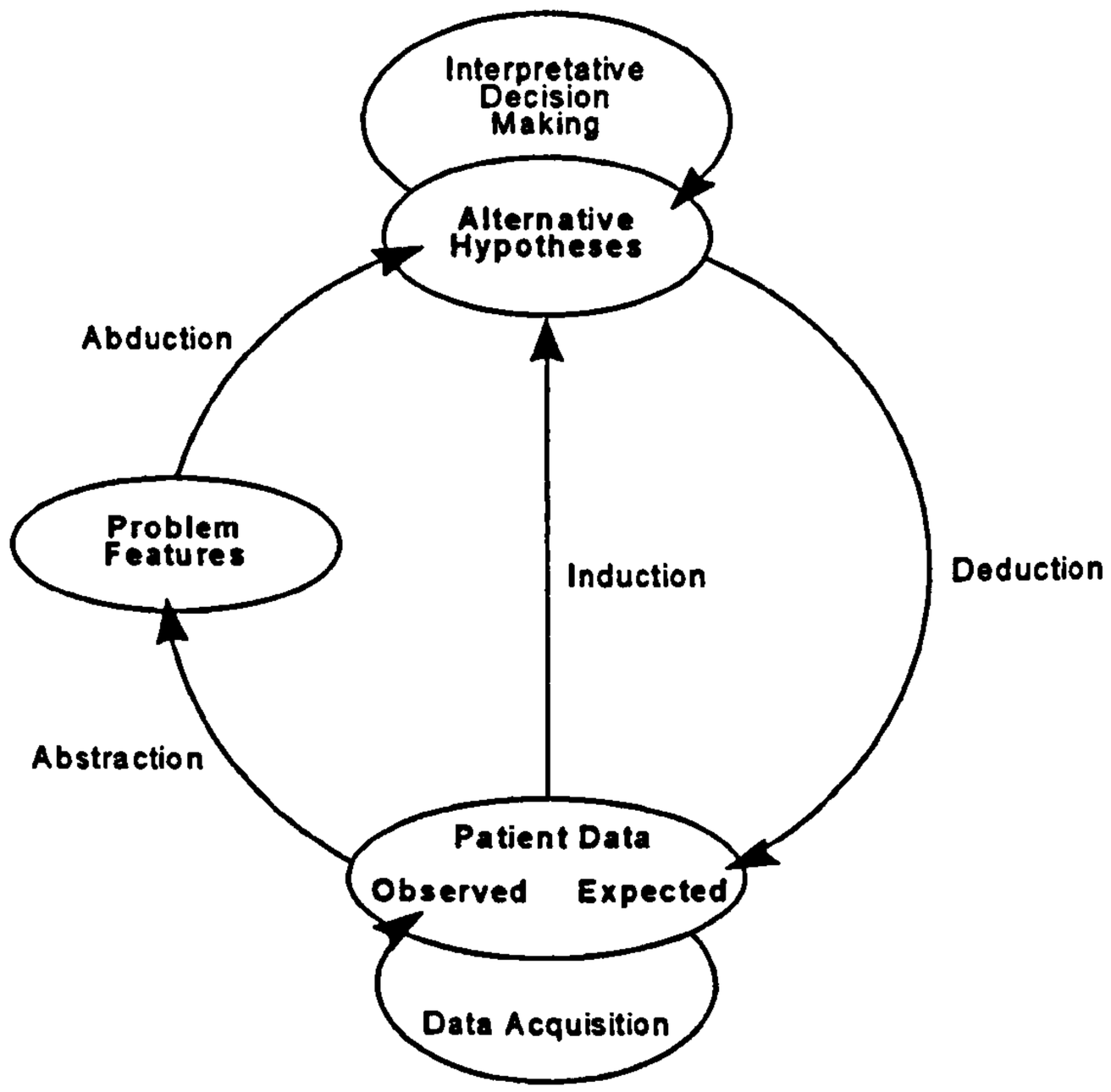


Figure 2-4. Elementary KBS problem-solving strategy (from Ramoni et al, 1992).

If more than one hypotheses were triggered, the hypotheses were rank ordered according to some numerical scoring method and the program pursued some strategy to explore and refine the list of hypotheses by asking questions and obtaining new information. The overall problem-solving cycle of hypothetico-deductive reasoning is depicted in Figure 2-4. Figure 2-5 depicts the application of the triggering heuristic combined with top-down and bottom-up search strategies to process an intermixed hierarchy of infectious diseases in NEOMYCIN.

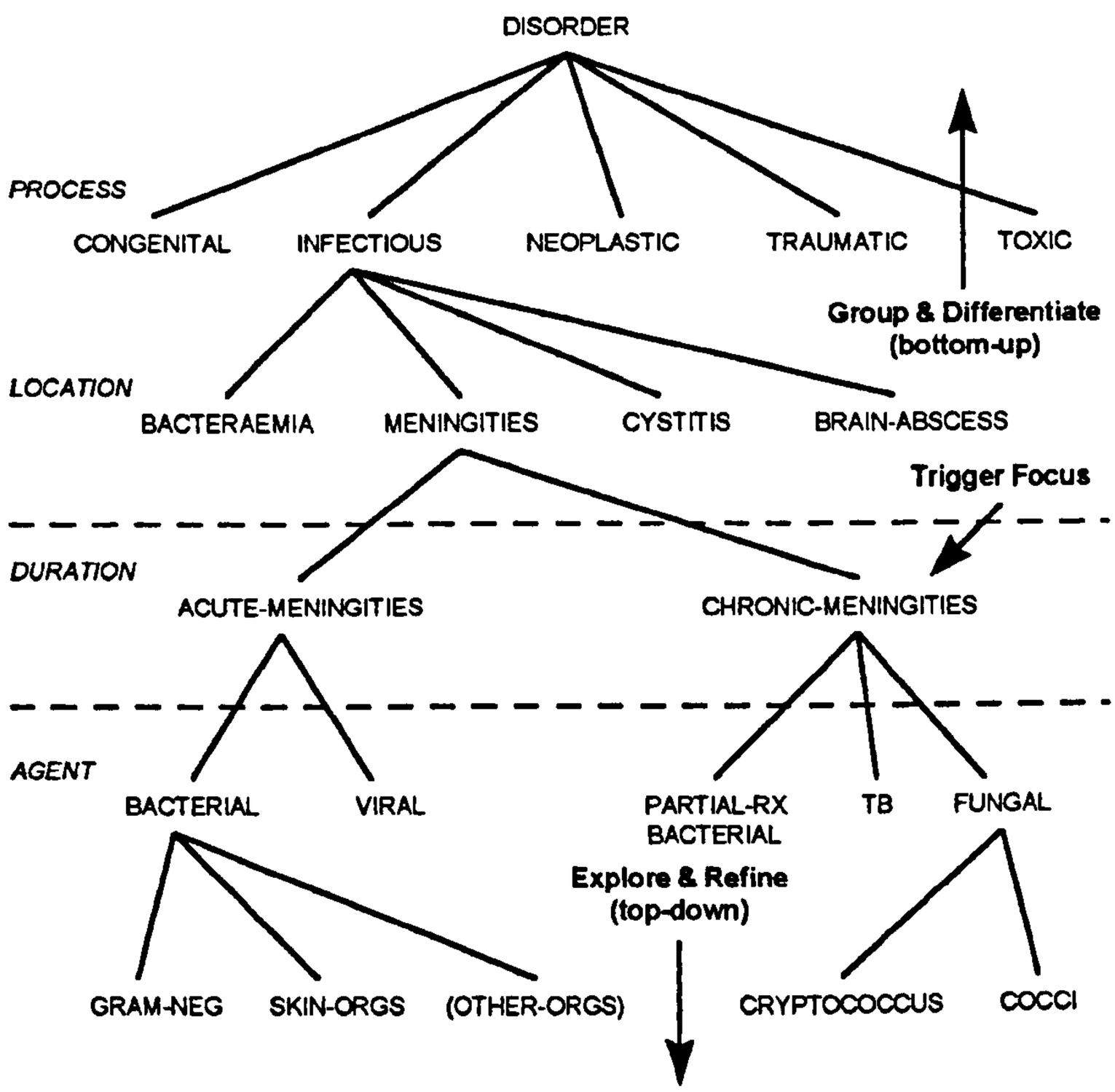


Figure 2-5. The application of search strategies in hierarchical knowledge representation structures (from Clancey, 1992)

2.3.4.1. The Present Illness Program

Figure 2-6 depicts one of the hypothesis frames that comprised PIP's knowledge base for renal disease consultation (Pauker et al, 1976). There were five classes of information within each frame of knowledge about renal diseases:

1. Relation to findings.
2. Complementary relation to other hypotheses.
3. Competing relation to other hypotheses.
4. Logical decision criteria.
5. Numerical likelihood estimation.

The system interpreted these relations to dynamically construct a patient-specific model (PSM) (Kulikowski, 1988; Clancey, 1992) of renal disease, through the instantiation of the prototypical knowledge frames that pertained to the problem solution state at any one time.

In addition to general heuristic strategies for limiting the sudden proliferation of the hypotheses generated by the available evidence, namely the two-stage limitation strategy for hypotheses generation and the principle of parsimony for problem abstraction (Pauker et al, 1976), PIP divided prototypical findings into trigger findings and non-trigger findings, to focus its reasoning. If a reported finding matched one of the triggers for a frame, the represented hypothesis was immediately brought into consideration. If a reported finding matched a non-trigger finding, its relevance to that hypothesis was only noticed if the hypothesis was already under consideration.

Interpretative hypotheses were also generated indirectly as complementary hypotheses. Complementary hypotheses identified other disorders which should be considered in addition to the hypothesis under consideration to account for the condition of the patient. This relationship might be:

1. Associative, if the two were related by some empirical association.
2. Complicational, if one disorder was a typical complication of the other.
3. Causal, if the physiology of the represented disorder was well understood.

NAME:	**nephrotic syndrome**
IS A TYPE OF:	**clinical state**

relation to findings

FINDINGS:

low serum albumin concentration
heavy proteinuria
> 5 gm/24Hrs proteinuria
massive, symmetrical edema
EITHER *facial* OR *peri-orbital*, AND *symmetrical* edema
high serum cholesterol concentration
urine lipids *present*

logical decision criteria

MUST NOT HAVE: proteinuria *absent*

IS SUFFICIENT: BOTH *massive* edema AND > 5gm/24Hrs proteinuria

numerical likelihood estimator

MAJOR SCORING:

serum albumin concentration:
- low: 1.0
- high: -1.0

proteinuria:
- > 5 gm/24Hrs: 1.0
- heavy: 0.5
- EITHER absent OR light: -1.0

edema:
- massive AND symmetrical: 1.0
- NOT massive BUT symmetrical: 0.3
- erythematous: -0.2
- asymmetrical: -0.5
- absent: -1.0

MINOR SCORING:

serum cholesterol concentration
- high: 1.0
- NOT high: -1.0

urine lipids:
- present: 1.0
- absent: -0.5

complementary relation to other hypotheses

MAY BE CAUSED BY:
acute glomerulonephritis,
chronic glomerulonephritis,
nephrotoxic drugs,
insect bite,
idiopathic nephrotic syndrome,
systemic lupus erythematosus, or diabetes mellitus

MAY BE COMPLICATED BY:
hypovolemia
cellulitis

MAY BE CAUSE OF: sodium retention

competing relation to other hypotheses

DIFFERENTIAL DIAGNOSIS:
IF neck veins elevated,
CONSIDER: constrictive pericarditis
IF ascites present,
CONSIDER: cirrhosis
IF pulmonary emboli present,
CONSIDER: renal vein thrombosis

Figure 2-6. Frame-based knowledge representation in the Present Illness Program (from Pauker et al, 1976).

All non-complementary hypotheses were considered competitors, however, a differential-diagnosis relationship to other frames was specified for evaluating such a condition. Complementary and competing relations to other hypotheses were also used in controlling the activation of hypotheses.

2.3.4.2. Interpretative Decision-Making in PIP

During the consultation, the interpretative hypotheses generated by PIP were evaluated to determine the extent to which they constituted reasonable explanations for the patient's clinical presentation as suggested by the findings, and to eventually make a decision on the most accurate hypothesis.

There were two aspects to the hypothesis testing task. First, an evaluation of the goodness of fit of the observed findings to the expectations of each generated hypothesis, facilitated decision making for the acceptance or rejection of an active frame on the basis of the available information, or the choice to obtain further information, based on the likelihood of the leading hypothesis. Second, each hypothesis was examined to determine the extent to which it accounted for all the facts in the case. The first estimate was represented in a matching score, whereas the second in a binding score (Szolovits and Pauker, 1978).

By contrast to the activation or triggering of hypotheses, which was purely categorical, once a hypothesis was under consideration, both categorical and evidential mechanisms existed to test the hypothesis. Logical decision criteria were used by the program to make categorical decisions about the likelihood of a hypothesis under consideration at any one time. Is-sufficient, covered the case of pathognomonic findings, in which the presence of a single finding was in itself sufficient to confirm the presence of the hypothesised disorder. Logical combinations were used to specify more complex criteria. Must-have and must-not-have specified necessary conditions, in the absence of which a hypothesis would not be accepted as confirmed. In the absence of information asserting the categorical validity of an hypothesis, PIP estimated the likelihood of each hypothesis by combining the matching and the binding scores.

The matching score was estimated in two parts. First, a local scoring process estimated the numerical likelihood of each hypothesis in terms of the major and minor features of the disorder. The local score of an hypothesis reflected the degree to which the observed evidence supported the hypothesis directly. Following that, PIP computed the overall matching score by revising local scores to include the effects of propagated information deriving from related hypotheses as described in the previous section. The local and updated scores were computed by adding or subtracting the contribution of each finding to the hypothesis and normalising by the maximum possible total score. If the score for an hypothesis exceeded a defined upper threshold (Pauker and Kassirer, 1980), the frame was accepted. If the score fell below a defined lower threshold, the hypothesis was forced into a semi-active state.

The evaluation of the ability of an hypothesis to account for the findings of the case, expressed in terms of the binding score, comprised the following process. The program computed for each frame in its knowledge base a value equal to the fraction of all findings in the patient profile which were explained by the hypothesis. This value and the matching score were averaged, and the hypotheses were assigned a rank order based on the average.

2.3.4.3. PIP's Performance Evaluation

Albeit its complex, combined deterministic and evidential interpretative decision making mechanism, PIP failed to achieve any level of competence in solving clinical problems. Too often, small variations in a borderline clinical case pushed the program's scoring just above or just below a threshold and affected its conclusions significantly. Furthermore, the program's questioning focus frequently shifted, as the scoring mechanism brought one and then another competing hypothesis to the top of its ranking list (Szolovits and Pauker, 1978).

Blois (1980) analysed the task undertaken by PIP, the INTERNIST-1 system described in the section that follows, and other similar programs, and concluded that it would be unlikely to deliver a computer program that emulates human expert behaviour in taking the history of a disease. However, the biggest limitation of the

Present Illness Program was that of any system that proceeded it. All disease entities were assumed to be competitors, i.e. mutually exclusive and collectively exhaustive, and as a consequence its performance degraded significantly when more than two diseases coexisted (Schwartz et al, 1987).

2.3.4.4. The INTERNIST-1 System

INTERNIST-1 was the first prototype system to be developed as part of a long standing project cited in section 1.1, which was initiated in the early 1970s at the University of Pittsburgh in the USA, with the ambitious goal (Blois, 1980) of encoding the whole of internal medicine in a frame-based KBS to support clinical decisions.

The INTERNIST-1 knowledge base, which by 1984 covered 75% of internal medicine, by containing 600 disease profiles, 4000 independent manifestations of disease, and a number of conditional dependencies between manifestations, (Miller, 1984), was organised in a tree-structured hierarchy of disease profiles, maintained as frames. General descriptions of diseases, or aggregates, were maintained toward the top of the hierarchy, or toward the root of the problem taxonomy, whereas refined descriptions of diseases were obtained by traversing the hierarchy towards the leafs, or the ultimate aetiology. Figure 2-7 depicts a section of the top of the hierarchy.

Initially, it was attempted to represent all diseases in a hierarchy of mutually exclusive and collectively exhaustive sets of diagnostic hypotheses which would be formulated on the basis of probabilistic evidence. This simple hierarchy could be used to implicitly represent the reasoning processes involved in the diagnostic task, but simple hierarchical classification was found to be impossible due to the vast amount of general medical knowledge the system was built to represent, and the fact that so much knowledge exposed a lack of organisation that generally characterises medicine. Instead heuristic reasoning programs were built to create mutually exclusive and collectively exhaustive sets of differential diagnoses, from sets of findings assembled into independent problem areas (Miller et al, 1982). Thus, a problem area was defined as a selected group of observed findings, the differential diagnosis of which formed

what was assumed to be a mutually exclusive and exhaustive set of diagnoses. By constructing specific differential diagnoses to address identified problem areas, INTERNIST-1 could narrow the set of possible diagnoses from all known diseases to well-defined collections of competing diagnoses in a small number of categories. Heuristic principles would then be applied to resolve each differential diagnosis. Thus in a sense, problem areas were used to construct virtual hierarchies of mutually exclusive and collectively exhaustive diagnoses which could then be refined by evaluating the actual independent hierarchies using data of a probabilistic nature.

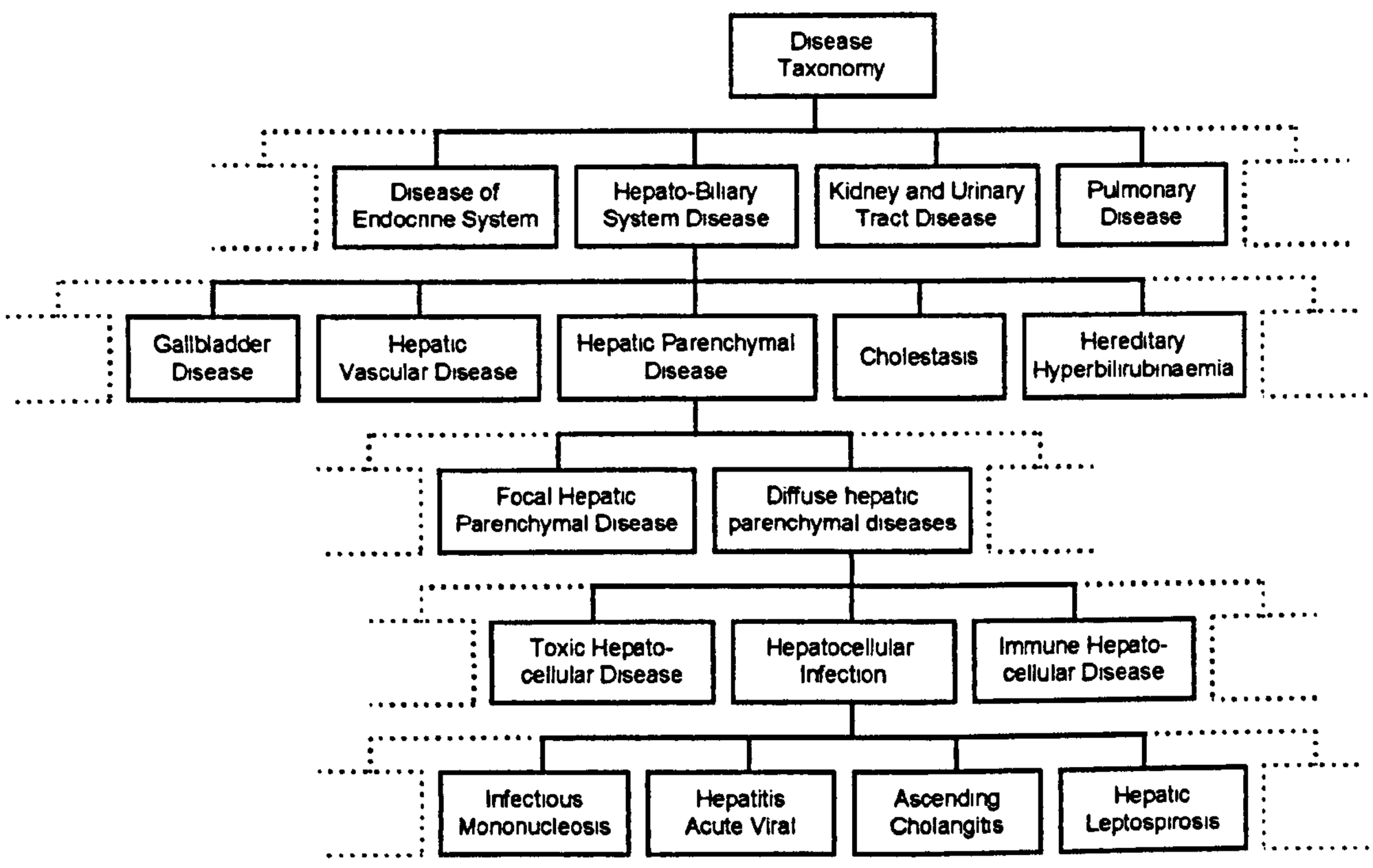


Figure 2-7. Organisation of disease profiles represented in frames which form a hierarchy of disease taxonomy (from Miller et al, 1982 and Miller et al, 1986).

DISPLAY WHICH MANIFESTATION LIST?
ALCOHOLIC HEPATITIS

AGE 16 TO 25 ... 0 1
AGE 26 TO 55 ... 0 3
AGE GTR THAN 55 ... 0 2
ALCOHOL INGESTION RECENT HX ... 2 4
ALCOHOLISM CHRONIC HX ... 2 4
SEX FEMALE ... 0 2
SEX MALE ... 0 4
URINE DARK HX ... 1 3
WEIGHT LOSS GTR THAN 10 PERCENT ... 0 3
ABDOMEN PAIN ACUTE ... 1 2
ABDOMEN PAIN COLICKY ... 1 1
ABDOMEN PAIN EPIGASTRIUM ... 1
ABDOMEN PAIN NON COLICKY ... 1 2
ABDOMEN PAIN RIGHT UPPER QUADRANT ... 1 3
ANOREXIA ... 0 4
DIARRHEA ACUTE ... 1 2
MYALGIA ... 0 3
VOMITING RECENT ... 0 4
ABDOMEN BRUIT CONTINUOUS RIGHT UPPER QUADRANT ... 1 2
...
...
BILIRUBIN BLOOD CONJUGATED INCREASED ... 2 4
BILIRUBIN URINE PRESENT ... 2 4
CHOLESTEROL BLOOD DECREASED ... 2 2
CHOLESTEROL BLOOD INCREASED ... 1 2
HAEMATOCRIT BLOOD LESS THAN 35 ... 1 3
HEMOGLOBIN BLOOD LESS THAN 12 ... 1 3
KETONURIA ... 1 2
PROTEINURIA ... 1 2
SGOT 120 TO 400 ... 2 3
...
...
LIVER BIOPSY BILE PLUGGING ... 1 2
LIVER BIOPSY FATTY METAMORPHOSIS ... 2
LIVER BIOPSY FOCAL NECROSIS AND INFLAMMATION ... 2
LIVER BIOPSY HEPATOCELLULAR NECROSIS MARKED ... 2
LIVER BIOPSY PERIPORTAL FIBROSIS MILD ... 1
...
...
LINKS FOR ALCOHOLIC HEPATITIS:

Predisposes to	MALLORY WEISS SYNDROME ... 1
Causes	SINUSOIDAL OR POSTSINUSOIDAL PORTAL HYPERTENSION ... 1 2
Causes	HEPATIC ENCEPHALOPATHY ... 2 2
Causes	RENAL FAILURE SECONDARY TO LIVER DISEASE <HEPATORENAL SYNDROME> ... 2 2
Coincident with	PANCREATITIS ACUTE ... 2 2
Precedes	MICRONODAL CIRRHOSIS <LAENNECS> ... 2 3

Figure 2-8. Excerpt from a sample manifestations list maintained by INTERNIST-1 (From Miller et al, 1982). The first number after each manifestation is its evoking strength for the diagnosis; the second is the frequency of the manifestation in the disease.

The main information stored in the INTERNIST-1 knowledge frames, an example being shown in Figure 2-8, was a list of manifestations expected to appear in a patient with a given disease. Each manifestation was accompanied by two clinical parameters: the evoking strength and the frequency of manifestation. The evoking strength of a manifestation was meant to be analogous to the predictive value of that manifestation for the particular disease, whereas the frequency of a finding in a particular disease was analogous to the sensitivity of that finding for the particular disease (Bankowitz et al, 1989). In probabilistic terms, these two parameters correspond to the a posteriori probability of a disease given the finding has been observed and the probability of the finding being observed in the disease. The latter measure has two interpretations in probability theory, namely the frequency of manifestation and personal belief, and is the source of dispute concerning its proper use (Feinstein, 1977; Cheeseman, 1985; Horvitz et al, 1988; Aliferis and Miller, 1995). Their particular interpretation and use for interpretative decision making in the INTERNIST-1 system is described in the section that follows and further discussed in Section 2.7.

2.3.4.5. Interpretative Decision Making in INTERNIST-1

At the beginning of the consultation, the user would supply the system with a list of manifestations present in the patient and findings denied. The following steps were then taken by the program to provide a differential diagnosis of the supplied findings. As each positive manifestation was encountered, the program retrieved its complete differential diagnosis list from the inverted disease profiles stored in the knowledge base and added it to a master differential diagnosis list. Higher-level concepts from the classification hierarchy were retained on the list as long as the diagnoses they subsumed were indistinguishable in their ability to explain the observations. The manifestation-based differential diagnosis lists obtained in this manner were part of the solution state of the diagnostic problem (PSM), and as such were also maintained in the knowledge base to provide information for subsequent stages in the diagnostic process.

For each disease hypothesis in the master list, four further lists were maintained:

1. All positive manifestations in the patient that were explained by the disease hypothesis.

2. All manifestations that might occur in a patient with the disease but were known to be absent, or were findings denied, in the patient considered.

3. All manifestations that were present in the patient but were not explained by the disease hypothesis, possibly because there was a second underlying disorder.

4. All manifestations that were on the disease's profile but about which nothing was known and should therefore be asked.

During a diagnostic consultation, the program reasoned about a given diagnostic problem in two stages: problem-space formulation and problem-space structuring and resolution (Pople, 1982). Following the initial formulation of the problem space (PSM) in the form of the lists described above, INTERNIST-1 proceeded by evaluating the constituent hypotheses. Each hypothesis on the master list of diagnoses was given a score which was the sum of a positive and a negative component, as well a bonus score. The positive component included the weights of all manifestations that were explained by the hypothesis (List 1), based on the evoking strengths listed in Table 2-1. The negative component included the weight of all manifestations that were expected to occur in patients with the disease but were absent in the patient under consideration (List 2). This component was based on the frequency of manifestation classified in Table 2-2. Also included were the weights of all manifestations present in the patient but not explained by the hypothesised diagnosis (List 3). This was based on the clinical importance or import of each manifestation. Table 2-3 lists the interpretations of the import variable. A threshold value was used to decide which of the hypotheses to pursue further or to temporarily discard.

Evoking Strength	Interpretation
	0 Non-specific association. Manifestation occurs too commonly to be used to construct a differential diagnosis.
	1 Rare association. Diagnosis is a rare or unusual cause of listed manifestation.
	2 Non-dominant cause. Diagnosis causes a substantial minority of instances of listed manifestation.
	3 Equivocal cause. Diagnosis is the most common but not the overwhelming cause of listed manifestation.
	4 Dominant cause. Diagnosis is the overwhelming cause of listed manifestation.
	5 Pathognomonic association Listed manifestation is pathognomonic for the diagnosis.

Table 2-1. Interpretation of the Evoking Strength clinical parameter quantifying the relationship between a finding and a disease in INTERNIST-1 (from Miller et al, 1982 and Miller et al, 1986).

Frequency of Manifestation Estimate	Interpretation
	1 Manifestation occurs rarely in the disease
	2 Manifestation occurs in a substantial minority of cases of the disease.
	3 Manifestation occurs in roughly half the cases.
	4 Manifestation occurs in the substantial majority of cases.
	5 Manifestation occurs in essentially all cases and is therefore a prerequisite for the diagnosis.

Table 2-2. Interpretation of the Frequency of Manifestation clinical parameter quantifying the relationship between a finding and a disease in INTERNIST-1 (from Miller et al, 1982 and Miller et al, 1986).

Following the evaluation of the initially generated hypotheses, problem area construction was carried out by a simple partitioning heuristic rule. If any two diseases taken together could explain no more observations than either one did alone, the diseases were classified as competitors. The problem area at any one time was formed by the top-most ranking diagnosis and the competitors. Once the problem area containing the most attractive diagnosis was formed and selected, criteria for establishing a conclusive diagnosis were applied. Failing that, the program selected one of three questioning strategies to obtain the information required to further

explore the problem. Three heuristics were available for such cases: pursue, rule-out and discriminate. Finally, when a conclusive diagnosis was possible, all observed manifestations explained by the diagnosis were removed from future consideration and the cycle repeated using the remaining unexplained findings. Unexplained findings were assumed to be due to misinterpretation, or a concurrent disease. If not, then the program listed the top ranking hypotheses based on a threshold value.

Import	Interpretation
1	Manifestation is usually unimportant, occurs commonly in normal persons, and is easily disregarded.
2	Manifestation may be of importance, but can often be ignored; context is important,
3	Manifestation is of medium importance, but may be an unreliable indicator of any specific disease.
4	Manifestation is of high importance and can only rarely be disregarded as, for example, a false-positive result.
5	Manifestation absolutely must be explained by one of the final diagnoses

Table 2-3. Interpretation of the Import clinical parameter quantifying the relationship between a finding and a disease in INTERNIST-1 (from Miller et al, 1982).

2.3.4.6. Performance Evaluation of the INTERNIST-1 System

Following the validation of the INTERNIST-1 knowledge base, the system was retrospectively evaluated using data from clinicopathological conferences published in the New England Journal of Medicine as case records of the Massachusetts General Hospital. The INTERNIST-1 knowledge base was not altered during the course of this trial. The results of the evaluation indicated that the accuracy of the system was comparable to that of the clinicians caring for the patients, but not as good as that of the invited expert case discussants (Miller et al, 1982; Bankowitz et al, 1989). The developers also noted that because INTERNIST-1 defined problem areas in the above ad hoc manner, its differential diagnoses did not always resemble those constructed by clinicians. Furthermore, they observed that although the system included an ad hoc scoring scheme for evidential reasoning, the program's problem solving behaviour

resulted primarily from the application of the two heuristic principles of the formation of problem areas through the partitioning algorithm and the conclusion of diagnoses within problem areas, using strategies such as diagnosis by exclusion (Szolovits and Pauker, 1978; Miller et al, 1982). In fact, in a study of the INTERNIST-1 and PIP diagnostic algorithms at MIT, the partitioning heuristic was found to be a key to INTERNIST-1's superior performance over the Present Illness Program (Sherman, 1981). However, although designed to overcome the problem of multiple coexisting diseases with interacting mechanisms, the partitioning heuristic failed to deal adequately with the problem (Patil, 1987).

2.3.5. Summary and Conclusion

Summarising the above, classification KBS solve the problem of recognising an unknown clinical problem observed in a patient, by classifying the patient as a member of one or more known classes of patients. These classes can be thought of as disease stereotypes which are hierarchically organised, and the process of identification is one of matching features of an observed clinical problem against those of known classes of patients which are explicitly represented in a knowledge base. Thus, classification KBS essentially select the problem solution that best matches the data from a set of pre-enumerated and explicitly represented solutions (Clancey, 1985).

Since by definition both the knowledge as well as the evidential information employed in classification reasoning are imperfect and incomplete, the real-life to expected or represented case pattern matches are only partial, resulting in a list of uncertain interpretative hypotheses. Given that the method employed for the acquisition, representation and manipulation of the inherent interpretative uncertainty produces accurate results, it should suffice to select the top-most ranking hypothesis. However, since there may be more than one diseases present in the patient, and since these diseases may interact in their presentation, researchers attempted to develop ad hoc heuristic models of differential diagnosis, which combined deterministic and evidential reasoning strategies, in guiding the system to the acquisition of the information

required to refine the problem solution by reducing the list of interpretative hypotheses based on the likelihood of the top-most ranking hypothesis.

As discussed above, these strategies led to problem solving behaviours that were too unfocused and consequentially erratic. The triggering heuristic was found useful for limiting hypothesis activation in abductive reasoning, especially when combined with the hierarchical organisation of hypotheses. Nevertheless, even with a cluster size of two or three ordered findings, simple triggering often led to the generation of an unmanageably large set of hypotheses in broad domains (Sherman, 1981). In some cases, the resultant interpretative problem solving behaviour was a sole consequence of the heuristically defined measures of uncertainty they employed, while the underlying formal AI methodologies provided some form of program organisation. In other cases, the ad hoc methods for inexact reasoning were passive observers and the categorical behaviour that resulted was the sole consequence of the logical structure of the formal AI methods employed (Szolovits and Pauker, 1978).

In theory these hybrid models were designed to function as modular belief updating schemes, thereby facilitating the construction and maintenance of a knowledge base. However, it was shown that in reality they did not satisfy the property of modularity, that is the assumption that the observation of a new piece of evidence bearing on some hypothesis under consideration did not depend on previous observations. This is because in general, logical relationships represent complete models of interaction between knowledge modules. In contrast, uncertain relationships encoded invisible interactions which were summarised with numerical measures such as the certainty factors or evoking strengths. In the process of such summarisation, information was lost about the detailed categorical interaction. Therefore, when uncertain information was combined, unexpected non-modular interactions occurred (Heckerman et al, 1992; Aliferis and Miller, 1995). Furthermore, most of the programs discussed above focused their efforts on identifying the single most likely diagnosis that explained the majority of observations. Only after the first diagnosis was confirmed did they attempt to make a second diagnosis based on any residual findings, and the process repeated until either all findings were exhausted or the user explicitly terminated the diagnostic process. As a consequence, these systems failed to recognise diseases with interacting pathological mechanisms or diseases whose presentation overlapped. These problems

were addressed further by means of the method of causal simulation reasoning which is described below.

2.4. Causal Simulation Systems

2.4.1. Introduction

In order to solve the problem of recognising the presence of multiple coexisting diseases with interacting mechanisms, KBS developers pursued the use of improved structural rather than numerical methods for reasoning with partial belief and incomplete information. The new improved knowledge representations invariably exploited causality as causal models generally require a reduced number of the often erroneous assumptions encountered in classification reasoning as described above (Kuipers and Kassirer, 1984; Aliferis and Miller, 1995).

In the Present Illness Program causality was used both in the diagnostic direction as well as in the predictive inference direction. In the diagnostic or effect-to-cause direction, PIP reasoned causally for the instantiation of complementary hypotheses which identified other disorders that should be considered in addition to the hypothesis under consideration to account for the condition of the patient. Furthermore, this relationship was causal if the physiology of the represented disorder was well understood, complicational if one disorder was a typical complication of the other, or associative if the two were related by some empirical association. In the predictive or cause-to-effect direction, causality was exploited for hypothesis testing. Similarly, in the INTERNIST-1 system causal links represented non-hierarchical associations between disease frames which were quantified by an evoking strength and a frequency measure. However, the explicit use of causality as a pivotal mechanism in interpretative decision making was first explored in the systems described below.

2.4.2. CASNET

CASNET (Weiss et al, 1978; Kulikowski and Weiss, 1982), which was designed to assist in the management of the eye disease glaucoma, reasoned about an observed clinical problem using a variant of the heuristic classification method summarised in Figure 2-2, termed causal-process classification (Clancey, 1985). The variation in causal-process classification, which is summarised in Figure 2-9, is that patient data are linked to solution classes via dysfunctional processes which cause the observed disease manifestations. Thus, building on the method of simple hierarchical and heuristic classification, systems employing causal-process classification can support hypothetical solutions by an external process involving interlinked dysfunctional states. In this case, the inferred causal model of the disease, or the diagnosis, is said to explain the findings. Figure 2-10 depicts the use of causal-process classification to support NEOMYCIN's inference structure. Figure 2-11 depicts the use of causal links to support and explain the classification process in CASNET.

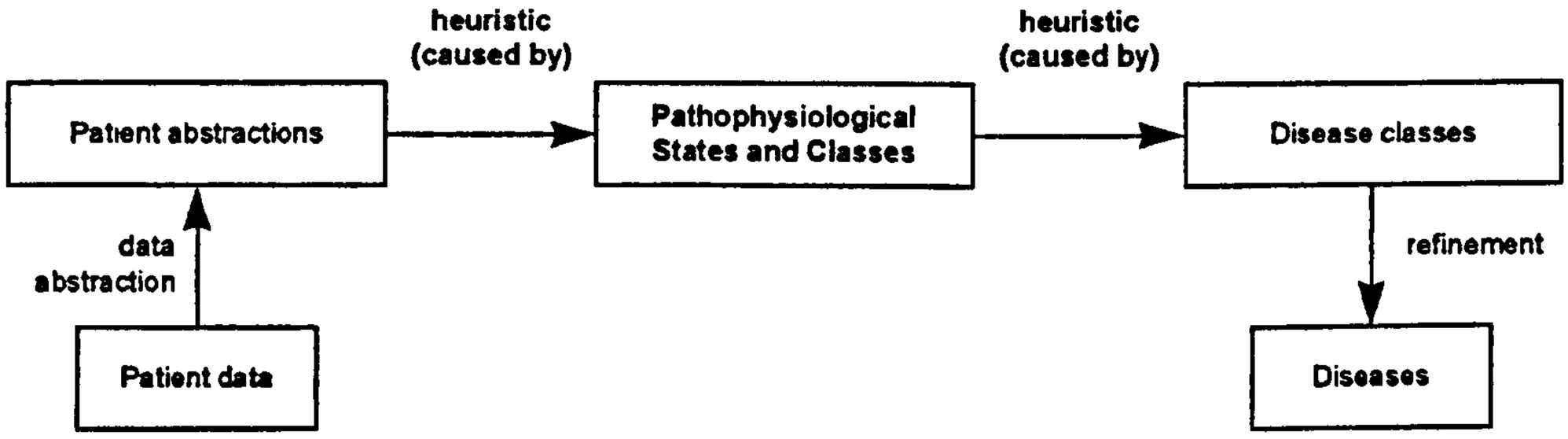


Figure 2-9. The inference structure of causal-process classification (from Clancey, 1985).

As in the case of heuristic classification systems, problem solutions were pre-enumerated, however, by contrast to other classification systems, the paths to them were constructed rather than selected. Since multiple causal explanations could be constructed for a given set of symptoms, an inference strategy was required that did not realise every possible association but reasoned about alternative chains of inference. Even though diagnostic solutions were pre-enumerated by definition, assertions could in principle be taken back, so reasoning was non-monotonic.

However, the most well-known programs that solved diagnostic problems by causal-process classification were monotonic, dealing with alternative lines of reasoning by assigning weights to the paths. Indeed, many programs did not even compare alternative explanations, but simply listed all solutions, rank-ordered (Clancey, 1985).

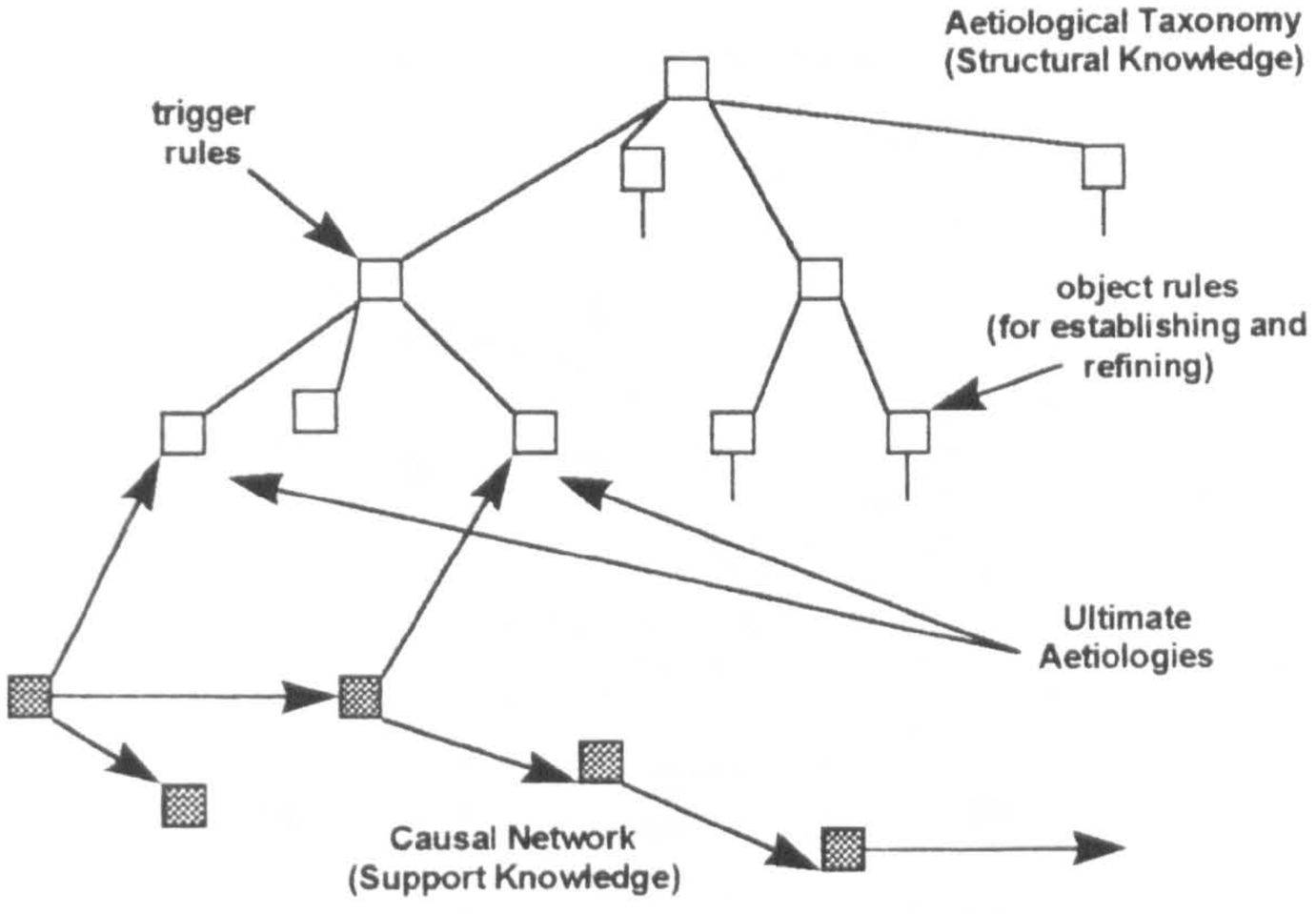


Figure 2-10. Representation of domain knowledge in NEOMYCIN (from Keravnou and Washbrook, 1989).

CASNET generated its causal pathways by calculating a likelihood value for each node in each plausible pathway. This likelihood value was a function of the confidence of an observation and of a heuristically derived weight for the associated causal link. In this way, CASNET had compiled into its knowledge base a set of disease indices that steered the causal pathway generator to the most probable causal links. The program used threshold values to determine whether a particular node's status was confirmed, denied, or uncertain. Again, the set of confirmed dysfunctional nodes in CASNET was viewed as an explanation of the patient's pathophysiology. In addition, CASNET took into consideration the temporal progression of an illness and used the notion of causal consistency in its diagnostic reasoning. However, because of CASNET's limited mechanism for manipulating causal relations, the program did not deal with multiple co-occurring disorders (Uckun, 1992).

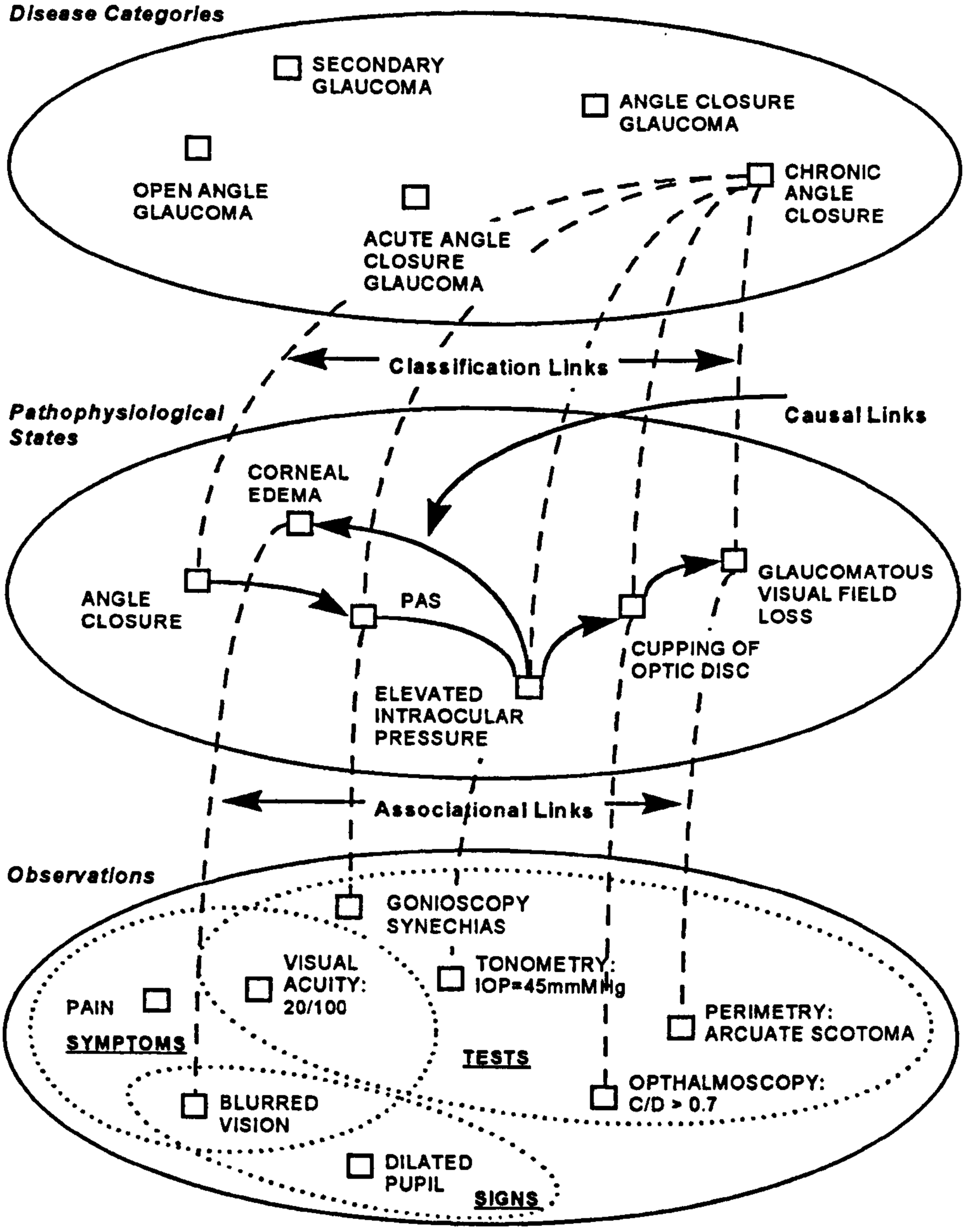


Figure 2-11. CASNET's use of causal relations in supporting classification reasoning (adapted from Weiss et al, 1978).

2.4.3. CADUCEUS

To improve the performance of INTERNIST-1, its designers explored new ways of reasoning about the causal relations already present in its knowledge base. This study led to the experimental implementation of a second prototype called INTERNIST-II

(Pople, 1977). The new prototype attempted to group the evoked hypotheses into causally related clusters of complementary hypotheses. However, INTERNIST-1, NEOMYCIN and the MDX hierarchical classification system (Chandrasekaran and Mittal, 1983) employed intermixed hierarchies for domain knowledge representation (eg. Figure 2-5); each level was organised around a different disease attribute such as duration, anatomical site, aetiology, pathophysiology, severity, and clinical presentation. It soon became apparent that processing the large causal network required to model a broad domain in such a manner using an intermixed hierarchy was computationally intractable without more sophisticated representational support and search strategies. To overcome the limitations of intermixed hierarchies (Patil, 1987; Szolovits et al, 1988), research emphasis was placed on the development of strategies for reasoning with multiple pure hierarchies, each of which facilitated problem analysis along one of the above orthogonal dimensions of a pathological process.

A substantially more sophisticated use of causal relations in forming composite hypotheses from individual hierarchies was proposed by Pople in the development of CADUCEUS (Pople, 1982). CADUCEUS used a number of hierarchies each organised around a different concept and which, unlike the INTERNIST systems, had the form of lattice structures, allowing each disease in the hierarchy to have more than one parent thereby permitting the representation of multi-system diseases. Superimposed on these taxonomic hierarchies were a number of different types of links that related nodes within each hierarchy and across different hierarchies. These links formed the basis for the synthesis operators, which constructed composite hypotheses, or causally connected graphs, through an appropriate choice of level of disease abstraction and causal constraints. Each such composite hypothesis consisted of a collection of diseases, clinical states, and causal relations that attempted to provide a causal explanation of the observed findings (PSM). The synthesis operators employed abductive reasoning to support existing hypotheses by extending them causally toward their ultimate aetiologies, while attempting to refine a composite hypothesis taxonomically, moving from general to more specific disease descriptions. This process is depicted in Figure 2-12.

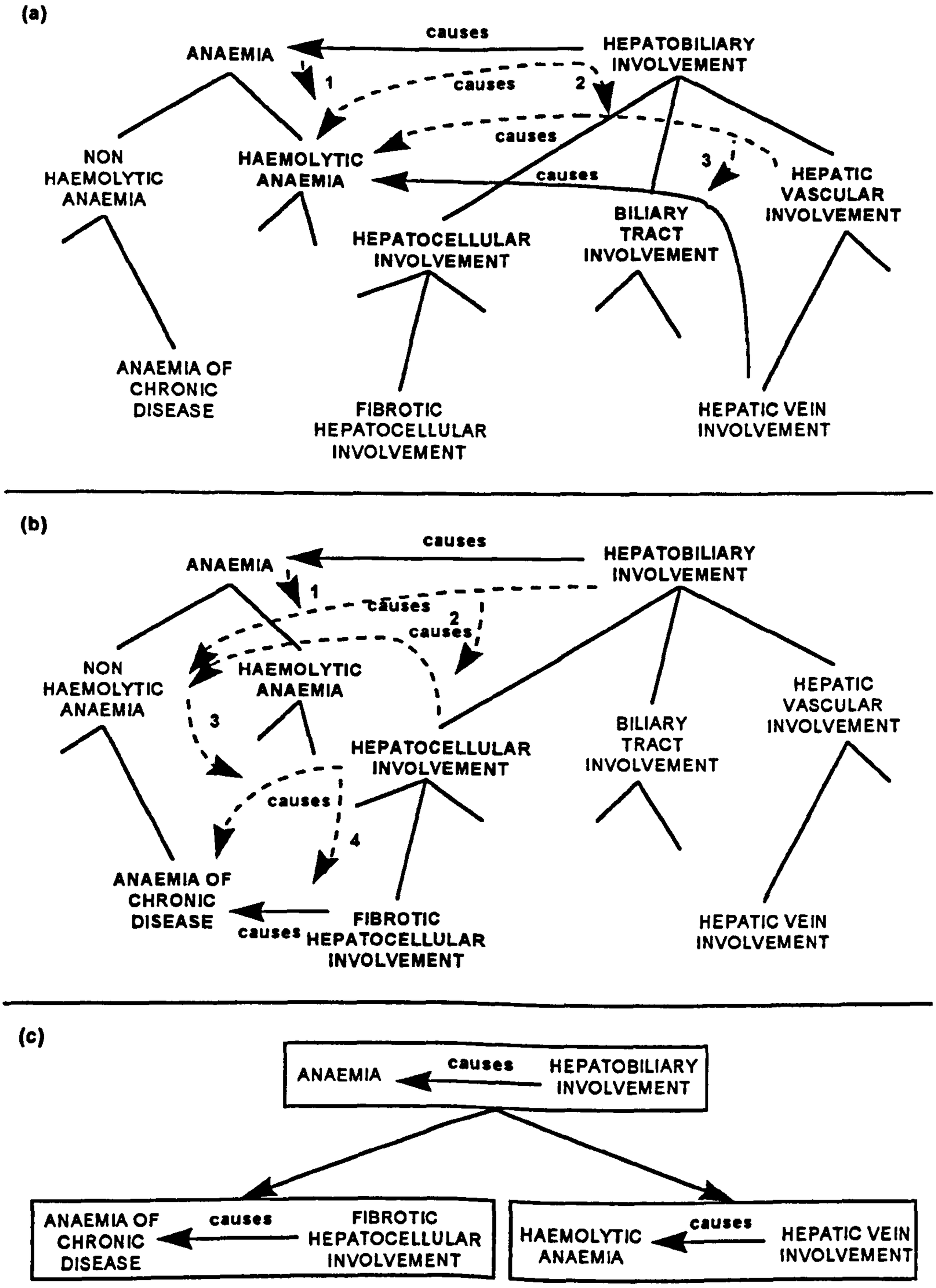


Figure 2-12. Two possible refinements of the causal relation between anaemia and hepatobiliary disease in CADUCEUS (from Patil, 1987). Numbers indicate the sequence of refinements (a and b). The resultant composite hypotheses are depicted in the lower section (c).

The following types of links were used in CADUCEUS for the exploration of composite hypotheses (Patil and Senyk, 1988):

1. Causal links were used to represent pathophysiological relations among nodes in the taxonomic hierarchies. One of the major uses of causal links was for forming aggregates in a differential diagnosis.

2. Planning links represented high-level causal relations. Planning links were used to summarise other planning links and chains of causal links. Unlike causal links, planning links represented possible causal relations. That is they were existentially quantified over the nodes in the hierarchy. The primary purpose of the planning links was the formation of causal hypotheses at an abstract level. These were subsequently refined in the process described above.

3. Spanning links were used in a manner similar to ABEL's composite links described below, to summarise chains of causal reasoning. Spanning links allowed the synthesis operators to limit the scope of causal search and to introduce additional hypothesised clinical states into a composite hypothesis.

4. Constrictor relations stood for strong or pathognomonic associations between observable manifestations and pathophysiological states. The use of such relations in CADUCEUS is the same as any triggering relation.

Although the composite hypotheses in CASNET and CADUCEUS were similar in form, the fundamental difference between the two programs is that CASNET used a static causal model to select alternative, weighted causal process paths. In other words CASNET was a classification or analytic system. By contrast, CADUCEUS was capable of hypothesising the presence or absence of a causal relation between any two states and as a result construct or synthesise alternate composite hypotheses providing different causal interpretations of the same set of pathophysiological states. However, in the final analysis, although both programs facilitated the recognition of multiple simultaneous diseases, neither of the programs addressed the problem of interactions among the diseases (Patil, 1987; Patil and Senyk, 1988). This problem was addressed for the first time in the development of ABEL, described below.

2.4.4. ABEL

The ABEL system (Patil et al, 1981, 1982) attempted to solve the problem of co-occurring diseases with interacting mechanisms by reasoning about clinical scenarios that arise from the disease interactions. Instead of pursuing each hypothesis separately, ABEL collected them into problem sets, and then applied planning techniques to extract a proposed sequence of information queries that optimally and systematically reduced uncertainty in each problem area. Problem solving plans were generated in a manner similar to NEOMYCIN, by decomposing the top-level information-gathering goal within each problem area into subgoals in the context of the evolving PSM.

Medical knowledge in ABEL consisted of hierarchical representations of anatomical, physiological, aetiological, and temporal knowledge about disorders of acid-base balance. The program described its knowledge of pathophysiology in terms of clinical states and causal relations at five different levels of detail, the most significant of which are shown in Figure 2-13. The most detailed level dealt explicitly with stores of electrolytes in various body compartments and with their movement from one compartment to another. Each state contained a number of attributes such as severity and duration. Each link describing a possible causal relation between two states also specified a set of constraints between the attributes of the cause and effect nodes. Finally each causal relation at a given level of detail was described at the next, more detailed level using a causal network to elaborate its underlying mechanisms.

Composite hypotheses in ABEL were described by a set of PSMs that attempted to explain all the facts known about the patient. Each of these PSMs was itself a multilevel structure which contained descriptions of the same diagnostic hypothesis at levels varying from a clinical summary to the detailed pathophysiological explanation of the patient's disorders.

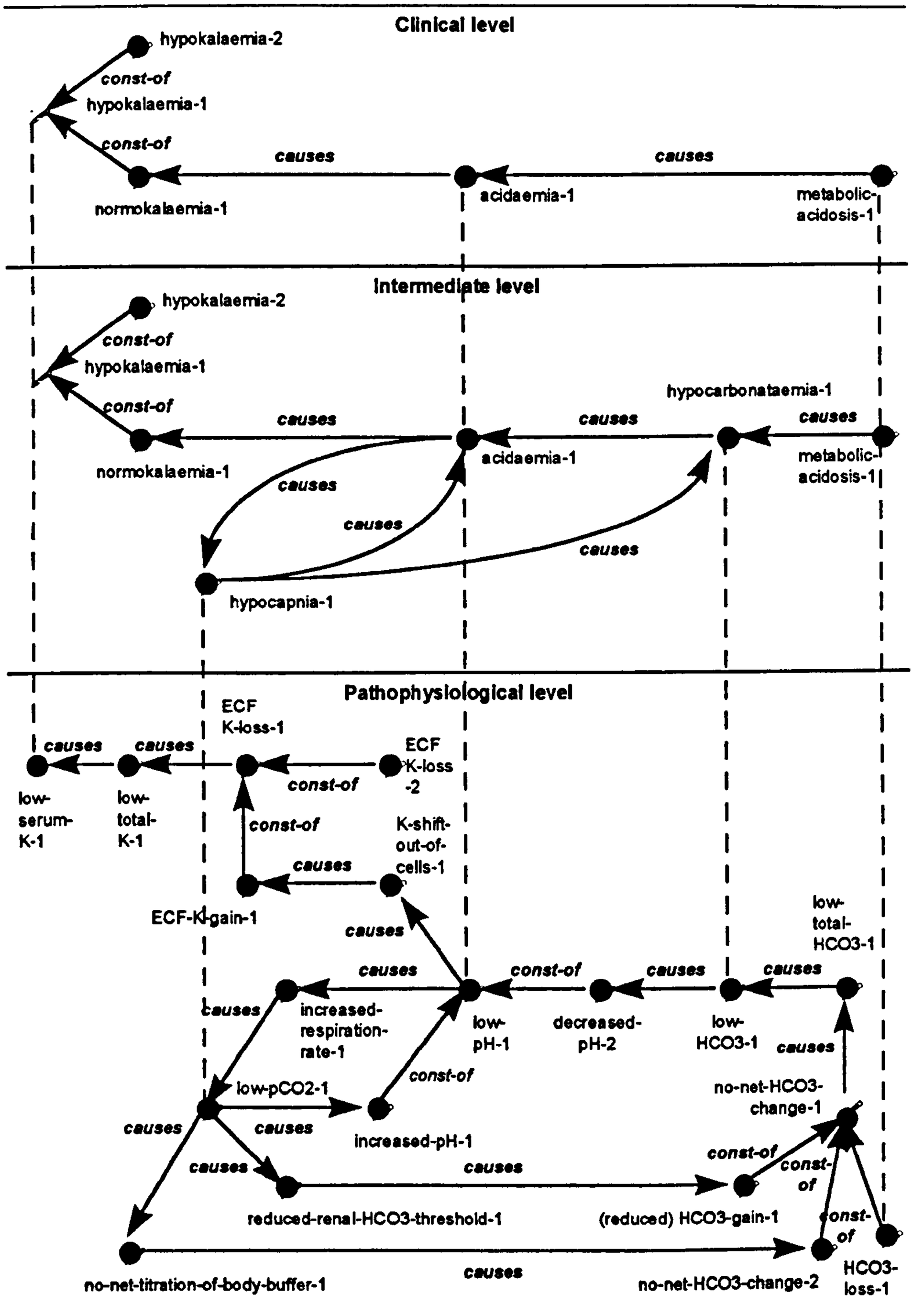


Figure 2-13. Causal relations in the domain of acid-base and electrolyte disorders, modelled in ABEL at multiple levels of detail (from Patil, 1981).

There were several differences between the representation of causal relations in ABEL and those used in CADUCEUS and CASNET (Patil and Senyk, 1988). In ABEL each causal relation represented a functional constraint between the attributes of the cause and effect nodes. Furthermore, each aggregate causal link was described at several levels of detail. As mentioned above, the critical feature of ABEL was its ability to determine and represent situations where a hypothesis is capable of explaining only part of an observed finding. This was achieved through ABEL's PSM construction operators. Component summation combined the effects of multiple causes, taking into account possible interactions among them. Component decomposition identified the components of a finding that had not been accounted for. This was achieved by taking the difference between the attributes of the finding, such as severity and duration, and those predicted by the known and hypothesised diseases in the composite hypothesis. ABEL's operators worked by translating the interactions perceived at the clinical level to the pathophysiological level, where they were analysed or elaborated. The results were then translated back, or aggregated, to the clinical level.

The diagnostic scenarios or diagnostic closures, such as the one depicted in Figure 2-14, were constructed by projecting the states of a PSM forward hypothetically, identifying the consequences predicted by these states, with a concomitant backward projection from the unexplained states of the PSM to identify diseases that could not account for them. In an informal analysis of the efficiency of various components of ABEL, it was found that nearly three-fourths of the time was spent constructing scenarios or closures, with the remainder of the time delegated to all other aspects of model construction and information gathering. For this reason, the developers proceeded with the design of a new system which used several types of links to capture possible relations among disease entities and manifestations, ranging from purely syndromic associations to detailed quantitative functional relations, thereby reasoning at different levels of granularity (Uckun, 1992), and compiled causal links to express clinical pathophysiological relations in order to achieve component summation and decomposition without the use of computationally expensive multiple levels of reasoning (Patil and Senyk, 1988). The Guardian system described below solves the problem of resource limitations for causal simulation reasoning by

employing both taxonomical as well as causal models for reactive and reasoned responses respectively.

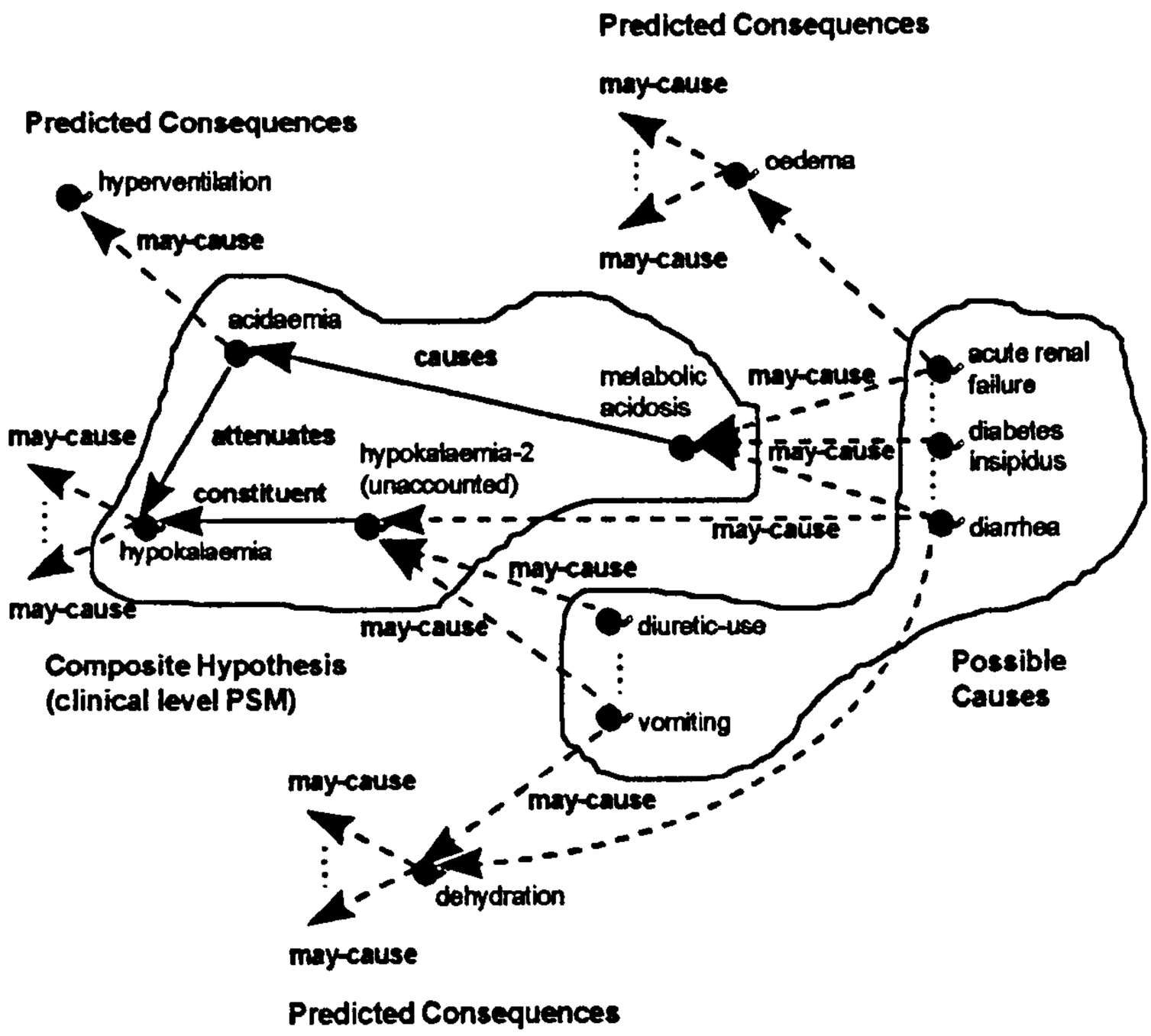


Figure 2-14. Use of diagnostic closures or scenarios in ABEL (from Patil, 1987).

2.4.5. GUARDIAN

The Guardian system was developed to function as a proof-of-concept prototype intelligent agent for the critical care environment (Hayes-Roth, 1990; Hayes-Roth, 1992). The aim was to develop a generic AI architecture for intelligent monitoring and control (IMC), suitable for application in multiple domains (Hayes-Roth et al, 1989). The chosen IMC application domain was patient monitoring in the surgical intensive care unit (SICU). Guardian has been characterised by its developers as a work in progress. In other words Guardian is an evolving intelligent agent and must

handle a growing variety of critical care scenarios. The Guardian Demonstration 4 scenario was completed in June, 1990 (Hayes-Roth et al, 1992).

Being a proof-of-concept system, Guardian monitors a simulated surgical ICU patient-ventilator-laboratory system. Thus, the primary objective underlying the development of Guardian was not to build a practical system suitable for near-term integration with existing information management systems but to develop an advanced prototype critical care consultant that demonstrably performed and co-ordinated the full range of intelligent behaviour required for effective critical care monitoring, did so reliably in a significant range of medical situations, and would arguably scale up to meet the comprehensive set of practical requirements with an appropriate development effort.

SICU patients have been subjected to major surgery and suffer temporary failure of one or more organ systems. Therefore, life-support devices assume the fundamental functions of the ailing organ system until it can heal and resume its normal function. Because life-support devices injure as well as sustain patients, one objective of SICU monitoring is to wean the patient from the device as rapidly as possible and consistent with other therapeutic objectives (Hayes-Roth et al, 1992).

Based on a model of the patient's physiological impairment and expected rate of recovery, partially based on the APACHE-II severity of disease classification system (Knaus et al, 1985), the physician orders an initial configuration of device settings that substantially augment the patient's own function, followed by a program of modifications to those settings that gradually reduce the level of assistance to zero, followed by device withdrawal. ICU staff monitors the patient's response to the prescribed treatment by observation of biomedical measurements to validate the pathophysiological assumptions underlying the weaning plan, refine the plan when the underlying assumptions are proven incorrect, and perform additional actions to diagnose and correct other unanticipated problems. In supporting such activity Guardian would have to interact with ICU team members in a number of ways, for example to:

1. Summarise the patient's progress and condition for clinicians and physicians on rounds.
2. Alert clinicians to imminent problems before they might otherwise be noticed.
3. Suggest and critique alternative therapies.

In doing so, the system would have to satisfy the following reasoning requirements:

1. Reasoning about complex time-varying behaviours.
2. Integration of multiple reasoning activities for interpretation of patient data, diagnosis of observed signs and symptoms, prediction and explanation of the patient's progress, reaction to urgent patient conditions, and planning of longer-term actions.
3. Integration of perception, reasoning, and action.
4. Dynamic allocation of limited computational resources.
5. Co-ordination of multiple response modes: immediate reactive responses to emergencies, prompt associative responses where clinical knowledge is applicable, and deliberate reasoned responses to complex or evolving patient conditions.

Guardian's reasoning system instantiates a domain-independent dynamic control architecture, implemented as the BB1 blackboard system (Hayes-Roth, 1985, 1990), to interpret perceived information from the environment, perform all knowledge-based reasoning and problem solving, that is detection, diagnosis, prediction, planning, explanation, and decide what actions to perform. The reasoning system also constructs and modifies dynamic global control plans to co-ordinate its perception, reasoning, and action. Each monitored datum, if passed to the reasoning system for being non-physiological, will trigger several reasoning operations whose execution produces several cognitive events and triggers new operations, and so on. Guardian's cognitive state at any one time is determined by the state of a global memory of facts. The global memory includes the knowledge base and the transient information used to drive reasoning processes. Reasoning operations are triggered by clinical events, signified by changes to the global memory, and, when executed, make new changes to the global memory and produce new events. Reasoning results include the intermediate and final products of reasoning tasks, such as observations, hypotheses,

diagnoses, predictions, plans, etc. These are related temporally and in relation to each other. A task scheduler decides which task to perform when, based on the current cognitive state, real-time constraints on the utility of actions, information stored in the knowledge base on instantiations of such constraints, and the strategy pursued as a result of these considerations, ie. reaction, associative reasoning, reasoned response, etc. Action systems control the execution of actions to affect the environment based on perception and reasoning. In 1992, Guardian's actions included: directly changing some ventilator settings, recommending other interventions to correct diagnosed problems or avoid predicted problems, and giving explanations of its monitoring strategy, its reasoning about particular problems, and the biological and physical phenomena underlying the patient's conditions.

Guardian's knowledge base is summarised in Table 2-4. It is organised in interconnected layers and divided into epistemological and generic reasoning knowledge. Clinical knowledge represents common diagnoses of disorders and diseases in a classification hierarchy, a part of which is depicted in Figure 2-15 for the case of hypoxia (O_2 delivery disorder). Diagnoses have attributes for:

1. Credibility: conditional probability given current evidence.
2. Criticality: cost of not correcting a problem when it exists.
3. Urgency: rate of decrease in utility of corrective action over time.

Factual Knowledge	
Clinical knowledge	hypoxia problems
Biological and device knowledge	respiratory, circulation, pulmonary exchange tissue metabolism, tissue exchange, ventilator
First-principles knowledge	flow, diffusion, exchange, equilibrium, metabolism
Generic Reasoning Knowledge	
Associative reasoning	diagnosis, reaction, prediction, planning, explanation
Model-based reasoning	diagnosis, reaction, prediction, inference, planning, explanation
Control reasoning	prioritisation, focus of attention, task co-ordination, reasoning strategies, real-time performance

Table 2-4. Summary of Guardian's knowledge base (from Hayes-Roth et al, 1992).

Each diagnosis is connected by probabilistic links via intervening variables to relevant types of evidence, and standard therapy actions, with attributes for:

1. Observability: frequency, or cost of gathering the evidence.
2. Resources: time and other resources required to execute an action.
3. Consequences: predictable effects of executing the action.
4. Reversibility: capability and cost of reversing the action.

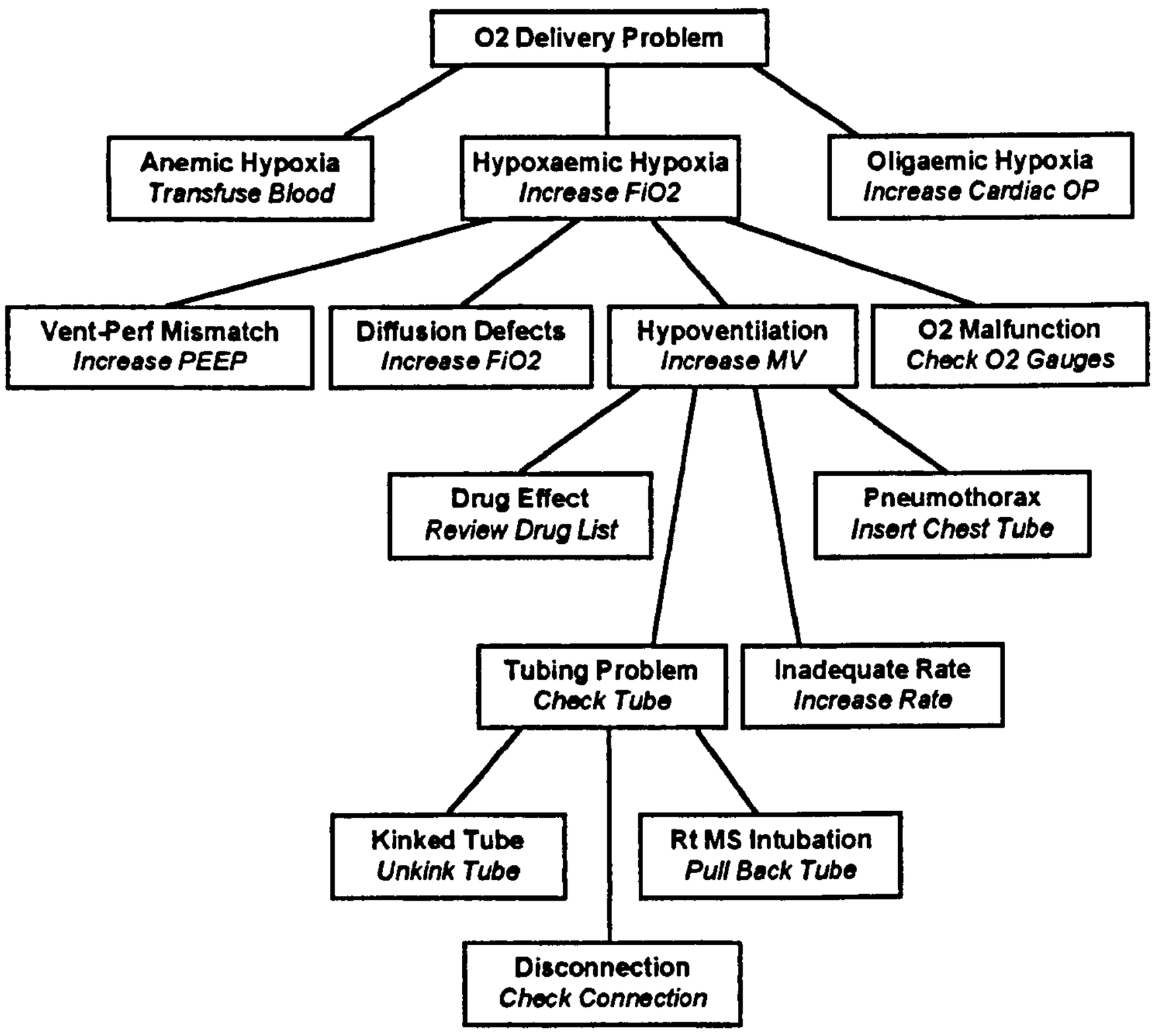


Figure 2-15. Excerpt from Guardian's hierarchical representation of knowledge about Hypoxia problems (from Hayes-Roth et al, 1992).

Biological and device knowledge represents anatomy and physiology in the isomorphic symbolic form of implication. For example in its structure-function knowledge base a section of which is depicted in Figure 2-16, Guardian includes anatomical facts such as: the lung is a respiratory structure; the lung includes the bronchi and alveoli; the bronchi and alveoli are structurally connected. It also includes

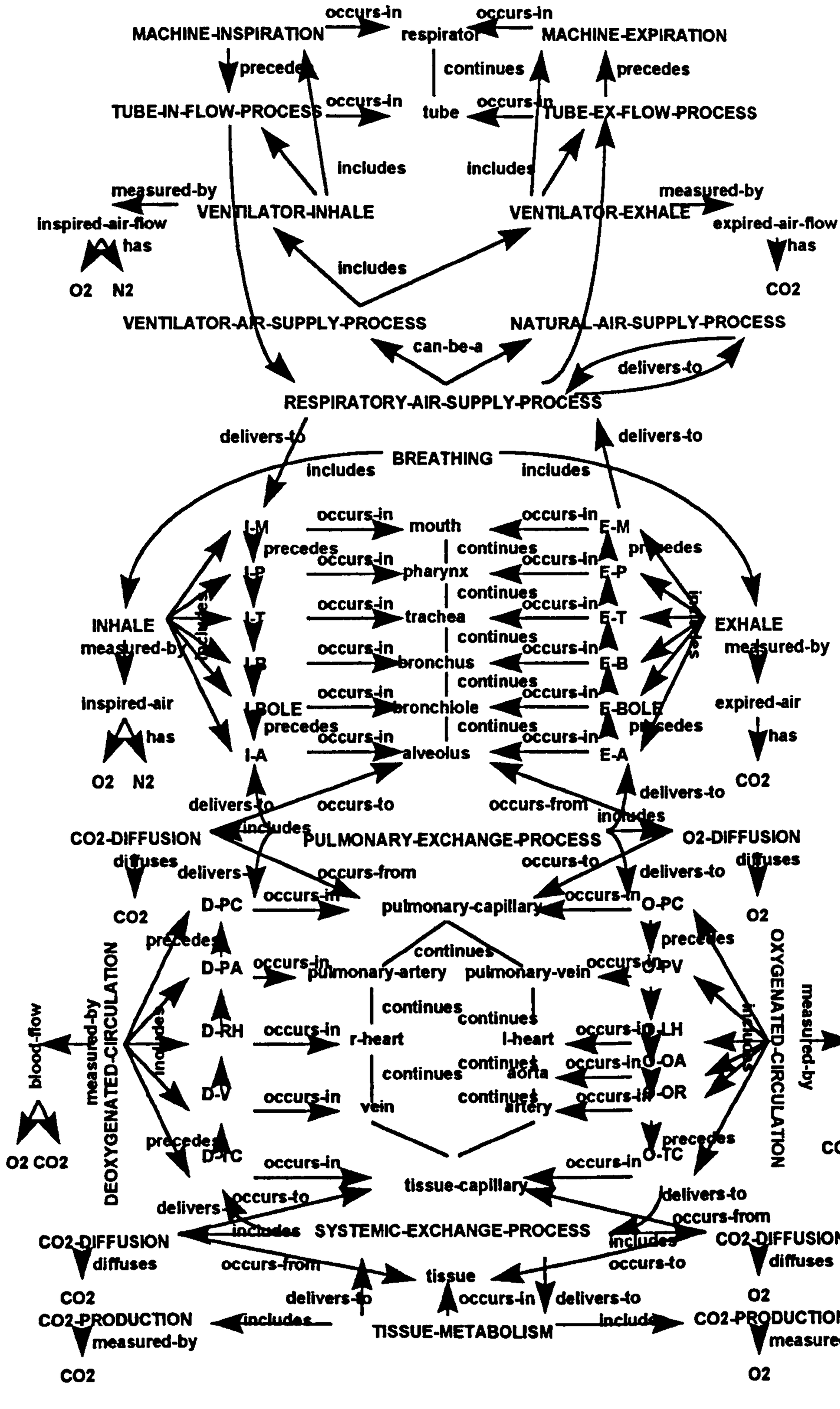


Figure 2-16. Excerpt from Guardian's knowledge of O_2-transportation physiology represented in a semantic structure-function network (from Hayes-Roth et al, 1992).

physiological facts such as: breathing is a respiratory process; breathing includes inhaling and exhaling; breathing occurs in the respiratory system structures; inhaling through the mouth precedes inhaling through the throat, and so on. First-principles knowledge is represented in similar relational graph structures which model the normal and abnormal structure and function of known types of physical systems, such as relations that hold among pressure, resistance, and flow in flow systems, and blockages or leaks causing predictable changes.

Generic reasoning knowledge represents potential reasoning operations and strategies for organising sequences of operations to achieve goals. Guardian has reasoning knowledge for several tasks: diagnosis of observed signs and symptoms; reaction to problems; prediction of patient condition; causal inference of precursors and consequences of observations and problems; planning of longer-term course of action; explanation of underlying causal phenomena. Furthermore, Guardian can perform each of these tasks using associative or model-based reasoning methods. Associative diagnosis involves the propagation of asynchronously arriving evidence through a diagnosis hierarchy. For example, Guardian responds to an observed rise in PIP by quickly diagnosing a hypoventilation problem and increasing the patient's ventilation.

2.4.6. Summary and Conclusion

There are three types of causal simulation models: behavioural, functional and structural simulation models (Uckun, 1992). By contrast to classification models which simply classify an observed behaviour, causal simulation models describe how organ systems produce an observed behaviour (Clancey, 1989). On the other hand, classification and behavioural simulation models do not characterise on any analysis level the full state of a system being reasoned about, but rather describe hidden internal states, observed manifestations, and causal relations among these attributes, without complete descriptions of the purpose of transitions or how transitions follow from system structure. By contrast, functional models provide a complete explanation or system description, since they capture the system's purpose procedurally on multiple abstraction levels. Thus, functional models can relate states to functional

goals. Furthermore, a structure-function model fully accounts for each organ system component in terms of that component's role in fulfilling the design or simulation system's function.

Simulation models offer a number of advantages over classification models, including:

1. Better handling of the problems in the periphery of the knowledge base, therefore, robustness and graceful degradation.
2. Better explanation of reasoning processes.
3. Prediction of future states.
4. Easier truth maintenance.
5. Being able to determine what is happening in a situation at a particular time.

Although simulation systems are expected to outperform classification systems based on associative knowledge in complicated situations, disadvantages of these systems are also pointed out (Uckun, 1992):

1. Increased knowledge acquisition load.
2. Lack of detailed domain knowledge.
3. Inherent uncertainty resulting in increased ambiguity.
4. Computational overheads of simulating complex domain models.
5. Introduction of undue complexity in simple problem-solving cases.

However, taxonomical classification and causal simulation are orthogonal knowledge representations and systems have been developed to exploit either or both to varied degrees (Clancey, 1985, 1989, 1992; Uckun, 1992; Ramoni et al, 1992; Stefanelli, 1993; Uckun, 1994).

2.5. Belief Networks

2.5.1. Introduction

Section 2.3 discussed the reasons why both early probabilistic as well as heuristic AI-based methods failed to provide effective solutions to the problem of reasoning with partial belief and incomplete information in classification KBS. Section 2.4 described what may be termed the pure AI approach to the support of classification reasoning, which employs qualitative causal models of disease processes for the de novo construction of alternative scenarios of behavioural, functional and structural dependencies among clinical events, linking patient data to the underlying taxonomical structures, thereby avoiding the use of numerical information. However, because of resource limitations such an approach may not be feasible for certain medical decision support problems.

An alternative approach emerged in the 1980s, when medical AI researchers united methodologically the two aspects of reasoning with partial belief and incomplete information and developed methods for the acquisition, representation, manipulation and explanation of uncertain medical knowledge that combined the structure of the formal AI representations described in this chapter and summarised in Table 2-5, with the formal uncertainty calculus provided by probability theory. The resultant knowledge representation structures were termed belief networks and are described below. This section describes hierarchical classification belief networks, however, a number of methods of inference with belief networks have been explored which permit more complex structural representations (Horvitz et al, 1988).

	Representation Epistemology	Reasoning & Strategic Knowledge	Patient-Specific Model
MYCIN	Definitional and heuristic associations between goals.	Implicit inference structure of simple and heuristic classification.	Assertions organised in a context tree for the application of rules.
CASNET	Hierarchical taxonomy of disease categories linked to causal network of dysfunctional states (observations, empirical associations, treatment plan). Multi-level representation.	Explicit bottom-up (data-driven) causal-process classification. Prediction (reversal).	Symptom explanations as alternative causal-process pathways with assigned weights.
NEOMYCIN	Hierarchical taxonomy of aetiologies (triggers, observations, heuristic associations, temporal). Causal network linking observations to aetiologies via dysfunctional states (caused-by).	Task-specific architecture (explicates MYCIN's inference structure). Focusing strategy. Abstract hierarchical operators use backward deduction to confirm a hypothesised solution.	Differential diagnosis plan for task instantiation.
INTERNIST-1	Single intermixed hierarchy of diseases with basic causal relations (causes, precedes, predisposes-to coincident-with).	Heuristic reasoning strategies (problem area formation and partitioning) directed focusing and information acquisition.	Lists of hypotheses and manifestations with associated scores directed focusing on problem areas.
CADUCEUS	Multiple orthogonal pure hierarchies (organ system involvement, aetiology, etc) with superimposed causal network links (causal, planning, spanning, constrictors).	Dynamic synthesis operators constructed alternative composite hypotheses. Searching the PSM space.	Differential diagnosis. Graphs as models of processes. Composite hypotheses without interactive analysis.
ABEL	Causal networks on multiple levels of detail (pathophysiological, intermediate, clinical).	Dynamic synthesis operators on different levels of detail (aggregation, elaboration, summation, projection, decomposition). Prediction.	Subgraphs treated as new process descriptions related to each other.

Table 2-5. Overview of KBS models for the acquisition, representation and manipulation of uncertain knowledge (from Keravnou and Washbrook, 1989 and Clancey, 1992).

2.5.2. Probabilistic Interpretative Decision-Making Revisited

Until the early 1990s, the most popular of the heuristic uncertain knowledge engineering environments described in section 2.3 was the rule-based CF model

(Lauritzen and Spiegelhalter, 1988; Heckerman et al, 1992). As discussed in section 2.3.3, the particular approach to medical knowledge modelling had appeal as providing a general and flexible scheme for acquiring and representing expert knowledge in a declarative and modular form, conferring the ability to add or remove rules from a knowledge base without modifying other rules. Furthermore, the CF model was meant to represent, combine, and propagate the effects of multiple sources of evidence in terms of their joint degree of confirmation or disconfirmation of each hypothesis of interest. Thus, certainty factors were meant to represent an update or change in belief induced by the evidence, rather than an absolute degree of belief, as did probability (Horvitz et al, 1988). Thus, the approach avoided the need for a priori information, which was believed to assume more information than given and to be hard to acquire. Instead, the CF model and the other schemes described in section 2.3 assumed equal a priori probabilities (Heckerman and Miller, 1986). The handling of a priori probabilistic knowledge in the CF and INTERNIST models was meant to be consistent with studies which showed that people tend to ignore priors (Tversky and Kahneman, 1974; Kahneman et al, 1982).

Section 2.3.3.4 cited and briefly discussed evaluations of the CF and other heuristic models of inexact reasoning which have indicated that, although designed as such, these schemes did not satisfy the property of modularity. In fact the assumptions underlying these models were obscured and stronger than those of the simple-Bayes model, thereby promoting errors in judgement, computer-based reasoning, and leading to overall less accuracy. Furthermore, the errors that accrue from assuming equal a priori probabilities may be less serious in domains where the quantity and quality of evidence typically overwhelms the a priori knowledge, however, in general, even approximate information about a priori probabilities may be valuable knowledge and discarding this information may lead to serious errors. In addition to these theoretical problems, section 2.3.3.4 also discussed a serious practical limitation common to these models, regarding the acquisition of belief estimates and the direction in which they must be specified. Another practical limitation of the CF model was that the model did not provide a basis for consistent management of evidence bearing on hypotheses that are hierarchically related, and thus did not facilitate the narrowing of focus, i.e. the currently considered set of hypotheses, with

accumulation of evidence; a process which characterises diagnostic reasoning in medicine and expert reasoning in general (Gordon and Shortliffe, 1985).

Finally, in response to Feinstein's criticism, researchers have argued that logic-based reasoning methods may also be considered normative in that they prescribe a set of rules for correct inference under uncertainty; that is, a system that reasons or makes recommendations using these rules may be viewed as normative with respect to deterministic knowledge. By contrast, decision theory is not generally proposed as a descriptive theory; it does not purport to provide a description of how people actually behave when reasoning under uncertainty. Indeed, Tversky and Kahneman have demonstrated that people frequently do not behave in accordance with decision theory. In fact, the characteristic biases exhibited in intuitive judgement are part of the justification for applying decision sciences to assist people with decision making (Horvitz et al, 1988; Heckerman et al, 1992).

For these reasons, the research community has abandoned the use of heuristic models of inexact reasoning in preference to belief networks which employ probability theory as a system of personal belief, in providing a flexible and operational means of uncertainty assessment, representation and manipulation, which helps physicians avoid the errors promoted during knowledge elicitation, reduces errors in reasoning with partial belief and incomplete information, and generates decision supporting information of higher accuracy. Belief networks, also known as causal probabilistic networks (Andreassen et al, 1991), are directed, acyclic graphs which support local computations for uncertain inference by means of evidence propagation. A key advantage of belief networks over the early formal probabilistic approaches is their ability to represent probabilistic relationships precisely and concisely, while preserving the rich representational semantics of more structured AI methods. The new representations can facilitate assessment of coherent a priori probability distributions, make assumptions explicit, and allow assumptions to be manipulated easily by knowledge engineers and experts (Horvitz et al, 1988).

2.5.3. Evidence Propagation in Hierarchical Belief Networks

A formal probabilistic method for handling the impact and propagation of evidence on the belief of hypotheses which avoids the problems discussed above was suggested by Kim and Pearl (1983), provided the hypotheses were organised in a strict or singly connected hierarchy. A strict hierarchy is a tree structured hierarchy, in which the root node describes a class of diagnoses, and leaf nodes form a mutually exclusive and collectively exhaustive set of hypotheses, $H = \{h_1, h_2, h_m\}$, for that class. Any intermediate-level hypothesis, S, is the disjunction of its immediate descendants and can be thought of as a subset of H, whose members are the leaf nodes, or singleton hypotheses, or one-element subsets, which are descendants of S.

Initially, each singleton hypothesis h_i is quantified with a measure of belief Bel(h_i), reflecting the probability that h_i is true given all previous evidence. By mutual exclusivity, the belief in each intermediate-level hypothesis is the sum of the beliefs given to its constituents. Furthermore, given the assumption of conditional independence between elements of each subset, each new piece of evidence bearing directly upon one of the subsets, contributes no information about its individual elements. Now, suppose that a new piece of evidence e arrives which directly bears upon one of the subsets, say S, but says nothing about its constituents. The impact of e on the belief of every hypothesis in the hierarchy can be calculated with the following three-step process, based on the simple-Bayes model described in section 2.3.2 (Pearl, 1986).

Step 1: Estimation. An expert determines the hypothesis set S upon which the evidence bears directly, and estimates the degree λ_s to which the evidence confirms or disconfirms S. λ_s is the likelihood ratio:

$$\lambda_S = \frac{p(e|S)}{p(e|\neg S)} \tag{2.11}$$

Confirmation is expressed by $\lambda_s > 1$ and disconfirmation by $\lambda_s < 1$.

Step 2: Weight distribution. Each singleton hypothesis $h_i \in S$ obtains the weight $W_i = \lambda_s$ while every hypothesis outside S receives a unity weight $W_i = 1$.

Step 3: Belief updating. The belief in each singleton hypothesis h_i is updated from the initial value of Bel(h_i) to:

$$\text{Bel}'(h_i) = p(h_i|e) = \alpha_s W_i \text{Bel}(h_i) \tag{2.12}$$

where α_s is a normalising factor:

$$\alpha_s = \left[\sum_i W_i \, \text{Bel}'(h_i)\right]^{-1} \tag{2.13}$$

The belief in each intermediate-level hypothesis is computed from the sum of the beliefs of its singleton elements.

This three phased process may be conducted recursively, where the updated beliefs calculated for evidence e_k serve as a priori beliefs for the next evidence e_{k+1}. The normalisation phase can be postponed until several pieces of evidence e_1, e_2, ..., e_n exert their impacts on their corresponding hypotheses S_1, S_2, ..., S_n. In this case, the weights are combined multiplicatively via $W_i(e_1, e_2, ..., e_n) = W_i^1 W_i^2 ... W_i^n$, where

$$\begin{aligned} W_i^k &= \lambda_{sk}, \quad && \text{if } h_i \in S_k \\ &= 1, && \text{if } h_i \in \neg S_k \end{aligned} \tag{2.14}$$

An implementation of this method for evidence impact, propagation and aggregation in hierarchical belief networks is described in the chapter which follows, applied to the interpretation of evidence on disorders of acid-base metabolism.

2.6. Conclusion

This chapter described the fundamental problem of reasoning with partial belief and incomplete information that characterises clinical decision making, and a number of methods for uncertainty management in medical KBS. Section 2.3 was a critical review of heuristic AI-based methods designed to replace probability theory for the acquisition, representation and manipulation of uncertain medical knowledge, due to misconceived limitations of the theory for the task. The review exposed that in fact these methods promote errors in judgement and lead to interpretative decision making of poorer performance and accuracy. Following that, section 2.4 described what may

be termed the pure AI approach to the support of classification reasoning, which employs causal models of disease processes for the simulation of alternative scenarios of organ system behaviour, function and structure given a set of observations, avoiding the use of numerical information. Finally, section 2.5 described belief networks as a method of inexact reasoning which unites the two aspects of reasoning with partial belief and incomplete information by combining the structure of formal AI representations with the formal uncertainty calculus provided by probability theory as a system of personal belief, justifying its use both from a theoretical and a pragmatic perspective, in providing a flexible and operational means of uncertainty assessment, representation and manipulation, which helps physicians avoid the errors promoted during knowledge elicitation, reduces errors in reasoning with partial belief and incomplete information, and generates decision supporting information of higher accuracy.

The chapter which follows describes the implementation of the method described above for evidence propagation in a hierarchical belief network, used for the interpretation of laboratory investigation data and in particular those concerning disorders of acid-base balance. The particular implementation was used as the KBS integration prototype for the development of the ICIMS system.

- 3 -

The Prototype Knowledge-Based Clinical Evidence Interpretation System

3.1. Introduction

The previous chapter discussed the reasons why the medical AI research community has returned to the calculus of probability theory as a system of belief for the acquisition, representation and manipulation of uncertain medical knowledge in the form of belief networks. This chapter describes an application of the method described in section 2.5.3, to the interpretation of laboratory investigation data regarding abnormalities of acid-base balance, which was used as the KBS integration prototype for the incorporation of the computational intelligence required for the development of the ICIMS system. The chapter which follows describes the development of an object-oriented clinical information management system which was designed to combine the tasks of acquisition, organisation, storage, update and review of the domain knowledge base required for the interpretation of the clinical information acquired in the process of monitoring the ICU patient with disturbed acid-base homeostasis, as well as of the information generated in the process, within a singular system architecture.

3.2. System Architecture

3.2.1. Purpose

The KBS integration prototype, named Bgas (Chelsom, 1990), was developed in a logic-based environment (Prolog) to provide tools for the acquisition, representation and manipulation of the domain knowledge-base required for interpretative decision-making in the domain of acid-base balance, and to thereby assess the performance of a singly connected hierarchical belief network in providing assistance with the interpretation of blood-gas laboratory analysis data.

3.2.2. System Organisation

A knowledge-base editing environment named Framebuilder, was developed to enable clinicians to construct a strict hierarchy of probabilistic classification knowledge frames, and to specify expected patterns of evidence for the recognition of 16 simple and complex disorders of acid-base metabolism, by choosing clinical parameters from a vocabulary of laboratory data, signs and symptoms, relations between data variables and clinical history. Figure 3-1 depicts the prototype belief network which was constructed using Framebuilder for the probabilistic classification of the evidential information generated by blood-gas analyses.

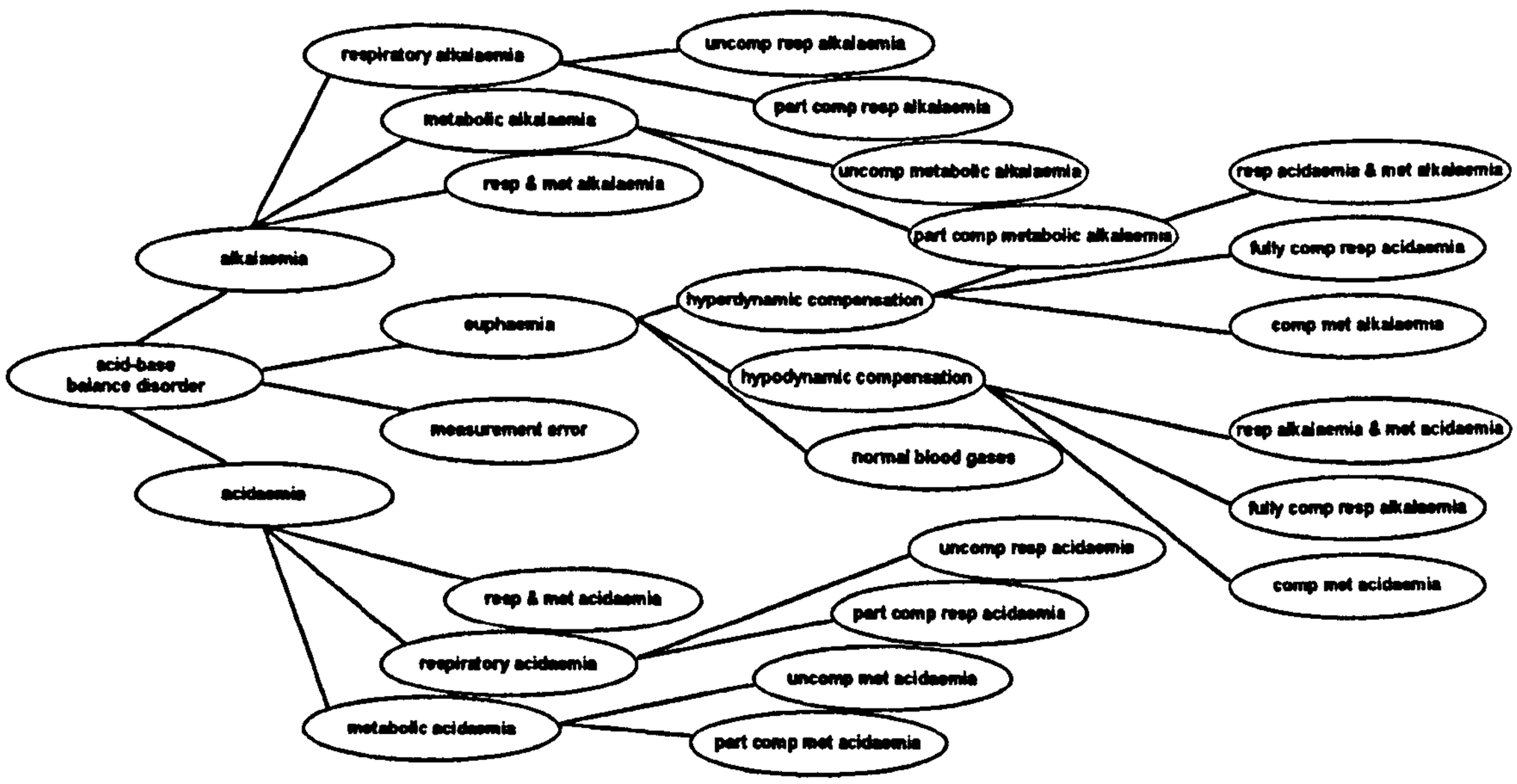


Figure 3-1. The prototype KBS knowledge-base for interpretative decision-making support in acid-base balance, organised in a strict hierarchy of disorder profile frames.

Each clinical parameter specified in the disorder profile frames was accompanied by the conditional probability of the particular piece of evidence being observed, given the disorder represented in the frame. Furthermore, each frame was assigned an a priori value of the probability of the occurrence of the represented disorder given no evidence had been observed. Table 3-1 lists the 16 disorder profiles represented in the prototype knowledge base, with their basic definitional features (Section 2.3.3.1). The relationship between these features, namely acidity (pH), partial pressure of carbon

Acid-Base Balance Disorder	Definitional Features					
	Hypocapnia (low pCO_2)	Normocapnia (normal pCO_2)	Hypercapnia (high pCO_2)	Hypobicarbonataemia (low $[HCO_3^-]$)	Normobicarbonataemia (normal $[HCO_3^-]$)	Hyperbicarbonataemia (high $[HCO_3^-]$)
Alkalaemia (high pH)						
Respiratory	√					
1. Uncompensated					√	
2. Partially Compensated				√		
Metabolic						√
3. Uncompensated		√				
4. Partially Compensated			√			
5. Respiratory & Metabolic	√					√
Acidaemia (low pH)						
Respiratory			√			
6. Uncompensated					√	
7. Partially Compensated						√
Metabolic				√		
8. Uncompensated		√				
9. Partially Compensated	√					
10. Respiratory & Metabolic			√	√		
Euphaemia (physiological pH)						
Hyperdynamic Compensation			√			√
11. Resp Acidaemia & Met Alkalemia						
12. Fully Comp Resp Acidaemia						
13. Comp Met Alkalaemia						
Hypodynamic Compensation	√			√		
14. Resp Alkalemia & Met Acidaemia						
15. Fully Comp Resp Alkalemia						
16. Comp Met Acidaemia						
17. Physiological Blood Gases		√			√	

Table 3-1. Specification of the definitional features of the 17 disorders of acid-base metabolism recognised by the KBS prototype. Numbered disorders were represented as the leafs of the hierarchical belief network.

dioxide (pCO_2), and bicarbonate ion concentration ($[HCO_3^-]$), is given by the Henderson-Hasselbach equation:

$$pH = 6.1 + \log \frac{[HCO3-]}{\alpha(pCO2)} \tag{3.1}$$

where $\alpha = 0.3$ (mmol/L)/mmHg at 38°C.

The hierarchical belief network constructed using Framebuilder was processed in the manner suggested by Pearl (Section 2.5.3), in order to assess and propagate the effect of each piece of evidence given in a case, using a blackboard controlled, task-specific reasoning module for the construction of patient-specific models (PSM) of probabilistic classification, from the general hierarchical model (Clancey, 1992). Figure 3-2 depicts the resultant dual-panelled blackboard architecture with its associated interpretative task-domain knowledge sources. PSM blackboard entries were split into five levels of abstraction for the physiological diagnosis panel and four levels of abstraction for the clinical diagnosis panel. The physiological diagnosis panel was designed to function in a bottom-up manner, starting from raw patient data and proceeding up toward the root of the virtual PSM hierarchy, to produce a differential diagnosis of disorders of acid-base metabolism. The clinical diagnosis panel was designed to work in the opposite direction, starting with a clinical diagnosis entered by the user and proceeding down toward the leafs of the virtual PSM hierarchy to generate expected consequences, which were latter used to critique the results from the physiological diagnosis panel in order to refine complex interpretative hypotheses which could not be differentiated in the light of measurement data alone.

The following section describes the implementation of the cognitive information processing knowledge sources used in the Bgas system. Figure 3-3 depicts an example PSM of interpretative decision-making, represented in terms of multi-level blackboard entries. Table 3-2 provides a summary of the knowledge sources (KS), specifying the blackboard (BB) levels at which each was triggered, the levels on which it operated, and further actions caused by its instantiation.

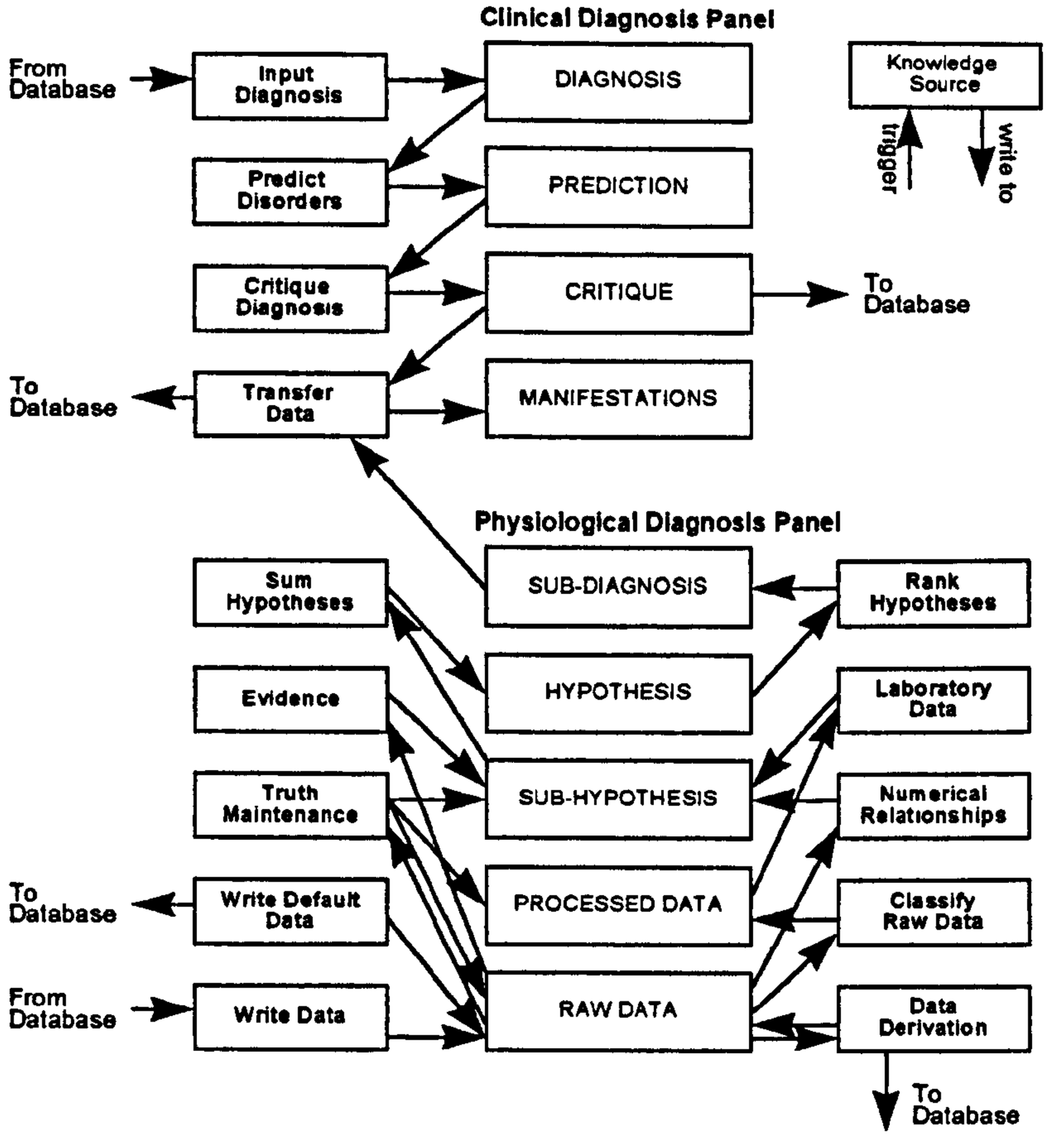


Figure 3-2. Implementation of the blackboard model of problem-solving task-domain organisation and representation in the KBS prototype (from Chelsom, 1990).

3.2.3. Task-Domain Knowledge-Sources

3.2.3.1. Data Transfer from Database to Blackboard

During a consultation session, apart from the BB entries, the system maintained a patient-specific database which comprised four data types: laboratory data variables, signs and symptoms, history and diseases. The RawData KS monitored the database for these data types and was triggered by any such data encountered in the database but not on the BB. The WriteRawData KS replaced any existing value of the data at the RawData BB level with the new value found in the database. The WriteDiseaseDiagnosis KS operated in the same way, monitoring the database for

data of the disease type and transferring them to the Diagnosis BB level. The WriteDefaultData KS monitored laboratory data variables which had default values specified in the knowledge base and was triggered after the WriteRawData KS and the DeriveData KS if neither had been able to provide a data value.

Disease **Pleural Effusion**	**Diagnosis**
Disease, Disorder **pleural effusion, compensated respiratory acidosis**	**Prediction**
Disease, Disorder, Critique **pleural effusion, comp respiratory acidosis, expected**	**Critique**
Disorder **compensated metabolic acidosis** **partially compensated metabolic acidosis**	**Manifestations**
Disorder, Belief **compensated metabolic acidosis, 0.5** **partially compensated metabolic acidosis, 0.5**	**SubDiagnosis**
Hypothesis, Belief **compensated metabolic acidosis, 1.525** **partially compensated metabolic acidosis, 1.525**	**Hypothesis**
Hypothesis, Evidence, Belief **dominant acidosis, pH, 1.024** **compensated metabolic acidosis, HCO3, pCO2, 1.007** **compensated metabolic acidosis, coma, 1.115**	**SubHypothesis**
Data Name, Classification, Probability **pH, high, 0.000** **pH, usual, 0.655** **pH, low, 0.355**	**ClassifiedData**
Data Type, Data Name, Value, Status **variable, pH, 7.350, measurement** **variable, Base Excess, -2.000, derivation** **variable, FIO2, 21.000, default** **symptom, coma, present, measurement**	**RawData**

Figure 3-3. Example of a patient-specific model of interpretative decision-making in the domain of acid-base balance, represented in terms of blackboard entries in the KBS prototype (from Chelsom, 1990).

3.2.3.2. Truth Maintenance

Any changes in the values of data in the database were detected by the WriteRawData KS and the entries at the RawData BB level were updated accordingly. Entries at the ClassifiedData and SubHypothesis BB levels were automatically updated because the KSs that produced them were re-triggered by the changes at the RawData BB level. Similarly, any necessary changes to default or derived data were made by simple re-triggering of the appropriate KS. Thus, truth maintenance was largely achieved by a passive process of KS re-triggering. If, however, the user changed a data value to unknown, then it was completely removed from the database and the WriteRawData KS did not detect the change. This problem was solved by the TruthMaintenance KS, which was triggered by data that had entries as RawData on the BB but no corresponding entry in the database. The TruthMaintenance KS removed any relevant entries from the ClassifiedData and SubHypothesis BB levels and also removed RawData entries that had been derived using data that were not known any more.

3.2.3.3. Data Derivation

The Bgas knowledge base contained expressions of the relationships between variables that could be used to derive data values to support interpretative decisions. The DataDerivation KS was triggered when all of the variables involved in the right-hand side of the data derivation expression appeared as entries at the RawData BB level and the result was written at the same level. For example, given the equation:

$$\text{Anion Gap} = [N_a^+] + [K^+] - [HCO_3^-] - [Cl^-] \qquad (3.2)$$

and values of $[N_a^+]$, $[K^+]$, $[HCO_3^-]$, $[Cl^-]$ specified at the RawData BB level, DataDerivation calculated Anion Gap and made an appropriate entry at the RawData level.

3.2.3.4. Data Transfer from Blackboard to Database

Default and derived data were written to the database, as well as to the BB, by the appropriate KS. The only other transfer of data from the BB to the database was achieved by the TransferData KS. The TransferData KS took each diagnosed disorder from the SubDiagnosis BB level and wrote it at the Manifestation level of the clinical diagnosis BB panel. In addition, any disorders that had been diagnosed with a belief greater than 0.95 were written to the database as patient history.

3.2.3.5. Data Classification

The Bgas knowledge base contained probability assignments regarding the expected level of data variables for particular interpretative hypotheses, as shown in Table 3-1, against which the available patient data should be matched. Thus, for some laboratory data variable V being a feature of some hypothesis H, the following conditional probability assignments were stored in the knowledge base:

$$p(V \text{ is low}|H), \quad p(V \text{ is normal}|H), \quad p(V \text{ is high}|H) \tag{3.3}$$

For example, in the case of partially compensated metabolic alkalaemia, the expected definitional features are shown in Table 3-1 as, high pH (alkalaemia), high [HCO3-] (metabolic), and high pCO2 (partial respiratory compensation). However, since patient data were supplied in the form of instrumentation measurements, these definitional features had to inferred by qualitative abstraction classification (Section 2.3.3.1).

In order to calculate the probability that a particular measurement being classified as evidence into the hierarchy of Figure 3-1, for the recognition of the underlying disorder, was low, normal or high, the ClassifyRawData KS implemented the following method of qualitative abstraction classification. Assuming that the probability distribution of V is known for a reference population of healthy individuals, a measurement V was defined as low if it fell further than two standard deviations (2σ) below the mean value (μ), high if it fell further than 2σ above the mean and normal if it lied between these limits.

Thus, if for example, the normal value of V for a particular patient when healthy, was more than 2σ above a measurement value X for that patient, then X had fallen by more than 2σ and it was classified as low. The probability that X was high was given by the area of the probability distribution more than 2σ below X, and the probability that X was normal was given by the area within +/- 2σ of X. That is,

$$p(X \text{ is low}) = p(u|u \geq X+2\sigma)$$

$$= 1 - \Phi(\frac{X+2\sigma-\mu}{\sigma}) \quad (3.4)$$

$$p(X \text{ is normal}) = p(u|X-2\sigma \leq u \leq X+2\sigma)$$

$$= \Phi(\frac{X+2\sigma-\mu}{\sigma}) - \Phi(\frac{X-2\sigma-\mu}{\sigma}) \quad (3.5)$$

$$p(X \text{ is high}) = p(u|u \leq X-2\sigma)$$

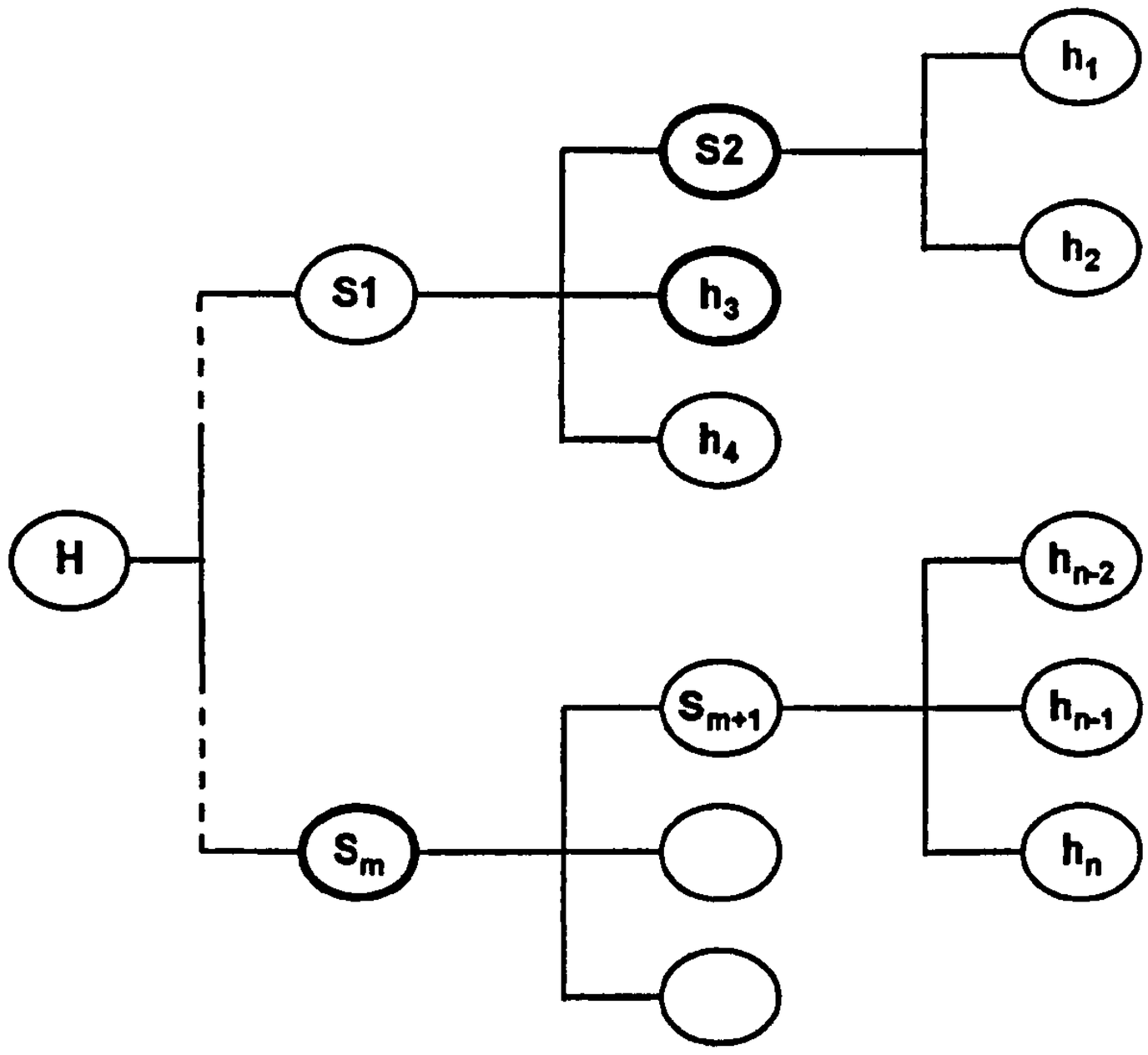
$$= \Phi(\frac{X-2\sigma-\mu}{\sigma}) \quad (3.6)$$

where Φ is the standard Gaussian distribution, which was stored in the Bgas knowledge base in the form of a look-up table. The ClassifyRawData KS calculated these probabilities and made an appropriate entry at the ClassifiedData BB level shown in Figure 3-3. These entries were subsequently used as evidence as described below.

3.2.3.6. Evidence Impact on Interpretative Hypotheses

Three knowledge sources, collectively called evidence handlers, implemented the method of belief propagation described in section 2.5.3. These were the Evidence, VariableEvidence and RelationEvidence KSs, which calculated the updating factors W_i^j (2.14) for each hypothesis h_i which had a probability assignment $p(e_j|h_i)$ (3.3) for the particular piece of evidence e_j under consideration.

Figure 3-4 depicts an example hierarchical belief propagation network indicating the interpretative hypotheses which have evidence, i.e. $p(e_j|h_i)$, assigned in a knowledge



S_m Nodes with assignment $p(e|S_m)$ in a knowledge base.

$H = \{h_1, h_2, ..., h_n\}$

$S2 = \{h_1, h_2\}$

Figure 3-4. Example of a hierarchical belief network indicating interpretative hypotheses with assigned evidence (adapted from Chelsom, 1990).

The required updating factors were calculated in the following manner. Since there was no indication in the knowledge base about how to treat unassigned nodes h_u, given evidence e, it was assumed that the observation of e had no effect on the belief in h_u. Hence,

$$p(h_u|e) = p(h_u) \tag{3.7}$$

where $p(h_u)$ is the probability of h_u prior to the evidence e.

Bayes' theorem states (2.6):

$$p(h_u|e) = \frac{p(e|h_u)p(h_u)}{\sum_i p(e|h_i)p(h_i)}, \quad i = 1 \text{ to } n \text{ leaf nodes.} \tag{3.8}$$

Combining (3.7) and (3.8):

$$p(e|h_u) = \Sigma_i\, p(e|h_i)p(h_i) \tag{3.9}$$

which implies that $p(e|h_u)$ is constant for all unassigned nodes. The summation in (3.9) can be split into the summation over the assigned nodes, denoted by Σ_a, and the sum over the unassigned nodes, denoted by Σ_u, so that (3.9) can be rewritten as:

$$p(e|h_u) = \Sigma_a\, p(e|h_i)p(h_i) + \Sigma_u\, p(e|h_i)p(h_i) \tag{3.10}$$

Since $p(e|h_u)$ is constant, the summation of $p(e|h_i)p(h_i)$ over the unassigned nodes is given by:

$$\Sigma_u\, p(e|h_i)p(h_i) = p(e|h_u)\, \Sigma_u\, p(h_i) \tag{3.11}$$

$$= p(e|h_u)(1 - \Sigma_a\, p(h_i))$$

since the leaf nodes are mutually exclusive and collectively exhaustive within each subset of hypotheses, and so $\Sigma_u\, p(h_i) = 1 - \Sigma_a\, p(h_i)$.

Substituting for $p(e|h_u)$ from (3.10) and rearranging:

$$\sum_u p(e|h_i)p(h_i) = \frac{(\sum_a p(e|h_i)p(h_i))(1 - \sum_a p(h_i))}{\sum_a p(h_i)} \tag{3.12}$$

Regarding the impact of evidence e on an assigned node, Bayes' theorem states:

$$p(h_a|e) = \frac{p(e|h_a)p(h_a)}{\sum_i p(e|h_i)p(h_i)}, \quad i = 1 \text{ to } n \text{ leaf nodes.} \tag{3.13}$$

where $p(h_a)$ is the probability of h_a prior to the evidence e.

Splitting the summation in the denominator of (3.13), as before, into summation over the assigned and unassigned nodes:

$$p(h_a|e) = \frac{p(e|h_a)p(h_a)}{\sum_a p(e|h_i)p(h_i) + \sum_u p(e|h_i)p(h_i)} \tag{3.14}$$

Substituting for $\Sigma_u\, p(e|h_i)p(h_i)$ from (3.12):

$$p(h_a|e) = \frac{p(e|h_a)p(h_a)}{\sum_a p(e|h_i)p(h_i) + (\sum_a p(e|h_i)p(h_i))(1 - \sum_a p(h_i)\sum_a p(h_i))} \quad (3.15)$$

So,

$$p(h_a|e) = \frac{p(e|h_a)\sum_a p(h_i)}{\sum_a p(e|h_i)p(h_i)} p(h_a) \quad (3.16)$$

or Bel'(h_a) = w.Bel(h_a)

where,

$$w = \frac{p(e|h_a)\sum_a p(h_i)}{\sum_a p(e|h_i)p(h_i)} \quad (3.17)$$

Figure 3-5 shows the algorithm used to implement the updating function in (3.16). Operating separately on each hypothesis class, the algorithm first formed a list of all hypotheses that had a probability assignment in the knowledge base for the piece of evidence under consideration. The a priori probability of each hypothesis, $p(h_i)$, were then summed to S1, the probability of the evidence given each hypothesis, $p(e|h_i)$, were found and the products $p(e|h_i)p(h_i)$ were summed to S2. The updating factor in (3.17) was then calculated as $w_i = p(e|h_i)$.S1/S2 and an appropriate entry was then made at the SubHypothesis BB level.

For variable evidence, $p(e|h_i)$ was derived from:

$$p(e|h_i) = p(\text{low}|h_i)p(\text{low}|e) + p(\text{normal}|h_i)p(\text{normal}|e) + p(\text{high}|h_i)p(\text{high}|e) \quad (3.18)$$

where $p(\text{low}|e)$, $p(\text{normal}|e)$ and $p(\text{high}|e)$ were found at the ClassifiedData BB level, derived by (3.4) to (3.6), and $p(\text{low}|h_i)$, $p(\text{normal}|h_i)$ and $p(\text{high}|h_i)$ were stored in the knowledge base as in (3.3).

The a priori belief in each singleton hypothesis before any evidence had been impacted on the hierarchy was calculated by multiplying the a priori weights assigned to the nodes leading from the root to the hypothesis concerned. For example, assuming equal weights were assigned to the members of each set of hypotheses, i.e. no

prevalence information is used, the a priori weight for uncompensated respiratory alkalaemia in Figure 3-1 would be: 0.2500 x 0.3330 x 0.5000 = 0.0416.

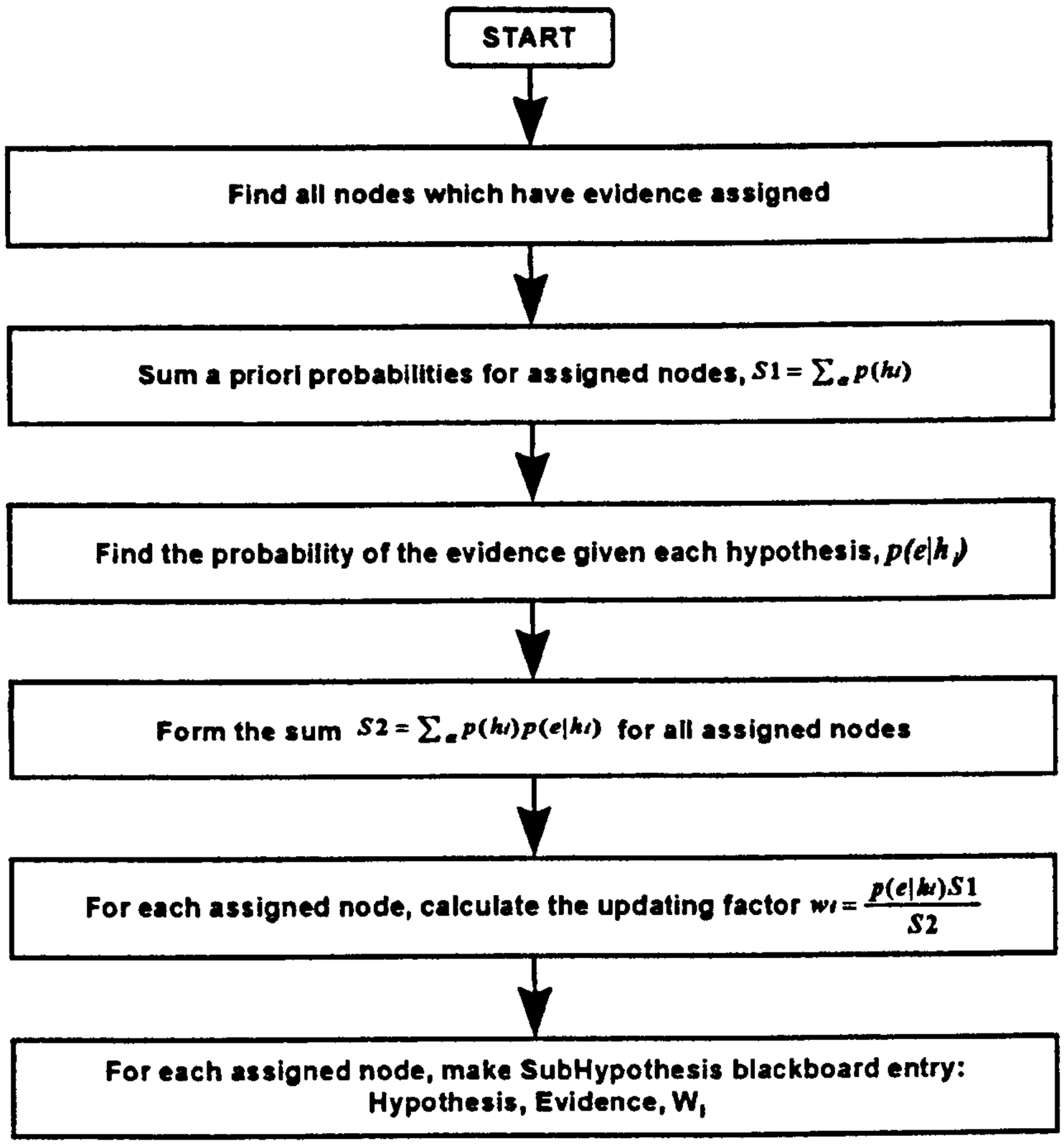


Figure 3-5. Algorithm for handling the impact of evidence on interpretative hypotheses in the KBS prototype (from Chelsom, 1990).

3.2.3.7. Evidence Aggregation and Propagation

The three KSs described in the previous section calculated the updating factors for the belief in the interpretative hypotheses generated by the evidence and wrote them to the SubHypothesis BB level. This is because the generated hypotheses may originate from any level in the domain hierarchy and there may be more than one entry for each hypothesis, that is, a piece of evidence may bear directly on more than one hypotheses. The SumHypotheses KS was triggered by such entries and wrote the

updated belief of leaf nodes in the hierarchy at the Hypothesis BB level. The algorithm used by SumHypotheses for the aggregation of evidence and the propagation of belief to the leaf nodes is shown in Figure 3-6.

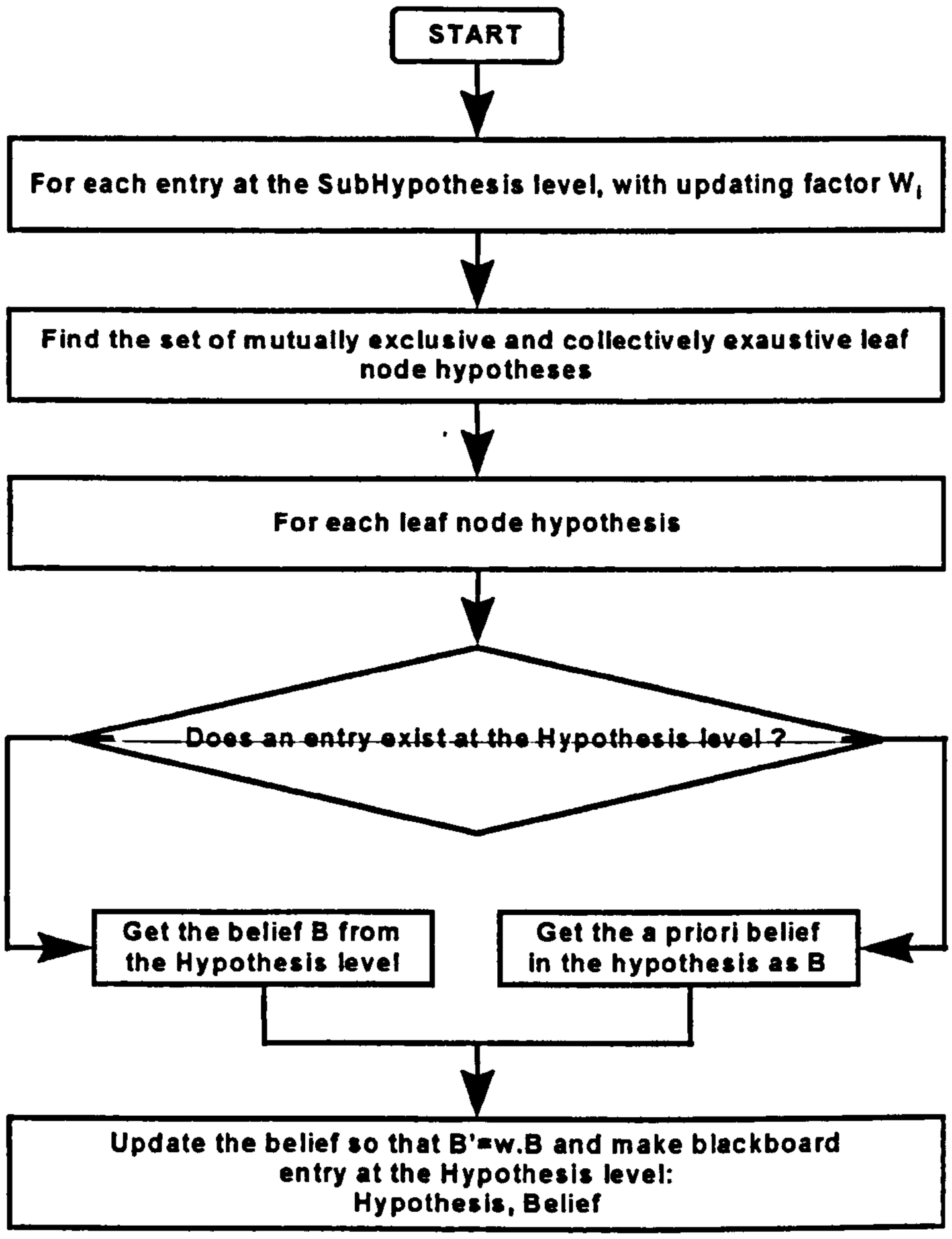


Figure 3-6. Algorithm for evidence aggregation and propagation in the KBS prototype (from Chelsom, 1990).

For each hypothesis at the SubHypothesis BB level, with associated updating factor W, the belief in each of its descendent leaf nodes was updated by W. If an entry for the leaf node already existed at the Hypothesis BB level, then it was the belief specified there that was updated (section 2.5.3). Otherwise, the a priori belief was updated and written to the Hypothesis level. This method of updating ensured that

however high its a priori probability might be, a hypothesis did not appear at the Hypothesis level until some evidence was observed that affected its belief.

The RankHypotheses KS was triggered by entries at the Hypothesis BB level and reported interpretative hypotheses in rank order at the SubDiagnosis level. RankHypotheses ensured that interpretations were reported at the most appropriate level of abstraction in the disorder class hierarchy. If each of the descendent leaf nodes of an hypothesis appeared at the Hypothesis level with belief increased from its a priori value, then they were combined to form a single interpretative hypothesis with belief equal to the sum of the beliefs of the leaf nodes, since the singleton descendants of each class of hypotheses are mutually exclusive and collectively exhaustive.

3.3. Summative Evaluation

3.3.1. Evaluation Method

Following a series of tests performed in order to evaluate the translation of the theoretical design into a working prototype, Bgas was retrospectively evaluated with 60 cases of disturbed acid-base metabolism, of which 51 were selected from records of patients in the ICU of the West Middlesex Hospital, London, and 9 were selected from the literature to include the more unusual cases. These 60 cases were transcribed to a standard format, which displayed the patient data and the list of the 17 interpretative decisions represented in the system, and presented to the expert responsible for the knowledge base, a group of three senior clinicians (senior registrars), and a group of three junior clinicians (senior house officers). For the groups of senior and junior clinicians, the evaluation was split into three sets of 20 cases and one set was given to each clinician. In this way, a representative diagnosis was obtained for each case from a senior and a junior clinician , but each participant was only required to diagnose 20 cases. The details of each case and the corresponding interpretations may be found in Chelsom (1990).

3.3.2. Evaluation Results

Chapter 2 described classification reasoning and its limitations in solving interpretative problems which involve complex disorders with interacting mechanisms. However fundamental to the development of medical knowledge-bases and interpretative decision support systems, classification reasoning on its own has been shown to be insufficient for the task of recognising multiple coexisting disorders with overlapping presentations. This expected performance was verified in the retrospective evaluation of the BGAS system, which failed to distinguish between the members of the two classes of complex disorders shown in Figure 3-1 based on measurement classification data alone (Table 3-1). This is because the sets of disorders $\{d_{11}, d_{12}, d_{13}\}$ and $\{d_{14}, d_{15}, d_{16}\}$ have no evidence assigned in terms of the fundamental definitional features. Table 3-3 provides a summary of the interpretative decisions made by the system and the human decision makers involved in the study.

INTERPRETATION	SYSTEM	EXPERT	SENIOR	JUNIOR
Uncomp metabolic acidosis	1	5	2	3
Part comp metabolic acidosis	6	5	9	12
Compensated metabolic acidosis	0	4	2	2
Metabolic acidosis	3	-	-	-
Uncomp metabolic alkalosis	4	5	5	2
Part comp metabolic alkalosis	2	2	2	1
Compensated metabolic alkalosis	0	0	0	1
Metabolic alkalosis	0	-	-	-
Uncomp respiratory alkalosis	6	5	5	10
Part comp respiratory alkalosis	3	4	6	2
Compensated respiratory alkalosis	4	4	3	4
Respiratory alkalosis	4	-	-	-
Uncomp respiratory acidosis	4	5	6	4
Part comp respiratory acidosis	1	1	1	2
Compensated respiratory acidosis	0	1	1	0
Respiratory acidosis	1	-	-	-
Resp acidosis & met alkalosis	1	0	0	2
Resp alkalosis & met alkalosis	3	0	0	3
Resp alkalosis & met acidosis	0	3	2	3
Resp acidosis & met acidosis	11	10	10	7
Normal blood gases	4	6	6	2
Low HCO_3^- with pCO_2	2	-	-	-
TOTAL	60	60	60	60

Table 3-3. Summary of interpretative decisions made by the KBS prototype and human decision makers in the domain of acid-base balance (from Chelsom, 1990).

Bgas attempted to solve the problem of recognising and classifying complex disorders using derived data as evidence. However, the evidence provided by such data contributes no more information than the three definitional features listed in Table 3-1. Another mechanism which could solve the problem was implemented in the form of CritiqueDiagnosis KS. CritiqueDiagnosis, the action of which is depicted in Figure 3-7, operated in the following manner.

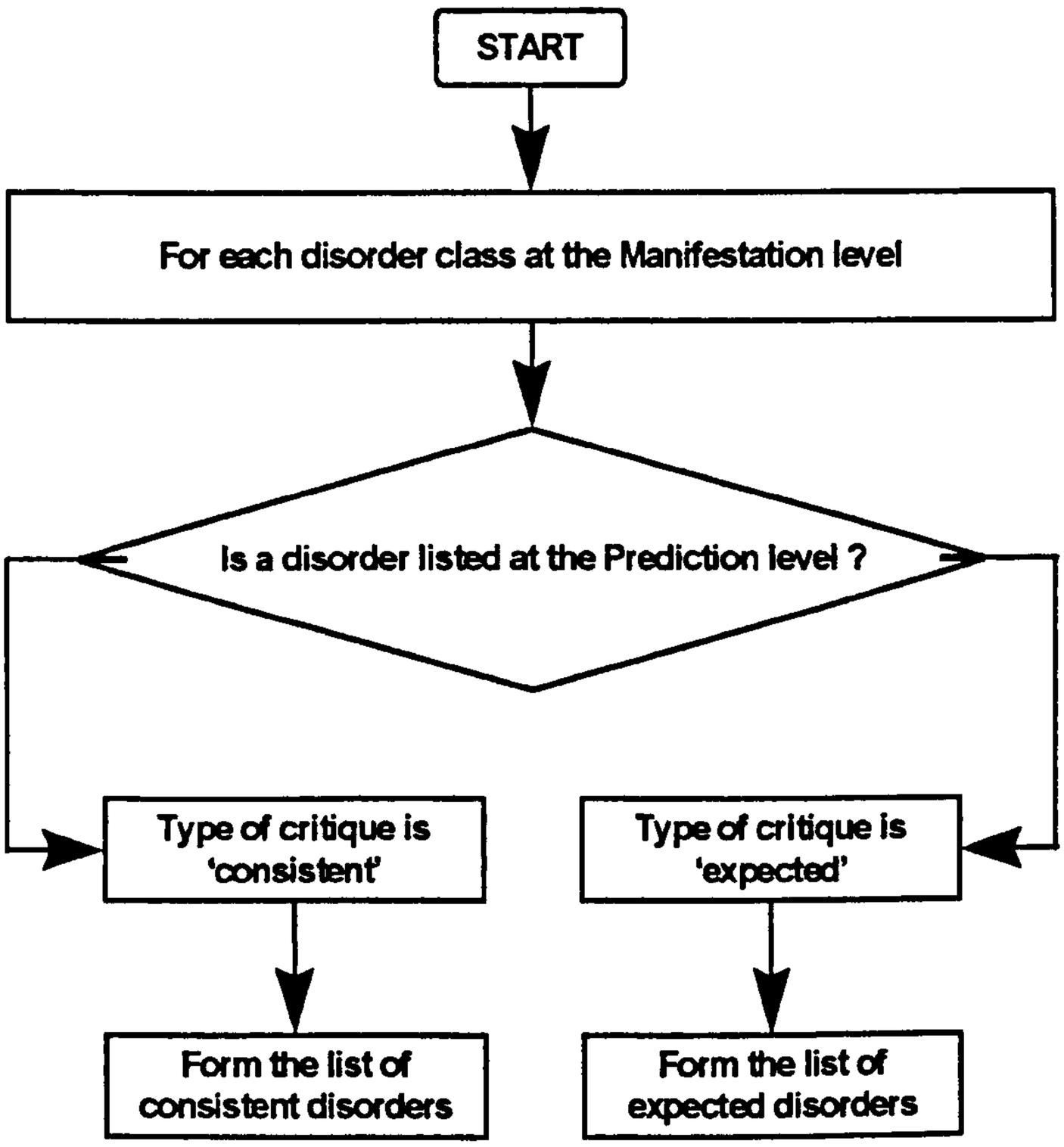


Figure 3-7. Action of the CritiqueDiagnosis Knowledge Source (from Chelsom, 1990).

The aetiology of the patient's underlying clinical condition was entered into the system and the WriteDiseaseDiagnosis KS made an appropriate entry at the Diagnosis BB level. Furthermore, the TransferData KS transferred the final data interpretations from the SubDiagnosis level to the Manifestations level of the clinical diagnosis panel. The PredictDisorders KS searched the knowledge base for any disorders that were

associated with the patient's disease(s) and wrote them at the Prediction BB level (Figure 3-3). The CritiqueDiagnoses KS then compared the Prediction and Manifestations levels, and produced a list of consistent disorders and a list of expected disorders which were not found at the Manifestations level. Thus, in the example of Figure 3-8, the complex interpretative hypothesis of high pCO_2 and [HCO_3^-], representing the combined states of euphaemia, hypercapnia and hyperbicarbonataemia, could be refined given the information that the patient is accidentally hypoventilated (high pCO_2) and is being administered diuretic therapy for an underlying cardiac, renal or hepatic disorder. The first clinical feature causes an expected respiratory acidosis, whereas the second causes an expected metabolic alkalosis. By projecting the interpretative hypothesis of hyperdynamic compensation onto this clinical context, the system could select the case of combined respiratory acidosis and metabolic alkalosis from the set $\{d_{11}, d_{12}, d_{13}\}$. However, this performance was not evaluated in the KBS prototype.

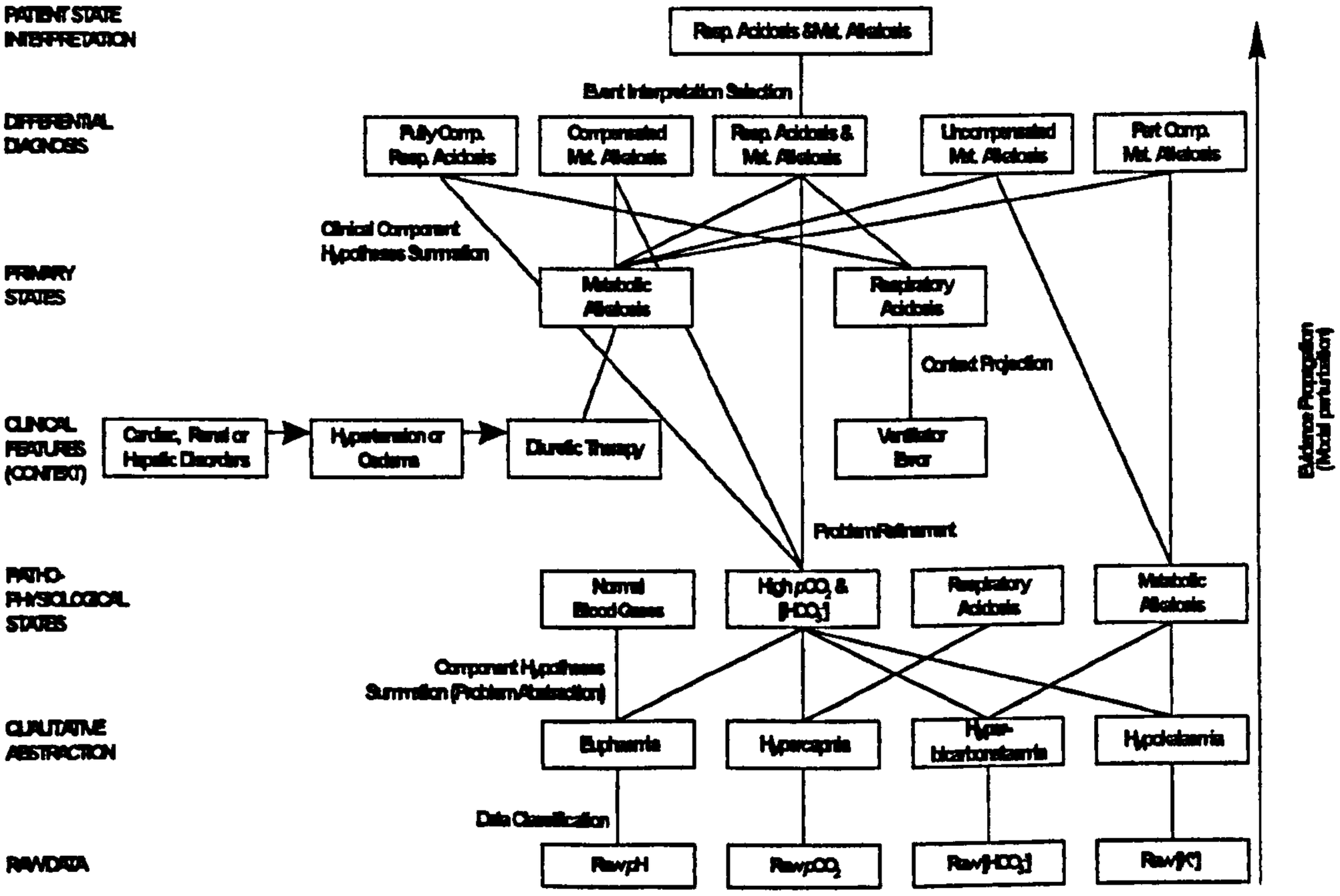


Figure 3-8. The use of clinical features to refine a complex interpretative hypothesis in the form of causal-process classification.

The summative evaluation study, also as expected, showed that overall there was considerable disagreement between the human decision makers, even between the expert and the senior clinicians. More specifically, the system was in agreement with the expert clinician in 55% of the cases, and 53% of the cases with the senior clinicians, where the agreement between the expert and the senior clinicians was only 55%. Thus, the KBS prototype performed at the level of the expert who designed the knowledge-base. Table 3-4 shows the agreement between the decision makers.

	EXPERT	SENIOR	JUNIOR	SYSTEM
EXPERT		33	24	33 (39)
SENIOR	55 %		23	32 (38)
JUNIOR	40 %	37 %		26 (35)
SYSTEM	**55%** (65%)	**53%** (63%)	**43%** (58%)	

Table 3-4. Agreement on interpretation (from Chelsom, 1990). The figures in parentheses include cases in which the clinician's interpretation was subsumed by the less specific interpretation in the system (see Section 3.2.2.7).

These results are further analysed in Table 3-5 and Table 3-6, which display the performance of the system broken down by diagnosis. In cases where the system made an interpretation at an intermediate level of the hypothesis hierarchy which subsumed the clinician's diagnosis, a false +ve was recorded. When it did not subsume the clinician's interpretation, a false +ve was recorded for each of the descendent leaf node interpretations.

DISORDER	CASES	TRUE +ve	FALSE -ve	TRUE -ve	FALSE +ve	sensitivity	specificity	diagnostic index
Uncomp met acid	5	2	3	54	1	0,40	0,98	1,38
Part comp met acid	5	4	1	52	3	0,80	0,95	1,75
Comp met acid	4	0	4	54	2	0,00	0,96	0,96
Uncomp met alk	5	3	2	54	1	0,60	0,98	1,58
Part comp met alk	2	2	0	58	0	1,00	1,00	2,00
Uncomp resp alk	5	5	0	51	4	1,00	0,93	1,93
Part comp resp alk	4	2	2	51	5	0,50	0,91	1,41
Comp resp alk	4	2	2	52	4	0,50	0,93	1,43
Uncomp resp acid	5	5	0	55	0	1,00	1,00	2,00
Part comp resp acid	1	0	1	58	1	0,00	0,98	0,98
Comp resp acid	1	0	1	59	0	0,00	1,00	1,00
Resp alk & met acid	3	0	1	57	2	0,00	0,96	0,96
Resp acid & met acid	10	9	1	48	2	0,90	0,96	1,86
Normal blood gases	6	4	2	54	0	0,66	1,00	1,66

Table 3-5. Accuracy with development expert as gold standard (from Chelsom, 1990).

DISORDER	CASES	TRUE +ve	FALSE -ve	TRUE -ve	FALSE +ve	sensitivity	specificity	diagnostic index
Uncomp met acid	2	2	1	57	1	1,00	0,98	1,98
Part comp met acid	9	5	5	48	3	0,55	0,94	1,49
Comp met acid	2	1	2	57	1	0,50	0,98	1,48
Uncomp met alk	5	2	3	53	2	0,40	0,96	1,36
Part comp met alk	2	1	1	57	1	0,50	0,98	1,48
Uncomp resp alk	5	3	4	50	5	0,60	0,91	1,51
Part comp resp alk	6	3	3	52	2	0,50	0,96	1,46
Comp resp alk	3	1	2	54	3	0,33	0,95	1,28
Uncomp resp acid	6	5	2	54	0	0,83	1,00	1,83
Part comp resp acid	1	1	0	59	0	1,00	1,00	2,00
Comp resp acid	1	0	1	59	0	0,00	1,00	1,00
Resp alk & met acid	2	0	2	57	1	0,00	0,98	0,98
Resp acid & met acid	10	10	0	49	1	1,00	0,98	1,98
Normal blood gases	6	4	2	54	0	0,66	1,00	1,66

Table 3-6. Accuracy with senior clinician as gold standard (from Chelsom, 1990).

The only significant difference between the pattern of interpretations made by the expert and senior clinicians is that senior clinicians tended to make more diagnoses of partially compensated metabolic acidosis than the expert, whose interpretations of metabolic acidosis were split fairly evenly between compensated, partially compensated and uncompensated.

Overall, not counting the cases of complex disorders, the evaluation study showed that BGAS was in agreement with either the expert or senior clinician involved in the study in 44 of the 53 cases (83%). Nonetheless, as discussed above, these results are on their own insufficient as criteria for the decision whether to admit the system or reject it from further evaluation.

3.4. Conclusion

This chapter described an application of the method described in section 2.5.3 for belief propagation in a tree structured hierarchical belief network, to interpretative decision-making in the domain of acid-base metabolism. The evaluation study summarised in section 3.3, indicated that the KBS prototype was in agreement with the expert or senior clinicians involved in the study in 83% of the cases. However, the prototype failed to recognise complex disorders, that is, disorders which are fully compensated or accompanied by a second disorder of another organ system involved

in acid-base homeostasis. The prototype included a version of the method of causal-process classification described in section 2.4.2, to support its interpretative decisions when presented with such cases, however this performance was not evaluated.

Nonetheless, it was assumed that the system's performance could be improved by training the belief network, given data in sufficient quantity and quality, which was not possible since the prototype was not allowed in routine use for the reasons described in chapter 1. Furthermore, the integration of the KBS prototype into the clinical information processing activity of the user, would facilitate an analysis of the activity involved in recognising the presence of complex disorders and the development of an extended task domain to perform this task. The chapter which follows describes the development of a clinical information management environment, designed to support the integration of the KBS prototype into the routine information processing activity of the clinician-user, by combining the acquisition, organisation, storage, update and review of the knowledge and patient data required for interpretative decision support in the domain of monitoring the ICU patient with disturbed acid-base homeostasis.

Object-Oriented Design of an Intelligent ICU Clinical Information Management Support System

4.1. Introduction

Chapter 1 described how albeit having demonstrated an ability to perform at or near the level of human decision-makers, the overwhelming majority of medical KBS prototypes have failed to provide effective solutions to key medical decision support problems. More specifically, the chapter discussed the methodological inadequacies in the evaluation and dissemination of knowledge-based and other clinical opinion and decision support tools, the criteria by which the introduction of support systems into routine clinical practice is assessed and decided, and the necessity for a well designed approach to the integration of pertinent tools within the users' routine information processing activity. Following this background material, a subsequent analysis of the clinical decision-making task and methods for reasoning with partial belief and incomplete information in Chapter 2, and a knowledge description of the KBS prototype which was to be integrated within routine clinical practice in Chapter 3, this chapter describes an incremental, object-oriented approach to the functional integration of the KBS prototype, geared toward the solution of the problems encountered in the combined management of the clinical information generated and utilised in the process of monitoring and supporting the ICU patient with acid-base balance disorders.

4.2. Functional Integration Requirements

Figure 4-1 depicts the process of formative and constructive assessment by which, as discussed in Chapter 1, the prototype KBS described in the last chapter should integrated into the cognitive information processing activity observed in the ICU in order to deliver effective solutions to actual medical decision support problems. The indicates the incremental, iterative nature of the approach and its role in guiding the integration development process. More specifically, as discussed in Chapter 1, the figure shows that beta-testing focuses on assessing the compatibility of the integrated system with the functional requirements elicited during the phase of exploration,

which are discussed in this section. The object-oriented system development and integration techniques and methods employed in the process are described in the section which follows.

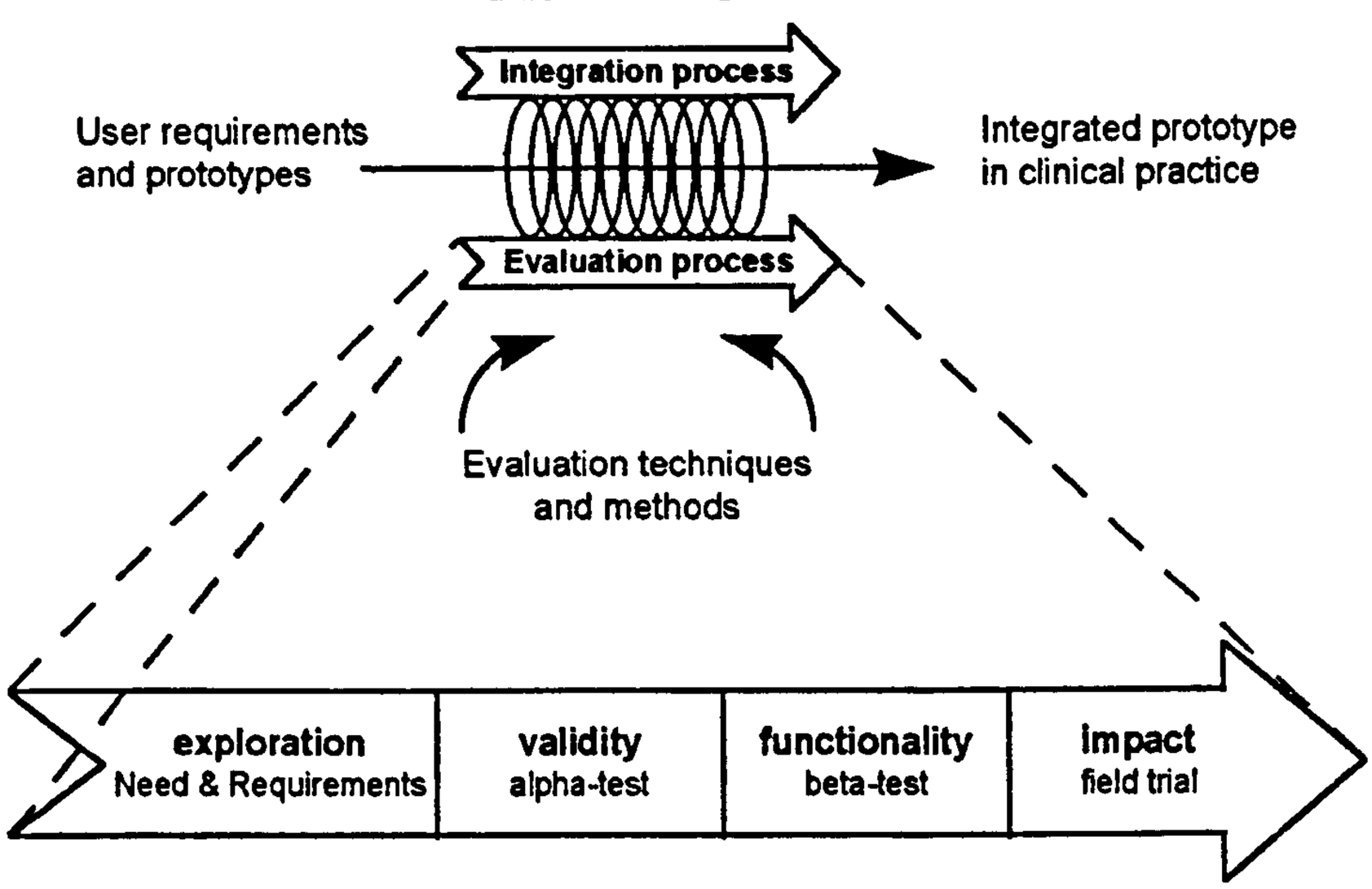


Figure 4-1. Constructive assessment in the clinical decision-support system integration-development process (from Brender et al, 1995).

4.2.1. The Critical Care Environment

Half way through the 18th century, Florence Nightingale expressed the need for a place in the hospital where post-operative and other patients needing close attention could be watched. Almost a century later, an American Anaesthesia Study Commission concluded that one-third of post-operative deaths in the first 24-hours of recovery could have been prevented by better nursing care (Jennett, 1986). By 1960, ICUs were still almost unknown in the US, yet by 1979 there were 55,000 ICU beds

(Jennett, 1986). By the mid-1980s this figure had risen to 75,000 beds (Gardner, 1990).

Thus, the purpose of the ICU is to reduce mortality and morbidity and they are used for the management of several quite distinct kinds of hospitalised patients who are in a critical condition (Jennett, 1986; Gardner, 1990). To manage these patients, an ICU uses a range of high technology measurement instrumentation and requires specialised staff. Consequentially, an ICU has a high cost and limited availability. In the mid-1980s, in US hospitals ICU beds accounted for as many as 15% of the total number of beds. In the UK the proportion at the time was 1%. It was estimated that more than half the difference between the cost of hospital care in the two countries derived from the difference in the provision of intensive care (Jennett, 1986).

Since, by contrast to the activity observed in other hospital units, the majority of what goes on in an ICU is intensive nursing care, the majority of the technology encountered there is for patient monitoring and life support, not therapy (Jennett, 1986). Patient monitoring is defined as "the repeated and continuous observation or measurement of the patient parameters, his or her physiological function, and the function of life support equipment, for the purpose of guiding management decisions, including when to make therapeutic interventions, and assessment of those interventions" (Hudson, 1985). Changes in vital patient parameters can indicate the progression of a condition, response to therapy, development of complications, and the need for further action. Thus, patient monitoring is regarded as a subset of clinical investigation, contributing not only to initial diagnosis but also to prognosis and management. The aim of monitoring the ICU patient is to detect the advent of significant pathophysiological events and prevent their consequences by timely intervention, thereby buying time until an underlying disease process can be reversed. In this context, monitoring is prevention on a short time scale, preventing complications rather than diseases, but contributing significantly to the reduction of avoidable mortality and morbidity (Jennett, 1986).

In the process of monitoring the ICU patient, clinical information indicating the patient's state must be collected from a multitude of sources, integrated, recorded, and interpreted within the clinical context to provide interpretations appropriate to the

context. Finally, patient data along with their contextual interpretations must be made available to the clinician decision-maker in a timely fashion in order to effect appropriate patient-state control decisions. Clinical information may be acquired from vital-sign monitoring equipment (ie. temperature, respiratory rate, heart rate, and arterial blood pressure monitors), life-support and therapeutic equipment (ie. ventilators, dialysis machines and drug infusion pumps), nurse observations, and the clinical laboratory, including the rapidly evolving class of off-site point-of-care clinical laboratory instrumentation, for example blood-gas analysers and blood oxymeters. Following acquisition, the available information forms the input to patient management decision-making processes involving the prescription and adjustment of ventilation, drug infusion, and nutrition regimes, to ensure maintenance of body fluid volume, circulatory stability and metabolic homeostasis, together with optimisation of oxygen delivery (Carson, 1989).

4.2.2. The Need for Decision Support in the Data-Overloaded ICU

As new high technology medical procedures become available for routine use, the number of management alternatives increases and complex integration of information is implicit in many patient management decisions (Speicher and Smith, 1983). For example, advanced life-support equipment may have many controls that must be regulated in co-ordination with other interventions. As a consequence, the management strategies adopted in current critical care clinical practice are rarely optimal. The difficulty faced by the clinician is that of integrating and interpreting a range of monitored variables which span the respiratory, circulatory, and metabolic systems of the patient in the context of the indicated higher level processes (Carson et al, 1988; Blois, 1988), and because of the resultant complexity of monitoring and controlling the ICU patient, even skilled clinicians may make errors that limit the quality of care and which may cause life-threatening occurrences. Thus, although the availability of patient data from the available sources aims to improve the quality of care, the large number of variables and frequency of measurement can overwhelm clinicians (Blois, 1984; Fallat and Osborn, 1984; Price and Mason, 1986).

At the beginning of 1990s, a study published in the Lancet indicated that 80% of preventable critical incidents in a British ICU were found to be due to human error, while only 20% could be ascribed to equipment failure (Wright et al, 1991). Transcription and execution errors were found to be significant factors contributing to preventable mishaps in the unit, however, errors frequently also occurred in the interpretation of data. The factors felt to contribute to the detection of incidents included increased vigilance, presence of alarms on equipment and the presence of experienced staff. Failing to suspect more than one cause was one of the most common errors that lead to the misinterpretation of an observed clinical problem. For instance, in relation to laboratory medicine and particularly in the context of managing the patient with disturbed acid-base homeostasis, although the chemical pathology of the disorders is well understood by experienced clinicians, the correct interpretation of the clinical data is subject to individual success rates as low as 20% (Schreck et al, 1986).

4.2.3. Required ICU Decision Support Functions

In order to assist decision-making in the data-overloaded, high-technology critical care environment described above, IMC systems such as the proof-of-concept system described in Section 2.4.5 should perform the following decision support functions:

1. Summarise the patient's progress and condition for clinicians and physicians on rounds.
2. Alert clinicians to imminent problems before they might otherwise be noticed.
3. Suggest, critique and possibly execute alternative therapies.

Figure 4-2 depicts the tasks required for the execution of the above described decision support functions required to assist in monitoring and supporting the ICU patient. Interpretative inference is inference in the effect-to-cause direction of causality and was extensively described in Chapter 2. Predictive inference is inference in the cause-to-effect direction, leading from the set of interpretative hypotheses to plausible explanations of the data available in a case, or to corresponding disease prognoses. The latter refer to a set of plausible predictive inferences of outcome expressed in

terms of expected consequences, complications and overall disease progression, with or without intervention. By contrast to explanatory predictive inferences which are used for the static justification of interpretative hypotheses, as shown in Figure 2-4 for the case of consultation systems, temporal prognostic predictive inferences are used primarily for treatment planning and management, including evaluation and optimisation of patient-state control plans and predictive alarming (Uckun, 1994).

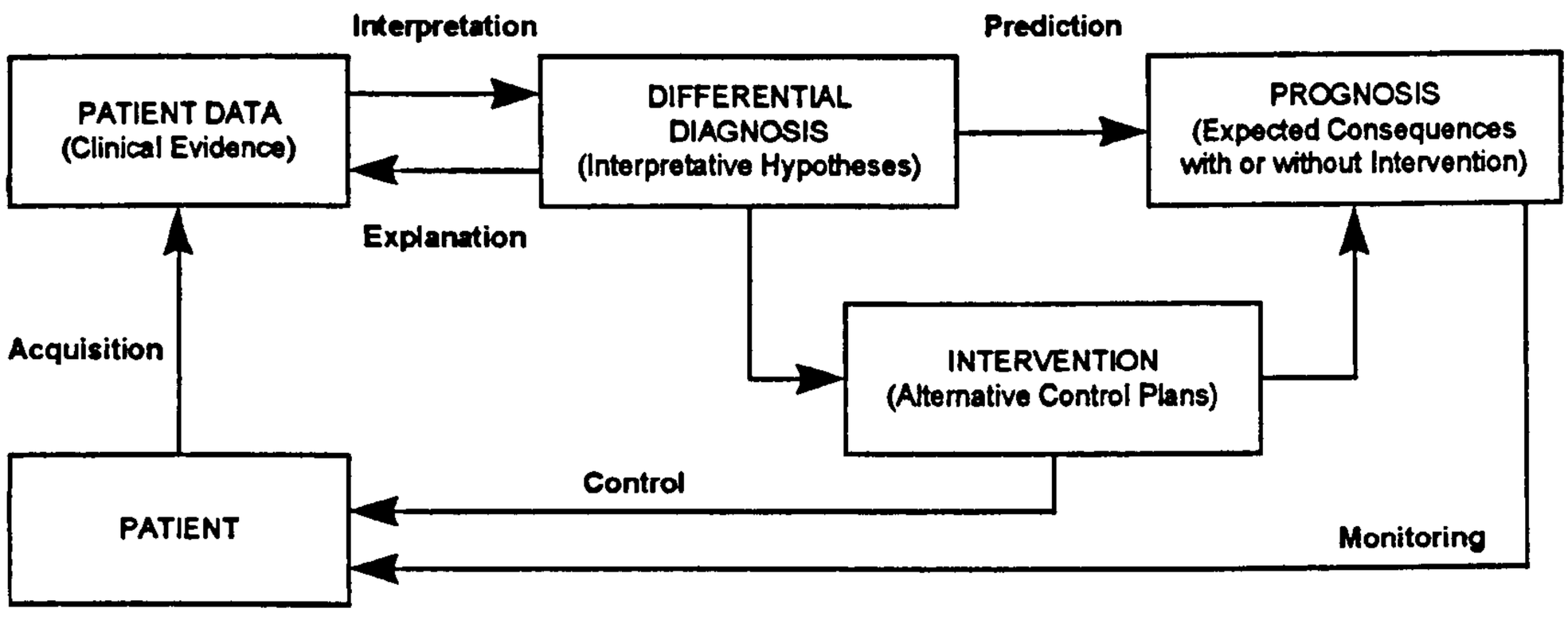


Figure 4-2. Clinical activity observed in the process of monitoring and supporting the ICU patient.

There are two ways in which to support patient-state control plans: open-loop and closed loop (Carson, 1989; Uckun, 1994). Open-loop control or execution monitoring involves the temporal confirmation of constructed PSMs from continued observation of patient state, primarily to assist in planning and management. Closed-loop control means a system is able to execute control plans directly, for example, by modifying ventilator settings and drip infusion rates. For ethical and legal reasons, there are very few clinical systems that operate in closed-loop mode either in clinical use or under evaluation (Uckun, 1994). An interesting approach to ICU patient management is the use of the critiquing model to generate action-oriented alarms (van der Lei et al, 1990; van der Lei and Musen, 1991). Action-oriented alarms or suggestions such as changing a drip rate of a drug, changing the rate of volume of ventilator, and detecting adverse drug interactions, are considered the ultimate in critical care

decision support, as at this level, change in medical care can best be initiated and consequentially, evaluation of the assistance offered by the computer can be assessed.

Finally, in order for these tasks to performed in the context of IMC as shown in Figure 4-2 and described in Section 2.4.5, support systems must provide the means for communication with environment, including facilities for the acquisition, temporal organisation, maintenance, update and review of the information generated in the process of monitoring the ICU patient.

4.3. Elements of the Object-Oriented Development Methodology

The increase in the modelling complexity that computer programs have been dealing with, as manifested in the medical AI research described in this thesis and the development of generic task-specific KBS architectures (Clancey, 1992), such as the one described in Section 2.4.5, has prompted a significant amount of applied research in software engineering, particularly with regard to decomposition, abstraction, encapsulation and hierarchy (Factor et al, 1989, 1991; Stefanelli, 1993). The development of increasingly more expressive and modular programming languages has complemented these advances. Overall, the trend has been a move away from imperative languages and algorithms that tell the computer what to do, toward declarative languages that describe the key abstractions in the problem domain (Booch, 1991; Graham, 1994). This trend of abstraction-object-orientation corresponds to the paradigm shift away from the underlying machine and closer to the problem space (Newell, 1982).

Object-oriented software engineering provides methods for the analysis, design, and implementation of computer systems, based on the principles of abstraction, encapsulation, modularity, hierarchy, typing, concurrency, and persistence, which are described below in this section. What makes object-orientation particularly suited to the implementation of complex integrated systems is that the methodology brings these elements together into an incremental, unified decomposition, representation

and implementation framework for modelling complex systems, with a structure-preserving transformation of concepts to maintainable implementations (Booch, 1991). Such an application is described in the section which follows.

4.3.1. Abstraction

Object-oriented systems are organised as co-operative collections of objects, each of which represents an instance of some class of objects, which in turn corresponds to some problem domain abstraction and which is a member of a hierarchy of classes united via inheritance relationships.

Once instantiated, objects exist for some time, during which time they can act on other objects and be acted upon by other objects, thereby be changed, shared and destroyed. Thus, objects encompass two types of abstraction: entity abstractions and action abstractions. These correspond, respectively, to an object that represents a useful model of a problem domain entity and to an object that provides generalised sets of operations, all of which perform the same kind of function. In other words, objects have a structure and thus a state, a behaviour, and an identity. The structure and behaviour of similar objects are defined in their common class. The identity of an object is defined upon the creation of a class instance.

The fact that every object has a state means that it takes up some space, be it in the physical world or in computer memory; i.e. all objects within a system encapsulate some state, and all of the state within a system is encapsulated by objects. The state of an object encompasses all of the static properties of the object, inherited from its class, plus the current dynamic values of each of these properties. The behaviour of an object is defined by its actions, or how an object acts and reacts, in terms of its state changes and message passing. Any object that uses the resources or expertise of another object is termed a client. For a given class there are usually two kinds of clients: objects that invoke operations upon instances of the class by message passing, and subclasses that inherit from the class. In the latter case, objects are related by structural relationships, which are described below, whereas in the former by using relationships. There are three types of uses for an object:

1. Actor. An object is an actor when it can operate upon other objects but is never operated upon by other objects.

2. Server. An object is a server when it never operates upon other objects but is only operated upon by other objects.

3. Agent. An object is an agent when it can both operate upon other objects and be operated upon by other objects. An agent is usually created to do some work on behalf of an actor or another agent.

4.3.2. Encapsulation

In order that an abstraction contributes effectively to the decomposition of a problem domain, its implementation details must be encapsulated so that no part of a complex system should depend on the internal details of another part. Thus abstraction and encapsulation are complementary concepts. Abstraction focuses upon the outside view of an object, and encapsulation, also known as the principle of information hiding, prevents clients from seeing the inside view. In practice, this means that each class must have two parts. These are the interface to the class and the implementation of the class. The interface of a class captures only its outside view, encompassing the designer's abstraction of the behaviour common to all instances of the class. The implementation of a class comprises the representation of the abstraction as well as the mechanisms that achieve the desired behaviour. Thus, encapsulation is the process of hiding all of the details of an object that do not contribute to its essential characteristics.

The abstraction of an object should precede the decisions about its implementation. Once an implementation is selected, it should be treated as a secret of the abstraction and hidden from the object's clients. The interface of a class is the one place where the designer asserts all of the assumptions that a client may make about any instances of the class. The implementation encapsulates details about which no client may make assumptions. It is through this property of encapsulation that object-orientation allows the development of open systems. Another benefit is the ability to change the representation of an abstraction without disturbing any of its clients, whether internal

or external. Thus, object-oriented systems are maintainable and can evolve through their use and re-use.

In object-oriented languages, operations that clients may perform upon an object are typically declared as methods, which are part of the interface and the implementation of the class. C++ uses the term member function to denote this concept. An object's behaviour is determined by the implementation of its class. Furthermore, the access to an object's behaviour is determined by the level of the member method's declaration in the class' interface. In C++ there are three levels of encapsulation:

1. Private. A member of an object (method or attribute) is declared private if it is meant to be available to the class's own methods and cannot be accessed by other objects, but only those created from that class.

2. Protected. A member of an object is declared protected when it is not visible to any other classes except its subclasses.

3. Public. A member is declared public if it is meant to be available to any other object.

4.3.3. Modularity

A software architecture comprises a logical as well as a physical structure. In the same way that abstractions, classes and objects, provide a means of decomposing the problem space and form the logical structure or architecture of a system, program modules provide a means for decomposing the system's physical architecture. Modularisation consists of dividing a program into modules which can be compiled separately, but which have connections with other modules. As with abstractions, the connections between the modules are the assumptions which the modules make about each other. As with classes of objects, most languages that support the program module as a separate concept also distinguish between the interface of a module and its implementation. Again, program modules communicate through their interfaces. One program module does not have to now the implementation of the other. Thus modularity and encapsulation are nested levels of abstraction.

In complex applications the use of modules is essential for managing complexity. In production languages and systems rules play a fundamental role in the modularity of the KBS (Section 2.3.3). For example, modules may comprise top-level goals and one module needs to only know the top-level goal of the module, rather than its whole logical structure. In object-oriented C, for example in C++, modules are connected by declaring in one module the file containing the interface to another module, or the header file. The overall goal of the decomposition into modules is the reduction of development effort by allowing modules to be designed and revised independently. However, it is often the case that at the beginning of the design stage modules are not clear distinctions and they are only formed after the key abstractions of the represented domain have been identified. Modularity is the property of a system that has been decomposed into a set of cohesive and loosely coupled modules. Cohesive means grouping logically related abstractions and loosely coupled means there are minimal dependencies among modules. Furthermore, modularisation should make a system reusable. For example, generic tasks are reusable. In fact, abstraction, encapsulation and modularisation, are the key advantage of the blackboard architecture.

4.3.4. Hierarchy

Abstraction, encapsulation, and modularity are three ways in which to deal with complexity in software systems. As discussed in Chapter 2, hierarchy is the backbone of structural organisation because it provides a ranking or ordering of abstractions in such a way that greatly facilitates the process and effectiveness of abstraction. The two most important hierarchies in a complex system are its class structure, which takes the form of a kind-of hierarchy, and its object structure, which takes the form of a part-of hierarchy.

Inheritance is the most important kind-of hierarchy. As stated above, inheritance defines a relationship among classes, wherein one class shares the structure or behaviour defined in one or more classes. A one-to-many relationship between classes is termed single inheritance, which is the defining characteristic of tree structured hierarchies such as the one shown in Figure 3-1. A many-to-one relationship is termed

multiple inheritance. Whereas kind-of hierarchies denote generalisation-specialisation relationships, part-of hierarchies describe decomposition-aggregation relationships. Inheritance relationships are essentially class relationships and aggregation, or composition, relationships are essentially object relationships. Aggregation or composition hierarchies are said to be defined on multiple levels of abstraction. The solution blackboard, or the short term memory, is essentially an aggregation hierarchy which also reflects the original kind-of decomposition of the problem domain. Operators decompose a problem by posting solutions on such a hierarchy, the aggregation of which should reveal the final solution.

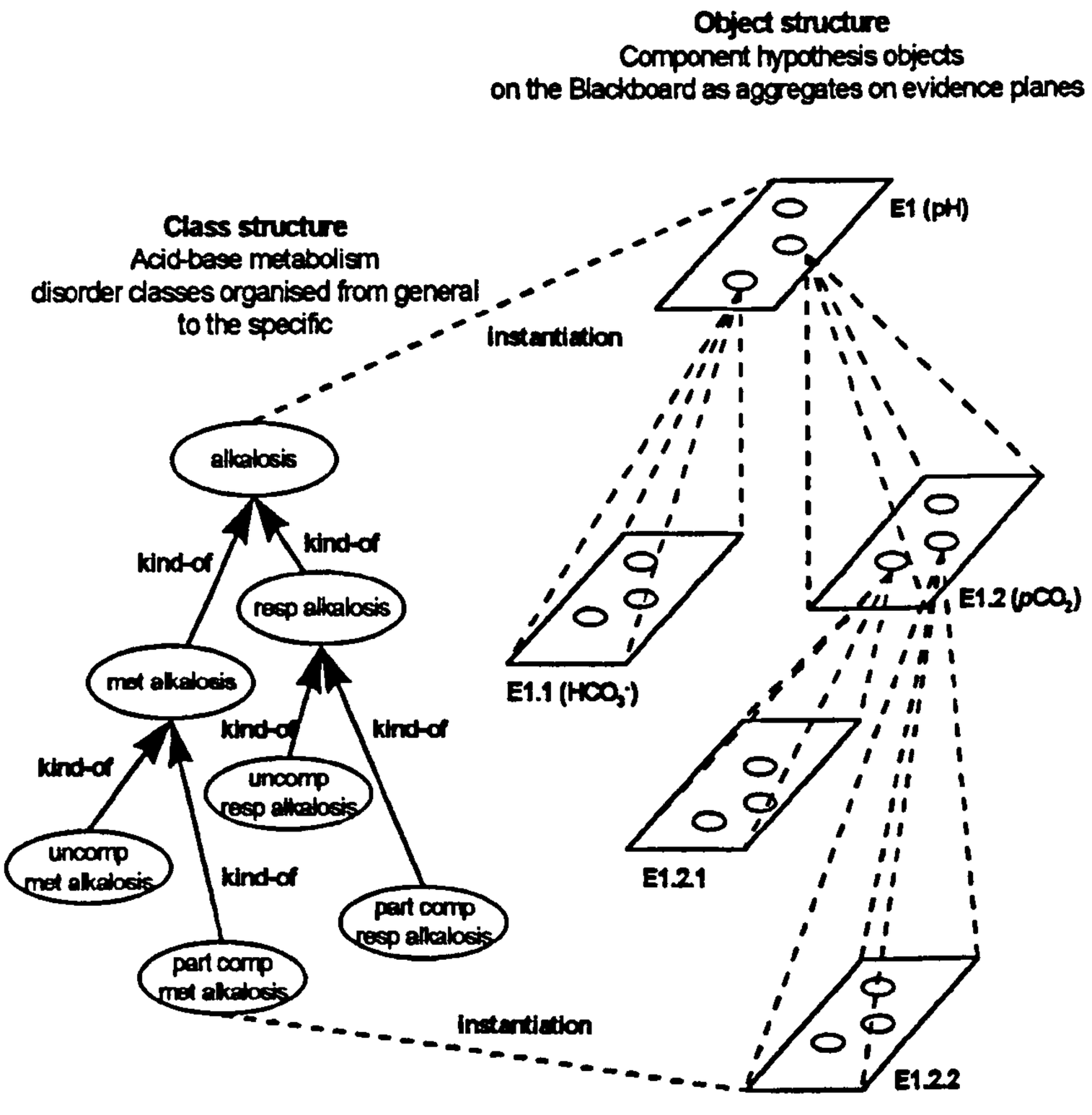


Figure 4-3. The organisation of classes and objects as orthogonal structures (adapted from Booch, 1991)

The organisation of classes and objects as orthogonal structures is depicted in Figure 4-3. The diagram depicts a hierarchy of disorder hypotheses, represented as classes $\{D_1, D_2, \ldots, D_7\}$ united via inheritance or kind-of relationships, as for example in

Figure 3-1, and an orthogonal PSM hierarchy of object aggregates {E1, E1.1, ..., E1.2.2}, representing evidence planes on a PSM blackboard, as for example in Figure 3-3, which may be instantiated by message passing between interpretative classes or by a separate blackboard-controlled reasoning module (Figure 3-2). A multilevel object aggregation hierarchy formed by the instantiation of classes continues to reflect the class structure only the inheritance relationships are implicit and weaker than aggregation relationships.

4.3.5. Persistence

A software object of some type, once created, takes up a certain amount of space and exists for a particular amount of time, until it is destroyed. There is a range of object persistence, ranging from transitory objects that arise within the evaluation of a simple expression or statement, to objects in a database that outlive the execution of a single program. The spectrum of object persistence encompasses the following (Booch, 1991):

1. Transient results in expression or statement evaluation.
2. Local variables in procedure activation.
3. Global variables and heap items whose extent is different from their scope.
4. Data that exist between executions of a program.
5. Data that exist between various versions of a program.
6. Data that outlive the program.

Traditional programming languages usually address only the first three kinds of object persistence. Persistence of the last three kinds is typically the domain of database technology. Introducing the concept of persistence to the object model, gives rise to object-oriented databases, and enables the development of integrated systems with superior performance in data handling both in terms of representational as well as reasoning power and efficiency (Graham, 1994). In practice, such databases build upon proven technology, such as sequential, indexed, hierarchical, network, or relational data bases or knowledge bases, but then offer to the programmer the abstraction of an object-oriented interface, through which database queries and other

operations, such as KBS reasoning, are completed in terms of objects whose lifetime transcends the lifetime of an individual program. Furthermore, object persistence deals with more than just the lifetime of data. In object-oriented databases, not only does the state of an object persist, but its class must also transcend any individual program so that every program interprets this saved state in the same way.

4.4. Top-Level View of the Integrated System

The ICIMS system was developed using the above described methodological elements of object-oriented design and implementation, in order to adopt a layered, incremental development approach to the integration of the KBS prototype, in accordance to the functional requirements specified in Section 4.2. This process, which is depicted in Figure 4-1, started with the design of an object handling system and proceeded upwards and closer to the user, guided by the constructive assessment of a clinical advisor, to eventually include the cognitive task-domain model of the KBS prototype, which was described in Chapter 3, and ergonomic interfaces for the management of the clinical information acquired and generated in the process of monitoring the ICU patient.

ICIMS was developed within the Borland object-oriented C++ environment, using the standard Windows Application Programmers Interface (API) for the construction of the users' access interfaces. C++ is an object-oriented software development environment, however, it does not in itself provide mechanisms for creating and handling persistent objects, that is, as described above, objects whose class instantiation inheritance, class structure inheritance, and state is saved, and transcends the lifetime of an individual program, thereby providing the ability to create and manipulate persistent world models. This property of object-oriented system development was added by means of the POET pre-compiler (Persistent Objects Extended Database Technology), which reads class interfaces and creates persistent objects from classes and class structures, or models, which are declared persistent.

Figure 4-4 presents a top-level view of the layered, modular ICIMS system architecture. The clinical object base (COB) module shown in the middle, forms the kernel of the domain abstraction and support system integration process, by functioning as a global memory of persistent model-derived objects. There are three types of persistent model or persistent class structures in the system. The patient record model (PRM) class structure was designed for the derivation of the persistent objects required to support the management of the clinical information generated in the process of monitoring the ICU patient. The domain knowledge model (DKM) class structure was designed for the derivation of the persistent objects required to support the management of the knowledge-base utilised in the interpretation of the acquired clinical information. Finally, the patient-specific model (PSM) class structure was designed for the derivation of the blackboard objects required to support the application of the task-domain model (TDM) for evidence propagation in the hierarchical belief network contained in the COB.

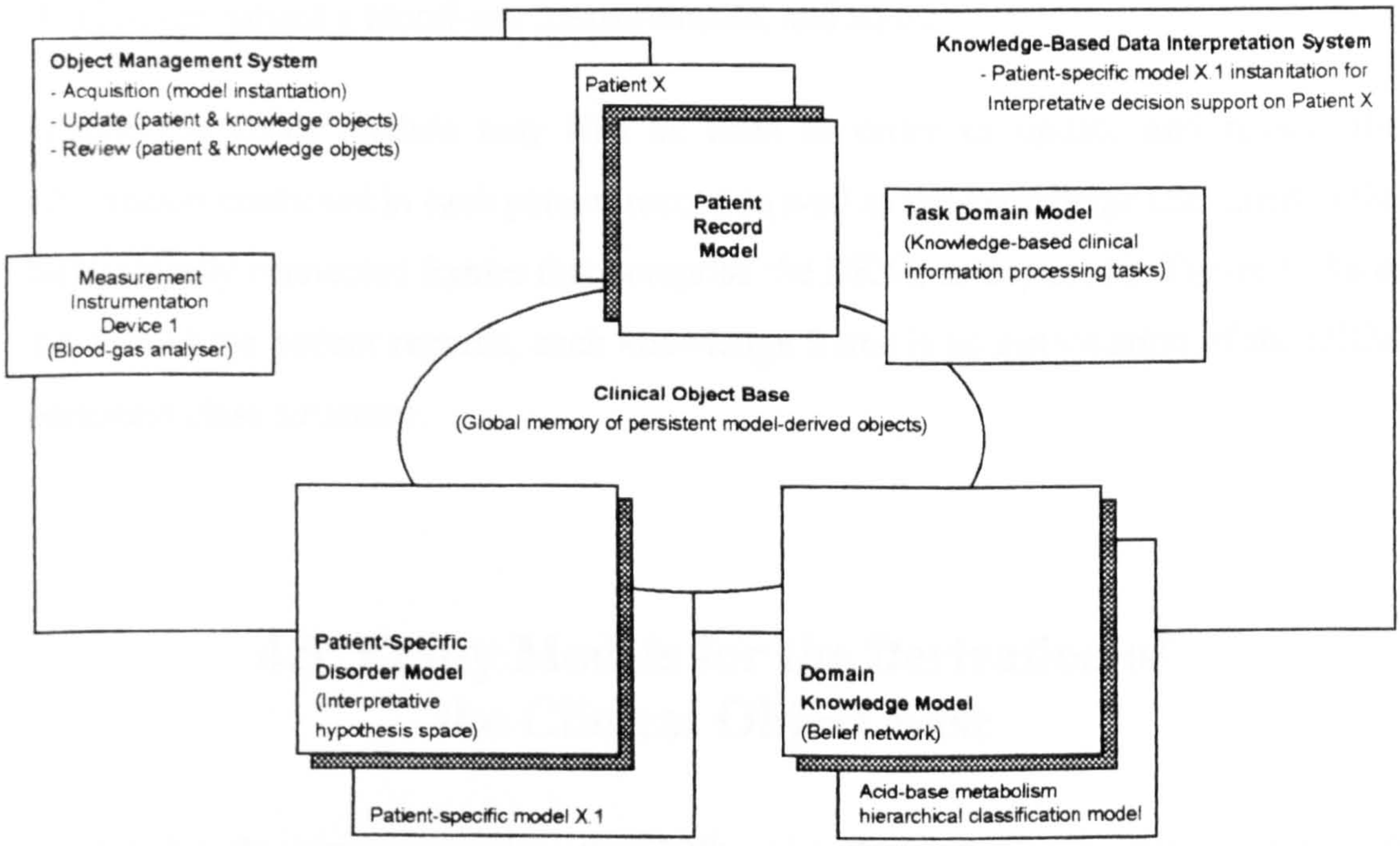


Figure 4-4. Top-level view of the Intelligent Clinical Information Management Support System modular architecture.

The object management system (OMS), which includes an object communication system (OCS) module and the knowledge-based data interpretation system (DIS) module, was designed to monitor the sources of patient data connected to the ICIMS system via serial data communication interfaces, integrate, store and organise the acquired patient data, in this case the results of the blood-gas analysis, by creating instances of the PRM class structure, interpret the data stored in the PRM model-derived object structures, and display their contents in a manner which converts it to information, thereby providing the clinical decision support required to avoid the misinterpretation and consequentially mismanagement of an observed clinical problem under conditions of information overload and contextual complexity.

Thus, for each set of evidence generated by the blood-gas analyser, the DIS module will instantiate a PSM class structure, by applying the reasoning operators that comprise the TDM of the KBS prototype (Figure 2). So, as shown in Figure 3, for patient [X], ICIMS will construct the patient-specific model [X.1] of the patient's interpretative hypothesis space for disorders of acid-base metabolism, given the set [E.1] of the patient's blood-gas measurements, and so on.

Finally, the OMS module may also be used in order to update and review the information contained in each patient record as well as the knowledge contained in the hierarchically connected frames that comprise the DKM, as depicted in Figure 1. As in the case of the patient records, each knowledge frame is an instantiation of the DKM persistent class structure.

4.5. Entity Models for the Derivation of the Clinical Object Base

As stated in the previous section, the COB which forms the heart of the incremental KBS integration process and of the development of the resultant modular ICIMS system architecture, comprises a number of persistent objects which are derived from three models or class structures represented in the system. These are the PRM which corresponds to a prototypical data base schema, the DKM which corresponds to a

prototypical knowledge base schema, and the PSM which corresponds to the interpretative problem solution blackboard of the KBS prototype. This section shows the class structure of these models and the interfaces to the model classes, by which objects are created and maintained. The sections which follow describe the OMS modules which were designed for the acquisition, update and review of the information stored in these objects, and the TDM for the instantiation of the knowledge-based clinical information processing tasks required for the interpretation of the acquired clinical information. The diagrams included in this section and the sections which follow show only the segments of the ICIMS class structure which are required to gain an understanding of the integration-development process. Furthermore, the class interfaces that accompany the class diagrams are meant to indicate the attribute and method members of the classes and are therefore shown in pseudo-code, thereby concealing much of the C++ interface and implementation detail. Figure 4-5 shows the key to the class diagrams which follow.

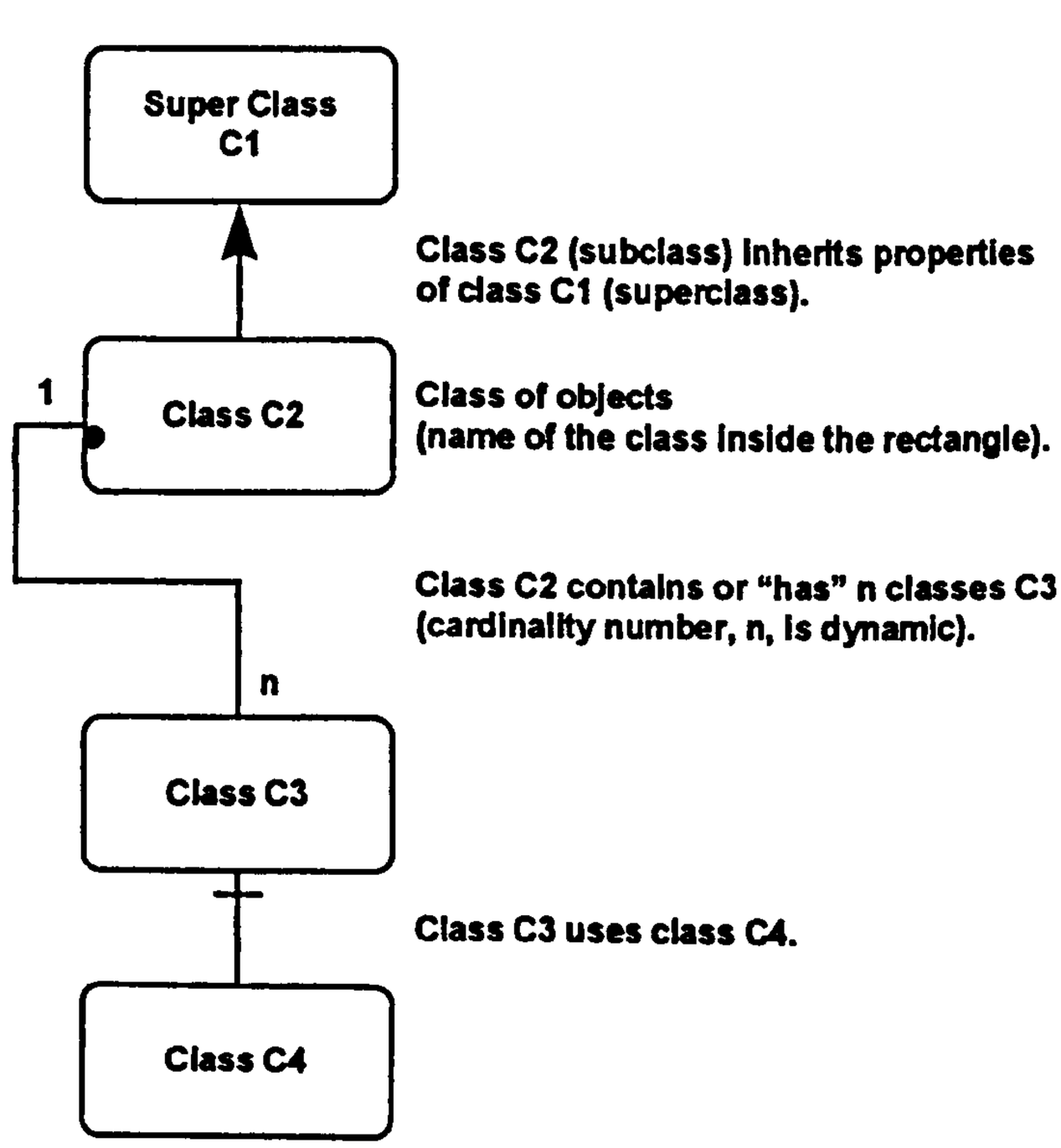


Figure 4-5. Key to the Intelligent Clinical Information Management Support System class structure diagrams.

4.5.1. The Persistent Class Administration System

Each class of objects in the ICIMS system which is declared and made persistent using the POET pre-compiler inherits the properties of the class of COB objects which are constructed and managed by the POET persistent class administration system (PCAS) whose class structure is depicted in Figure 4-6.

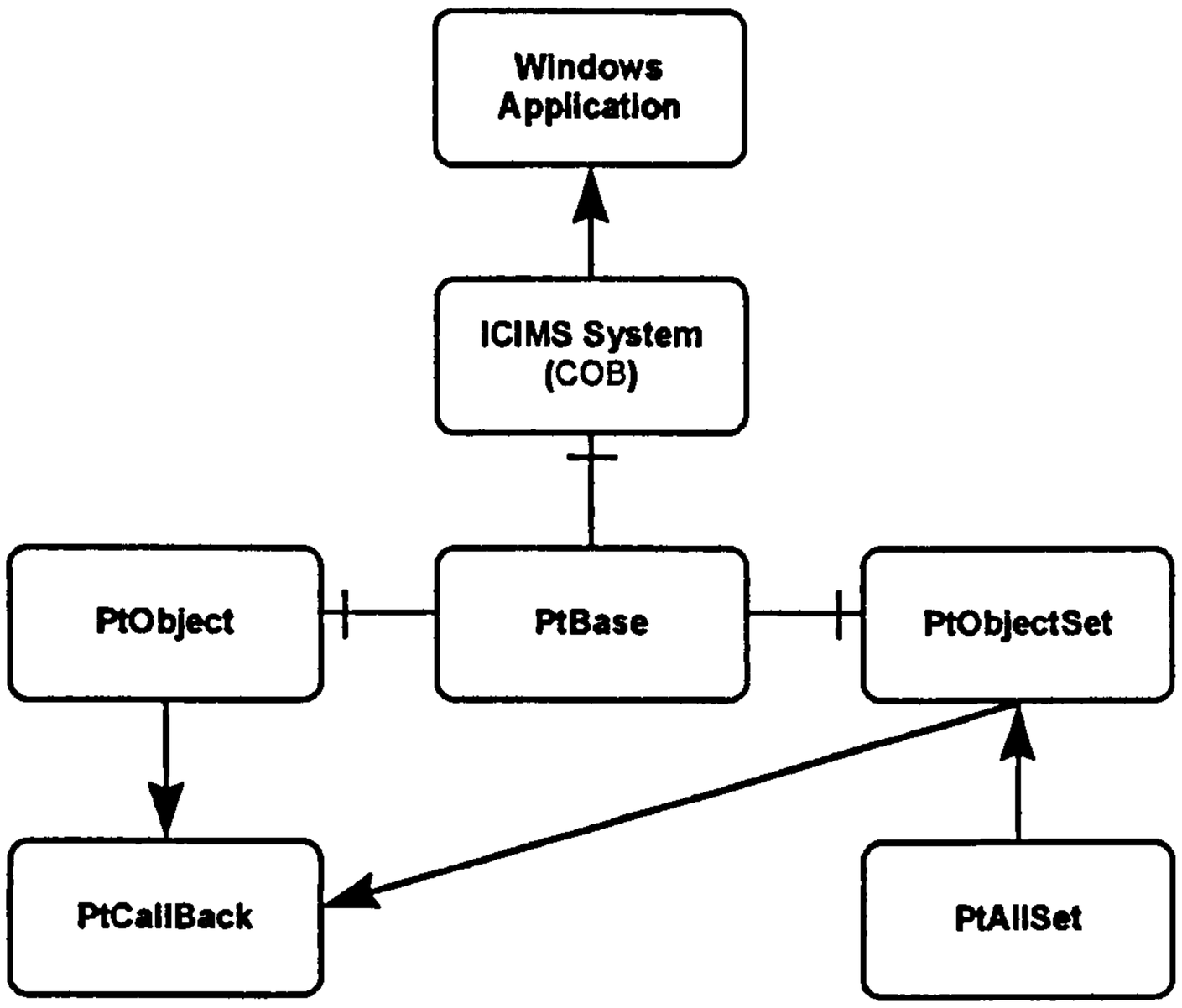


Figure 4-6. Class diagram of the persistent class administration system.

The class PtBase is used to construct an object which represents the object base itself. Thus, PtBase is the class which defines methods for opening and closing a number of concurrent object bases and contains data which POET uses for internal management. Once an object base has been opened, PtBase is used to assign other objects to the object base. The public PtBase methods are shown in Table 4-1.

The class PtObjectSet, whose object handling methods are shown in Table 4-2, is the base container class for all sets of persistent objects. The PtAllSet class provides the means for accessing the objects contained in the object base. As shown in Figure 4-6, AllSets are derived from PtObjectSet and thus inherit the methods listed in Table 4-2. The POET pre-compiler creates an AllSet for each persistent class it encounters.

Thus, for example, the declaration 'persistent class patient' will result in the generation of a patientAllSet, which may be used to retrieve an object from the COB, insert an object, append an object, and in doing so may be searched sequentially, or by performing a query, etc.

class **PtBase** public member methods	**Function**
PtBase	Constructor
~PtBase	Destructor
Open	Opens an object base
Close	Closes the object base
Connect	Establishes a connection to the object server
DisConnect	Breaks off the connection to the server

Table 4-1. Member methods declared as public in the PCAS PtBase class interface.

class **PtObjectSet** public member methods	**Function**
PtObjectSet	Constructor
~PtObjectSet	Destructor
Append	Append an element to the end of the set
Assign	Assign the set to an object base
Clear	Delete all elements from the set
Delete	Delete the element at the current position
Get	Get an element from the set
GetNum	Return the number of elements in the set
Insert	Insert an element at the current position
Put	Overwrite the element at the current position
Query	Perform a query
Seek	Change the internal position within the set
Unget	Clean up after Get

Table 4-2. Member methods declared as public in the PCAS PtObjectSet class interface.

Finally, PtObject is a class which contains data members for object identity and link counts, and the methods needed to assign objects to an object base, store them, delete them, lock them, watch them, or manage the object's link count. These methods are inherited by each and every persistent class. The methods derived from PtObject for

handling persistent objects are listed in Table 4-3. The use of these methods in the ICIMS system is described in the sections which follow.

class **PtObject** public member methods	**Function**
PtObject	Constructor
~PtObject	Destructor
Assign	Gives an object an identity and assigns it to an object base
Delete	Deletes the object from the object base
Forget	Decrements the object's link count; if the object link count is zero it calls the object's destructor to remove it from the memory.
Remember	Increments the objects link count
Store	Stores the object

Table 4-3. Member methods declared as public in the PCAS PtObject class interface.

4.5.2. The Patient Record Model

Figure 4-7 shows the class diagram of the PRM, used for the derivation of the persistent object structure required to support the management of the clinical information generated in the process of monitoring the ICU patient. Section 4.4 briefly described how the OMS module, which apart from updating and reviewing the information stored in the COB is also responsible for raw data acquisition by means of the OCS module, monitors the sources of patient data and creates instances of the PRM class structure, which are identified by means of the name and hospital number of the patient whose information is stored in each derived persistent object structure.

As shown in Figure 4-7, each patient object is a kind-of COB object and is linked via a containment relationship to a number of clinical feature objects and a number of blood-gas analysis objects. The latter is one of the sets of patient parameters which may be required to monitor a particular ICU patient. Other sets may include electro-cardiograph data, blood oxygenation data, etc. Once these objects are created by the COB PCAS described above, they are filled-in either automatically by the OCS module or manually by means of the OMS module dialogues described below, which make use of the methods described in Section 4.5.1 for this purpose.

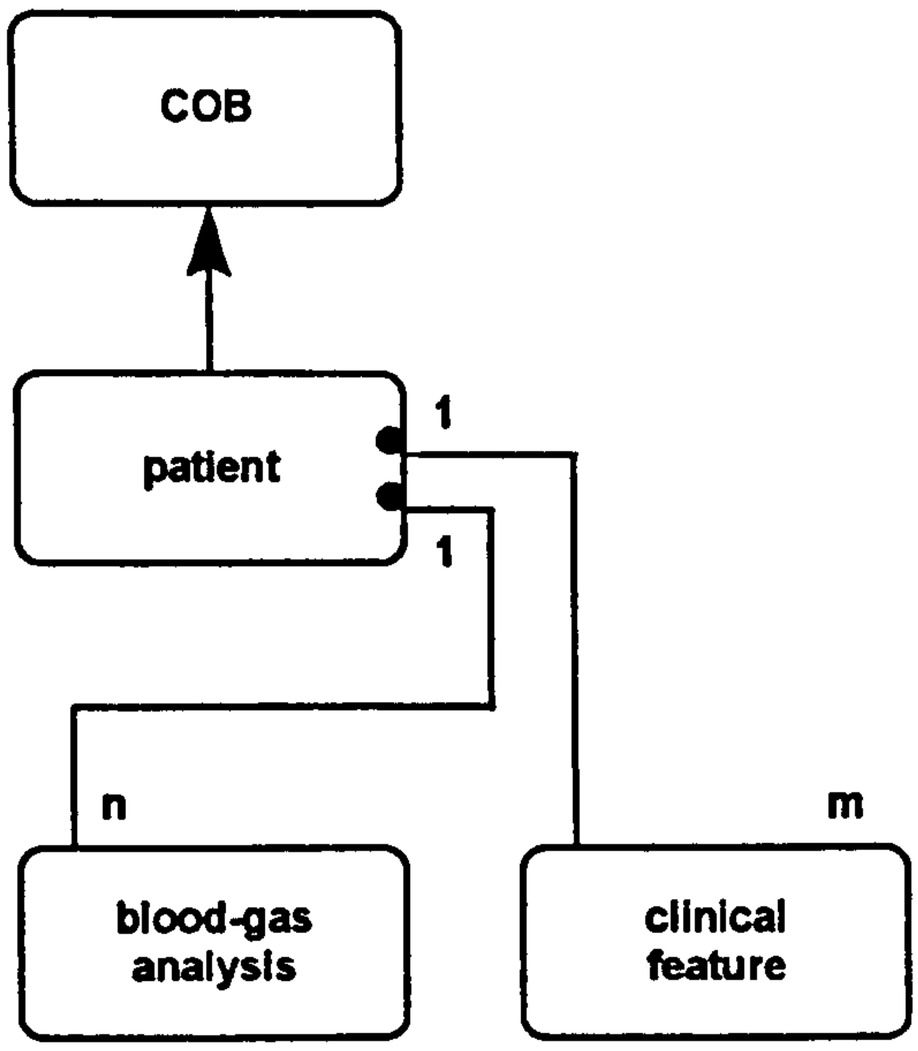


Figure 4-7. Class diagram of the patient record model for the derivation of the persistent object structures required to support the management of the clinical information generated in the process of monitoring the ICU patient.

Listing 4-1 shows the class interface used by the OMS module to derive and manage persistent patient object structures via the PCAS. As described in Section 4.3, the patient class encapsulates only the behaviour of the total system which is pertinent to the abstraction it represents. Thus, any client object wishing to manipulate patient objects may do no more than to construct a patient object structure using the public patient constructor method, initialise the object attributes using the InitialiseObject method, including the contained sets created automatically by the instantiation of the patient class, destroy the object structure using the ~patient destructor method, display the private object attributes using the DisplayObject method, and finally set and retrieve its private members using SetMember and GetMember methods respectively.

```
persistent class patient
{
private:
   name, firstName, hospNumber, ward;
protected:
   cset<bloodGasAnalysis*>     bloodGases;
   cset<clinicalFeature*>      patientFeature;
public:
   patient();
   ~patient();
   SetMember();
```

```
    GetMember();
    InitialiseObject();
    DisplayObject();
};
```

Listing 4-1. Class interface for the construction and manipulation of persistent patient object structures.

Apart from the private attributes required to identify each object derived from this class, the patient class interface also specifies containment links to the two sets of objects as shown in Figure 4-7. The behaviour of these two sets is specified in their respective interfaces. Listing 4-2 shows the interface to the class of blood-gas analysis objects and Listing 4-3 the interface to the class of clinical feature objects. Although these two classes have their own identifiers, their behaviour is the same as that of the patient class. However, the implementation of this seemingly common behaviour may differ. Furthermore, as stated in Section 4.3, the implementations may be changed without affecting any other parts of the system. So for example, in creating bloodGasAnalysis objects, the SetMembers method implementation may be altered so that the OMS module which uses it, does not specify a value for the AnGap parameter, but instead the object calculates that value from the other values it contains, as described in Section 3.2.2.3. In fact, even if the private members of the class are altered, the POET PCAS provides a mechanism for versioning the COB, so that no data is lost, provided no member is deleted.

```
persistent class bloodGasAnalysis
{
private:
   dataSource, accNumber; date, time;
   pH, pCO2, pO2, HCO3, BP, FiO2, Hct,
   Beecf, AnGap,Ca, Cl, K, Na, ...;
   interpretation;
public:
   bloodGasAnalysis();
   ~bloodGasAnalysis();
   ...
};
```

Listing 4-2. Class interface for the construction and manipulation of persistent blood-gas analysis objects.

```
persistent class clinicalFeature
{
private:
   feature;
public:
   clinicalFeature();
   ~clinicalFeature();
   ...
}
```

Listing 4-3. Class interface for the construction and manipulation of persistent clinical feature objects.

4.5.3. The Domain Knowledge Model Structure

The DKM class structure, designed for the derivation of the persistent object structure required to support the management of the knowledge base utilised for the interpretation of the acquired clinical information, is shown in Figure 4-8. This class structure reflects the frame structure used by the KBS prototype for the representation of the belief network shown in Figure 3-1, and is augmented by the methods required to access its attribute members. The class interfaces are shown in Listings 4-4 to 4-7.

```
persistent class disorder
{
private:
   disorderName;
   parentTaxonomy;
   aPrioriProbability;
   set<linkedDisorder>   descendentDisorder;
   set<evidence*>        expectedLevelOfParameter;
   set<aetiology*>       patientFeatureEvidence;
public:
   ...
};
```

Listing 4-4. Class interface for the construction and manipulation of persistent disorder object structures.

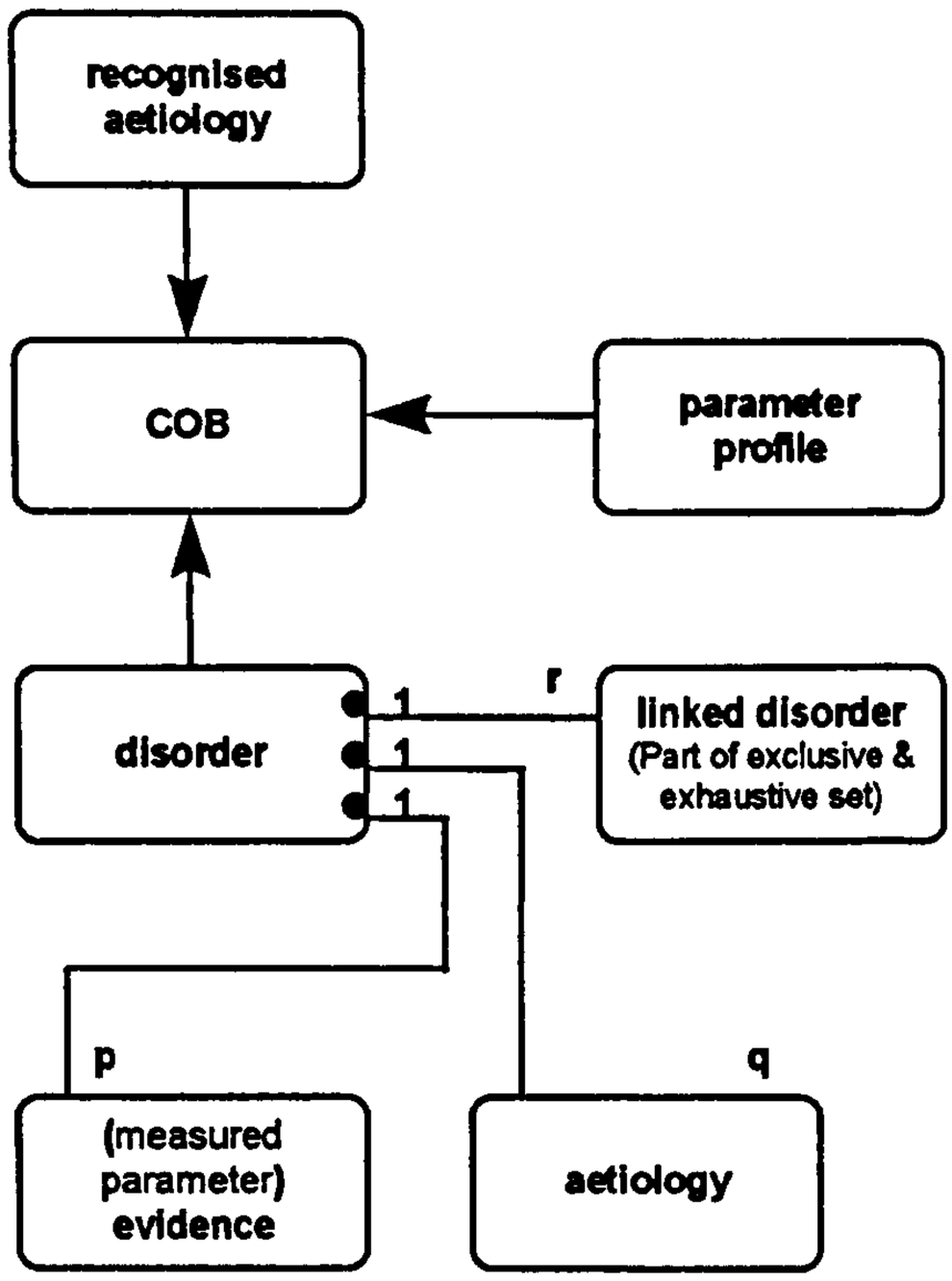


Figure 4-8. Class diagram of the domain knowledge model for the derivation of the persistent object structures required to support the management of the knowledge base utilised for the interpretation of the acquired clinical information.

Listing 4-4 states that disorder objects belong to some general class of disorders, in this case acid-base metabolism, and are assigned an a priori probability of their occurrence given no evidence has been observed. Furthermore, each disorder object contains a number of disorder names which are linked to it as shown in Figure 3-1, a number of objects containing probability assignments for each piece of evidence being observed at a given level given the disorder (Section 3.2.2.5), and a set of aetiologies. Listings 4-5 and 4-6 show the class interfaces for these two classes of objects. Listing 4-7 shows the class interface for the parameter profile objects which are used for the classification of patient measurements.

```
persistent class evidence
{
private:
   parameter;
   probLowGivenDisorder;
   probNormalGivenDisorder;
   probHighGivenDisorder;
public:
   evidence();
   ~evidence();
```

```
   ...
};
```

Listing 4-5. Class interface for the construction and manipulation of persistent disorder evidence objects.

```
persistent class aetiology
{
private:
   feature;
public:
   aetiology();
   ~aetiology();
   ...
};
```

Listing 4-6. Class interface for the construction and manipulation of persistent disorder aetiology objects.

```
persistent class parameterProfile
{
public:
   PtString  name;
   PtString  upperLimit;
   PtString  lowerLimit;
   PtString  mean;
   PtString  standardDeviation;
   PtString  default;
public:
   ...
};
```

Listing 4-7. Class interface for the construction and manipulation of persistent parameter profile objects.

The set of recognised aetiologies contains the vocabulary of terms and since its interface is identical with Listing 4-6, it is not shown here. Again, the public interface of these classes is the same as with patient objects and is therefore also not shown here.

4.5.4. The Patient Specific Model Structure

The patient-specific model (PSM) class structure was designed for the derivation of the persistent blackboard objects required to support the application of the task-domain model (TDM) for evidence propagation in the hierarchical belief network as

described in Section 3.2.2. Figure 4-9 shows the ICIMS version of the prototype blackboard shown in Figure 3-3. The class interfaces to the persistent blackboard objects derived from the general PSM for the construction of situation-specific PSMs are identical with those shown in the previous two sections. Their private members were designed to contain the information specified in Figure 3-3. The objects which use these classes of blackboard objects in the process of interpreting acquired patient data are described in the Section 4.7.

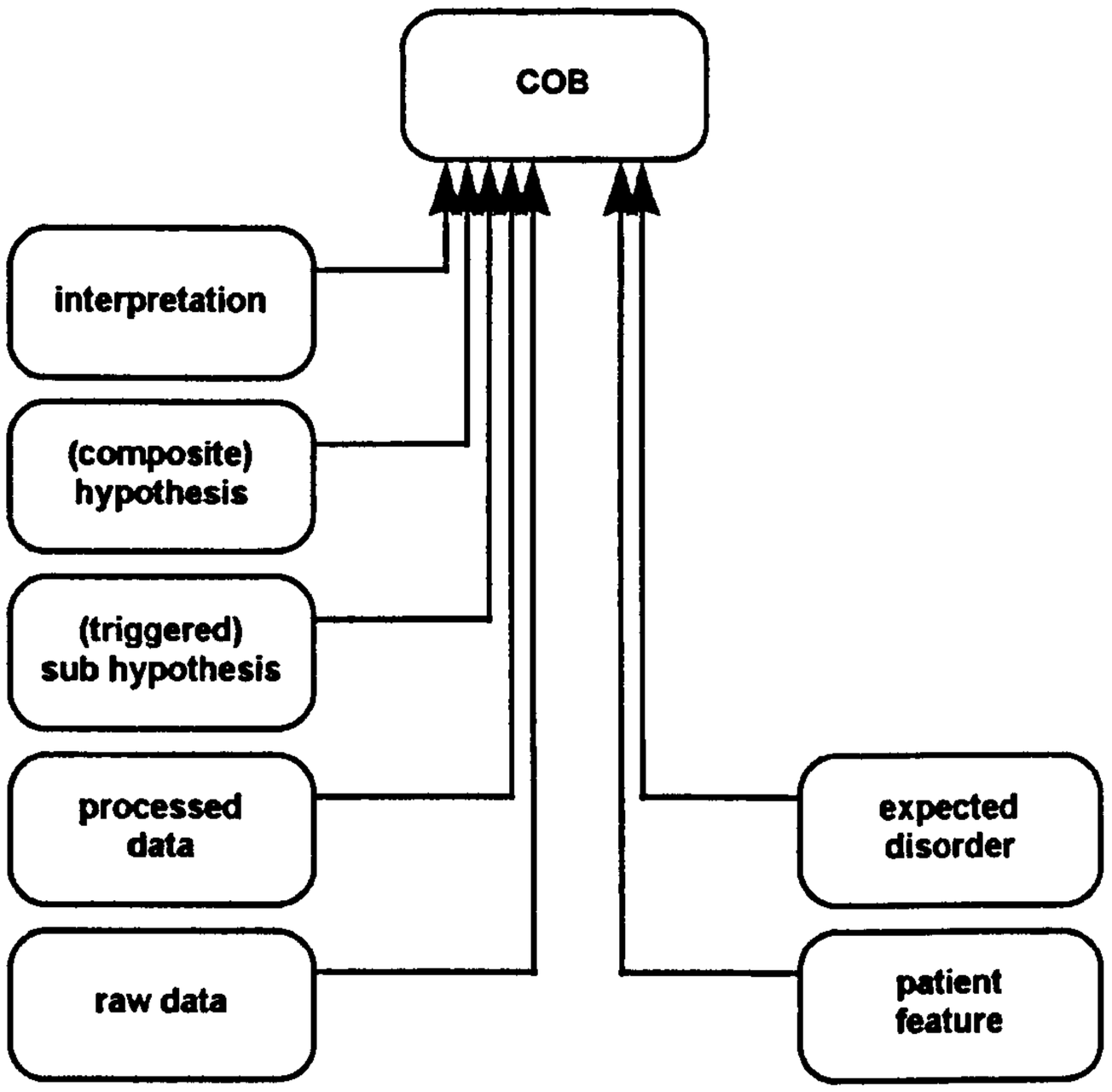


Figure 4-9. Class diagram of the patient-specific model for the derivation of the blackboard objects required to support the application of the prototype task-domain model.

4.6. The Object Management System

In the ICIMS system, the persistent object structures contained in the COB constitute the majority of the entity abstractions required to support the integration-development process. This architectural feature confers the required ability to develop, append, and constructively assess layers of action abstractions, which make use of the underlying COB entities in order to provide the required clinical information management and decision support functionality, in a manner which is ergonomically and cognitively compatible with the patient care activity of the user, without affecting the underlying object structures or their contents in the process.

Thus, as shown in Figure 4-4, the first layer of the incremental ICIMS system development comprises the COB module, which uses the PCAS for handling the persistent object structures derived and maintained by means of the second layer, which comprises the object derivation models. Similarly, the OMS module is one level closer to the user since it provides the required functionality of patient data and domain knowledge acquisition, update and review, and a step closer to the integration of the KBS prototype, since it is at this stage that most of the clinical advisor's constructive assessment is translated into the evolutionary modifications pertaining to the KBS prototype integration process.

The first layer of action modelling which was appended onto the basic object handling system was the OMS module. As stated above, the OMS module of the ICIMS system performs the instantiation, update and review of the persistent object structures described above via the PCAS. The class interfaces shown in Section 4.5 indicate that apart from constructing and destroying objects, each class of objects provides methods for initialising, setting, and retrieving the information stored in each object derived from that class. During the initial stages of the integration development process, these interfaces were used to test the data acquisition process as well as the data base and knowledge base schemata. Following that, the OMS module was developed to use these methods in order to manipulate the persistent objects contained in the COB and derived from the models described above, via Windows

API dialogues with the user, which were designed to reflect the anatomy of the persistent object structures contained in the COB, thereby facilitating the review and update of their contents.

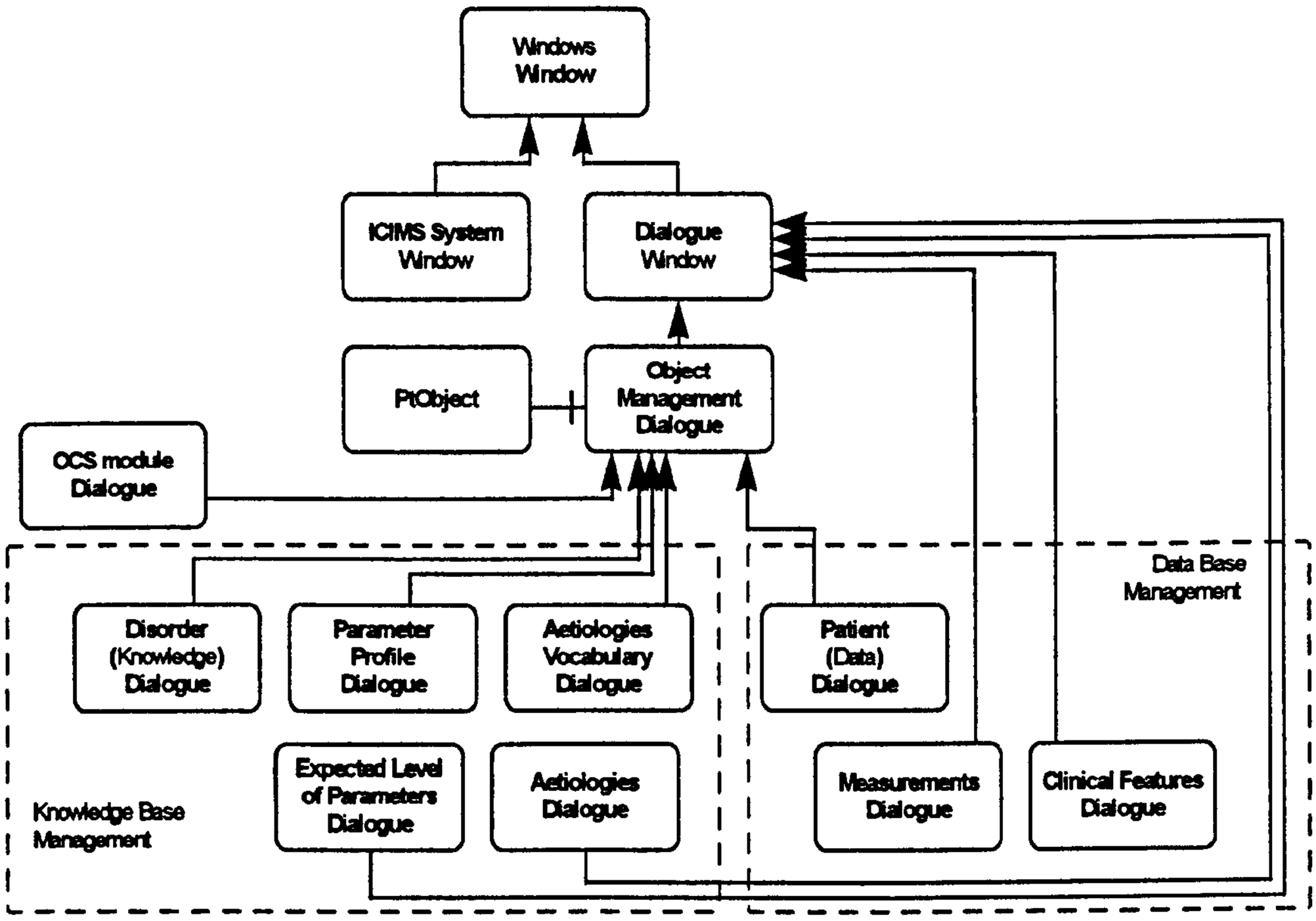


Figure 4-10. Class structure diagram linking the ICIMS Clinical Object Base to the Object Management System module for the acquisition, update and review of the data and knowledge model derived objects.

Figure 4-10 shows the class structure diagram of the OMS module of the ICIMS system. The module comprises the main ICIMS window class, which upon constructing and displaying a system window object using the ICIMSSystemWindow() method shown in Listing 4-8, uses the methods inherited from its Windows window superclass to control the main system menu.

```
class ICIMSSystemWindow: public WindowsWindow
{
protected:
   Commands();
public:
   ICIMSSystemWindow();
```

```
};
```

Listing 4-8. Class interface for the construction and manipulation of ICIMS system windows.

The class of dialogue window objects also inherits the properties of the Windows window class in order to instantiate, display and control dialogues with the user, which are superimposed on the main window for the purpose of providing access to the COB objects. As the diagram shows, there are two types of dialogue windows in the system. Those which inherit the properties of a standard Windows dialogue window and those which inherit the properties of the class of ICIMS object management dialogues. The interface to the latter class of objects is shown in Listing 4-9.

```
class objectManagementDialogue : public dialogueWindow
{
protected:
   Commands();
   Init();
   Show();
   Fill();
   InitObject();
   NextObject();
   PrevObject();
   StepObject();
   GetPtObject();
   GetAccessCode();
   ...
public:
   objectManagementDialogue() : dialogueWindow()
                                 ...
};
```

Listing 4-9. Class interface for the construction and manipulation of object management dialogue objects.

Thus, objectManagementDialogue objects are responsible for creating, initialising and displaying dialogue screens, filling in the COB objects, and processing a number of pertinent menu commands which include showing, storing and deleting an object, and the sequential search through the object AllSets described above. As described in Section 4.5.1, the class of ICIMS system applications shown in Figure 4-6 uses PtBase and the member methods listed in Table 4-1 to open and close a COB. Once this is done, the class of objectManagementDialogue objects uses the PtObject member methods listed in Table 4-3 to store and delete the objects contained in the

COB. The dialogue window classes which directly inherit the properties of the dialogueWindow class do not require the PtObject methods for storing and deleting objects since they are used to modify objects which are linked with those modified by object-ManagementDialogue objects and are hence stored automatically by the PCAS. Finally, since the class objectManagementDialogue objects provides the means for displaying, storing and deleting COB objects, the method GetAccessCode is used to control access privileges.

The 9 dialogue windows shown in Figure 4-10 belong to one of the three general categories of OMS function described in Section 4.5. These are the OCS dialogue for data acquisition, the data base management dialogues and the knowledge base management dialogues for updating and reviewing the information stored in the data and knowledge COB objects respectively. The OCS module dialogue displays in a separate window the information acquired by the system module via the RS232 interfaces. The user has no control over this dialogue, apart from supplying the information required for registering the acquired patient measurements. Once this is done, the OCS dialogue creates, initialises, fills-in and stores the acquired and supplied patient information using PRM derived objects. The other two types of dialogue are described in the following two sections.

4.6.1. Data Base Management Dialogues

Listing 4-10 shows the class interface used by the OMS module in order to create a patient dialogue object for the purpose described above, upon receiving an appropriate message from the ICIMSSystemWindow menu Commands method shown in Listing 4-8. Since patient dialogue objects inherit the properties of the class of objectManagementDialogue objects, the methods declared in the protected interface of Listing 4-9 are declared as virtual, meaning that their implementation is defined in the superclass. The class of patient dialogue objects also contains a number of EditBox and ListBox members which are used to display and modify the contents of members of the object being updated and reviewed.

```
class patientDialogue : public objectManagementDialogue
{
private:
 EditBox      accessCode;
 EditBox      numberOfPatients;
protected:
 patient                    *pPatient;
 recognisedAetiology        *pRecognisedAetiology
 patientAllSet                    *pSetOfAllPatients;
 recognisedAetiologyAllSet        *pSetOfAllRecognisedAetiologies;
protected:
 EditBox      name;
 EditBox      firstName;
 EditBox      hospNumber;
 ...
 ListBox      patientFeatureList;
 ListBox      systemFeatureList;
 virtual      Show();
 virtual      Fill();
 virtual      Commands();
 virtual      Init();
 virtual      InitObject();
 virtual      NextObject();
 virtual      PrevObject();
 ...
public:
 patientDialogue () : objectManagementDialogue(),
                      ...
 virtual ~patientDialogue ();
};
```

Listing 4-10. Class interface for the construction and manipulation of patient dialogue objects.

Furthermore, by means of pointers to objects, denoted by the asterisk which is used in the C programming language for this purpose, patientDialogue objects may use patient objects to review and update their members, recognisedAetiology objects to specify the patient's clinical features from the available persistent vocabulary objects, and the corresponding AllSets which are required to search through the COB as described in Section 4.5.1. These relationships are depicted in Figure 4-11, which also shows the PCAS link between patient and data dialogues as described below. The data base management module of the OMS also contains dialogue classes for updating and reviewing the blood-gas analysis objects contained in patient objects as shown in Figure 4-7. Again, the interface of this dialogue class reflects the contents of the blood-gas analysis object it is designed to update and review, with the exception of its window control inheritance as described above. Finally, as shown in Listing 4-10, the patient's clinical features are supplied directly via the patient dialogue.

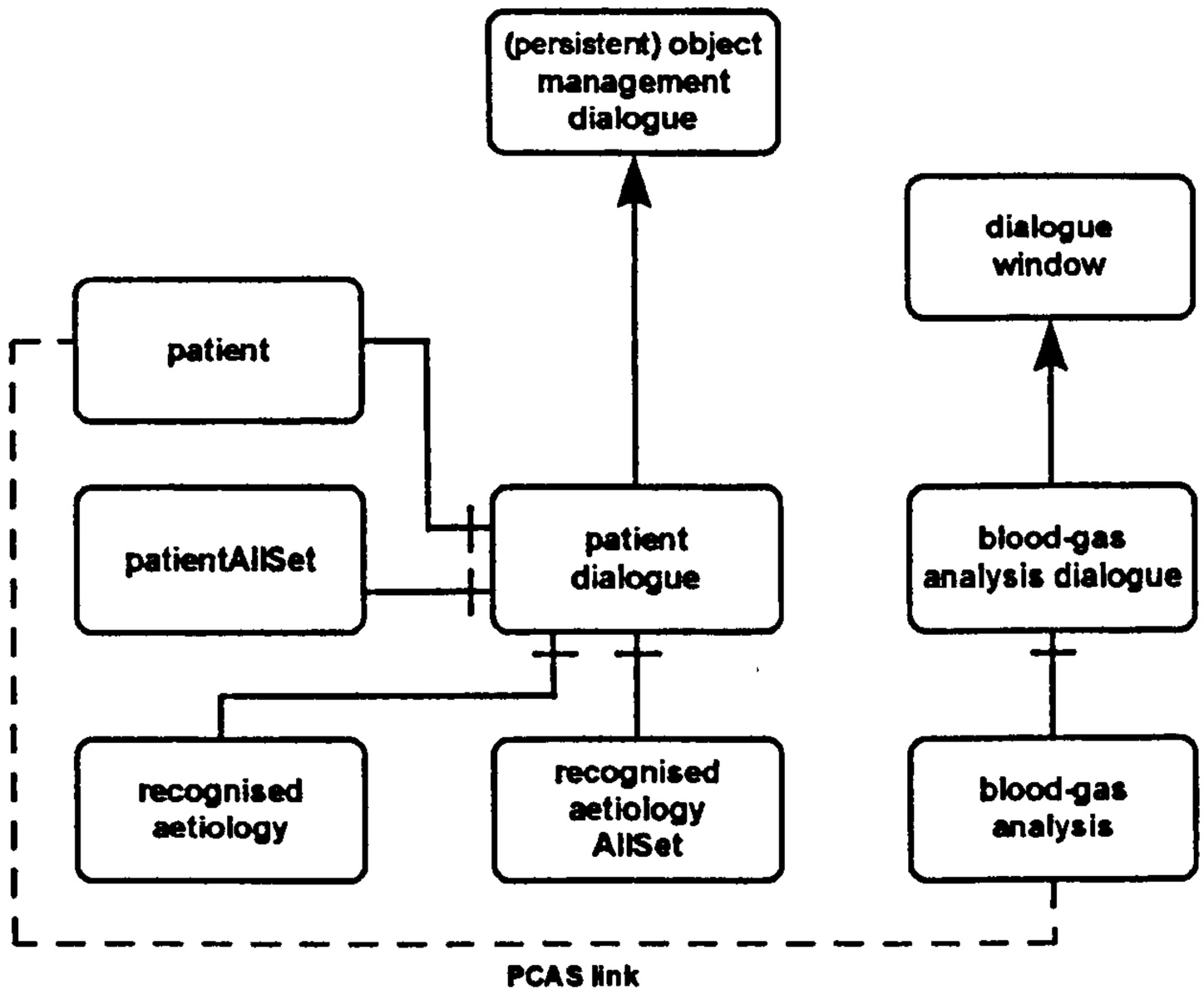


Figure 4-11. Class diagram of the patient data dialogue object structure created by the object management system.

4.6.2. Knowledge Base Management Dialogues

Listing 4-11 shows the class interface used by the OMS module in order to create disorder knowledge dialogue objects. The disorderDialogue class interface is identical to that of patientDialogue objects, in the sense that it reflects the contents of disorder objects and it inherits the properties of the objectManagementDialogue class. The class diagram of disorder dialogues is shown in Figure 4-12.

```
class disorderDialogue: public objectManagementDialogue
{
private:
   SingleEdit      accessCode;
protected:
   disorderAllSet          *allDisorders;
   disorder                *disorder;
protected:
   EditBox   name;
   ...
   virtual   Show ();
   ...
public:
   disorderDialogue (): objectManagementDialogue(),
```

```
                                    ...
    virtual ~disorderDialogue ();
};
```

Listing 4-11. Class interface for the construction and manipulation of disorder dialogue objects.

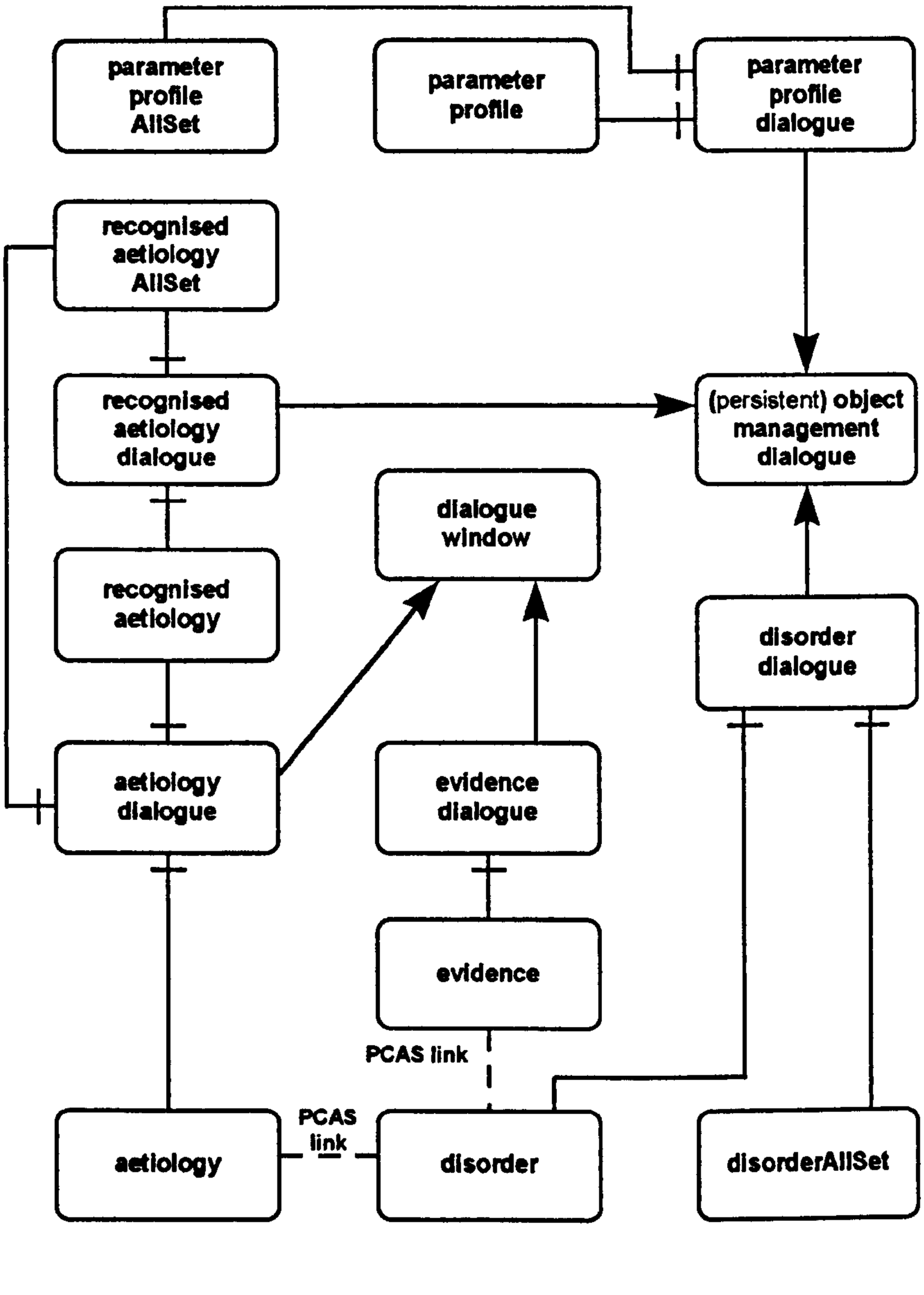


Figure 4-12. Class diagram of the disorder knowledge dialogue object structure created by the object management system.

Listing 4-12 shows the class interface for evidence dialogue objects. That is, dialogue objects which were designed to update and review the information stored in the evidence objects contained in disorder objects as shown in Figure 4-8. Again, the class

of evidence dialogue objects contains the members required for editing the evidence members shown in Listing 4-5. However, since the particular dialogue is not responsible for registering any changes made to the corresponding COB object contents, as described above, it directly inherits the properties of dialogue window objects as shown in Figure 4-10. For this reason the class of evidenceDialogue objects specifies its own user interface commands, as do all dialogue objects with the same inheritance.

```
class evidenceDialogue: public dialogueWindow
{
private:
   EditBox   variable;
   EditBox   plow;
   EditBox   pnormal;
   EditBox   phigh;
protected:
   virtual   Init();
   virtual   Commands();
public:
   evidence  evidenceObject;
   evidenceDialogue(): dialogueWindow (),
                              ...
};
```

Listing 4-12. Class interface for the construction and manipulation of evidence dialogue objects.

Finally, as shown in Figure 4-12, the knowledge base management module of the OMS also contains the class of recognised aetiology dialogue objects for the maintenance of a vocabulary of clinical terms recognised by the system, the class of aetiology dialogue objects, used by the OMS to construct and maintain known patient features as causes of the represented disorders, and the class of parameter profile dialogue objects for specifying the expected level of monitored parameters given each represented disorder.

4.7. The Data Interpretation System

As stated above, the OMS module layer was appended onto the COB layer in order to satisfy the user's fundamental requirements for clinical information management and decision support. More specifically, the OMS module was designed to generate ergonomic dialogues with the user, examples of which are shown in the next chapter, in order to create the persistent data and knowledge object structures contained in the COB, by deriving such object structures from the patient record and domain knowledge models, and to thereby also support and facilitate the integration of the prototype interpretative TDM required for the contextual interpretation of the data acquired in the process of monitoring patients with disorders of acid-base balance. This corresponds to the third and final stage of the prototype KBS integration process.

The DIS module was thus designed to be appended onto the OMS ICIMS system layer, in order to instantiate a number of interpretative dialogues with the user, which dialogues represent the cognitive information processing tasks comprising the prototype TDM (Section 3.2.3), and to thereby derive the third class of persistent object structure, that of the PSM, which as described above was designed to take the place of the prototype KBS blackboard.

4.7.1. Data Classification Dialogue

Data classification dialogues display the operations performed by the classification knowledge source of the prototype KBS task domain described in Section 3.2.2.5 for the qualitative abstraction of the acquired raw patient data. Figure 4-13 shows the class diagram of classification dialogue objects and Listing 4-13 the class interface.

```
class classificationDialogue : public dialogueWindow
{
protected:
   ...
   ListBox                    classificationBox;
   rawData                    *pRawData;
```

```
   processedData          *pProcessedData;
   parameterProfile             *pProfile;
   rawDataAllSet                *pSetOfAllRawData;
   parameterProfileAllSet       *pSetOfAllProfiles;
protected:
   Classify();
   virtual          Init();
   virtual          Commands();
public:
   classificationDialogue (): dialogueWindow(),
                              ...
   virtual ~classificationDialogue();
};
```

Listing 4-13. Class interface for the construction and manipulation of data classification dialogue objects.

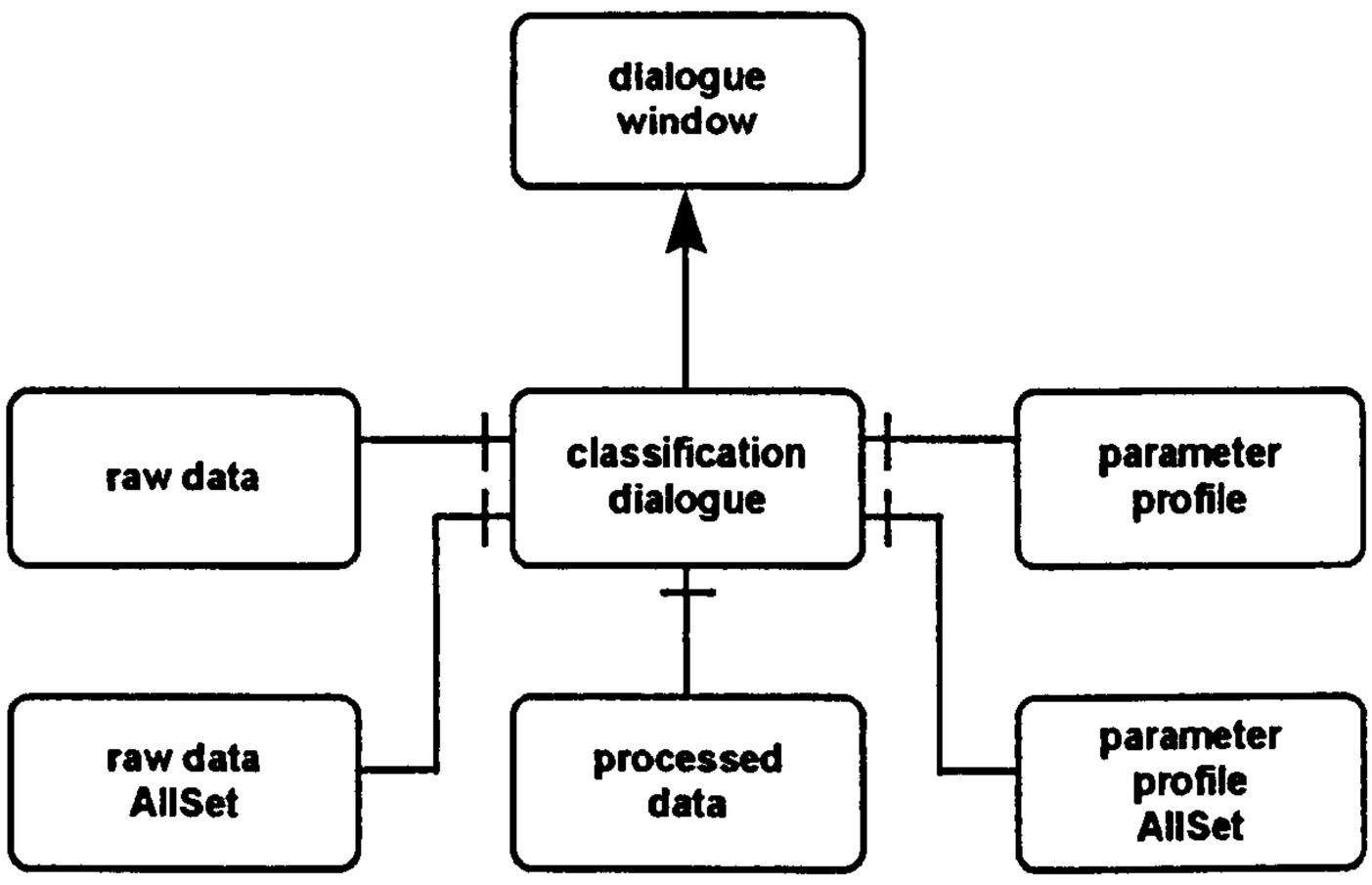


Figure 4-13. Class diagram of data classification dialogues.

As shown in Figure 4-13, since classification dialogues are a kind of dialogue window, once a data classification dialogue object is created, it inherits the properties of the general class of dialogue windows for display and user control. Furthermore, the classification dialogue class uses persistent objects from the rawData class of the blackboard object base and persistent parameterProfile objects from the knowledge object base. The class of processedData of the blackboard object base is used to create corresponding objects and to store the results of the classification operation. The initialisation method clears the contents of the display box used by the object to show the results of the data processing it performs and creates the rawData and

parameterProfile AllSets used by the object to search the spaces of the corresponding blackboard and knowledge-base persistent objects maintained in the COB and required for the operation represented in this class of objects. Once this is done, the Init method calls the Classify method, which incidentally cannot be called by other objects since it is not public, in order to carry out the data classification process. Again the Commands method processes the dialogue control messages used by the Windows API system. One of these messages is the cancel message, which calls the object destructor ~classificationDialogue to destroy the AllSets created by the initialisation method.

Again, following the principles of abstraction, encapsulation or information-hiding, and modularity, for the development of open, maintainable and re-usable systems, the classes of interpretative behaviour abstraction for raw data classification, and evidence impact, aggregation and propagation, were designed so that their client objects (the DIS module or other interpretative KS objects) are not required to know any of the implementation details of the represented behaviour. For example, classification dialogue objects declare in their class interface 1) as public (ie. visible to client objects), only the object's constructor and destructor, and 2) as private (ie. visible only to the objects constructed from the class), the parts of the COB accessed by the represented KS and the methods the class uses internally in order to perform the represented interpretative action. Thus, the DIS is called to construct the required KS object using its class description, and to destroy the object following its application. In the meantime, the state of the system's problem-solving behaviour is encapsulated within the KS object. This means that although implemented to support and facilitate the integration validation of the BGAS TDM within the ICIMS system and its environment, by progressively generating and consolidating interpretative hypotheses given the evidence available in a case, the classes of objects comprising the ICIMS TDM may be re-implemented without disturbing any parts of the system, and thus evolve into an integrated intelligent monitoring and control (IMC) TDM (Sections 2.4.5 & 4.2.3).

4.7.2. Evidence Impact Dialogue

The operation of the evidence impact knowledge source of the KBS prototype TDM as described in Section 3.2.2.6, is represented in the ICIMS system by the class of evidence dialogue objects depicted in Figure 4-14. This class of dialogue objects is slightly more complicated than that of classification dialogue objects since calculating the weight of evidence on the interpretative hypotheses contained in the knowledge object base involves a number of object AllSet searches. More specifically, as shown in Figure 4-14 and Listing 4-14, evidence dialogue objects use disorder AllSet objects to search the belief network shown in Figure 3-1, and disorder objects to retrieve the information stored in each node and required in order to calculate the weight of the evidence given by processed data objects. The class of evidence dialogue objects uses two methods to search the disorder AllSet. The first method is the sequential search which performed using the Seek method shown in Table 4-1 and the second is the Query method also shown in Table 4-1.

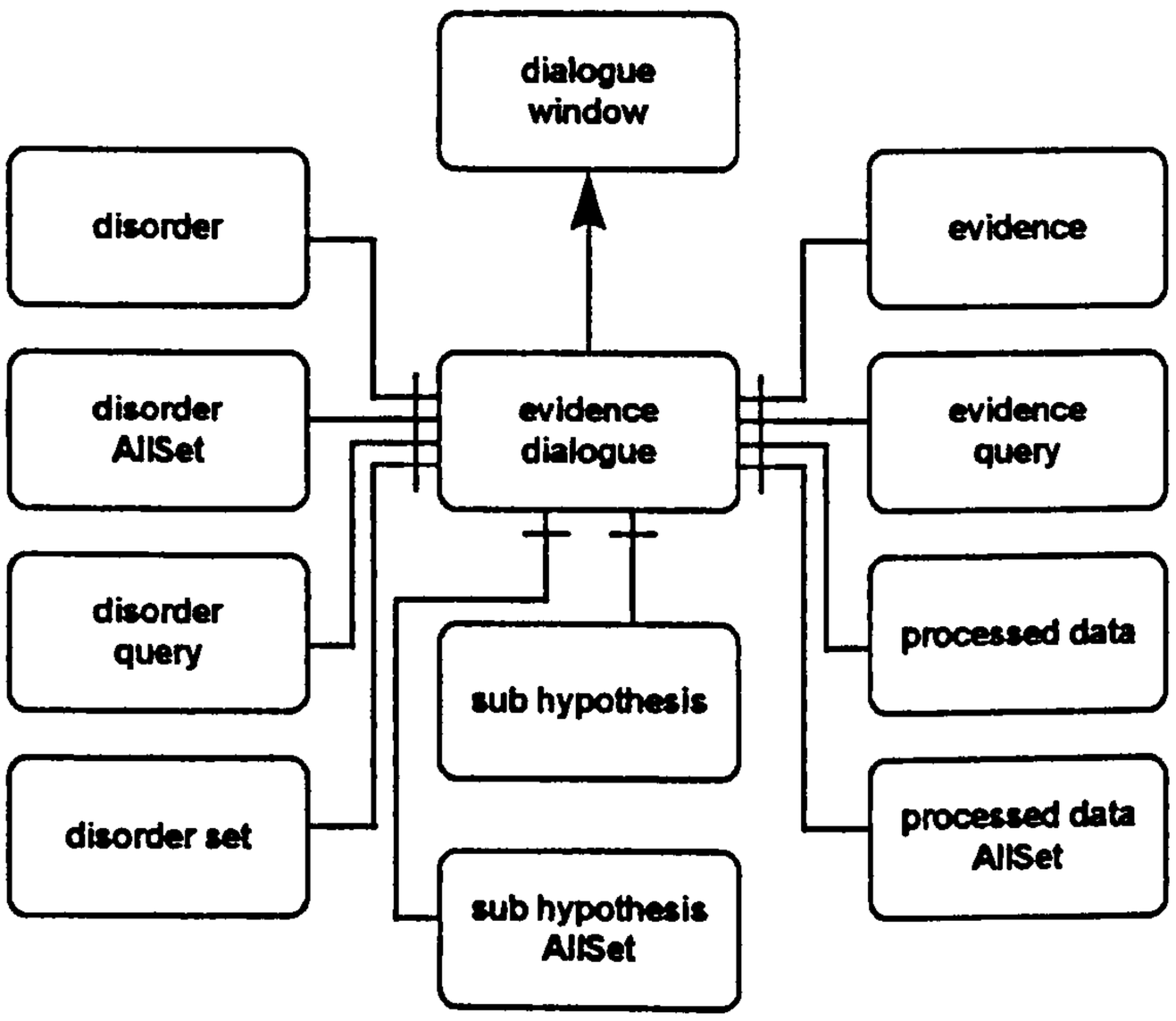


Figure 4-14. Class diagram of evidence impact dialogues.

```
class evidenceDialogue : public dialogueWindow
{
protected:
   ...
   ListBox   subHypothesesBox;
   EditBox   countBox;
   disorder                *pDisorder ;
   disorderAllSet          *pAllDisorders ;
   disorderQuery           *pTheQSpec ;
   disorderSet             *pTheResultSet ;
   evidenceQueryCondition;
   evidenceQuery           *pTheQDataSpec;
   evidence                *pEvidence;
   processedData           *pProcessedData;
   processedDataAllSet     *pSetOfAllProcessedData;
   subHypothesis           *pSubHypothesis;
   subHypothesisAllSet     *pSetOfAllSubHypotheses;
   s1;
   s2;
protected:
   SetEvidenceQueryCondition();
   SetEvidenceQuerySpec();
   SetDisQuerySpec();
   CreateResultSet();
   FindAssignedDisorders();
   GenerateSubHypotheses();
   UpdateWeights();
   ProcessEvidence();
   Show();
   virtual   Init();
   virtual   Commands();
public:
   evidenceDialogue ():    dialogueWindow(),
                           ...
   virtual ~evidenceDialogue();
};
```

Listing 4-14. Class interface for the construction and manipulation of evidence impact dialogue objects.

In order to search the disorder AllSet using the Query method, evidence dialogue objects construct and use the following PCAS objects. Disorder query objects perform a query to satisfy certain conditions within the disorder AllSet, some of which pertain to disorder objects, such as the disorder name and linked disorders, and some to evidence objects. The latter are satisfied via evidence query objects. The results of the query, comprising pointers to knowledge objects, are stored in disorder set objects. Finally, evidence dialogue objects use processed data objects to retrieve the qualitative abstractions of raw data evidence they contain and create sub hypothesis objects to store the results of the application of the knowledge source represented in the particular dialogue.

As with the classification dialogue objects, the object initialisation method constructs the necessary AllSet objects in memory for searching the problem space, and subsequently calls the constructed object's protected ProcessEvidence method in order to perform the represented knowledge source operations. These comprise the query methods SetEvidenceQueryCondition, SetEvidenceQuerySpec, SetDisQuerySpec and CreateResultSet, as well as the methods FindAssignedDisorders, GenerateSub-Hypotheses and UpdateWeights, described in Section 3.2.2.6. The Show method displays the results of the application of the knowledge source in the subHypothesis ListBox.

4.7.3. Evidence Aggregation and Propagation Dialogue

The operation of the evidence aggregation and propagation knowledge source of the KBS prototype TDM described in Section 3.2.2.7, is represented in the ICIMS system by the class of sum hypotheses dialogue objects depicted in Figure 4-15. Listing 4-15 shows the class interface used by the DIS to generate sum hypotheses dialogues with the user and to thereby apply the represented operator to the solution of an interpretative problem.

```
class sumHypothesesDialogue : public dialogueWindow
{
protected:
   ...
   belief;
   leaf;
   ListBox          hypothesesBox;
   disorder         *pDisorder;
   disorderAllSet   *pSetOfAllDisorders;
   disorder         *pChild;
   disorderAllSet   *pSetOfAllChildren;
   subHypothesis    *pSubHypothesis;
   subHypothesisAllSet   *pSetOfAllSubHypotheses;
   hypothesis            *pHypothesis;
   hypothesisAllSet      *pSetOfAllHypotheses;
   interpretation        *pInterpretation;
   interpretationAllSet  *pSetOfAllInterpretations;
protected:
   RetrieveDisorder();
   RetrieveChild();
   RetrieveHypothesis();
   SumHypotheses();
   ShowAndCopy();
   virtual   Init();
   virtual   Commands();
public:
```

```
    sumHypothesesDialogue(): dialogueWindow(),
                                ...
    virtual ~sumHypothesesDialogue();
} ;
```

Listing 4-15. Class interface for the construction and manipulation of evidence aggregation and propagation dialogue objects.

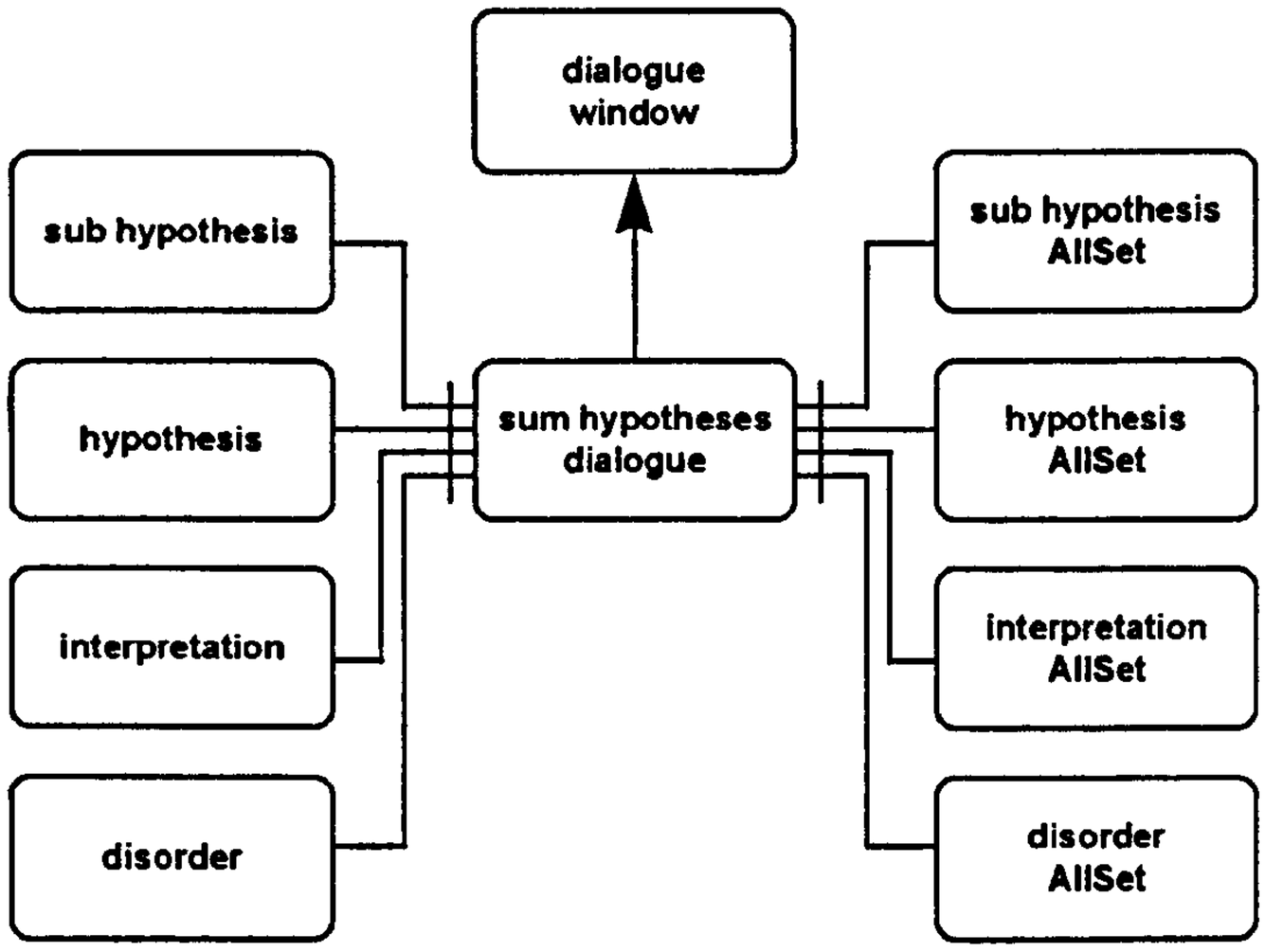


Figure 4-15. Class diagram of sum hypotheses dialogues.

As before, the class of dialogues uses the subHypothesis AllSet to search for solution elements at the sub-hypothesis blackboard level shown in Figure 4-9, and subHypothesis objects in order to retrieve the information stored in such objects at the level concerned. The results of the application of the knowledge source to that level of the blackboard are written one level higher and closer to the solution, by instantiating and storing hypothesis objects. In order to perform the represented cognitive information processing task, depicted in Figure 3-6, the SumHypotheses method of the class of sumHypothesesDialogue objects calls RetrieveDisorder for each subHypothesis object, and RetrieveChild to search the disorder AllSet and to retrieve the names of the descendent leaf node disorder hypotheses and to thereby propagate the effect of evidence on the set of mutually exclusive and collectively exhaustive hypothesis set. For each leaf node disorder of each sub-hypothesis object, the SumHypotheses method searches the hypothesis AllSet to find out if entries have

been made and to therefore aggregate the effect of various pieces of evidence bearing on the same hypothesis. Finally, the ShowAndCopy method displays the results of the application of the represented knowledge source and copies the blackboard entries made by the knowledge source to the interpretation level AllSet for the application of the knowledge source dialogue which follows. All other member methods are the same as before.

4.7.4. Hypotheses Ranking Dialogue

Following the aggregation and propagation of the acquired evidence in the hierarchical belief network as described in the previous section, the DIS will instantiate the rank hypotheses dialogue class, the interface to its objects being shown in Listing 4-16. As shown in the class interface and Figure 4-16, rank hypotheses dialogues use hypothesis objects in order to retrieve the results of the application of the sum hypotheses dialogues to the sub hypothesis level of the blackboard, and interpretation objects in order to make appropriate entries at the interpretation level. The corresponding AllSet objects are used to search the COB as before. Disorder objects and disorderAllSets are used by the dialogue objects in order to search and retrieve the leaf disorders of each disorder.

As before, the Init method creates the required AllSet objects and calls the RankHypotheses member method. The RankHypotheses method searches the interpretation AllSet to check if at any stage in the interpretation process the belief in each of the singleton descendants of an hypothesis is the same. If so, the hypotheses are aggregated, removed from the interpretation level, and an entry is made to replace the set with the aggregated belief in the parent hypothesis. This process corresponds to the rank hypotheses knowledge of the KBS prototype, which ensures that interpretative hypotheses are reported at the most appropriate level of abstraction in the disorder class hierarchies. Finally, the Show method is used to display the contents of the interpretation AllSet in the appropriate dialogue window. Init and Commands are the same as before.

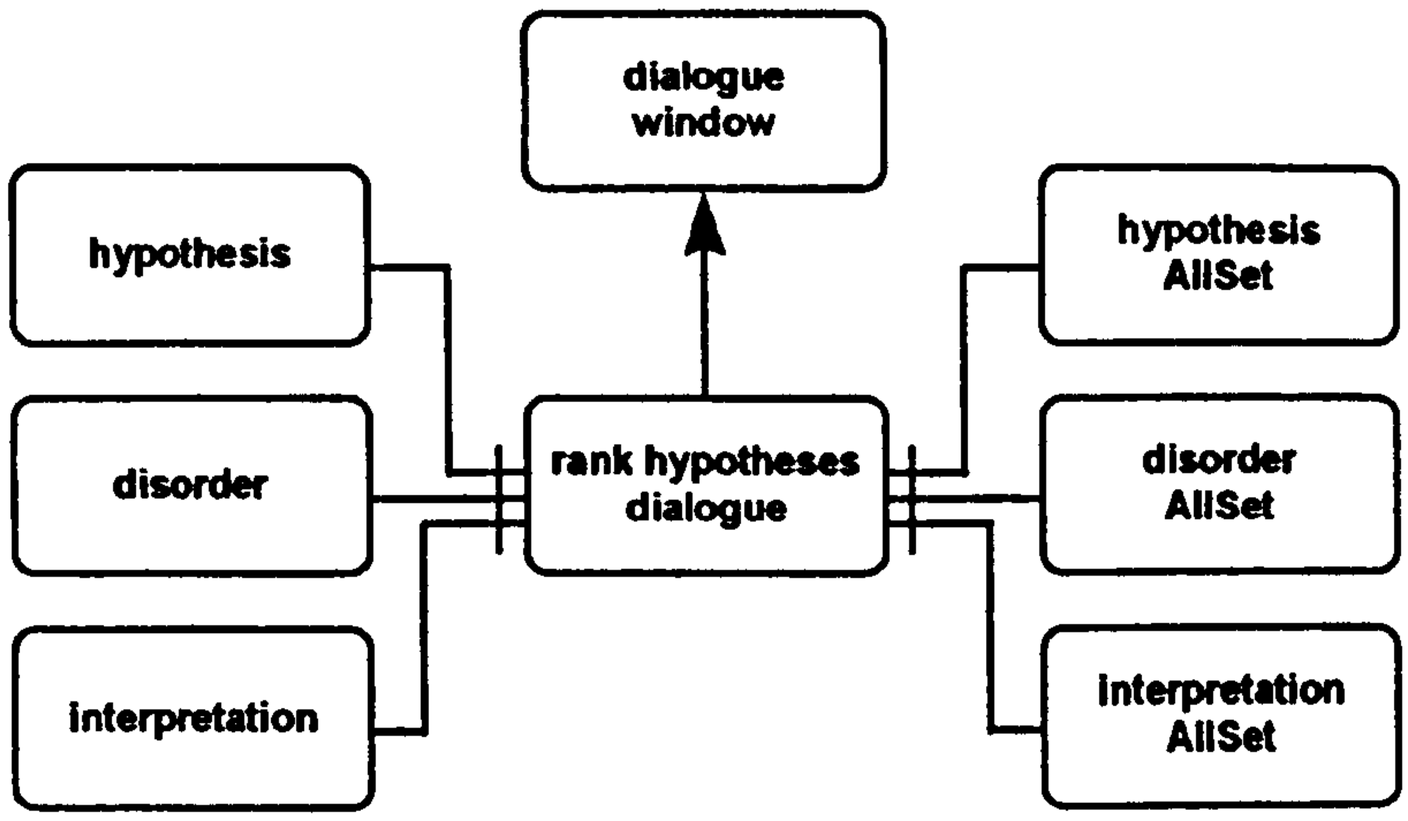


Figure 4-16. Class diagram of rank hypotheses dialogues.

```
class rankHypothesesDialogue : public dialogueWindow
{
  protected:
     ...
     ListBox interpretationsBox;
     ...
     Hypothesis          *pHypothesis;
     Interpretation      *pInterpretation;
     Disorder            *pDisorder;
     DisorderAllSet            *pSetOfAllDisorders;
     HypothesisAllSet          *pSetOfAllHypotheses;
     InterpretationAllSet      *pSetOfAllInterpretations;
protected:
     RankHypotheses();
     SearchInterpretations();
     Aggregate();
     RemoveInterpretation();
     Show();
     virtual Init();
     virtual Commands();
public:
     rankHypothesesDialogue() : dialogueWindow(),
                                ...
     virtual ~rankHypothesesDialogue();
};
```

Listing 4-16. Class interface for the construction and manipulation of normalisation and interpretation selection dialogue objects.

4.7.5. Interpretation Selection Dialogue

Figure 4-17 shows the class diagram of DIS dialogues for the normalisation and selection of interpretative hypotheses from the set generated in the manner described in the last section. More specifically, as shown in Listing 4-17, selectInterpretation-Dialogue objects use the protected Normalise method, via the public Init method, in order to perform the normalisation step described in Section 2.5.3 and implied in the development of the prototype TDM. The class of normalisation dialogue objects will finally select the two top-most ranking hypotheses and store them in the corresponding blood-gas analysis object of the patient data base using the two methods listed in the class interface.

```
class selectInterpretationDialogue : public dialogueWindow
{
  protected:
     ...
     ListBox finalInterpretationBox;
     Interpretation            *pInterpretation;
     InterpretationAllSet      *pSetOfAllInterpretations;
  protected:
     Normalise();
     Select();
     virtual Init();
     virtual Commands();
   public:
     selectInterpretationDialogue(): dialogueWindow(),
                                                ...
     virtual ~selectInterpretationDialogue();
     GetPrimaryDiagnosis();
     GetSecondaryDiagnosis();
} ;
```

Listing 4-17. Class interface for the construction and manipulation of normalisation and interpretation selection dialogue objects.

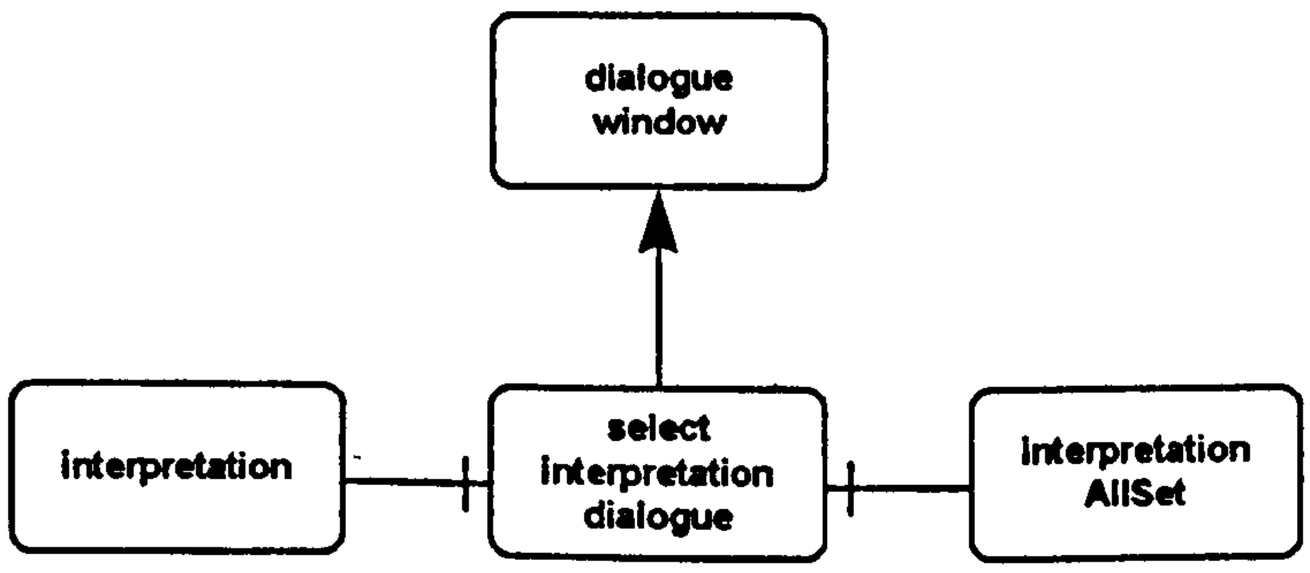


Figure 4-17. Class diagram of select interpretation dialogues.

At this stage, as discussed in Section 3.3, the KBS integration prototype will not be able to resolve complex interpretative hypotheses in order to distinguish their components. The representation and application of the tasks required for this purpose is described in the chapter which follows.

4.8. Conclusion

Following the description of the KBS integration prototype, which implemented a method for the interpretation of blood-gas data using a hierarchical belief classification network, this chapter proceeded to describe the incremental object-oriented design and development process by which the KBS prototype cognitive task-domain model was integrated into the information processing activity of the clinician user. This was achieved by means of the development of global memory system of persistent model derived object structures which were designed to support the management of the clinical information generated in the process of monitoring the ICU patient with disturbed acid-base metabolism, and of the knowledge utilised in the process of interpreting such information in order to support the patient.

The particular approach allowed the development of a uniform information management support structure, which was used to acquire and maintain the data and knowledge required for the functional integration of the KBS prototype, a process which was completed by appending layers for interaction with the clinical advisor, who then constructively assessed the functional, cognitive and ergonomic compatibility of the integrated system with the activity observed in the ICU. The chapter which follows describes these features along with the underlying implementation details of the system's class structure, which was designed as described in the present chapter to facilitate the evolution of such features.

Implementation of the Intelligent ICU Clinical Information Management Support System

5.1. Introduction

In the last chapter, this thesis described the design of an object-oriented clinical information management support system, which was developed to support and facilitate the incremental functional, cognitive and ergonomic integration of the KBS prototype described in the previous chapter, into the clinical information processing activity observed in the data-overloaded ICU, guided by the constructive assessment of the clinician-user-advisor.

Instead of focusing onto the entity abstractions required to support the integration-development process, as did the last chapter, this chapter describes some of the ICIMS system implementation detail, particularly the parts concerning those system features which were developed to make the integrated system functionally, cognitively and ergonomically compatible with the aforementioned activity. The chapter also describes the ways in which the integration-development process may proceed further toward satisfying the user's requirements for clinical information management and decision making support.

5.2. Patient Data Acquisition

5.2.1. User Requirements

As described in Section 4.2, in the process of monitoring and supporting the ICU patient, clinical information bearing evidence as to the patient's state must be collected from a multitude of sources, integrated, recorded, and interpreted within the wider clinical context, in order to provide interpretations appropriate to the context, and thereby guide the clinician decision-maker to effect appropriate patient-state control decisions. Thus, as shown in Figure 4-2 and described further in this chapter,

patient data acquisition is a primary functional requirement for the development of integrated and thus useful ICU clinical decision support KBS.

5.2.2. Integration Implementation

In the ICIMS system, patient data acquisition is performed by the OCS module of the OMS, which, as shown in the top-level architectural view of Figure 4-4 and further in the class diagram of Figure 4-10, was designed to generate dialogues with the user in order to display and control the process of object communication between the available measurement instrumentation, in this case the off-site blood-gas analyser, and the ICIMS COB, via RS232 interfaces.

Figure 5-1 shows the dialogue window generated by the OCS for this purpose. As shown in the figure, the standard Windows WIMP (Windows, Icons, Mice and Pull-down menus) API interface OCS menu bar displays the following five options. If selected, the File option will display a pull-down menu with one further option: Exit; the View option will display the blood-gas analysis assay results shown in the View display window, and Registration the corresponding dialogue window for entering the assay registration details. The Communication Setup menu option displays a pull-down menu with further options for setting up the communication protocol between the ICIMS system and the sources of patient data. Since the ICIMS system was developed for the integration of the blood-gas analysis interpretation KBS prototype, the OCS module monitors only one RS232 line connected to an off-site blood-gas analyser. Finally, as is the case with all ICIMS system modules, the Help option has not yet been implemented.

As soon as the results of the analysis of a patient blood sample are made available by the off-site blood-gas analyser, the OCS will display the View window shown with the communicated results along with the Registration dialogue. Once the details have been supplied, the user clicks the OK button with the mouse and the screen is reset. In order to ensure that no data is lost by failing to specify the patient details

corresponding to the blood sample results being acquired by the OCS, the system checks to see if at least the name and or the hospital number have been specified.

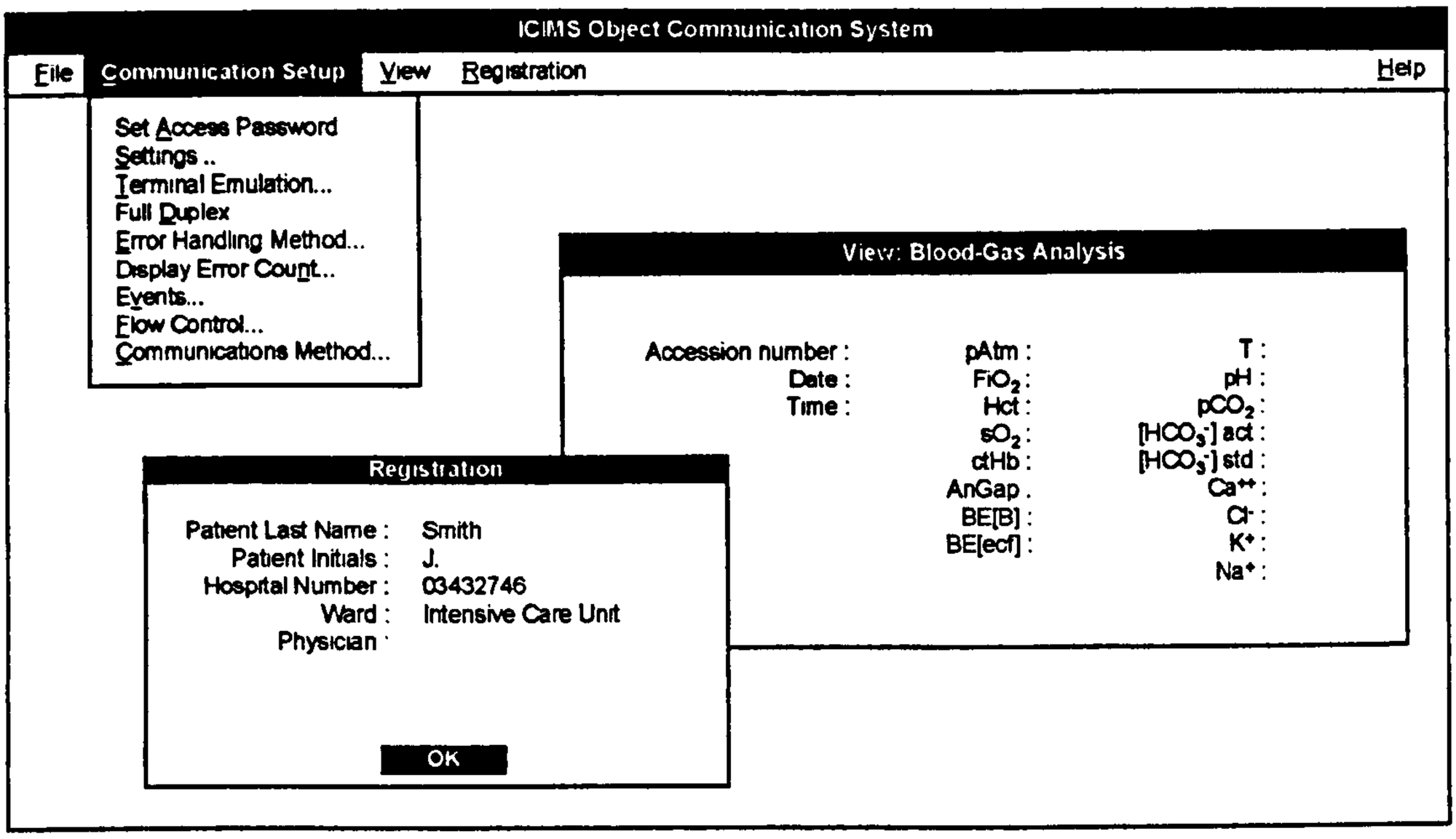


Figure 5-1. The Object Communication System module user-interface window.

Following the registration of the acquired blood-gas analysis results, the OCS will construct a Store object, the implementation details of which are described below, which will perform the necessary actions in order to connect to the PCAS object server, open the ICIMS COB, store the acquired blood-gas analysis results in a patient object structure which it will create for this purpose, close the COB, and finally disconnect from the server. The Store object will then self-destruct and pass control back to the OCS which will await the arrival of further patient data.

```
class ObjectBase
{
private:
   PtBase *pObjectBase;
public:
   ObjectBase();
   virtual ~ObjectBase();
   virtual Execute();
   OpenCOB();
   CloseCOB();
};
```

```
class Store : public ObjectBase
{
private:
   patient                 *pPatient;
   bloodGasAnalysis        *pAnalysis;
   patientAllSet           *pSetOfAllPatients;
public:
   Store();
   ~Store();
   Execute();
};
```

Listing 5-1. Class interface for Store and ObjectBase objects of the object communication system module.

The class interface of Store objects is shown in Listing 5-1. Again, for simplicity, the ICIMS system program listings shown in this chapter are not complete. Furthermore, as stated in Section 5.1, the implementation details are meant to link the decision support functions of the integrated system with the ICIMS class design described in Chapter 4. As shown in Listing 5-1, the class of Store objects inherits the properties of the class of ObjectBase objects and thus an object of the latter class will be constructed before the former. The ObjectBase object's constructor is called as soon as the class is instantiated, resulting in the execution of the ObjectBase constructor method, the implementation of which is shown in Listing 5-2.

```
ObjectBase :: ObjectBase()
{
   pObjectBase = new PtBase;
   OpenCOB();
}
ObjectBase :: ~ObjectBase()
{
   CloseCOB();
   delete pObjectBase ;
}
int ObjectBase :: OpenCOB()
{
   pObjectBase -> Connect();
   pObjectBase -> Open();
}
int ObjectBase :: CloseCOB()
{
   pObjectBase -> Close();
   pObjectBase -> DisConnect();
}
```

Listing 5-2. Class implementation for ObjectBase objects.

The ObjectBase object constructor will assign a pointer to the COB, declared as a private member in the class interface, via the PtBase class shown in Figure 4-6 and then call the OpenCOB method. The OpenCOB method will then use the PtBase methods listed in Table 4-1 in order to connect to the PCAS object server and open the COB. Following its construction, the OCS ObjectBase object will initialise the subclass of Store objects. The implementation details of the latter are shown in Listing 5-3. The Store object constructor will create a patient AllSet, required for searching the COB for a previous registration, and the OCS will then call the Execute method, which for this reason is declared in the public class interface. As shown in Listing 5-3 for the case of a new registration, the Execute member method uses the PtObjectSet and PtObject methods shown in Tables 4-2 and 4-3, as well as the constructed patient object structure's methods, in order to create a patient object structure, initialise it, assign it to the object base, set the registration details, and then proceed to do the same with a blood-gas analysis object which will be appended at the end of the ObjectSet contained in the patient object being created.

```
Store :: Store()
{
   pSetOfAllPatients = new PatientAllSet();
}
Store :: ~Store()
{
   delete pSetOfAllPatients;
}
int Store :: Execute()
{
   ...
   pPatient = new patient;
   pPatient -> Init();
   pPatient -> Assign();
   pPatient -> SetMembers();
   ...
   pAnalysis = new bloodGasAnalysis;
   pAnalysis -> Assign();
   pAnalysis -> SetMembers();
   pPatient -> bloodGasAnalyses.Append ( pAnalysis );
   pPatient -> Store();
   ...
   delete pPatient;
   delete pAnalysis;
}
```

Listing 5-3. Class implementation for Store objects.

Finally, the Store object will store the new, in this case, patient object structure, and delete the temporary objects used during the construction in order to free the memory used in the process. As stated above, control then passes onto the ObjectStore destructor method and subsequently to the OCS module.

5.2.3. Further Development

Due to the vast amount of development effort involved in the integration process, the ICIMS system decision supporting functions were implemented to satisfy base requirements. In the context of supporting the process of collecting and collating patient data, the ICIMS system may be further developed so that the OCS may function independently of other system operations. If so, the OCS module may monitor the execution of the various clinical information management support tasks, and interrupt any process upon the arrival of further data, in order to update the state of the system at any one time. However, such a clinical information processing behaviour would have to be co-ordinated with a blackboard module KBS task-scheduler, as in the case of the Guardian IMC system described in section 2.4.5, with real-time constraints on the application of the represented cognitive task-domain. Furthermore, since most operations performed by the ICIMS system involve accessing the COB, the OMS module would have to be further developed to introduce concurrency, transaction management and event handling, using the available advanced PCAS methods.

5.3. Clinical Information Management

5.3.1. User Requirements

By the definition which emerges from the material presented in this thesis, an integrated clinical decision support KBS is a system which provides facilities for the management of the clinical information comprising the knowledge base required for

the application of the represented cognitive task-domain, as well as of the clinical information acquired in order to perform the represented decision supporting functions and the decision supporting information generated in the process.

As described in the thesis introduction, such system architectures have a number of advantages over the traditional standalone consultation systems. Firstly, they are developed to provide essential and effective solutions to actual medical decision support problems, encountered in the data-overloaded environment of high-technology medicine, rather than to function as human expert consultation replacements. This is so, both in the sense of gearing the application of the underlying knowledge-based techniques toward this objective, as well as in the sense that they are designed to integrate, record, organise, and display the clinical information acquired and generated in the process, in a manner which provides further valuable clinical decision support. Secondly, by means of the provision of effective decision support in the latter sense, integrated KBS prototypes can justify their introduction into routine clinical practice, thereby being exposed to the clinical information processing environment and the amount and quality of data required for formatively and constructively assessing the evolution of the represented knowledge base and cognitive task-domain. Thirdly, by being able to maintain temporal records of the patient's progression and of the results of the application of the represented task-domain, integrated systems provide the framework for the development, representation, application and evaluation of advanced cognitive task-domains, as for example in the case of the Guardian system.

5.3.2. Integration Implementation

Figure 5-2 shows the ICIMS system main window screen, which, as described in Section 4.6 and shown in Figure 4-10, forms the link between the user, the OMS module, and the underlying persistent COB structure. Thus, there are options for generating patient data management dialogues with the user, in order to update and review the information stored in the persistent patient COB object structures described in Section 4.5.2, and disorder knowledge management dialogues, in order

to update and review the information stored in the persistent disorder COB object structures described in Section 4.5.3.

```
ICIMSWindowsApp :: ICIMSWindowsApp() : WindowsApplication()
{
   ObjectServer = new PtBase();
   ObjectServer -> Connect();
   ObjectServer -> Open(COB);
}

ICIMSWindowsApp :: ~ICIMSWindowsApp()
{
   ObjectServer -> Close (COB);
   ObjectServer -> DisConnect();
}

ICIMSSystemWindow :: Commands ()
{
   case EDIT_PATIENT:
         patientDialogue editPatientObject();
         editPatientObject.Execute();
   case EDIT_DISORDER:
         disorderDialogue editDisorderObject();
         editDisorderObject.Execute();
   case EDIT_PROFILE:
         parameterProfileDialogue editProfileObject();
         editProfileObject.Execute();
   case EDIT_AETIOLOGY_VOCABULARY:
         recognisedAetiologyDialogue editVocabulary();
         editVocabulary.Execute();
     case STATE_TREND:
         trendDialogue trend();
         trend.Execute ();
}
```

Listing 5-4. Implementation of the ICIMS system superclasses.

Once the system is started, the main control thread of the Windows application environment will create instances of the two top-level ICIMS superclasses. The main ICIMS system window shown in Figure 5-2 is the result of the construction of an ICIMS system superclass instance, the interface of which class is shown in Listing 4-8. The other superclass, shown in Figure 4-6, is the ICIMS system COB superclass. The implementation of both superclasses is shown in Listing 5-4. As shown in Listing 5-4, upon construction, the COB superclass will assign a new pointer to the object base server via the PtBase class described in Section 4.5.1, and proceed to connect to the server and open the COB using the methods shown in Table 4-1. At the same time, the Windows system will create an instance of the main ICIMS windows display

and dialogue control superclass and await for the user commands shown in pull-down windows in Figure 5-2.

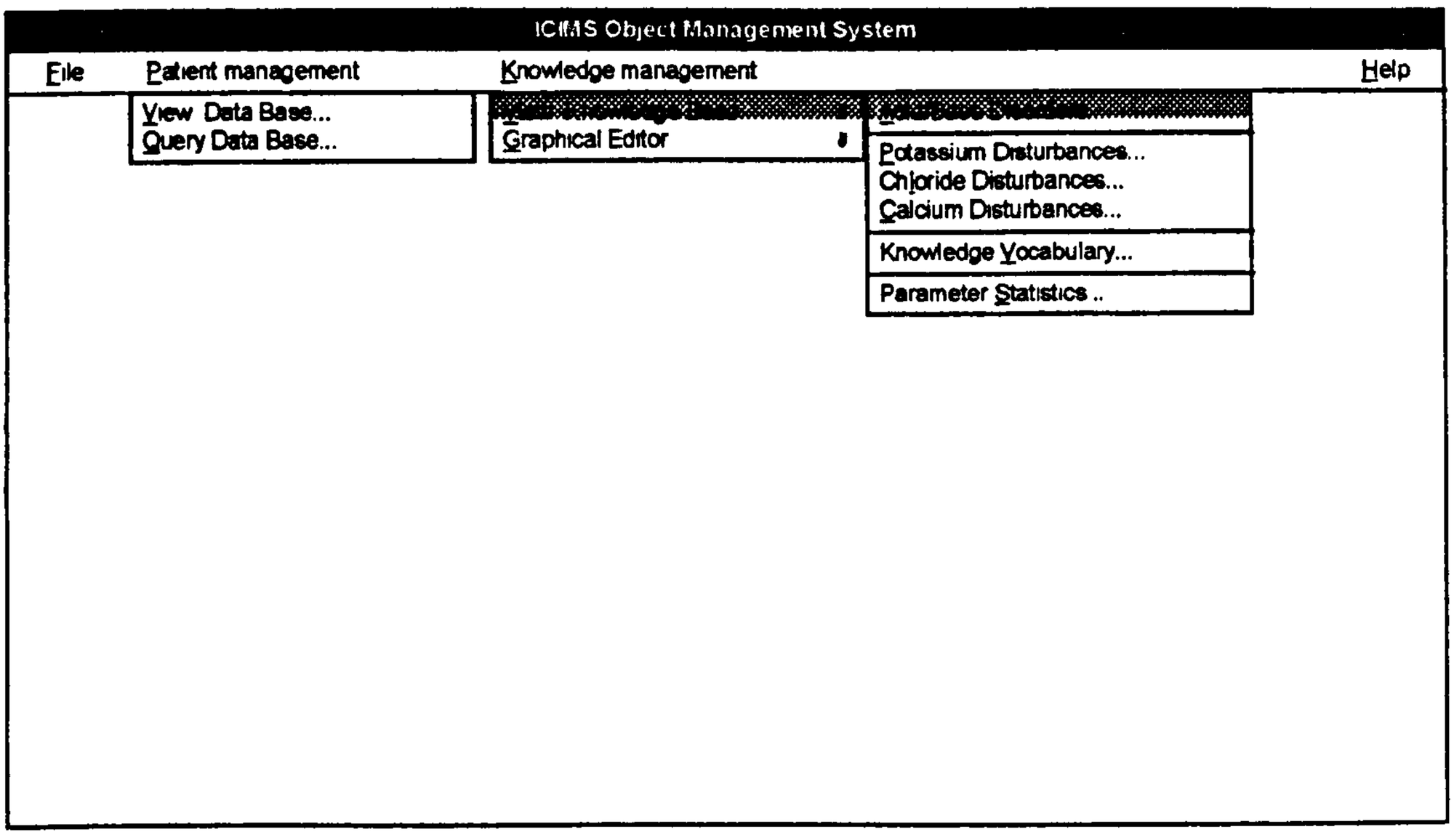


Figure 5-2. The Object Management System main user-interface window.

```
objectManagementDialogue :: Commands()
{
   case STORE:
         if GetAccessCode() = correct
         Fill();
         GetPtObject() -> Store();
   case DELETE:
         if GetAccessCode() = correct
         GetPtObject() -> Delete();
   case CLEAR:
         if GetAccessCode() = correct
         InitObject();
         GetPtObject() -> UnAssign();
         GetPtObject() -> Assign();
         Show();
case NEXT:
         NextObject();
         Show();
case PREVIOUS:
         PrevObject();
         Show();
case STEP:
         StepObject();
```

```
                Show();
        }
```

Listing 5-5. Class implementation for persistent object management dialogues.

The dialogue objects created by the various options or cases in the Commands method of the class ICIMSSystemWindow were described in Section 4.6. The implementation of the classes and the generated dialogues are described below, along with the new class of patient state trend dialogues.

As described in Section 4.6, each class of persistent object review and update dialogues, whether of the data base or the knowledge base OMS module, inherits the properties of the persistent object management dialogue class, the implementation of which is shown in Listing 5-5. Thus, patient, disorder, parameter profile, and recognised aetiology dialogue windows will display and process command buttons which respond as shown in Listing 5-5.

5.3.2.1. Patient Data Management Dialogue

An example dialogue generated by the OMS to reflect the anatomy of the persistent patient data object structures contained in the ICIMS COB, and to thereby facilitate the review and update of the contents of such object structures, is shown in Figure 5-3. The vertically arranged buttons on the right correspond to the command cases described above. The top of the patient dialogue pop-up window shown in the figure, displays the contents of the private members shown in the patient class interface in Section 4.5.2, within edit boxes as shown in the dialogue class interface of Listing 4-10. Finally, there are list boxes which list the contents of the aetiologies vocabulary, the patient's clinical features, the acquired blood-gas analysis results, and the interpretation of the acquired patient data. The aetiologies vocabulary is part of the knowledge COB and is used here to add recognised aetiologies to the list of clinical features for each patient. The other list boxes reflect the contents of the sets of objects contained in each patient object as shown in Figure 4-7. By pressing the New, Update and Delete buttons, the user may append a new set of patient data manually, update an existing set, or delete an existing set, respectively, if granted access to the COB by

entering the correct code using the access code edit box shown at the bottom right-hand corner of the window. Finally, the Evaluation button calls the DIS in order to instantiate the TDM as described in Section 4.7, in order to interpret a particular data set. This is done by selecting one of the lines in the list box, as shown in the figure by the shaded area, and pressing the Evaluate button. The same holds for the Update and Delete buttons. Figure 5-4 shows the pop-up window which appears upon selecting a set of blood-gas analysis results and pressing the update button. Again, this dialogue window screen reflects the contents of the blood-gas analysis objects, the class interface of which is shown in Listing 4-2.

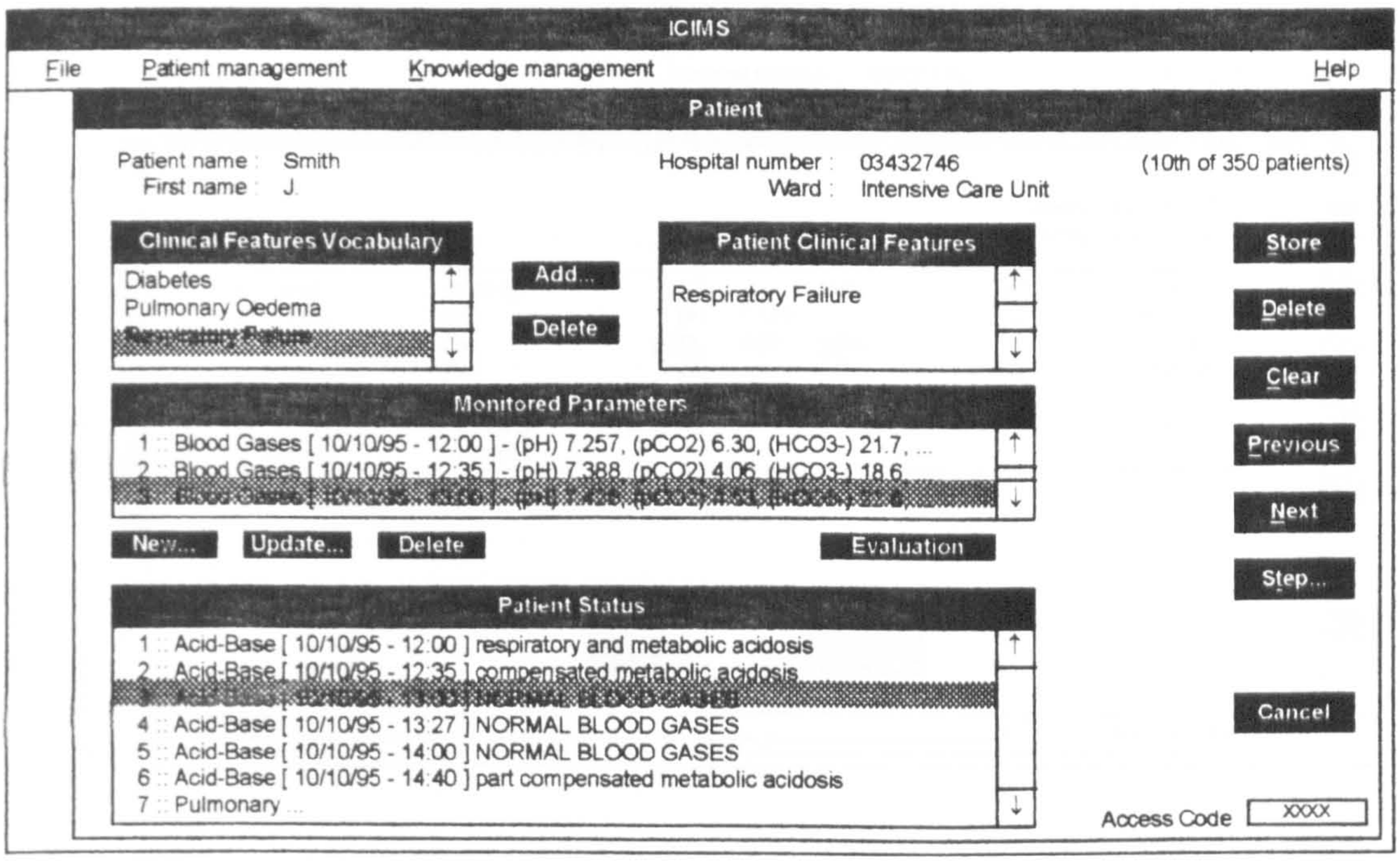


Figure 5-3. Patient dialogue window.

```
patientDialogue :: Init()
{
   accessCode.DeleteContent ();
   pSetOfAllPatients = new patientAllSet();
   pSetOfAllRecognisedAetiologies = new
       recognisedAetiologyAllSet();
   ...
   pSetOfAllPatients -> Seek(last patient);
   pSetOfAllPatients -> Get(pPatient);
   if patientAllSet empty
```

```
        pPatient = new Patient();
        InitObject();
        pPatient -> Assign();
    Show();
    ...
}

void patientDialogue :: InitObject()
{
    pPatient -> InitialiseObject();
}
```

Listing 5-6. Implementation of the object initialisation method for the class of patient data management dialogues.

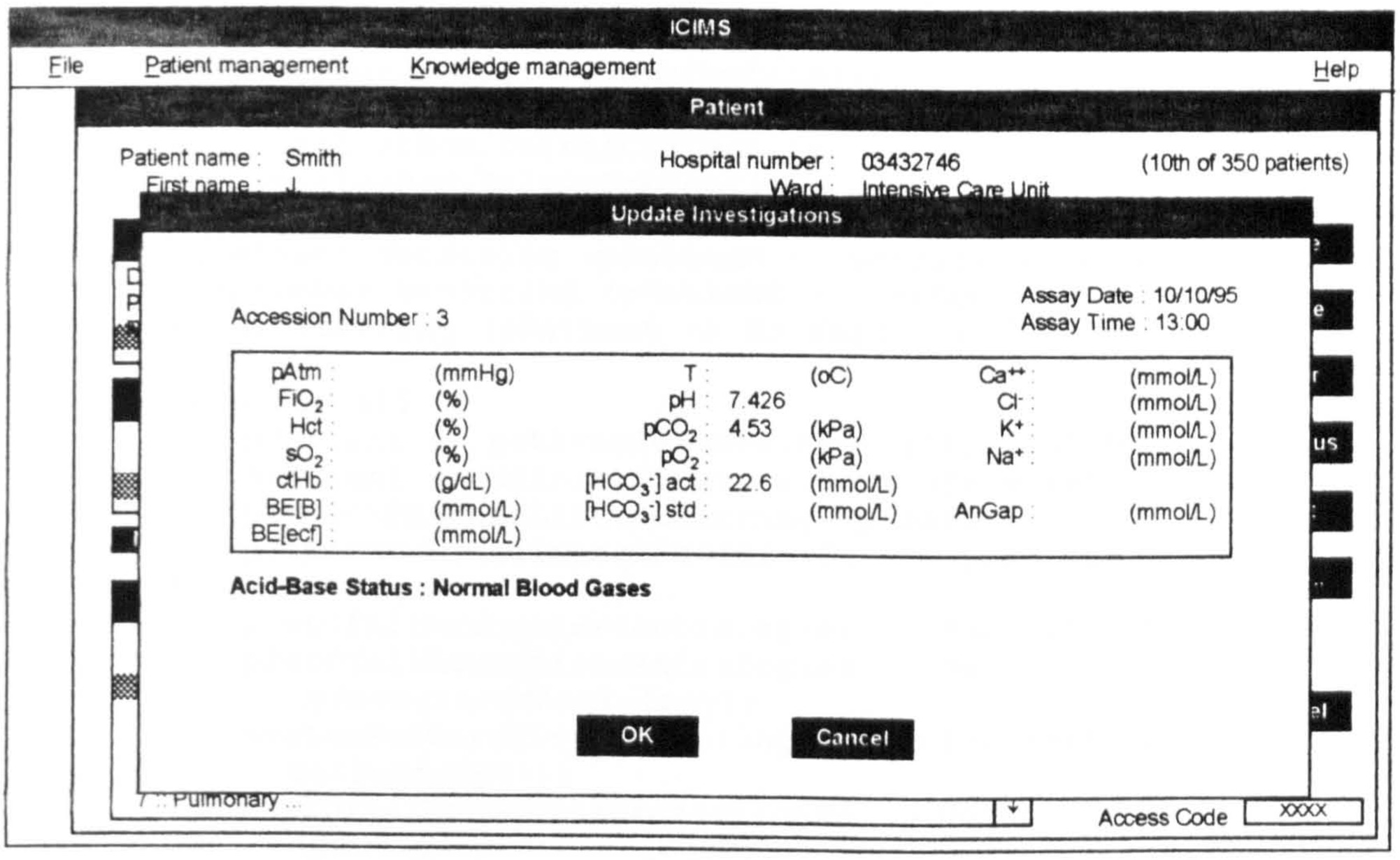


Figure 5-4. Blood-gas analysis dialogue window.

Listing 5-6 shows the implementation details of the patient dialogue object initialisation method shown in the class interface of Listing 4-10. As shown in the method implementation, upon construction of the dialogue object by the OMS and the corresponding menu selection user command shown in Figure 5-2 and Listing 5-4, the initialisation method will construct the two AllSet objects declared as protected in the class interface, namely the patientAllSet required to search the patient COB and the clinicalFeatures-AllSet required to display the patient features recognised by the

system. If the patientAllSet is empty, then the dialogue object will construct a new patient object as described above, initialise the object by indirectly calling the object's method shown in Listing 4-1, and assign the new object to the object base. Finally, the dialogue object will use the Show method described below to display the contents of the new patient object, or the last patient object for the case where the patientAllSet was not empty. Listing 5-7 shows the implementation of the patient dialogue Show method.

```
void patientDialogue :: Show()
{
   name.DeleteContent ();
   firstName.DeleteContent ();
   hospNumber.DeleteContent ();
   ward.DeleteContent ();
   patientFeatureList.DeleteContent();
   systemFeatureList.DeleteContent();
   analysisListBox.DeleteContent();
   analysisLogBox.DeleteContent();
   name.SetString (pPatient -> GetName());
   firstName.SetString (pPatient -> GetFirstName());
   hospNumber.SetString (pPatient -> GetHospNumber());
   ward.SetString (pPatient -> GetWard());

   loop for all
      pPatient -> patientFeature.Seek (1L, PtSTART);
      pPatient -> clinicalFeature.Get (pfeature);
      patientFeatureList.AddString (pfeature);
      pPatient -> clinicalFeature.Unget (pfeature);
loop for all
      pSetOfAllRecognisedAetiologies -> Seek (1L, PtSTART);
      pSetOfAllRecognisedAetiologies -> Get
        (pRecognisedAetiology);
      systemFeatureList.AddString (pClinicalFeature ->
        GetFeature());
      pSetOfAllRecognisedAetiologies -> Unget
        (pRecognisedAetiology);

BloodGasAnalysis *pAnalysis;
loop for all
      pPatient -> bloodGases.Seek (1L, PtSTART);
      pPatient -> bloodGases.Get (pAnalysis);
      analysisListBox.AddString (pAnalysis ->
      DisplayOBject());
            analysisLogBox.AddString (pAnalysis -> PrintLabel());
            pPatient -> bloodGasAnalyses.Unget ( pAnalysis ),
}
```

Listing 5-7. Implementation of the patient data management dialogue Show method.

As shown in Listing 4-10, the Show method was declared virtual in the class interface since its implementation was to be defined in one of the persistent object management

dialogues. Listing 5-7 shows the particular implementation. The Show method will first clear the contents of all the edit boxes and list boxes used by the class of OMS dialogues and display the reviewed object's contents. Following that, the dialogue object will loop through the patient feature and recognised aetiology AllSets in order to display the two list boxes shown at the top of Figure 5-3. Finally, using a pointer to blood-gas analysis objects, the method will loop through the corresponding AllSet and display the other two list boxes shown in Figure 5-3. Finally, Listing 5-8 shows the implementation of the Fill, NextObject, PrevObject methods used in the implementation of the object management dialogues as shown in Listing 5-5, declared in Listing 4-9, and re-declared in Listing 4-10.

```
void patientDialogue :: Fill()
{
...
pPatient -> SetMembers(Editbox,GetContents());
pPatient -> patientFeatures.Clear();
for all features in the list box
   patientFeaturesList.GetString();
   pPatient -> patientFeatures.Append(feature);
}

patientDialogue :: NextObject()
{
...
pSetOfAllPatients -> Seek (1L, PtCURRENT);
pPatient -> Forget ();
pSetOfAllPatients -> Get (pPatient);
}


patientDialogue :: PrevObject()
{
...
pSetOfAllPatients -> Seek (-1L, PtCURRENT);
pPatient -> Forget ();
pSetOfAllPatients -> Get (pPatient);
}

patientDialogue :: StepObject()
{
pSetOfAllPatients -> Seek (step);
pPatient -> Forget ();
pSetOfAllPatients -> Get (pPatient);
}
```

Listing 5-8. Implementation of the patient data management dialogue Fill, NextObject, PreviousObject and StepObject methods.

The user commands implementation for controlling the access to the objects contained within patient objects is shown in Listing 5-9.

```
patientDialogue :: Commands()
{
   case FEATURE_DEL:
      item = patientFeatureList.SelectedItem
      patientFeatureList.DeleteItem ();
      pPatient -> clinicalFeatures.Seek (item, PtSTART );
      pPatient -> clinicalFeatures.Delete ();
   case FEATURE NEW:
      item = systemFeatureList.SelectedItem
      systemFeatureList.GetString (buffer,
        systemFeatureList.SelectedItem());
      patientFeatureList.AddString (buffer);
case BLOODGASDATA_NEW:
      bloodGasAnalysisDialogue analysisDialogue ();
      analysisDialogue.Execute ();
      analysisLogBox.AddString (analysisDialogue.GetMembers());
      analysisListBox.AddString
      (analysisDialogue.GetMembers());
      bloodGasAnalysis *pAnalysis = new BloodGasAnalysis;
      pAnalysis -> Assign ();
      pPatient -> bloodGases.Append (pAnalysis);
      pAnalysis -> Forget ();
case BLOODGASDATA_UPD:
      item = analysisListBox.SelectedItem ();
      bloodGasAnalysisDialogue analysisDialogue ();
      bloodGasAnalysis *pAnalysis;
      pPatient -> bloodGases.Seek (item, PtSTART );
      pPatient -> bloodGases.Get (pAnalysis);
      analysisDialogue.SetMembers (pAnalysis -> GetMembers());
      analysisDialogue.Execute ();
      analysisLogBox.
        ChangeString (analysisDialogue.GetMembers());
      analysisListBox.
        ChangeString (analysisDialogue.GetMembers());
      *pAnalysis = analysisDialogue.dialogueAnalysis;
      pPatient -> bloodGases.Put (pAnalysis);
      pPatient -> bloodGases.Unget (pAnalysis);
   case BLOODGASDATA_DEL:
      item = analysisLogBox.SelectedItem ();
      analysisLogBox.DeleteItem (item);
      analysisListBox.DeleteItem (item);
      pPatient -> bloodGases.Seek (item, PtSTART);
      pPatient -> bloodGases.Delete ();
}
```

Listing 5-9. Implementation of the patient data management dialogue contained objects user command message processing method.

As discussed in Section 4.6, any modifications made using the OMS commands shown in Listing 5-9, will be stored automatically by the PCAS as shown in Figures 4-11 and 4-12, and further in Listing 5-5.

```
bloodGasAnalysisDialogue :: Init()
{
   accNumber.SetString (dialogueAnalysis.GetMember());
   ph.SetString (dialogueAnalysis.GetMember());
   pco2.SetString (dialogueAnalysis.GetMember());
   ...
}

bloodGasAnalysisDialogue :: Commands()
{
   dialogueAnalysis.SetMember (accNumber.GetString());
   dialogueAnalysis.SetMember (pH.GetString());
   ...
}
```

Listing 5-10. Implementation of the blood-gas analysis data management dialogue class methods.

Finally, Listing 5-10 shows the relatively simple implementation of blood-gas analysis dialogues, specifically, the methods for dialogue initialisation, which involves displaying the currently selected blood-gas object, and command message processing, which as shown in Figure 5-4 refers to the OK and Cancel buttons. Cancel will not make any changes, whereas OK will result in setting the private members of the blood-gas analysis object, as shown in Listing 4-2, to the new values taken from the edit boxes shown in Figure 5-4.

5.3.2.2. Disorder Knowledge Management Dialogue

Figure 5-5 shows an example dialogue generated by the OMS to reflect the anatomy of the persistent disorder knowledge object structures contained in the ICIMS COB, and to thereby facilitate the review and update of the contents of such object structures, in a manner identical to that described in Section 5.3.2.1 above. The top of the acid-base balance disorder knowledge dialogue shown in Figure 5-5 displays the name of the disorder being reviewed and possibly updated, the a priori belief in the disorder, and the links to other disorders, required to form the belief network shown

in Figure 3-1. As shown in Listing 4-11 and Figure 4-12, the class of disorder dialogues makes use of the recognised aetiologies AllSet in order for the user to specify the patient's features, as in the case of patient dialogues. The operation, ie. implementation, of the buttons corresponding to the two list boxes for reviewing and updating the contents of the evidence and aetiological patient features, which are part of the disorder knowledge model shown in Figure 4-8, is the same as that described in Section 5.3.2.1 for the case of the blood-gas analysis objects contained in persistent COB patient object structures.

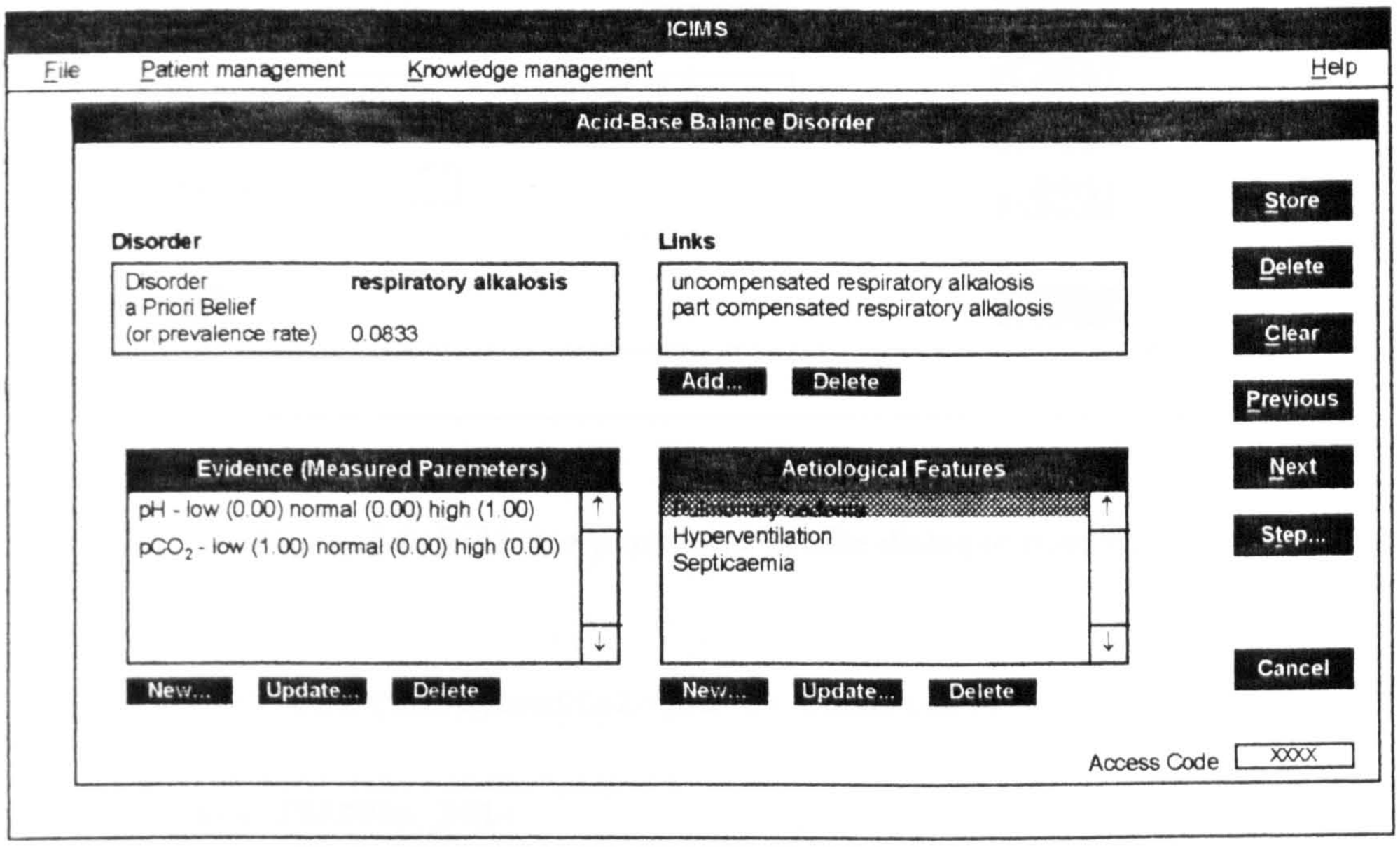


Figure 5-5. Disorder knowledge dialogue window.

Figure 5-6 shows an example of the evidence parameter profile update and review dialogue window, which is created and displayed by the OMS via the instantiation of the class of persistent object management dialogues as shown in Figure 4-12 and further in Listing 5-4. Finally, Figure 5-7 shows an example of the recognised aetiologies dialogue, which is also instantiated via a persistent object management dialogue object as shown in Figure 4-12. In this case, the only difference from the implementation of the other persistent object management dialogue subclasses is the

implementation of the user control command message processing method, which for this reason is shown in Listing 5-11.

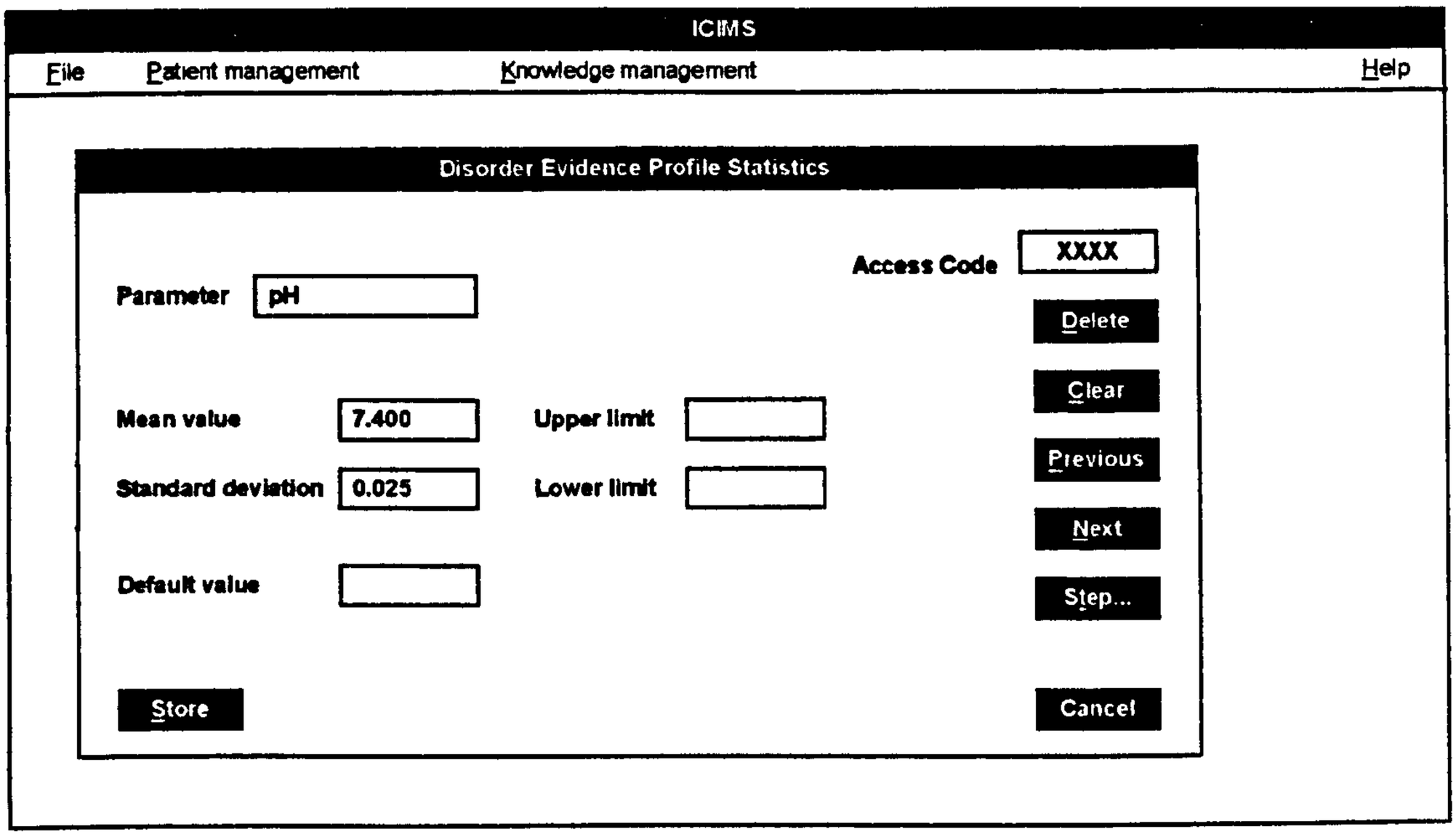


Figure 5-6. Evidence parameter profile dialogue window.

```
recognisedAetiologiesDialogue :: Commands()
{
   ...
   case FEATURE_DEL:
       item = featuresBox.SelectedItem ();
       featuresBox.DeleteItem ( item );
       pSetOfAllRecognisedAetiologies -> Seek (item, PtSTART);
       pSetOfAllRecognisedAetiologies -> Get (pFeature);
       pFeature -> Delete();
       pSetOfAllRecognisedAetiologies -> Unget (pFeature);
   case FEATURE_ADD:
       feature.GetString ();
       featuresBox.AddString ();
       pFeature = new recognisedAetiology ();
       pFeature -> Assign ();
       pFeature -> SetMember ();
       pFeature -> Store ();
       delete pFeature;
   }
```

Listing 5-11. Implementation of the Commands method of the class of recognised aetiologies dialogues.

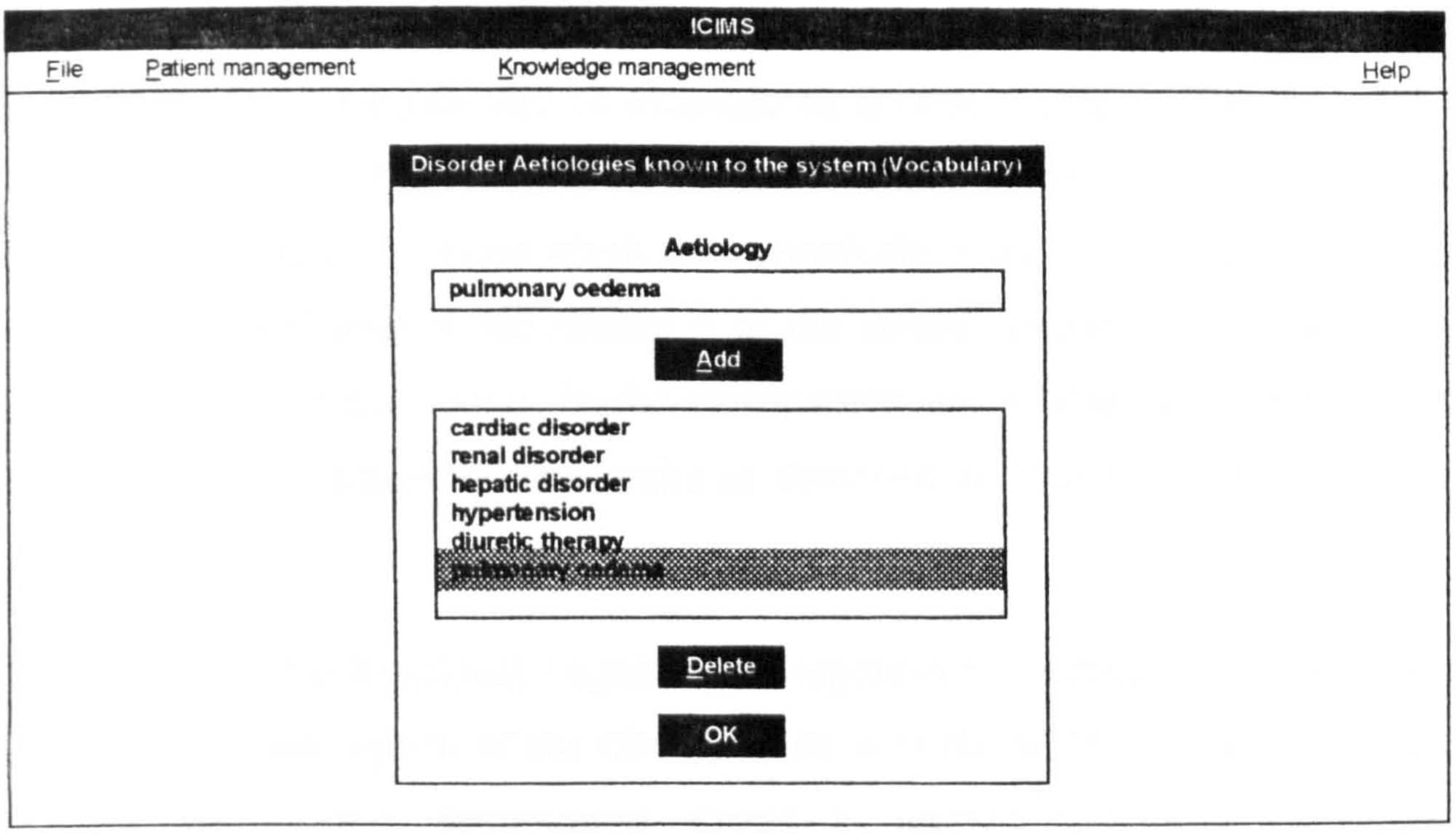


Figure 5-7. Recognised aetiologies dialogue window.

5.3.3. Further Development

As stated in Chapter 4, and further in Section 5.3.1, the OMS module layer of the ICIMS system was designed to support and facilitate the functional integration of the prototype knowledge-based clinical decision support system, within the clinical information processing activity observed in the process of monitoring and supporting the ICU patient. Thus, the patient data and disorder knowledge OMS module dialogues were designed to reflect the structure of the corresponding classes of COB objects, and to thereby be used to derive the underlying persistent object structures and to provide a visual representation of their contents for update and review.

The implementation of the OMS module may be extended by appending further layers to the ICIMS system architecture shown in Figure 4-4, which will make use of the existing OMS layer in order to further the integration-development process and augment the functional, cognitive and ergonomic compatibility of the system with the clinical activity of the user, thereby providing further clinical information management

and decision support. For example, as shown in Figure 5-2, the disorder knowledge management OMS module may be extended to provide a graphical display of the corresponding COB object structures, thereby providing the users with a knowledge review and update environment which is ergonomically and cognitively closer to their understanding and internal representation of the clinical taxonomy of disease. This tool will be of greater utility in the construction and evaluation of aetiological pathways to the represented taxonomies as described in Section 2.4 and further in Section 5.4 below.

With respect to the functional, cognitive and ergonomic compatibility of the patient (data) management option of the OMS module with the health care activity of the clinician user, further development should be geared toward providing any information management and decision support features required to satisfy the following two objectives (Groth and Collinson, 1993):

1. Data must be complete, accurate, timely and easily accessible.

2. Data must be displayed and transformed in a manner which converts it to information.

The existing health care delivery system uses paper records for this purpose, that is, in order to summarise the medical history of the patient being treated, and to document the observations, diagnostic conclusions, and management plans made by health care personnel. However, logistical and practical limitations reduce the effectiveness of the existing medical record system for collating, storing, organising and displaying large numbers of diverse data, often leading to the misinterpretation of an observed clinical problem. As described in the introduction to this thesis and further in Section 4.2.2, this is particularly true in the critical care medical environment which can be regarded as an exemplar of what holds true for the wider health care system.

Evaluation studies indicate that computer-based medical record systems (MRS) offer solutions to many of the medical decision support problems that arise from the limitations of the paper record system in dealing with the increasing volume of clinical information (Whiting-O'Keefe, 1985; McDonald and Barnett, 1990). MRS may

provide effective clinical decision support both in the sense that they can facilitate access to the information contained in the acquired patient data by performing elaborate data base queries, as well as in that they may function as surveillance or monitoring systems, using the knowledge-based techniques described in Chapter 2 in order to detect and flag patient conditions that need medical attention.

Well known surveillance systems include the HELP system (Warner, 1979; Pryor et al, 1983; Pryor, 1988), MQL which runs under COSTAR (Barnett et al, 1979), and CARE (McDonald, 1981), which runs under the Regenstrief Medical Record System (McDonald et al, 1983, 1988). The Regenstrief Medical Record System (RMRS) was developed by McDonald and his colleagues at the Indiana University Medical Centre and has been in operation at Wishard Memorial Hospital since 1974. By 1988, it maintained medical histories for over 250,000 patients, containing almost 25 million separate patient observations, all of which are encoded and fully retrievable. RMRS is part of a larger administrative support system that handles appointment scheduling and charge capture. The unique feature of the MRS component is a reminder system that actively reviews patient data and produces reminder notes for the physician based on 1400 encoded protocol rules. An evaluation study of the initial version of the system demonstrated that the reminders significantly improved the behaviour of physicians in remembering to order laboratory tests when appropriate, and in prescribing or modifying medication plans. For example, physicians given computer reminders quadrupled the use of certain vaccines in eligible patients, compared to those who did not receive reminders (McDonald et al, 1984). The system has since been extended to incorporate the HyperCritic critiquing task-specific KBS architecture (van der Lei et al, 1990; van der Lei and Musen, 1991).

One of the ways in which the patient record OMS module may possibly be extended was described in Section 4.7 and is demonstrated in Section 5.4 below, in the context of appending the DIS layer onto the basic OMS layer for the knowledge-based manipulation of the COB contents in order to generate high-level patient summaries and alarms, and interpretative trend displays. Furthermore, as shown in Figure 5-2, in each class of persistent patient object management dialogue there are two possible selections: view (and update) the underlying persistent object structures and query the

structures. The second option provides access to a number of query methods provided by the PCAS, an example of which was given in Section 4.7.2, in order to perform more elaborate searches of the COB contents than those possible via the patient data dialogues.

5.4. Contextual Interpretation of Acquired Patient Data

5.4.1. User Requirements

As stated in the last section, the functional, cognitive and ergonomic integration of computer-based clinical decision support tools within the information processing activity of the health care professional, should be geared toward making the patient data set recorded in routine clinical practice complete, accurate, timely and easily accessible. Furthermore, patient data must be displayed and transformed in a manner which converts it to information, thereby avoiding the misinterpretation and consequentially mismanagement of an observed clinical problem under conditions of information overload and contextual complexity.

The first part has been addressed via the development of the automated OCS and comprehensive OMS modules described above. With respect to the second part, Figure 5-8 shows the range of interpretative abstraction reasoning against the range of temporal abstraction reasoning employed by KBS in the representation and application of the cognitive task-domain of intelligent ICU patient monitoring and control (Uckun, 1994), which was described in Section 4.2. Other orthogonal dimensions, such as classification or simulation reasoning and the method of abstraction, were described in Chapter 2. At one end of the interpretation spectrum shown in Figure 5-8, inference methods interpret values of simple parameters. If the system considers only single data points, the inference operation is called classification, which corresponds to a number-to-symbol transformation. If the system interprets sequences of data, the inference method is referred to as trend detection. The next level of abstraction in the interpretation of an observed clinical problem is

state-based abstraction or contextual parameter interpretation. This level is the domain of KBS, which synthesise interpretations of parameter values into qualitative interpretations of physiological and pathophysiological states. If the interpretation is extended into sequences of states in time, a state-space trajectory is obtained. The decomposition of an observed clinical problem into sequences of clinical events may subsequently be used to synthesise interpretations of complex problems which may not be recognised otherwise (for example, see Cohn et al, 1990). Finally, KBS may synthesise causally or taxonomicaly ordered collections of pathophysiological states into diseases and complications.

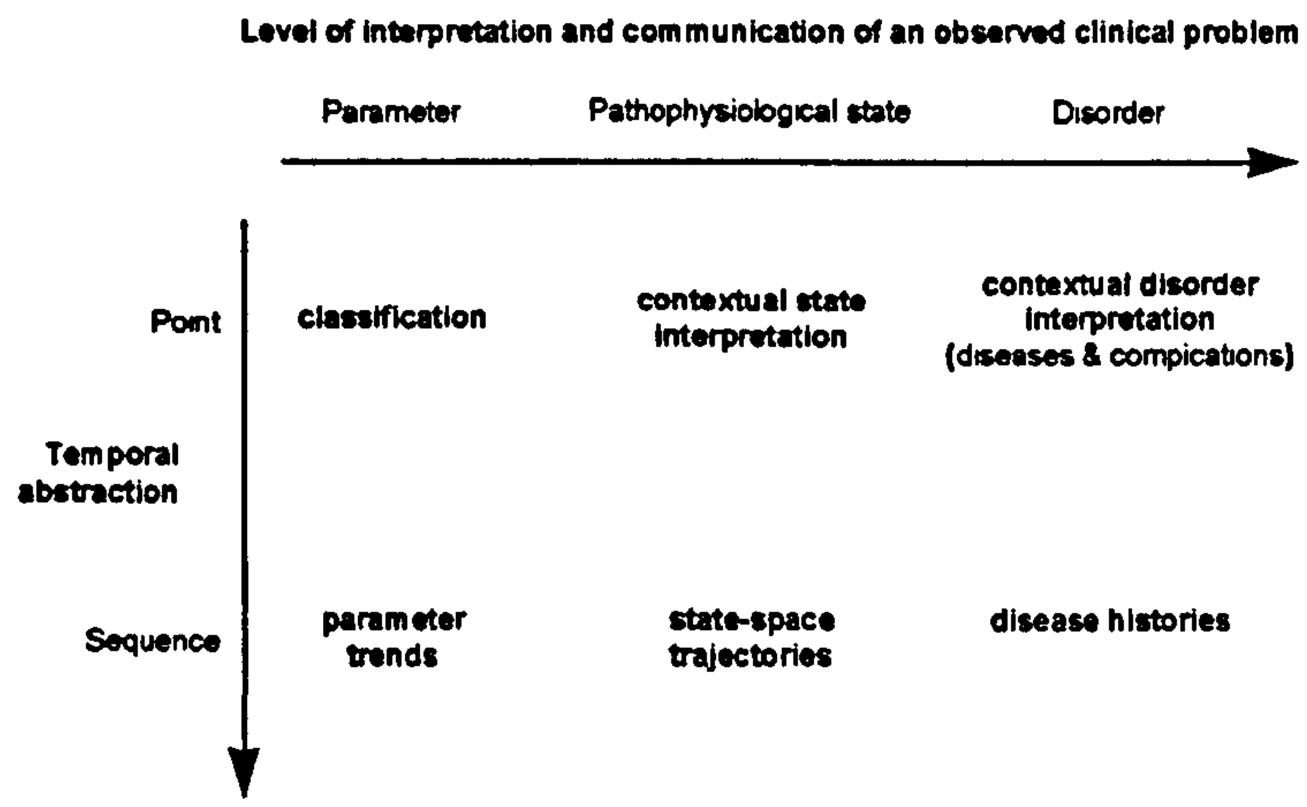


Figure 5-8. Interpretative inferences in the task domain of intelligent monitoring and control (from Uckun, 1994).

Existing ICU patient monitoring systems employ rudimentary inferencing strategies for the purpose of surveillance, in the form of what may be termed "smart" alarms, based on methods of algorithmic inference, such as high and low critical values, calculation-adjusted critical values, and the recognition of adverse critical trends independent of critical limits (Shabot et al, 1989, 1990). However, normal value limits or reference ranges are extremely difficult to define due to both inter-individual as well as intra-individual variations in human physiology (Chelsom, 1990). Thus, the interpretation of a single variable in isolation often leads to false alarms, further confusion and pre-emption against such tools. By contrast, the term intelligent alarms

is used to refer to the contextual, knowledge-based interpretation of monitored patient parameters, as described above.

5.4.2. Integration Implementation

Section 4.4 briefly described how, for each set of acquired patient data (in the case of the integration prototype, blood-gas data), the DIS module of the ICIMS system will instantiate a PSM class structure by applying the reasoning operators that comprise the represented TDM knowledge sources described in Sections 3.2.2.5 through to 3.2.2.7. So, as shown in Figure 4-4 for patient [X], ICIMS will construct the patient-specific model [X.1] of the patient's interpretative hypothesis space for disorders of acid-base metabolism, given the corresponding set of blood-gas measurements. Section 4.7 described the design of the interfaces to the classes of DIS dialogue generated in the process, and the mechanisms employed in the search and manipulation of the COB objects required for the application of the reasoning operators represented in each class of dialogues. As shown in Figure 5-3, the DIS is called to generate an interpretation of a set of blood-gas measurements by selecting the set from the list box and pressing the Evaluation button under the box. Listing 5-12 shows the implementation of the Commands method of the class of patient dialogues, which is omitted in Listing 5-9, for the instantiation of the interpretative dialogue classes.

More specifically, following the selection of a blood-sample measurement to be interpreted as to the patient's acid-base status, the DIS module will construct dialogue objects from their class description and execute their implementation in the order in which they are presented in Section 4.7 and shown in Listing 5-12. Once the two topmost-ranking interpretative hypotheses have been selected by the select interpretation dialogue object, the DIS module, represented in the current implementation as the code shown in Listing 5-12, will set the corresponding member attributes of the selected blood-gas analysis object using the methods provided by the class of objects as shown in Listing 4-2. The dialogue windows generated by these interpretative method abstraction objects are shown in Figure 5-9. Finally, as shown in

the interpretation selection window, at this stage the user has two options: to either cancel the process and store the displayed interpretations, or to continue the interpretative process in order to either verify the hypotheses or to resolve complex hypotheses by means of causal-process classification as described in Section 3.3 and further in Section 5.4.6 below. The sections which follow also discuss the verification of the prototype KBS integration, the evaluation of the DIS against peer systems, and the high-level alarm and patient summary decision support functions provided by the system in the form of interpretative state trends.

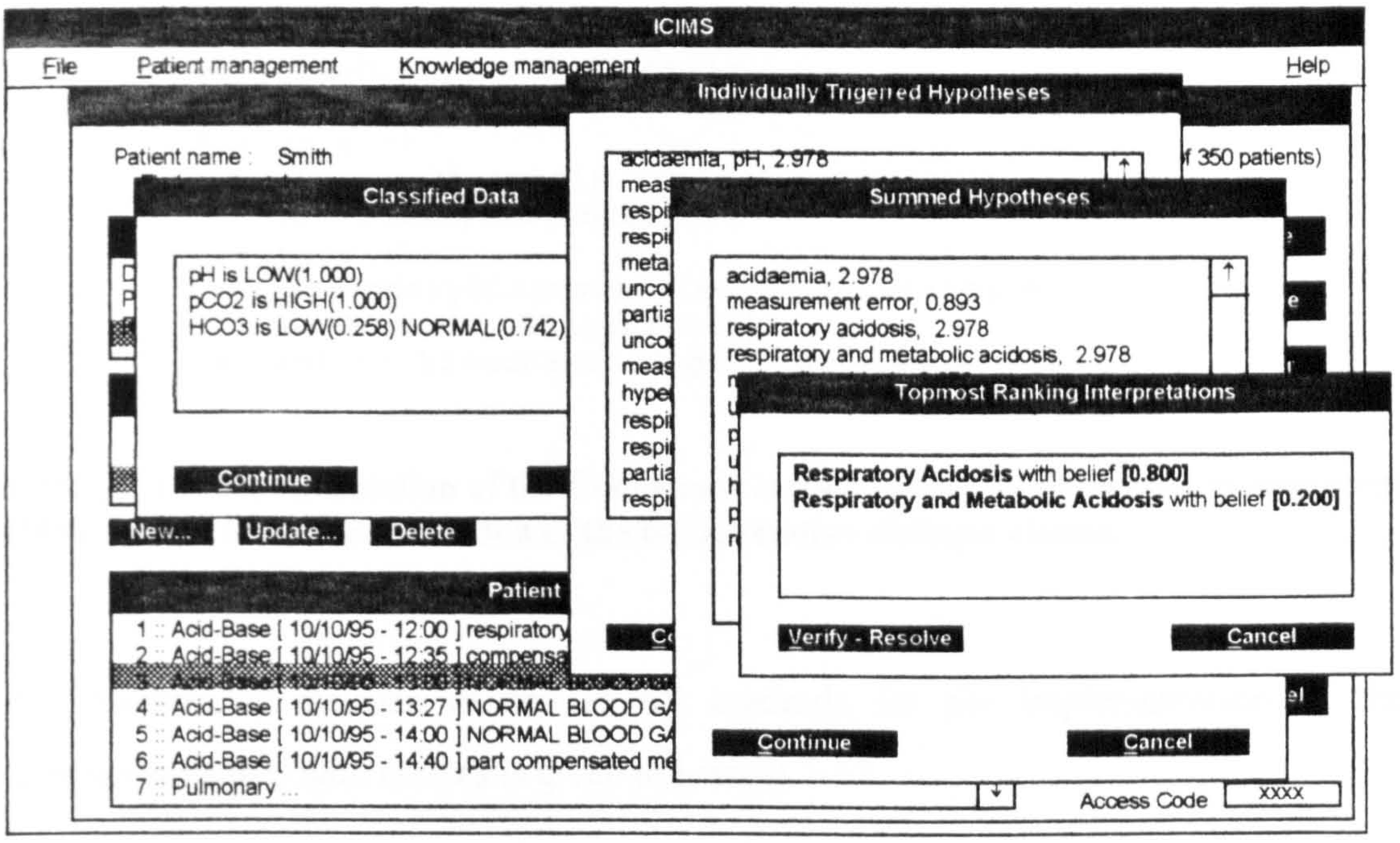


Figure 5-9. Data Interpretation System dialogue windows.

```
case EVALUATE:
{
   ...
   item = analysisLogBox.SelectedItem ();

   InitDialog initialisationDialogue ();
   initialisationDialogue.Execute ();

   bloodGasAnalysis      *pAnalysis;
   pPatient -> bloodGases.Seek (item);
   pPatient -> bloodGases.Get (pAnalysis);
```

```
loop for all data acquired
    pRawData = new rawData;
    pRawData -> Assign ();
    pRawData -> SetMembers ();
    pRawData -> Store ();
    delete pRawData;
classificationDialogue classifyDialogue ();
classifyDialogue.Execute();

evidenceDialogue processEvidenceDialogue ();
processEvidenceDialogue.Execute ();

sumHypothesesDialogue summationDialogue ();
summationDialogue.Execute();

while not complete
    rankDialogue = new rankHypothesesDialogue ();
    repeat = rankDialogue -> Execute ();
    delete rankDialogue;

selectInterpretationDialogue selectionDialogue ();
selectionDialogue.Execute();

pAnalysis ->
   SetPrimaryDiagnosis (selectionDialogue.
     GetPrimaryDiagnosis());
pAnalysis ->
   SetSecondaryDiagnosis (selectionDialogue.
     GetSecondaryDiagnosis());
pPatient -> bloodGases.Unget (pAnalysis);
}
```

Listing 5-12. Implementation of the Commands section of the patient record management OMS module for the instantiation of the interpretative dialogue classes.

An example of the use of the PCAS methods for the implementation of the interpretative dialogue classes is given in Listing 5-13.

```
classificationDialogue :: Init ()
{
   ...
   classificationBox.DeleteContent ();
   pSetOfAllRawData = new rawDataAllSet ();
   pSetOfAllParameterProfiles = new parameterProfileAllSet ();
   Classify();
}

classificationDialogue :: ~ClassificationDialogue ()
{
   delete pSetOfAllRawData;
   delete pSetOfAllParameterProfiles;
}


classificationDialogue :: Classify ()
{
   pSetOfAllRawData -> Seek (0, PtSTART );
```

```
loop for all raw data in the AllSet
   pSetOfAllRawData -> Seek (1L, PtSTART);
   pSetOfAllRawData -> Get (pRawData);

   loop for all parameter profiles
      pSetOfAllParameterProfiles -> Seek (1L, PtSTART);
      pSetOfAllParameterProfiles -> Get (pProfile);

      if pRawData -> GetVariable()= pProfile ->
      GetVariable();
         pProcessedData = new processedData;
         pProcessedData -> Assign ();
         pProcessedData -> Process
         pRawData -> GetVariable(), pRawData -> GetValue(),
         pProfile -> GetMean(), pProfile -> GetSdev());
         classificationBox.AddString(pProcessedData ->
         DisplayObject());
         pProcessedData -> Store();
         delete pProcessedData ;

      pSetOfAllParameterProfiles-> Unget (pProfile);

   pSetOfAllRawData -> Unget (pRawData);
}
```

Listing 5-13. Implementation of the class of classification dialogue objects.

5.4.3. Integrated KBS module Verification

Following the integration of the prototype KBS task-domain model within the clinical information management support system, the application of the acid-base balance disorder belief network processing model was verified using the knowledge contents with which the prototype KBS was evaluated (Table 3-1), which are reviewed in Table 5-1, the parameter classification profiles shown in Tables 5-2, 5-3 and 5-4, and two sets of patient data. The first set was obtained from textbook case studies (Walmsley and White, 1983), and the second from the original retrospective evaluation study (Chelsom, 1990).

Represented acid-base balance disorders	a priori belief	pCO_2	[$HCO3^-$]	pH
respiratory alkalosis, uncompensated	0.0833	L	N	H
respiratory alkalosis, partially compensated	0.0416	L	L-N	H
metabolic alkalosis, uncompensated	0.0833	N	H	H
metabolic alkalosis, partially compensated	0.0416	H-N	H	H
respiratory acidosis, uncompensated	0.0833	H	N	L
respiratory acidosis, partially compensated	0.0416	H	H-N	L
metabolic acidosis, uncompensated	0.0833	N	L	L
metabolic acidosis, partially compensated	0.0416	L-N	L	L
respiratory alkalosis + metabolic alkalosis	0.0833	L	H	H
respiratory acidosis + metabolic acidosis	0.0833	H	L	L
respiratory acidosis + metabolic alkalosis	0.0277	H	H	N
respiratory acidosis, compensated	0.0277	H	H	N
metabolic alkalosis, compensated	0.0277	H	H	N
respiratory alkalosis + metabolic acidosis	0.0277	L	L	N
respiratory alkalosis, compensated	0.0277	L	L	N
metabolic acidosis, compensated	0.0277	L	L	N

Table 5-1. Acid-base balance disorders represented in the ICIMS knowledge base with respective definitional features.

The results of the integrated system verification using the first data set are described in Appendix A.1. This data set is small and was used to test the interpretative dialogues and set the ground for the system verification using the original retrospective data set. With respect to the second data set, as expected, the integrated model behaved in the same way as in the prototype implementation, with a small deviation in borderline cases and a posteriori probability values, due to the modified parameter profiles. This deviation is discussed in Section 5.4.5.

Evidence Parameter	Units	Reference Range	Mean Value	Standard Deviation p(normal) = 0.5	Value when p(low)=1
pH	-	7.35 - 7.45	7.40	0.025	7.30
pCO_2	kPa	4.7 - 6.0	5.35	0.325	4.2
[HCO_3^-]	mmol/L	24 - 32	38	2.000	19
K^+	mmol/L	3.2 - 4.8			
Na	mmol/L	132 - 144			

Table 5-2. Parameter classification profiles (Walmsley and White, 1983).

Evidence Parameter	Units	Reference Range	Mean Value	Standard Deviation p(normal) = 0.5	Value when p(low)=1
pH	-	7.36 - 7.45	7.40	0.020	7.31
pCO_2	kPa	4.5 - 6.1	5.3	0.400	4.0
$[HCO_3^-]$	mmol/L	21.0 - 27.5	24.25	1.625	16.0
K^+	mmol/L	3.2 - 4.6			
Na	mmol/L	132 - 144			

Table 5-3. Parameter classification profiles (Whitby et al, 1984).

As stated in the introduction to this thesis and explained in section 5.4.5 below, the interpretative behaviour of the overall integrated belief network KBS can also be tuned with respect to the a priori belief in each disorder, using prevalence rates obtained from statistical studies rather than the experts experiential belief in the occurrence of each disorder given no clinical information, as for example in the case of de Dombal's system (Aliferis and Miller, 1995).

Evidence Parameter	Units	Reference Range	Mean Value	Standard Deviation p(normal) = 0.5	Value when p(low)=1
pH	-	7.37 - 7.43	7.40	0.015	7.32
pCO_2	kPa	4.86 - 6.10	5.48	0.310	4.36
$[HCO_3^-]$	mmol/L	22.1 - 28.2	25.15	1.525	17.1

Table 5-4. Parameter classification profiles (Siggaard-Andersen, 1990).

5.4.4. Evaluation against Peer Systems

Following the verification of the integrated KBS, the performance of the DIS module was evaluated against the results obtained from the application of the Siggaard-Andersen (1990) oxygen status algorithm. The results of this evaluation are given in Appendix A.2. Again the DIS was in agreement with the 10 cases used in the particular evaluation study.

5.4.5. Toward the Integration of the IMC Task Model

In contrast to the undesirable consultation mode of KBS operation as a source of expert advise (Miller and Masarie, 1990), IMC systems, that is, systems designed with the long-term goal of providing integrated real-time intelligent clinical information management support in critical care medical environments, should be geared toward satisfying 1) the need for complete, accurate, timely and easily accessible patient data sets, and 2) the need to display and transform the data generated in the process of monitoring and supporting patients, in a manner which converts it to information, thereby assisting in avoiding the misinterpretation and consequentially mismanagement of an observed clinical problem under conditions of information overload and contextual complexity (Groth and Collinson, 1993). In doing so, as discussed in Section 4.2.3, IMC systems are called to perform the following decision support functions: 1) To summarise the patient's progress and condition for clinicians and physicians on rounds; 2) To alert clinicians to imminent problems before they might otherwise be noticed; 3) To suggest, critique and possibly execute alternative patient state control decisions.

The development of integrated IMC systems is a long term prospect for applied AI research (Uckun, 1994). However, in the ICIMS system, the OMS module may be extended to incorporate those parts of the IMC TDM which are required to provide the first two functions listed above, namely the high-level patient summary display and intelligent alarm facilities. As opposed to expert consultation KBS, these two integrated clinical information management and decision support functions augment the effective information yield of traditional trend displays and parameter-based alarm facilities provided by existing ICU monitors, by synthesising interpretations of parameter values into qualitative interpretations of physiological and pathophysiological states, and interpretations of causally and/or temporally ordered collections of pathophysiological states into disease progressions and complications.

Figures 5-10 to 5-12 show examples of combined parameter-based and state-based trend display window, designed to augment the effective information yield generated by the patient record dialogues, and to provide the means for a preliminary assessment of the user's requirements for the integration of the BGAS TDM within the real-time

information processing activity supported by the ICIMS system. The data used to generate these interpretative-trend displays were acquired automatically by the ICIMS system following its pilot installation in the Mayday University Hospital ICU in Croydon, England, over a period of 6 months. The particular data set, which was selected from a total of approximately 1,800 blood-gas measurements taken from a total of 10 patients, corresponds to a ventilated patient with renal dysfunction during the period 12/2/96 to 7/3/96. The complete data set is given in Appendix A.3.

In combination, the two trend displays were designed to provide further valuable clinical information management and decision support, in that they can be used to detect changes in patient state, and to distinguish those which merit patient-state control attention from insignificant or erroneous indications, due to measurement, transcription or execution errors (Wright et al, 1991). For example, the measurement taken at 04:23 on the 14th of February 1996 for the patient with acute renal failure, may either be the result of a mechanical ventilation execution error, a measurement or transcription error, or a secondary acid-base balance disorder. However, the measurement taken at 17:50 on the 20th of February appears to be a misclassification error which, being part of an on-going interpretative state-space trajectory, and in conjunction with the superimposed parameter trend, does not affect the validity of the decision support generated by the system in as much as it does in the case of consultation systems, such as the BGAS prototype, which are required to give the 'correct' answer to an interpretative problem (Miller and Masarie, 1990).

Furthermore, the functional integration process may be extended to provide explanations of suspicious results such as those discussed above, in the following manner. Once the user has selected a particular "red-herring" parameter, or state interpretation, from the state-trend display which needs to be explained, the module opens an explanation window in which, using simple and knowledge-based queries, it attempts to causally explain the selected state. This may be done by the method of causal-process classification, involving the overall clinical picture, by temporal state-trajectory reasoning as described above, or simply by a simple statement of other

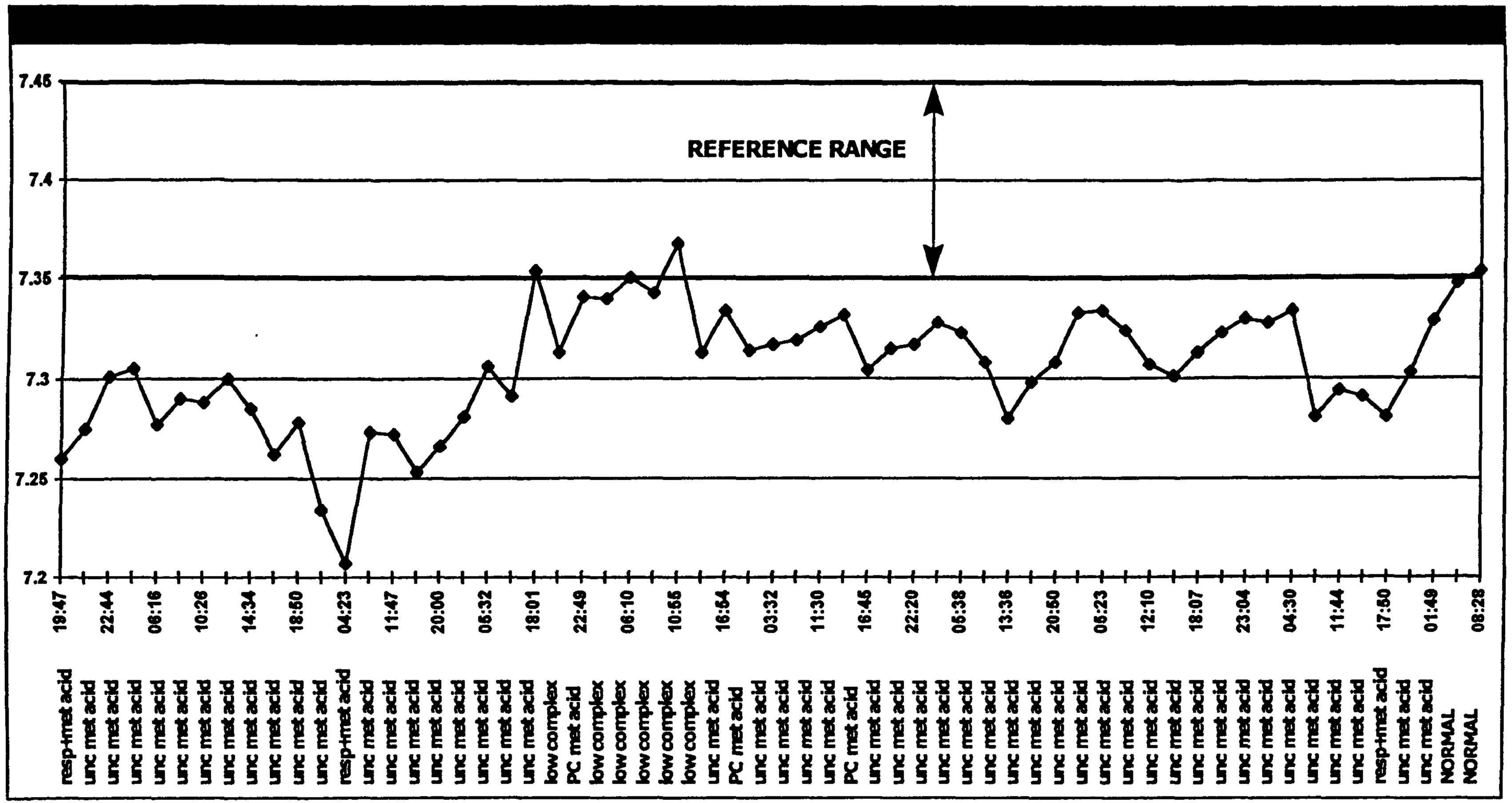


Figure 5-10. High-level patient summary interpretative trend display.

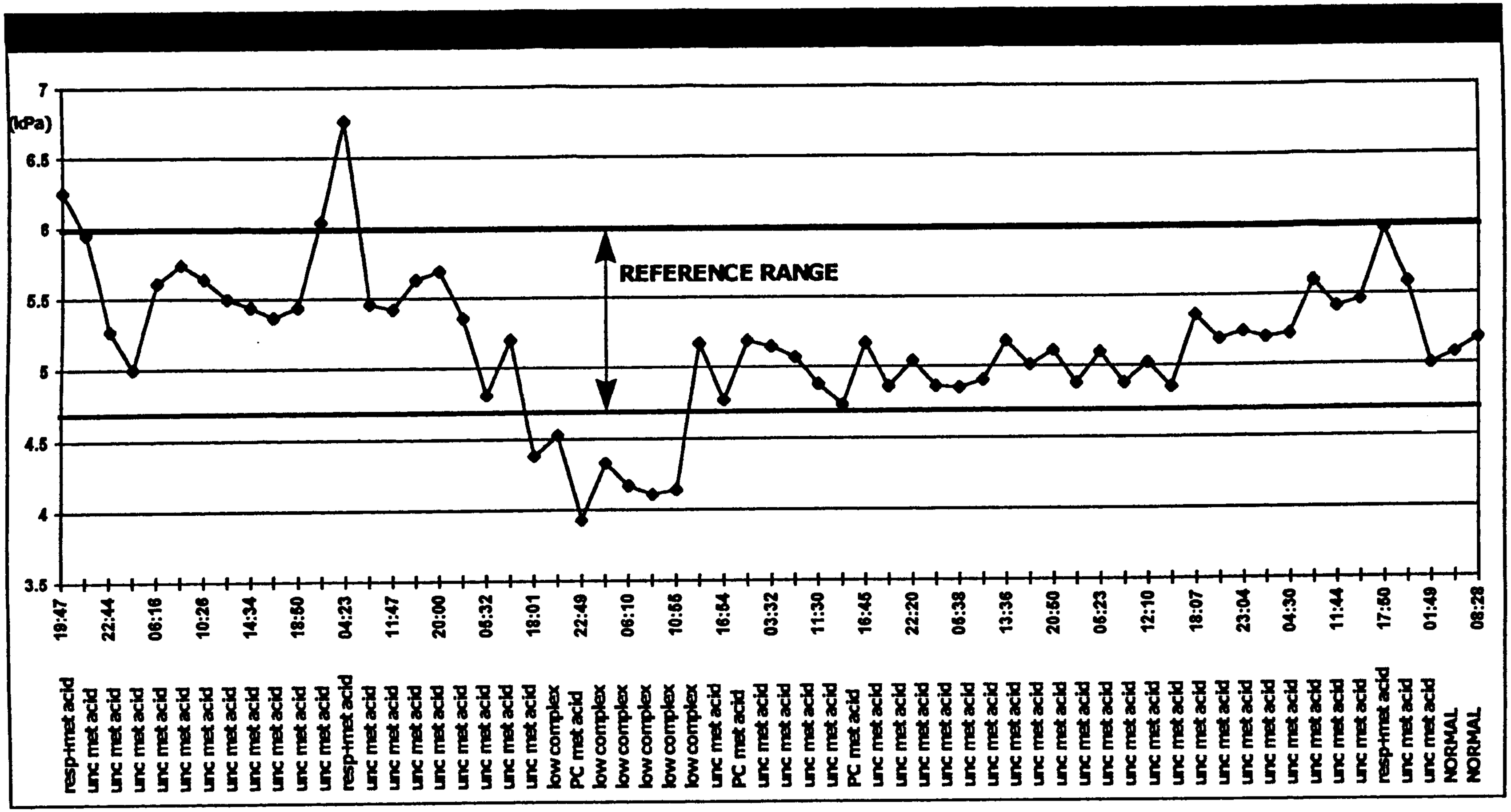


Figure 5-11. High-level patient summary interpretative trend display.

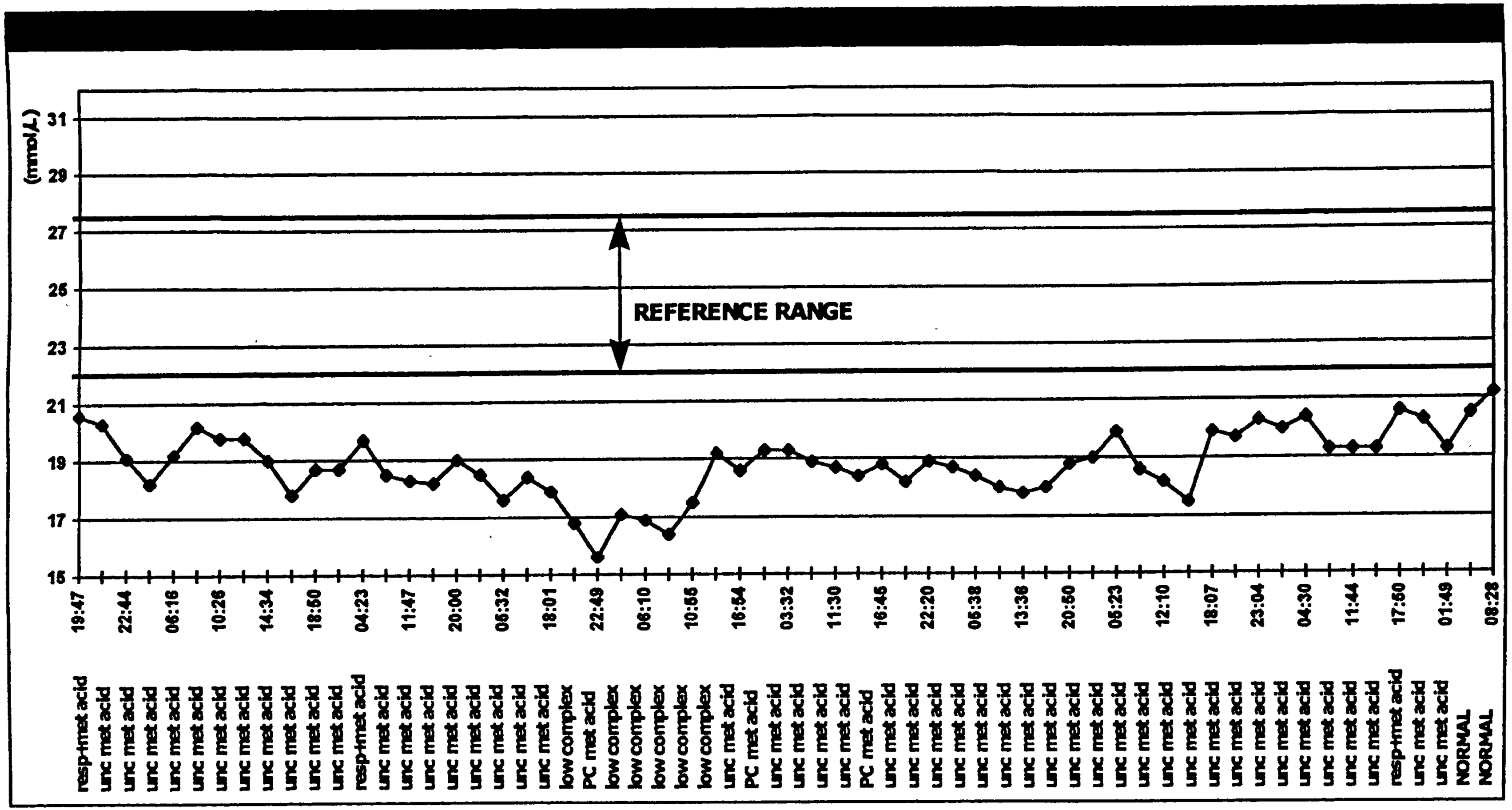


Figure 5-12. High-level patient summary interpretative trend display.

measurements and control settings. For example, the window might display ventilator settings which indicate that an execution error has taken place.

In order to extend the integration process in this direction, the ICIMS system must be further developed so that the OCS module may function independently of other system operations. If so, the OCS module may monitor the execution of the various clinical information management support tasks, and interrupt any process upon the arrival of further data, in order to update the state of the system at any one time. However, such a clinical information processing behaviour would have to be co-ordinated with a blackboard module KBS task-scheduler, as in the case of the Guardian IMC system (Section 2.4.5), with real-time constraints on the application of the represented cognitive task-domain. Furthermore, since most operations performed by the ICIMS system involve accessing the COB, the OMS module would have to be further developed to introduce concurrency, transaction management and event handling, using the available advanced PCAS methods.

5.5. Conclusion

Instead of focusing onto the entity abstractions required to support and facilitate the integration-development process, as did the last chapter, this chapter proceeded to describe some of the ICIMS system implementation detail, and in particular the parts concerning those system features which were developed to make the integrated system functionally, cognitively and ergonomically compatible with the clinical information processing activity observed in ICU clinical practice. Thus, the chapter described the implementation of computer-user dialogues for the acquisition, display, update and review of the clinical information utilised in the process of monitoring and supporting the ICU patient with disturbed acid-base balance. These dialogues were designed via a process of constructive assessment, to function as ergonomic interfaces to the underlying patient data and disorder knowledge object structures, for the maintenance of such structures, and to enable the incremental development of further clinical information management and clinical decision making support functions, such

as the contextual interpretation of the acquired patient data in order to produce high-level patient summaries and state alarms. The chapter also described the directions in which further development effort should take in order to bring the integrated system functionally, ergonomically, and cognitively closer to the user's clinical information processing and decision making activity.

- 6 -

Conclusions

6.1. Summary and Discussion

In the introduction to this thesis, it was argued that due to the methodological difficulties encountered in the evaluation of medical KBS, and in particular regarding the method employed for representing an expert's ability to reason with partial belief and incomplete information, most systems have not been independently evaluated for performance in clinical environments, and thus lack the evidence required to prove that they are effective in improving measurable parameters of the patient care delivery system. Thus, by failing to observe and assess the systems concerned at work, developers have for long omitted many of the relevant criteria for effectiveness and acceptance in the clinical setting, the most important being, functional, cognitive and ergonomic compatibility with the real needs and activities of the clinical user. It was argued that in order to address these problems, KBS must be integrated within the information processing activity of the clinical user, for the development of systems geared toward satisfying the primary requirement to support the management of clinical information in the modern high-technology health care environment.

Thus, the aim of the work described in this thesis was to overcome the aforementioned obstacles, which were also manifested in the development of the BGAS prototype, by designing a system that would support and facilitate the solution of the problems associated with the observation, interpretation, monitoring and management of complex clinical events, in the context of dealing with the wider problem of information management in the modern ICU. This was to be achieved by integrating an existing validated and retrospectively evaluated prototype belief network for the interpretation of blood-gas data within the information processing activity of the clinical user, and the development of a system designed to support the management of the clinical information generated in the process of monitoring the ICU patient with abnormalities of acid-base balance. Specific objectives were stated in section 1.2 as the following.

1. To develop a system which would combine the computer-based clinical decision support tasks of the acquisition, organisation, storage, update and review of the

information generated in the process of monitoring the ICU patient, as well as of the domain knowledge-base required for the contextual interpretation of the acquired clinical information, within a singular system architecture.

2. To use the clinical information management support system in order to develop and constructively assess the integration of the cognitive, clinical information processing tasks comprising the prototype KBS interpretative problem-solving task-domain into the ICIMS system, and consequentially into clinical practice, in order to incorporate the computational intelligence necessary for the interpretation of the patient data acquired in the process being supported.

3. To provide the means to assess specific problems encountered in the integration process, and to develop effective and usable solutions, by employing an approach which would enable the active participation of a clinical advisor who would act as an assessor of the functional, cognitive and ergonomic effectiveness of the KBS integration process, and of the overall decision support provided by the ICIMS system during its development.

As described in Chapter 4, these three objectives were pursued and accomplished via an incremental object-oriented design and implementation approach to the functional, cognitive and ergonomic integration of the validated KBS prototype. The integration-development process started with the design of the basic COB handling system, and proceeded upwards and closer to the user, guided by the constructive assessment of a clinical advisor, who was also involved in the construction of the integration prototype knowledge base, to include the validated task-domain model of the KBS prototype for interpretative evidence impact, aggregation and propagation in a hierarchical belief network of disorders of acid-base metabolism, and interfaces for the update and review of the information acquired and generated in the process of monitoring the ICU patient.

The COB formed the heart of this process, by containing the persistent object structures required to support the management of the clinical information utilised in the process of monitoring the ICU patient. More specifically, the ICIMS system uses

three models for deriving the COB. These are the PRM which corresponds to a prototypical data base schema, the DKM which corresponds to a prototypical knowledge base schema, and the PSM which corresponds to the interpretative problem solution blackboard of the KBS prototype. The OMS module was subsequently designed to generate dialogues with the user in order to create the persistent data and knowledge object structures contained in the COB, by deriving such object structures from the patient record and domain knowledge models.

Thus, the first layer of the incremental ICIMS system development, that is, the layer closer to the underlying machine, comprises the COB module, which uses the PCAS for handling the persistent object structures derived and maintained by means of the second layer, which comprises the object derivation models. Similarly, the OMS module is one level closer to the user since it provides the required functionality of patient data and domain knowledge acquisition, update and review, and a step closer to the integration of the KBS prototype, since it is at this stage that most of the clinical advisor's constructive assessment is translated into the evolutionary modifications pertaining to the KBS prototype integration process. Furthermore, since the COB level which was developed using the PCAS, is generic, and the model-based object derivation level comprises mostly basic entity abstractions, system modifications at the level of the OMS module can be made without affecting the underlying module levels. The final stage of the KBS integration process was the incorporation of the prototype cognitive task-domain for the interpretation of the acquired patient data.

This decision to use the prototype KBS described in Chapter 3, as the integration prototype for the incorporation of the computational intelligence required in the development of the ICIMS system was based on an analysis of the problem of reasoning with partial belief and incomplete information in clinical decision making and methods for the acquisition, representation and manipulation of uncertain medical knowledge. More specifically, Chapter 2 discussed a number of formal and heuristic methods designed to replace probability theory for the task, due to misconceived limitations of the theory, and exposed that these methods promote errors in judgement and lead to interpretative decision making of poorer performance and accuracy. The

functional, cognitive and ergonomic characteristics of the pilot ICIMS system implementation and of the clinical information management and decision making support provided by the integrated system were described in Chapter 5, along with some recommendations for future development, which are summarised below.

6.2. Contributions

The development of the ICIMS system has contributed to both the field of medical informatics as well as to the field of clinical medicine. To the field of medical informatics the contribution of the integration of the prototype knowledge-based system for the contextual interpretation of patient data within the patient care activity of the clinical user is threefold.

Firstly, the ICIMS system was developed to provide effective solutions to a number of essential medical decision support problems encountered in the data overloaded environment of modern clinical practice, rather than to function as a human expert consultation replacement. This is so, both in the sense of gearing the application of the underlying knowledge-based techniques toward this objective, as well as in the sense that ICIMS was designed to integrate, record, organise, and display the clinical information acquired and generated in the process, in a manner which provides further valuable clinical decision support. Secondly, by means of the provision of effective decision support in the latter sense, the integrated system reached a level of overall effectiveness which justifies its introduction into routine clinical practice, thereby being exposed to the clinical information processing environment and the amount and quality of data required for formatively and constructively assessing the evolution of the represented knowledge base and cognitive task-domain. Thirdly, by being able to maintain temporal records of the patient's progression and of the results of the application of the represented task-domain, ICIMS provides the framework for the development, representation, application and evaluation of advanced cognitive task-domains, as for example in the case of experimental closed-loop control IMC systems.

To the field of clinical medicine the ICIMS system contributes an integrated tool-set which assists in the management and improved utilisation of the clinical information

generated in the process of monitoring ICU patients in intensive care hospital units. Overall, the work described in this thesis demonstrates that both the fields of artificial intelligence in medicine as well as that of medical information systems engineering have matured to the point where, in combination, can provide effective solutions to the medical decision support problems encountered in the increasingly complex high-technology health care delivery system.

6.3. Further work

Chapter 5 described the object-oriented ICIMS system architecture implementation details, and in particular those which pertain to system features that were designed to make the system functionally, ergonomically and cognitively compatible with the information processing activity of the clinician user in a critical care environment. Thus, the chapter described the implementation of computer-user dialogues for the acquisition, display, update and review of the clinical information utilised in the process of monitoring and supporting the ICU patient with disturbed acid-base balance. These dialogues were designed via a process of constructive assessment, to function as ergonomic interfaces to the underlying patient data and disorder knowledge object structures, for the maintenance of such structures, and to enable the incremental development of further clinical information management and clinical decision making support functions, such as the contextual interpretation of the acquired patient data in order to produce high-level patient summaries and state alarms. The chapter also described the directions in which further development effort should take in order to bring the integrated system functionally, ergonomically, and cognitively closer to the user's clinical information processing and decision making activity.

Overall, further development effort should be directed toward the incorporation of a cognitive task model for causal-process classification and simulation reasoning in order to improve the system's interpretative performance with respect to complex disorders. It was stated that the presence of the ICIMS system in the ICU environment will support and facilitate the knowledge engineering and assessment

effort required in order to achieve this objective. This facility can be added to a repertory of explanatory functions as described in Chapter 2 and further in Sections 5.4.5 and 5.4.6 in order to augment the system's cognitive capacity and compatibility. Finally, by incorporating a comprehensive task-scheduling system, as described in Section 5.2.3, the ICIMS system may be further developed to function in the mode of IMC systems without the need for any computer-user interaction. This development will be complemented by the use of advanced peripheral devices such as touch screens.

References

ABRAMSON, N.S. and WALD, K.S. (1980). Adverse occurrences in intensive care units. Journal of the American Medical Association, 244, 1582-1584.

ADAMS, I.D., CHAN, M., CLIFFORD P.C., COOKE, W.M., DALLOS, V., De DOMBAL, F.T., EDWARDS, M.H., HANCOCK, D.M., HEWETT, D.J., McINTYRE, N., SOMERVILLE, P.G., SPIEGELHALTER, D.J., WELLWOOD, J. and WILSON, D.H. (1986). Computer aided diagnosis of acute abdominal pain: a multicentre study. British Medical Journal, 293(6550), 800-804.

ADLASSNIG, K.-P., KOLARZ, G. and SCHEITHAUER, W. (1985). Present state of the medical expert system CADIAG-2. Methods of Information in Medicine, 24, 13-20.

AIKINS, J.S., KUNZ, J.C., SHORTLIFFE, E.H. and FALLAT, R.J. (1983). PUFF: an expert system for interpretation of pulmonary function data. Computers and Biomedical Research, 16, 199-208.

ALIFERIS, C.F. and MILLER, R.A. (1995). On the heuristic nature of medical decision-support systems. Methods of Information in Medicine, 34(1-2), 5-14.

ANDERSON, I.D., WOODFORD, M., de DOMBAL, F.T. and IRVING, M. (1988). Retrospective study of 1000 deaths from injury in England and Wales. British Medical Journal, 296(6632), 1305-1308.

ANDREASSEN, S., JENSEN, F.V. and OLESEN, K.G. (1991). Medical expert systems based on causal probabilistic networks. International Journal of Biomedical Computing, 28(1-2), 1-30.

ASHLEY, J.S.A., PASKER, P. and BERESFORD, J.C. (1972). How much clinical investigation? Lancet, 1, 890-892.

BANKOWITZ, R.A., McNEIL, M.A., CHALLINOR, S.M., PARKER, R.C., KAPOOR, W.N. and MILLER, R.A. (1989a). A computer-assisted medical diagnostic consultation service. Implementation and prospective evaluation of a prototype. Annals of Internal Medicine, 110(10), 824-832.

BANKOWITZ, R.A., McNEIL, M.A., CHALLINOR, S.M. and MILLER, R.A. (1989b). Effect of a computer-assisted general medicine diagnostic consultation service on housestaff diagnostic strategy. Methods of Information in Medicine, 28(4), 352-356.

BARNETT, G. et al (1979). COSTAR - A computer-based medical information system for ambulatory care. In: Proceedings IEEE, 67, 1226.

BERLEUR, J. NGUYEN, N.T. and DELHAYE, R. (1995). Technology assessment in health care informatics: a clarification of the concept. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 21-34.

BEUSCART-ZEPHIR, M.C., BRENDER, J., BEUSCART, R. and MENAGER-DEPRIESTER, I. (1997). Cognitive evaluation: how to assess the usability of information technology in healthcare. Computer Methods and Programs in Biomedicine, 54(1-2), 19-28.

BLOIS, M.S. (1980). Clinical judgement and computers. New England Journal of Medicine, 303(4), 192-197.

BLOIS, M.S. (1983). Conceptual issues in computer-based diagnosis and the hierarchical nature of medical knowledge. Journal of Medical Philosophy, 8(1), 29-50.

BLOIS, M.S. (1984). Information and Medicine. Berkeley, CA: University of California Press.

BLOIS, M.S. (1988). Medicine and the nature of vertical reasoning. The New England Journal of Medicine, 318(13), 847-851.

BOOCH, G. (1991). Object oriented design. Redwood, CA: Benjamin/Cummings.

BRAHAMS, D. and WYATT, J. (1989). Decision aids and the law. The Lancet, September 9, 632-634.

BRENDER, J., TALMON, J., NYKÄNEN, P., McNAIR, P., DEMEESTER, M. and BEUSCART, R. (1995). On the evaluation of system integration. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 189-208.

BRENDER, J. and McNAIR, P. (1996). User requirements on the future laboratory information systems. Computer Methods and Programs in Biomedicine, 50(2), 87-93.

BRENDER, J. (1997). Assessment of IT-based solutions in healthcare: Trends in today's practice and justification of tomorrow's approach. In: C.Pappas, N.Maglaveras and J.-R.Scherrer (eds), Proc. Medical Informatics Europe 1997. Thessaloniki: IOS Press, 730-735.

BUCHANAN, B.G. and SHORTLIFFE, E.H. (1984). Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. Reading, MA: Addison-Wesley.

BUCHANAN, B.G. (1986). Expert systems: working systems and the research literature. Expert Systems, 3, 32-51.

CARSON, E.R. (1988). Artificial intelligence in critical care medicine: EC COST 13 project. Report of a mini-symposium held at City University, London, 10-11 December 1987. International Journal of Clinical Monitoring and Computing, 5, 251-257.

CARSON, E.R., CRAMP, D.G. and FINKELSTEIN, L. (1988). Intelligent measurement: developments in critical care medicine. In: ACTA IMECO XI. Budapest: International Measurement Confederation, 107-113.

CARSON, E.R., SHAMSOLMAALI, A., SUMMERS, R., LEANING, M.S. and CRAMP, D.G. (1988). Intelligent instrumentation in critical care medicine. In: E.R.Carson, P.Kneppo and I.Krekule (eds), Advances in Biomedical Measurement. New York: Plenum, 351-358.

CARSON, E.R. (1989). Systems thinking and knowledge technology in medicine. In: R.L.Hood, M.C.Jackson and P.Keys (eds), Systems Prospects. New York: Plenum, 207-214.

CARSON, E.R., CHELSOM, J.J.L. and SUMMERS, R. (1991). Progress with measurement, information and decision-making in critical care medicine. Measurement, 9(3), 104-10.

CARSON, E.R., KALOGEROPOULOS, D. and SUMMERS, R. (1992). Development of an integrated clinical information management system. In: J.P.Morucci, R.Plonsey, J.L.Coatrieux and S.Laxminarayan (eds), Proc. 14th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. New York: IEEE, 2848-2849.

CHANDRASEKARAN, B. and MITTAL, S. (1983). Conceptual representation of medical knowledge for diagnosis by computer: MDX and related systems. In: M.Yovitz (ed), Advances in Computers. New York, NY: Academic Press, 217-293

CHANDRASEKARAN, B. (1985). Generic tasks in knowledge-based reasoning: characterizing systems at the "right" level of complexity. In: Proceedings IEEE Second Conference on Artificial Intelligence Applications, Miami, FLA.

CHANDRASEKARAN, B. (1986). Generic tasks in knowledge-based reasoning: high-level building blocks for expert system design. IEEE Expert, 1, 23-30.

CHEESEMAN, P.C. (1985). In defence of probability. Proceedings of the International Joint Conference in Artificial Intelligence, 1002-9.

CHELSOM, J.J.L. (1990). The Interpretation of Data in Intensive Care Medicine: An Application of Knowledge-based Techniques. Ph.D. Thesis, City University, Department of Systems Science, 1990.

CLANCEY, W.J. and LETSINGER, R. (1981). NEOMYCIN: reconfiguring a rule-based expert system for application to teaching. In: Proc. 7th International Joint Conference on Artificial Intelligence. Los Altos, CA: Morgan-Kaufmann, 829-36.

CLANCEY, W.J. and SHORTLIFFE, E.H., eds. (1984). Readings in Medical Artificial Intelligence: the first decade. Reading, MA: Addison-Wesley.

CLANCEY, W.J. (1985). Heuristic classification. Artificial Intelligence, 27, 289-350.

CLANCEY, W.J. (1989). Viewing knowledge bases as qualitative models. IEEE Expert, Summer 1989, 9-23.

CLANCEY, W.J. (1992). Model construction operators. Artificial Intelligence, 53, 1-115.

COHEN, P.R. (1985). Heuristic Reasoning About Uncertainty: An AI Approach. London: Pitman.

COHEN, P.R. and HOWE, A.E. (1988). How evaluation guides AI research. AI Magazine, Winter, 35-43.

COHN, A., ROSENBAUM, S., FACTOR, M. and MILLER, P.L. (1990). DYNASCENE: an approach to computer-based intelligent cardiovascular monitoring using sequential clinical scenes. Methods of Information in Medicine, 29, 122-31.

COLLINSON, P.O., BORAN, G.P.R., GRAY, T.J., COCK, C. and HARRISON, L.A.W. (1990). Design, validation and evaluation of a clinical management and information system for fluid, electrolyte and nutritional therapy combining conventional and novel techniques. In: R.O'Moore, S.Bengtsson, J.R.Brynat and J.S.Bryden (eds), Lecture Notes on Medical Informatics 40 (Proc. MIE'90). London: Springer-Verlag, 285-287.

CONSORTI, F. (1995). Assessment of workstations and PACS in AIM: The experience of the MILORD Project. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 245-258.

COOPER, G.F. (1988). Computer-based medical diagnosis using belief networks and bounded probabilities. In: P.L.Miller (ed), Selected Topics in Medical Artificial Intelligence. New York: Springer-Verlag, 85-98.

COOPER, G. (1993). Probabilistic and decision-theoretic systems in medicine (editorial). Artificial Intelligence in Medicine, 5, 289-292.

CRAMP, D.G. and BARON, D.N. (1985). Clinical decision making in relation to laboratory investigation: the role of the computer. In: Computers and Control in Clinical Medicine, edited by: E.R.Carson and D.G.Cramp. New York: Plenum, 47-58.

CRAMP, D.G. and CARSON, E.R. (1985). The patient/clinician relationship, computing and the wider health care system. In: Computers and Control in Clinical Medicine, edited by: E.R.Carson and D.G.Cramp. New York: Plenum, 245-55.

CRAMP, D.G. and CARSON, E.R. (1985b). Design requirements for a user-friendly computer-aided decision support system in laboratory medicine. In: Proc. 2nd IFAC/IFORS/IEA Conference on Analysis, Design and Evaluation of Man-Machine

Systems, edited by: G.Johannsen, G.Mancini and L.Martenson. Oxford: Pergamon, 50-55.

CRAMP, D. and GOODYEAR, O.M. (1989). Expert systems in medicine - report on a European survey. Healthcare Informatics Foundation.

DAVIS, R. (1982). Expert systems: Where are we? And where do we go from here? Artificial Intelligence Magazine, 3(2), 3-22.

DAVIS, R., BUCHANAN, B. and SHORTLIFFE, E.H. (1984). Production rules as a representation for knowledge-based consultation program. In: W.J.Clancey and E.H.Shortliffe (eds), Readings in Medical Artificial Intelligence: The First Decade. Reading, MA: Addison-Wesley, 98-130.

DAVIS, R. (1986). Knowledge-based systems. Science, February 1986, 957-963.

de DOMBAL, F.T., LEAPER, D.J., STANILAND, J.R., McCANN, A.P. and HORROCKS, J.C. (1972). Computer-aided diagnosis of acute abdominal pain. British Medical Journal, 2(5804), 9-13.

de DOMBAL, F.T., LEAPER, D.J., HORROCKS, J.C., STANILAND, J.R. and McCANN, A.P. (1974). Human and computer-aided diagnosis of abdominal pain: further report with emphasis on performance of clinicians. British Medical Journal, 1(904), 376-380.

de DOMBAL, F.T., CLAMP, S.E., LEAPER, D.J., STANILAND, J.R. and HORROCKS, J.C. (1975). Computer-aided diagnosis of lower gastrointestinal tract disorders. Gastroenterology, 68(2), 252-260.

de DOMBAL, F.T. (1984). Computer-aided diagnosis of acute abdominal pain. The British experience. Review Epidimiologie Sante Publique, 32(1), 50-56.

de DOMBAL, F.T. (1991). The diagnosis of acute abdominal pain with computer assistance: worldwide perspective. Ann Chir, 45(4), 273-277.

de DOMBAL, F.T., DALLOS, V. and McADAM, W.A. (1991). Can computer aided teaching packages improve clinical care in patients with acute abdominal pain? British Medical Journal, 302(6791), 1495-1497.

DUDA, R.O. and SHORTLIFFE, H. (1983). Expert systems research. Science, 220, 261-68.

ENGELBRECHT, R., RECTOR, A. and MOSER, W. (1995). Verification and validation. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 51-66.

FACTOR, M., SITTIG, D.F., COHN, A.I., GELERNTER, D.H., MILLER, P.L. and ROSENBAUM, S. (1989). A parallel architecture for building intelligent medical

monitors. In: Proceedings 13th Annual Symposium on Computer Applications in Medical Care, 11-16.

FACTOR, M, GELERNTER, D.H., KOLB, C.E., MILLER, P.L. and SITTIG, D.F. (1991). Real-time data fusion in the intensive care unit. IEEE Computer, November 1991, 45-53.

FALLAT, R.J. and OSBORN, J.J. (1984). Patient monitoring techniques. In: G.C.Burton and J.E.Hodskin (eds), Respiratory Care. Philadelphia, PA: J.B.Lippincott.

FEINSTEIN, A.R. (1963). Boolean algebra and clinical taxonomy; I. Analytic Synthesis of the General Spectrum of a Human Disease. New England Journal of Medicine, 269(18), 929-938.

FEINSTEIN, A.R. (1977). The haze of Bayes, the aerial palaces of decision theory, and the computerised Ouija board. Clinical and Pharmacological Therapy, 21, 482-96.

GARDNER, R.M. (1990). Patient-monitoring systems. In: E.H.Shortliffe, L.Perrault, G.Wiederhold, and L.Fagan, (eds) Medical Informatics - Computer Applications in Health Care. Wokingham, England: Addison-Wesley.

van GENNIP, E.M.S.J. (1995). Approaches to experimental design. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 75-85.

GORDON, J. and SHORTLIFFE, E.H. (1985). A method for managing evidential reasoning in a hierarchical hypothesis space. Artificial Intelligence, 26, 323-357.

GORRY, G.A. and BARNETT, G.O. (1968). Experience with a model of sequential diagnosis. Computers and Biomedical Research, 1, 490-507.

GORRY, G.A. and BARNETT, G.O. (1968). Sequential diagnosis by computer. American Journal of Medicine, 55, 473-484.

GORRY, G.A. (1973). Computer-assisted clinical decision making. Methods of Information in Medicine, 12, 45-51.

GRAHAM, I. (1994). Object oriented methods. Cambridge, England: The University Press, Addison-Wesley.

GREENES, R.A. and SHORTLIFFE, E.H. (1990). Medical Informatics: an emerging academic discipline and institutional priority. Journal of the American Medical Association, 263 (8), 1114-1120.

GRÉMY, F. and BONNIN, M. (1995). Evaluation of automatic health information systems. What and how? In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment

and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 9-20.

GROTH, T. and COLLINSON, P.O. (1993). Strategies for decision support for fluid and electrolyte therapy in the intensive care unit - approaches and problems. International Journal of Clinical Monitoring and Computing, 10, 3-15.

GUNN, A.A. (1976). The diagnosis of acute abdominal pain with computer analysis. Journal of the Royal College of Surgeons Edinbourgh, 21, 170-172.

GUNN, A.A. (1991). The acute abdomen: the role of computer-assisted diagnosis. Baillieres Clin Gastroenterol, 5(3-1), 639-665.

HAYES-ROTH, B., WASHINGTON, R., HEWETT, R. and HEWETT, M. (1989). Intelligent monitoring and control. In: Proc. 11th International Joint Conference on Artificial Intelligence, 243-249.

HAYES-ROTH, B., WASHINGTON, R., ASH, D., HEWETT, R., COLLINOT, A., VINA, A. and SEIVER, A. (1992). Guardian: a prototype intelligent agent for intensive-care monitoring. Artificial Intelligence in Medicine, 4, 165-185.

HECKERMAN, D.E. and MILLER, R.A. (1986). Towards a better understanding of the INTERNIST-1 knowledge base. In: Proc. MEDINFO'86, Washington, DC.

HECKERMAN, D.E. and HORVITZ, E.J. (1987). On the expressiveness of rule-based systems for reasoning under uncertainty. In: Proc. AAAI-87 Sixth National Conference on Artificial Intelligence, Seattle, Washington, 121-126.

HECKERMAN, D.E. and SHORTLIFFE, E.H. (1992). From certainty factors to belief networks. Artificial Intelligence in Medicine, 4, 35-52.

HECKERMAN, D.E., HORVITZ, E.J. and NATHWANI, B.N. (1992). Toward normative expert systems: Part I. The Pathfinder project. Methods of Information in Medicine, 31(2), 90-105.

HECKERMAN, D.E. and NATHWANI, B.N. (1992). Toward normative expert systems: Part II. Probability-based representations for efficient knowledge acquisition and inference. Methods of Information in Medicine, 31(2), 106-116.

HENDERSON, S., EAST, T.D., MORRIS, A.H. and GARDNER, R.M. (1989). Performance evaluation of computerised clinical protocols for management of arterial hypoxaemia in ARDS patients. In: Proc. 13th Annual Symposium on Computer Applications in Medical Care, Washington, DC., 588-592.

HENDERSON, S., CRAPO, R.O., WALLACE, J. EAST, T.D. MORRIS, A.H., GARDNER, R.M. (1991). Performance of computerized protocols for the management of arterial oxygenation in an intensive care unit. International Journal of Clinical Monitoring and Computing, 8(4), 271-280.

HENKELMAN, M. (1995). Is there future for developing medical technologies in a restricted health care economy? With examples from magnetic resonance research. Keynote Address, 17th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Montréal, Canada, September 20th.

van der HOFSTEDE, J.W., QUAK, A.B.W.M., van GINNEKEN, A.M., MacFARLANE, P.W., WATSON, J., HENDRIKS, P.R.H. and ZEELENBERG, C. (1997). Evaluation plan for a cardiological multimedia workstation (I4C Project). In: C.Pappas, N.Maglaveras and J.-R.Scherrer (eds), Proc. Medical Informatics Europe 1997. Thessaloniki: IOS Press, 751-755.

HORROCKS, J.C., McCANN, A.P., STANILAND, J.R., LEAPER, D.J. and de DOMBAL, F.T. (1972). Computer-aided diagnosis: description of an adaptable system and operational experience with 2,034 cases. British Medical Journal, 2(5804), 5-9.

HORROCKS, J.C. and de DOMBAL, F.T. (1975). Computer-aided diagnosis of "'dyspepsia". American Journal of Digestive Diseases, 20(5), 397-406.

HORROCKS, J.C., DEVROEDE, G. and de DOMBAL, F.T. (1976). Computer-aided diagnosis of gastroenterologic diseases in Sherbrooke: preliminary report. Canadian Journal of Surgery, 19(2), 160-164.

HORROCKS, J.C., LAMBERT, D.E., McADAM, W.A., MORGAN, A.G., PACSOO, C., DARNBOROUGH, A. and de DOMBAL, F.T. (1976). Transfer of computer-aided diagnosis of dyspepsia from one geographical area to another. Gut, 17(8), 640-644.

HORVITZ, E.J., BREESE, J.S. and HENRION, M. (1988). Decision theory in expert systems and artificial intelligence. In: Special issue on Uncertainty in Artificial Intelligence, International Journal of Approximate Reasoning, 2(3), 247-302. Elsevier Science Publishing.

HUDSON, L. (1985). Monitoring of critically ill patients: Conference summary. Respiratory Care, 30, 628.

HUNTER, J., CHAMBRIN, M-C., COLLINSON, P.O., GROTH, T., HEDLUND, A., KALLI, S. KARI, A., LENOUDIAS, G., RAVAUX, P. ROSS, D., SALLE, J-M., SUKUVAARA, T., SUMMERS, R. and ZAAR, B. (1991). INFORM: integrated support for decisions and activities in intensive care. International Journal of Clinical Monitoring and Computing, 8, 189-199.

JENNETT, B. (1986). High technology medicine: benefits and burdens. Oxford, England: Oxford University Press.

JOHNSON, R.W. (1986). Independence and Bayesian updating methods. Artificial Intelligence, 29, 217-222.

JØRGENSEN, T. (1995). Measuring effects. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 99-109.

JØRGENSEN, T. (1995b). Methods for data acquisition. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 111-116.

KAHNEMAN, D., SLOVIC, P. and TVERSKY, A. (eds) (1982). Judgement under Uncertainty: Heuristics and Biases. New York, NY: Cambridge University Press.

KASSIRER, J.P. and GORRY, G.A. (1978). Clinical problem solving: A behavioral analysis. Annals of Internal Medicine, 89(2), 245-255.

KASSIRER, J.P. and KOPELMAN, R.I. (1989). Cognitive errors in diagnosis: instantiation, classification, and consequences. American Journal of Medicine, 86(4), 433-441.

KENT, D.L., SHORTLIFFE, E.H., CARLSON, R.W., BISCHOFF, M.B. and JACOBS, C.D. (1985). Improvements in data collection through physician use of a computer-based chemotherapy treatment consultant. Journal of Clinical Oncology, 3, 1409-1417.

KERAVNOU, E.T. and WASHBROOK, J. (1989). Deep and shallow models in medical expert systems. Artificial Intelligence in Medicine, 1, 11-28.

KIM, J.H. and PEARL, J. (1983). A computational model for causal and diagnostic reasoning in inference systems. In: Proc. 8th International Joint Conference on Artificial Intelligence, 190-193.

KNAUS, W.A., DRAPER, E.A., WAGNER, D.P. and ZIMMERMAN, J.E. (1985). APACHE-II: A severity of disease classification system. Critical Care Medicine, 13(10), 818-829.

KORVER, M. and LUCAS, P.J.F. (1993). Converting a rule-based expert system into a belief network. Medical Informatics, 18(3), 219-241.

KUIPERS, B.J. and KASSIRER, J.P. (1984). Causal reasoning in medicine: analysis of protocol. Cognitive Science, 8, 363-85.

KULIKOWSKI, C.A. and WEISS, S.M. (1982). Representation of expert knowledge for consultation: the CASNET and EXPERT projects. In: P.Szolovits (ed), Artificial Intelligence in Medicine. Boulder, CO: Westview Press.

KULIKOWSKI, C.A. (1988). Artificial intelligence in medical consultation systems: a review. IEEE Engineering in Medicine and Biology Society, 34-39.

LANGLOTZ, C.P. and SHORTLIFFE, E.H. (1983). Adapting a consultation system to critique user plans. International Journal of Man-Machine Studies, 19, 479-496.

LAURITZEN, S.L. and SPIEGELHALTER, D.J. (1988). Local computation on graphical structures and their application to expert systems. Journal of the Royal Statistics Society Bulletin, 50, 127-24.

LEDLEY, R.S. and LUSTED, L.B. (1959). Reasoning foundations of medical diagnosis. Science, 130, 9-21.

Van der LEI, J., MUSEN, M.A. and Van der DOES, E. (1990). Critiquing physician decision making using data from automated medical records: assessing the limitations. In: R.M.Miller (ed), Proc. 14th Symposium on Computer Applications in Medical Care. New York, NY: IEEE Computer Society Press, pp.559-563.

Van der LEI, J. and MUSEN, M. (1991). A model for critiquing based on automated medical records. Computers and Biomedical Research, 24, 344-78.

LODWICK, G.S., HAUN, C.L., SMITH, W.E., KELLER, R.F. and ROBERTSON, E.D. (1963). Computer diagnosis of primary bone tumors: a preliminary report. Radiology, 80.

van der LOO, R.P. (1995). Overview of published assessment and evaluation studies. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 261-282.

MASARIE, F.E., MILLER, R.A. and MYERS, J.D. (1985). INTERNIST-I properties: representing common sense and good medical practice in a computerised medical knowledge base. Computers and Biomedical Research, 18(5), 458-479.

MASARIE, F.E. and MILLER, R.A. (1987). INTERNIST-1 to quick medical reference (QMR): the transition from a mainframe to a microcomputer. In: Proc. IEEE 9th Annual Conference of the Engineering in Medicine and Biology Society, Boston. pp1521-1522.

McADAM, W.A., BROCK, B.M., ARMITAGE, T., DAVENPORT, P., CHAN, M. and de DOMBAL, F.T. (1990). Twelve years' experience of computer-aided diagnosis in a district general hospital. Annals of the Royal College of Surgeons England, 72(2), 140-146.

McDONALD, C. (1981). Action-Oriented Decisions in Ambulatory Medicine. Chicago, IL: Year Book Publishers.

McDONALD, C. et al (1983). Data base management, feedback control, and the Regenstrief medical record. Journal of Medical Systems, 7, 111.

McDONALD, C. et al (1984). Reminders to physicians from an introspective medical record: A two year randomized trial. Annals of Internal Medicine, 199, 130.

McDONALD, C. et al (1988). The Regenstrief medical records. M.D. Computing, 5(5), 34.

McDONALD, C.J. and BARNETT, G.O. (1990). Medical-record systems. In: Shortliffe, E.H., Perrault, L., Wiederhold, G. and Fagan, L. (eds), Medical Informatics - Computer Applications in Health Care. Wokingham, England: Addison-Wesley, 181-218.

MILLER, P.L. (1988). Selected Topics in Medical Artificial Intelligence. New York, NY: Springer-Verlag.

MILLER, P.L. (1988b). Artificial intelligence in medicine: an emerging discipline. In: P.L.Miller (ed), Selected Topics in Medical Artificial Intelligence. New York: Springer-Verlag, 11-24, 1988.

MILLER, P.L. and FISHER, P.R. (1988). Causal models for medical artificial intelligence. From: Proceedings, 11th Symposium on Computer Applications in Medical Care, IEEE, 1987, 17-22. In: P.L.Miller (ed), Selected Topics in Medical Artificial Intelligence. New York: Springer-Verlag, 11-24, 1988.

MILLER, R.A., POPLE, H.E. and MYERS, J.D. (1982). Internist-1, an experimental computer-based diagnostic consultant for general internal medicine. The New England Journal of Medicine, 307(8), 468-476.

MILLER, R.A. (1984). INTERNIST-1/CADUCEUS: Problems facing expert consultant programs. Methods of Information in Medicine, 23, 9-14.

MILLER, R.A., SCHAFFNER, K.F. and MEISEL, A. (1985). Ethical and legal issues related to the use of computer programs in clinical medicine. Annals of Internal Medicine, 102(4), 529-537.

MILLER, R., MASARIE, F.E. AND MYERS, J.D. (1986a). Quick medical reference (QMR) for diagnostic assistance. M.D. Computing, 3(5), 34-48.

MILLER, R.A., McNEIL, M., CHALLINOR, S., MASARIE, F. and MYERS, J. (1986b). The INTERNIST/QMR project - status report. Western Journal of Medicine, 145, 816-22.

MILLER, R.A. (1989). Legal issues related to medical decision support systems. International Journal of Clinical Monitoring and Computing, 6(2), 75-80.

MILLER, R.A. and MASARIE, F.E. (1989). Use of the Quick Medical Reference (QMR) program as a tool for medical education. Methods of Information in Medicine, 28(4), 340-345.

MILLER, R.A. and MASARIE, F.E. (1990). The demise of the "Greek oracle" model for medical diagnosis systems. Methods of Information in Medicine, 29, 1-2.

MILLER, R.A. (1994). Medical diagnostic decision support systems -- past, present, and future: a threaded bibliography and brief commentary. Journal of the American Medical Informatics Association, 1(1), 8-27.

O'MOORE, R. (1995). The conception of a medical computer system. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 45-49.

O'MOORE, R., GROTH, T., GRIMSON, W. and BORAN, G. (1996). The OpenLabs Project. Computer Methods and Programs in Biomedicine, 50 (2), 85-86.

MYERS, J.D., POPLE, H.E., MILLER, R.A. (1982). INTERNIST: can artificial intelligence help? In: Connelly, Benson, Burke, Fenderson (eds), Clinical decisions and laboratory use. Minneapolis: University of Minnesota Press, 251-269.

NEWELL, A. (1982). The knowledge level. Artificial Intelligence, 18, 87-127.

NØHR, C. (1995). From assessment to decision-making. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 117-126.

OHMANN, C., de DOMBAL, F.T., WINDING, O. and THE INTERNATIONAL EVALUATION PANEL (1995). Evaluation in the TELEGASTRO-Project. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 167-177.

OVERALL, J.E. and WILLIAMS, C.M. (1963). Conditional probability program for diagnosis of thyroid function. Journal of the American Medical Association, 183, 5.

PARKER, R.C. and MILLER, R.A. (1989). Creation of realistic appearing simulated patient cases using the INTERNIST-1/QMR knowledge base and interrelationship properties of manifestations. Methods of Information in Medicine, 28(4), 346-351.

PATIL, R.S., SZOLOVITZ, P. and SCHWARTZ, W.P. (1981). Causal understanding of patient illness in medical diagnosis. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence. Los Altos, California: Morgan Kaufmann, 893-899.

PATIL, R.S., SZOLOVITS, P. and SCHWARTZ, W.B. (1982). Modelling knowledge of the patient in acid-base and electrolyte disorders. In: P.Szolovits (ed), Artificial Intelligence in Medicine. AAAS Selected Symposia Series, Boulder, CO: Westview Press.

PATIL, R.S., SZOLOVITZ, P. and SCHWARTZ, W.P. (1982b). Information acquisition in diagnosis. In: Proceedings of the National Conference on Artificial Intelligence. Los Altos, California: M. Kaufmann, 345-8.

PATIL, R.S. (1987). A case study on evolution of system building expertise: medical diagnosis. In: E.W.L.Grinson et al. (eds), AI in the 1980's and Beyond. Boston: Massachusets Institute of Technology, 75-108.

PATIL, R. and SENYK, O. (1988). Compiling causal knowledge for diagnostic reasoning. In: P.L.Miller (ed), Selected Topics in Medical Artificial Intelligence. New York, NY: Springer-Verlag.

PAUKER, S.G., GORRY, G.A., KASSIRER, J.P. and SCHWARTZ, W.B. (1976). Towards the simulation of clinical cognition: taking a present illness by computer. American Journal of Medicine, 60, 981-96.

PAUKER, S.G. and SZOLOVITS, P. (1977). Analyzing and simulating taking the history of the present illness: context formation. In: W.Schneider and A.L.Sagvallhein (eds), Computational Linguistics in Medicine. Amsterdam, Netherlands: North-Holland. 109-118.

PAUKER, S.G. and KASSIRER, J.P. (1980). The threshold approach to clinical decision making. New England Journal of Medicine, 302(20), 1109-1117.

PEARL, J. (1982). Reverend Bayes on inference engines: a distributed hierarchical approach. In: Proc. AAAI National Conference on AI, Pittsburgh, 133-136.

PEARL, J. (1986). Fusion, propagation, and structuring in belief networks. Artificial Intelligence, 29, 241-88.

PEARL, J. (1986). On Evidential Reasoning in a Hierarchy of Hypotheses. Artificial Intelligence, 28, 9-15.

PETERSON, L.C., COBB, D.S. and REYNOLDS, D.C. (1995). ICOHR: intelligent computer based oral health record. Medinfo 1995, 8(2), 1709.

PIERSKALLA, P., and WOODS, D. (1988). Computers in hospital management and improvements in patient health care - new trends in the United States. Journal of Medical Systems, 12(6), 411-428.

POPLE, H.E. (1977). The formation of composite hypotheses in diagnostic problem solving: an exercise in synthetic reasoning. In: Proceedings of the Fifth International Joint Conference on Artificial Intelligence. Los Altos, California: M. Kaufmann: 1030-7.

POPLE, H.E. (1982). Heuristic methods for imposing structure on ill-structured problems: the structuring of medical diagnostics. In: Szolovitz, P. (ed), Artificial Intelligence in Medicine. AAAS Symposia Series. Boulder, CO: Westview Press, 119-190.

POPLE, H.E. (1982). The development of clinical expertise in the computer. In: Szolovitz, P. (ed), Artificial Intelligence in Medicine. AAAS Symposia Series. Boulder, CO: Westview Press, 79-117.

PRICE, D.J. and MASON, J. (1986). Resolving the numerical chaos at the bedside. In: J.R.Bryant, J.Roberts and P.Windsor (eds), Proc. Current Perspectives in Health Computing. British Journal of Healthcare Computing, Weybridge, 147-157.

PRYOR, T., GARDNER, R., CLAYTON, P. and WARNER, H. (1983). The HELP system. Journal of Medical Systems, 7, 87-102.

PRYOR, T.A. (1988). The HELP medical record. M.D.Computing, 5(5), 22-33.

PRYOR, T.A. (1990). Development of decision support systems. International Journal of Clinical Monitoring and Computing, 7, 137-46.

RAMONI, M., STAFANELLI, M., MAGNANI, L. and BAROSI, G. (1992). An epistemological framework for medical knowledge-based systems. In: IEEE Transactions on Systems, Man, and Cybernetics, 22(6).

REGGIA, J.A., NAU, D.S. and WANG, P.Y. (1983). Diagnostic expert systems based on a set covering model. International Journal of Man-Machine Studies, 19, 437-60.

RENNELS, G.D. and MILLER, P.L. (1988). Artificial intelligence research in anesthesia and intensive care. Journal of Clinical Monitoring, 4(4), 274-289.

SARANUMMI, N. (1995). Supporting System Development with Technology Assessment. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 35-44.

SCHMIDT, H.G., NORMAN, G.R. and BOSHUITZEN, H.P.A. (1990). A cognitive perspective on medical expertise: theory and implications. Academic Medicine, 65, 611-621.

SCHWARTZ, W.B. (1970). Medicine and the computer. The promise and problems of change. The New England Journal of Medicine, 283, 1257-64.

SCHWARTZ, W.B., PATIL, R.S. and SZOLOVITS, P. (1987). Artificial intelligence in medicine: where do we stand ? The New England Journal of Medicine, 316(11), 685-688.

SHABOT, M.M., LoBUE, M., LEYERLE, B.J. and DUBIN, S.B. (1989). Inferencing strategies for automated ALERTS on critically abnormal laboratory and blood-gas data. SCAMC Proceedings, 13, 54-57.

SHABOT, M.M., LoBUE, M., LEYERLE, B.J. and DUBIN, S.B. (1990). Decision support alerts for clinical laboratory and blood gas data. International Journal of Clinical Monitoring and Computing, 7, 27-31.

SHACHTER, R.D. and HECKERMAN, D.E. (1987). Thinking backward for knowledge acquisition. AI Magazine, 8, 55-63.

SHAFER, G. (1976). A mathematical theory of evidence. Princeton University Press.

SHAFER, G. (1987). Probability judgement in artificial intelligence and expert systems. Statistical Science, 3, 3-16.

SHERMAN, H.B. (1981). A comparative study of computer-aided clinical diagnosis of birth defects. (Technical Report TR-283). Cambridge, MA: MIT Laboratory for Computer Science.

SHORTLIFFE, E.H., AXLINE, S.G., BUCHANAN, B.G., MERIGAN, T.C. and COHEN S.N. (1973). An artificial intelligence program to advise physicians regarding antimicrobial therapy. Computers and Biomedical Research, 6, 544-60.

SHORTLIFFE, E.H., DAVIS, R., AXLINE, S.G., BUCHANAN, B.G., GREEN, C.C. and COHEN, S.N. (1975). Computer-based consultations in clinical therapeutics: explanation and rule acquisition capabilities of the MYCIN system. Computers and Biomedical Research, 8, 303-320.

SHORTLIFFE, E. and BUCHANAN, B. (1975). A model of inexact reasoning in medicine. Mathematical Biosciences, 23, 351-379.

SHORTLIFFE, E.H. (1976). Computer-Based Medical Consultations: MYCIN. New York: American Elsevier/North Holland.

SHORTLIFFE, E.H., BUCHANAN, B.G. and FEIGENBAUM, E.A. (1979). Knowledge engineering for medical decision making: A review of computer-based clinical decision aids. In: Proc. IEEE, 67, 1207-1224.

SHORTLIFFE, E.H. (1982). The computer and medical decision making: good advice is not enough. IEEE Engineering in Medicine and Biology Magazine, 1, 16-18.

SHORTLIFFE, E.H. (1984). The science of biomedical computing. Medical Informatics, 9(3/4), 185-193.

SHORTLIFFE, E.H. (1986). Medical expert systems - knowledge tools for physicians. Western Journal of Medicine, 145(6), 830-39.

SHORTLIFFE, E. (1987). Computer programs to support clinical decision-making. Journal of the American Medical Association, 258(1), 61-66.

SHORTLIFFE, E.H. and FAGAN, L.M. (1989). Research training in medical informatics: the Stanford experience. Academic Medicine, October 1989, 575-578.

SHORTLIFFE, E.H. (1989). Testing reality: the introduction of decision-support technologies for physicians. Methods of Information in Medicine, 28, 1-5.

SHORTLIFFE, E.H. (1990). Clinical decision-support systems. In: Shortliffe, E.H., Perrault, L., Wiederhold, G. and Fagan, L. (eds), Medical Informatics - Computer Applications in Health Care. Wokingham, England: Addison-Wesley, 466-502.

SHORTLIFFE, E.H. (1991). Medical informatics and clinical decision making: the science and the pragmatics. Medical Decision Making, 11(4), s2-14.

SHORTLIFFE, E.H. (1993). The adolescence of AI in medicine: will the field come of age in the '90s? Artificial Intelligence in Medicine, 5, 93-106.

SHULMAN, D., DONCHIN, Y., DEKEL, S., HELLER, O., GOPHER, D. and COTEV, S. (1987). Human errors in an intensive care unit: a pilot study. Critical Care Medicine, 15, 371.

SIEGEL, E.R., CUMMINGS, M.M. and WOODSMALL, R.M. (1990). Bibliographic-retrieval systems. In: Shortliffe, E.H., Perrault, L., Wiederhold, G. and Fagan, L. (eds), Medical Informatics - Computer Applications in Health Care. Wokingham, England: Addison-Wesley, 434-465.

SIGGAARD-ANDERSEN, O., GØTHGEN, I.H., WIMBERLEY, P.D. and FOGH-ANDERSEN, N. (1990). The oxygen status of the arterial blood revised: relevant oxygen parameters for monitoring the arterial oxygen availability. Scandinavian Journal of Clinical and Laboratory Investigation, 50(203), 17-28.

SIGGAARD-ANDERSEN, O. and SIGGAARD-ANDERSEN, M. (1990). The oxygen status algorithm: a computer program for calculating and displaying pH and blood gas data. Scandinavian Journal of Clinical and Laboratory Investigation, 50(203), 29-45.

SITTIG, D.F., PACE, N.L., GARDNER, R.M., BECK, E. and MORRIS, A.H. (1989). Implementation of a computerized patient advice system using the HELP clinical information system. Computers and Biomedical Research, 22, 474-487.

SITTIG, D.F., GARDNER, R.M., MORRIS, A.H. and WALLACE, C.J. (1990). Clinical evaluation of computer-based respiratory care algorithms. International Journal of Clinical Monitoring and Computing, 7(3), 177-185.

SPEICHER, C.D. and SMITH J.W.Jr. (1983). Choosing Effective Laboratory Tests. Philadelphia: W.B. Saunders, p.4.

SPIEGELHALTER, D.J. and KNILL-JONES, R. (1984). Statistical and knowledge-based approaches to clinical decision support systems, with an application in gastroenterology. Journal of the Royal Statistics Society Annals, 147, 35-77.

STEFANELLI, M., BELLAZZI, R., BERZUINI, C., IRONI, L. and QUAGLINI, S. (1992). Towards a general architecture for medical expert systems. In: M.Kahn, J.Smith, B.Buchanan, M.Musen, and P.Szolovits (eds), Artificial Intelligence in Medicine, AAAI Spring Symposia Series. Stanford, CA: Stanford University, 105-112.

STEFANELLI, M. (1993). European research efforts in medical knowledge-based systems. Artificial Intelligence in Medicine, 5, 107-124.

SWETS, J.A. (1988). Measuring the accuracy of diagnostic systems. Science, 240, 1285-93.

SZOLOVITS, P. and PAUKER, S.G. (1978). Categorical and probabilistic reasoning in medical diagnosis. Artificial Intelligence, 11, 115-144.

SZOLOVITS, P. (ed) (1982). Artificial Intelligence in Medicine. Boulder CO: Westview Press (AAAS Symposia Series).

SZOLOVITS, P., PATIL, R.S., SCHWARTZ, W.B. (1988). Artificial intelligence in medical diagnosis. Annals of Internal Medicine, 108, 80-87.

TALMON, J.L. and van der LOO, R. (1995). Literature on assessment of information technology and medical KBS evaluation: studies and methodologies. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 283-327.

TALMON, J. and SMEETS, R. (1995). Case acquisition for knowledge-based decision support system validation. In: E.M.S.J.van Gennip and J.L.Talmon (eds), Assessment and Evaluation of Information Technologies in Medicine. Amsterdam: IOS Press, 67-74.

TVERSKY, A. and KAHNEMAN, D. (1974). Judgement under uncertainty: heuristics and biases. Science, 185, 1124-31.

UCKUN, S. (1992). Model-based reasoning in biomedicine. Critical Reviews in Biomedical Engineering, 19(4), 261-292.

UCKUN, S. (1994). Intelligent systems in patient monitoring and therapy management: A survey of research projects. International Journal of Clinical Monitoring and Computing, 11, 241-253.

WALMSLEY, R.N. and WHITE, G.H. (1983). A Guide to Diagnostic Clinical Chemistry. Melbourne: Blackwell Scientific Publications.

WARNER, H.R., TORONTO, A.F., VEASY, L.G. and STEPHENSON, R. (1961). A mathematical approach to medical diagnosis: Application to congenital heart disease. Journal of the American Medical Association, 177, 177-83.

WARNER, H.R., TORONTO, A.F. and VEASY, L.G. (1964). Experience with Bayes Theorem for computer diagnosis of congenital heart disease. Annals of the New York Academy of Sciences, 115, 2.

WARNER, H. (1979). Computer-Assisted Medical Decision-Making. New York, NY: Academic Press.

WEISS, S.M., KULIKOWSKI, C.A., AMAREL, S. and SAFIR, A. (1978). A model-based method for computer-aided medical decision making. Artificial Intelligence, 11, 145-72.

van der WERFF, A. (1997). Organising health systems for better care and performance by open information technology. In: C.Pappas, N.Maglaveras and J.-R.Scherrer (eds), Proc. Medical Informatics Europe 1997. Thessaloniki: IOS Press, 123-127.

WHITBY, L.G., PERCY-ROBB, I.W. and SMITH, A.F. (1984). Acid-base balance and oxygen transport. Lecture Notes on Clinical Chemistry, Third edition. Oxford: Blackwell Scientific Publications.

WHITING-O'KEEFE, Q.E., SIMBORG, D.W., EPSTEIN, W.V. and WARGER, A. (1985). A computerized summary medical record system can provide more information than the standard medical record. Journal of the American Medical Association, 254(9), 1185-1192.

WIELINGA, B.J., SCHREIBER, A. Th. and BREUKER, J.A. (1992). KADS: a modelling approach to knowledge engineering. Knowledge Acquisition, 4, 5-53.

WILSON, P.D., HORROCKS, J.C., LYNDON, P.J., YEUNG, C.K., PAGE, R.E. and de DOMBAL, F.T. (1975). Simplified computer-aided diagnosis of acute abdominal pain. British Medical Journal, 2(5962), 73-75.

WONG, E.T. and ABENDROTH, T.W. (1996). Reaping the benefits of medical information systems. Academic Medicine, 71(4), 353-357.

WRAITH, S.M., AIKINS, J.S., BUCHANAN, B.G., CLANCEY, W.J., DAVIS, R., FAGAN, L.M., HANNIGAN, J.F., SCOTT, A.C., SHORTLIFFE, E.H., van MELLE, W.J., YU, V.L., AXLINE, S.G. and COHEN, S.N. (1976). Computerized consultation system for selection of antimicrobial therapy. American Journal of Hospital Pharmacology, 33(12), 1304-1308.

WRIGHT, D., MACKENZIE, S.J., BUCHAN, I., CAIRNS, C.S. and PRICE, L.E. (1991). Critical incidents in the intensive therapy unit. The Lancet, 338, 676-678.

WYATT, J. and SPIEGELHALTER, D. (1990). Evaluating medical expert systems: what to test and how ? Medical Informatics, 15, 205-217.

WYATT, J. (1991a). Use and sources of medical knowledge. The Lancet, 338, 1368-1373.

WYATT, J. (1991b). Computer-based knowledge systems. The Lancet, 338, 1431-1436.

WYATT, J. (1992). The evaluation of medical decision aids: Why, what, where, and how? In: 1992 American Medical Informatics Association Spring Congress, Portland, Oregon.

YU, V.L., BUCHANAN, B.G., SHORTLIFFE, E.H., WRAITH, S.M., DAVIS, R., SCOTT, A.C. and COHEN, S.N. (1979a). Evaluating the performance of a computer-based consultant. Computers Programs in Biomedical Research, 9(1), 95-102.

YU, V.L., FAGAN, L.M., WRAITH, S.M., CLANCEY, W.J., SCOTT, A.C., HANNIGAN, J., BLUM, R.L., BUCHANAN, B.G. and COHEN, S.N. (1979b). Antimicrobial selection by computer. A blinded evaluation by infectious disease experts. Journal of the American Medical Association, 242(12), 1279-1282.

YU, V.L. (1983). Conceptual obstacles in computerised medical diagnosis. Journal of Medicine and Philosophy, 8(1), 67-75.

ZADEH, L.A. (1965). Fuzzy sets. Information and Control, 8, 338-353.

ZADEH, L.A. (1972). A fuzzy-set theoretic interpretation of linguistic hedges. Journal of Cybernetics, 2(3), 4-34.

ZADEH, L.A. (1986). Is probability theory sufficient for dealing with uncertainty in AI: a negative view. In: L.N.Kanal and J.F.Lemmer (eds), Uncertainty in Artificial Intelligence. New York: North-Holland, 103-116.

Appendices

Appendix A.

Verification of the Interpretative Task-Domain Model

The ICIMS knowledge-base management system module of the OMS and the disorder knowledge model described in Chapter 4 were used to derive the COB objects which comprised the prototype knowledge-base. Following the construction of the knowledge-base using these tools, the integrated BGAS TDM described in Chapter 3 was applied to the textbook cases presented in this appendix in order to verify the TDM integration process. The parameter classification profiles are shown in Tables 5-2, 5-3 and 5-4. Two data sets were used for this purpose. The first set was obtained from textbook case studies (Walmsley and White, 1983), and the second from the original retrospective evaluation study (Chelsom, 1990). The cases examined in this appendix also serve to illustrate the mechanisms of acid-base balance pathophysiology as encapsulated in the GDM.

Case 1: Metabolic acidosis - diabetic ketoacidosis

The patient concerned is a known insulin dependent diabetic, presenting in coma. The laboratory findings for this patient are presented in Table A-1.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.09		acidaemia
pCO_2	4.7 - 6.0	2.7	kPa	hypocapnia
pO_2	10.6 - 13.3	14.8	kPa	hyperoxia
$[HCO_3^-]$	24 - 32	6	mmol/L	hypobicarbonataemia
Anion gap	8-17	30	mEq/L	high anion gap
Na	132 - 144	135	mmol/L	Normonatraemia
K^+	3.2 - 4.8	5.7	mmol/L	Hyperkalaemia
Cl^-	95 - 110	101	mmol/L	Normochloraemia

Table A-1. Laboratory findings for case 1.

Case 1: Pathophysiology

1. Increased production of non-volatile acids (acetoacetate, hydroxybutyrate) causes rapid consumption of buffer bicarbonate, and thus a decreased bicarbonate concentration.

$$pH(\downarrow\downarrow) \propto [HCO_3^-] (\downarrow\downarrow) / pCO_2 (N)$$

2. The kidney is unable to rapidly (a) excrete hydrogen ion and regenerate bicarbonate, (b) excrete the endogenous acid anions, so the low bicarbonate concentration continues and the acid anions build up in the plasma, causing an increased anion gap.

3. Respiratory compensation is attempted by increasing ventilation so as to lower the pCO_2, but in this case the response is inadequate and the hydrogen ion concentration remains above normal (partial compensation).

$$pH(\downarrow) \propto [HCO_3^-] (\downarrow\downarrow) / pCO_2 (\downarrow)$$

4. Hyperkalaemia is due to (a) increased release of K^+ from cellular protein which, in the absence of insulin, is now being catabolised to fuel gluconeogenesis, (b) acidaemia.

The main causes of metabolic acidosis are listed in Table A-2. Table A-3 lists the clinical classification of lactic acidosis.

Case 1: ICIMS Interpretation = Partially compensated metabolic acidosis.

These results are in agreement with the source interpretation. A partially compensated metabolic acidosis is characterised by a low pH, a low $[HCO_3^-]$, and a low to normal pCO_2. However, full compensation does not occur in metabolic acid-base disorders and consequentially the low pCO_2 is interpreted by the system as low to normal in the context of the low pH. In the context of a normal or low to normal pH, the clinical data would be interpreted as indicating a compensated metabolic acidosis.

Increased acid production (load)	Reduced acid excretion
Ketoacidosis	Renal insufficiency (failure, acute, chronic)
Diabetes	Renal tubular acidosis
Alcoholism (acute and chronic)	Hypoaldosteronism
Starvation	
Lactic acidosis	Loss of bicarbonate (bases)
Shock of any type (hypovolaemic, haemorrhagic, septic); Hypoxia (hypoxaemic, anoxic, circulatory, histotoxic); Drugs (e.g. phenformin); Toxic substances (methanol, salicylate, ethylene glycol); Failure of enzymatic pyruvate utilisation	Gastrointestinal tract
	Diarrhoea (intestinal fistula)
	Pancreatic fistula
	Renal
	Proximal renal tubular acidosis
	Ureterosigmoidostomy
	Inhibitors of carbonic anhydrase
Poisoning	
Methanol, salicylate, ethylene glycol, paraldehyde	Excess acid intake
	Sodium and ammonium chloride
	Aminoacids

Table A-2. Common causes of metabolic acidosis (source: WALMSLEY, R.N. and WHITE, G.H. (1983). A Guide To Diagnostic Clinical Chemistry. *Melbourne, Australia: Blackwell Scientific Publications*; Harrison's Principles of Internal Medicine, 11th Edition. New York, NY: McGraw-Hill).

Type A	Drugs/toxins
Poor tissue oxygenation	Biguanides
Shock (any cause)	Ethanol
Profound anaemia	Methanol
Respiratory failure	Fructose
Cyanide poisoning	Sorbitol
Carbon monoxide poisoning	Xylitol
	Streptozotocin
Type B	Salicylates
Common disorders	Isoniazid
Diabetes mellitus	Congenital enzyme defects
Renal failure	Type I Glycogen storage disease (glucose 6-phosphatase deficiency)
Hepatic failure	Fructose-1,6-diphosphatase deficiency
Severe infection	Pyruvate decarboxylase deficiency
Malignancies (lymphoma, leukaemia, sarcoma)	Pyruvate dehydrogenase deficiency
	Miscellaneous
Seizures	D-lactic acidosis
Alkaloses	

Table A-3. Clinical classification of lactic acidosis (source: ARIEFF, A.I. and DeFRONZO, R.A. Fluid electrolyte and acid-base disorders, Vol.1. New York, NY: Churchill Livingstone).

Case 2: Metabolic acidosis - high anion gap (renal failure)

Adult patient with chronic renal failure. The laboratory findings for this patient are presented in Table A-4.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.28		acidaemia
pCO_2	4.7 - 6.0	3.5	kPa	hypocapnia
pO_2	10.6 - 13.3	12.2	kPa	normoxia
$[HCO_3^-]$	24 - 32	12	mmol/L	hypobicarbonataemia
Anion gap	8-17	34	mEq/L	high anion gap
Na	132 - 144	135	mmol/L	Normonatraemia
K^+	3.2 - 4.8	5.6	mmol/L	Hyperkalaemia
Cl^-	95 - 110	96	mmol/L	Normochloraemia

Table A-4. Laboratory findings for case 2.

Case 2: Pathophysiology

Renal failure and diabetic ketoacidosis are examples of a high anion gap acidosis. Major causes of this condition are: (1) ketoacidosis, (2) lactic acidosis, (3) renal failure, (4) ingestion (see Table A-2 and A-3).

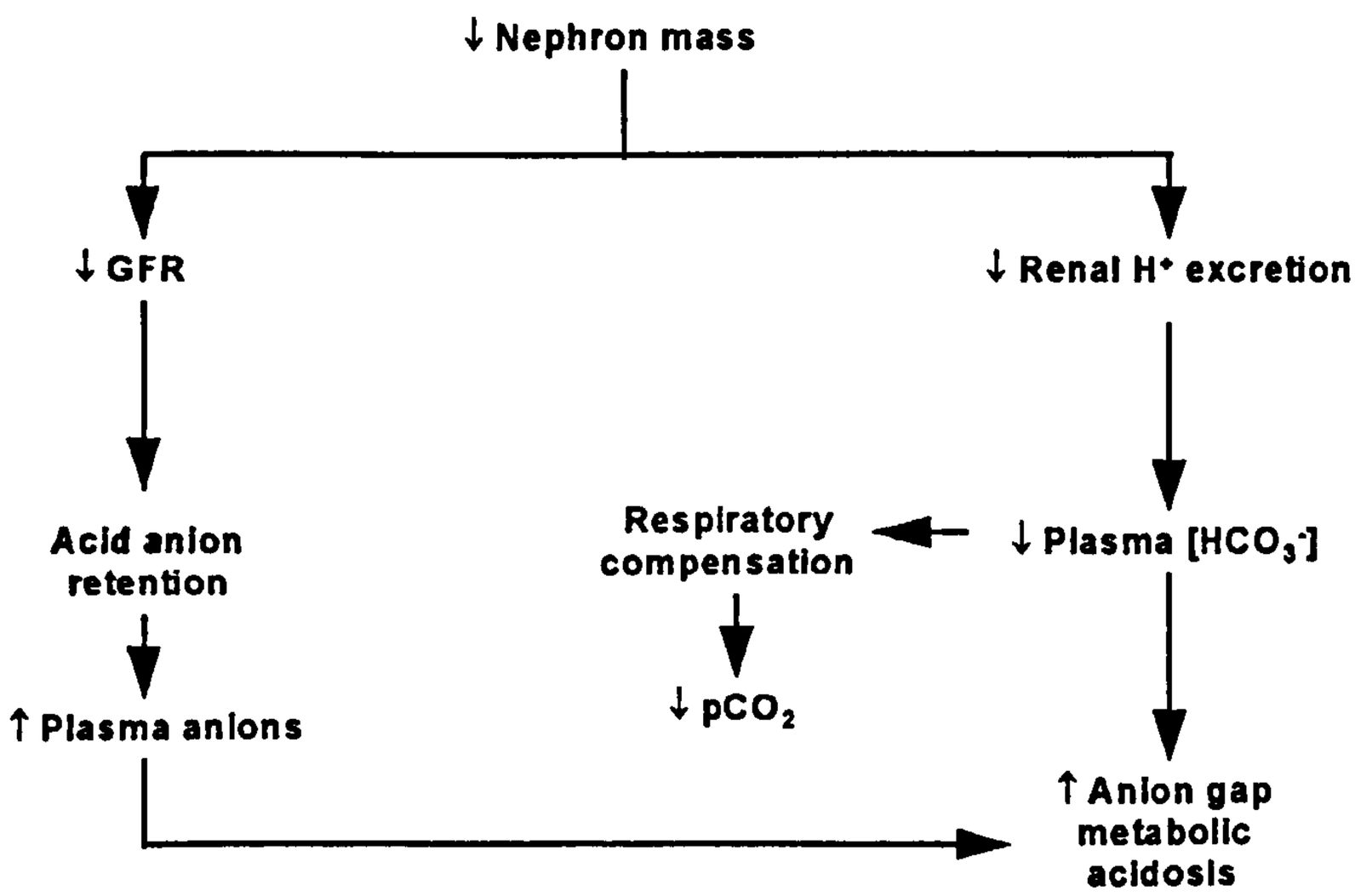


Figure A-1. Mechanism of high anion gap metabolic acidosis in chronic renal failure (from Walmsley and White, 1983).

The pathophysiological mechanism of high anion gap metabolic acidosis in chronic renal failure is depicted in Figure A-1. Table A-5 lists other manifestations of metabolic acidosis.

Electrolyte metabolism	Renal effects
Potassium	Sodium, potassium wasting
Calcium	Uric acid retention
Metabolic effects	Gastrointestinal effects
Protein wasting	Emesis (vomiting)
Altered organic acid synthesis	GI transport inhibition
Catecholamine secretion and action altered	
Aldosterone stimulation	Cardiovascular effects
Parathyroid hormone stimulation	Cardiac
1,25-dihydroxy vitamin D synthesis inhibited	Contraction defect
	Conduction defect
	Peripheral arteriolar vasodilation
Pulmonary effects	Venoconstriction
Kussmaul respiration	
Pulmonary vasoconstriction	
Oxygen transport: enhanced tissue delivery	

Table A-5. Manifestations of metabolic acidosis (source: ARIEFF, A.I. and DeFRONZO, R.A. Fluid electrolyte and acid-base disorders, Vol.1. New York, NY: Churchill Livingstone).

Case 2: ICIMS Interpretation = Partially compensated metabolic acidosis.

These results are in agreement with the source interpretation (see case 1).

Case 3: Metabolic acidosis - hyperchloraemic, RTA

Infant with chronic acidosis and failure to thrive (distal RTA). The laboratory findings for this patient are presented in Table A-6.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.24		acidaemia
pCO_2	4.7 - 6.0	3.2	kPa	hypocapnia
pO_2	10.6 - 13.3	10.7	kPa	normoxia
$[HCO_3^-]$	24 - 32	10	mmol/L	hypobicarbonataemia
Anion gap	8-17	15	mEq/L	normal anion gap
Na	132 - 144	139	mmol/L	Normonatraemia
K^+	3.2 - 4.8	2.9	mmol/L	Hypokalaemia
Cl^-	95 - 110	116	mmol/L	Hyperchloraemia

Table A-6. Laboratory findings for case 3.

Case 3: Pathophysiology

Renal tubular acidosis (RTA) results from ineffective tubular secretion of hydrogen ion, and depending on the location of the defect it may result in HCO_3^- wasting (proximal RTA, Type II) or failure to excrete the daily metabolic load of acid (distal RTA, Type I). In the proximal form, the HCO_3^- wasting is usually associated with other proximal tubular defects (phosphaturia, glycosuria, aminoaciduria). The distal H^+ secretory mechanism is intact and these patients are capable, during severe acidosis, of producing a urine of low pH (<5.3). Subjects with the distal defect are able to reabsorb HCO3- in the proximal tubule, but cannot under any circumstances acidify the urine, i.e. urine pH is always greater than 5.3, even during severe systemic acidosis. The pathophysiological mechanism of hyperchloraemic metabolic acidosis in renal tubular acidosis is depicted in Figure A-2. Table A-7 lists the differential diagnosis of normal anion gap metabolic acidosis.

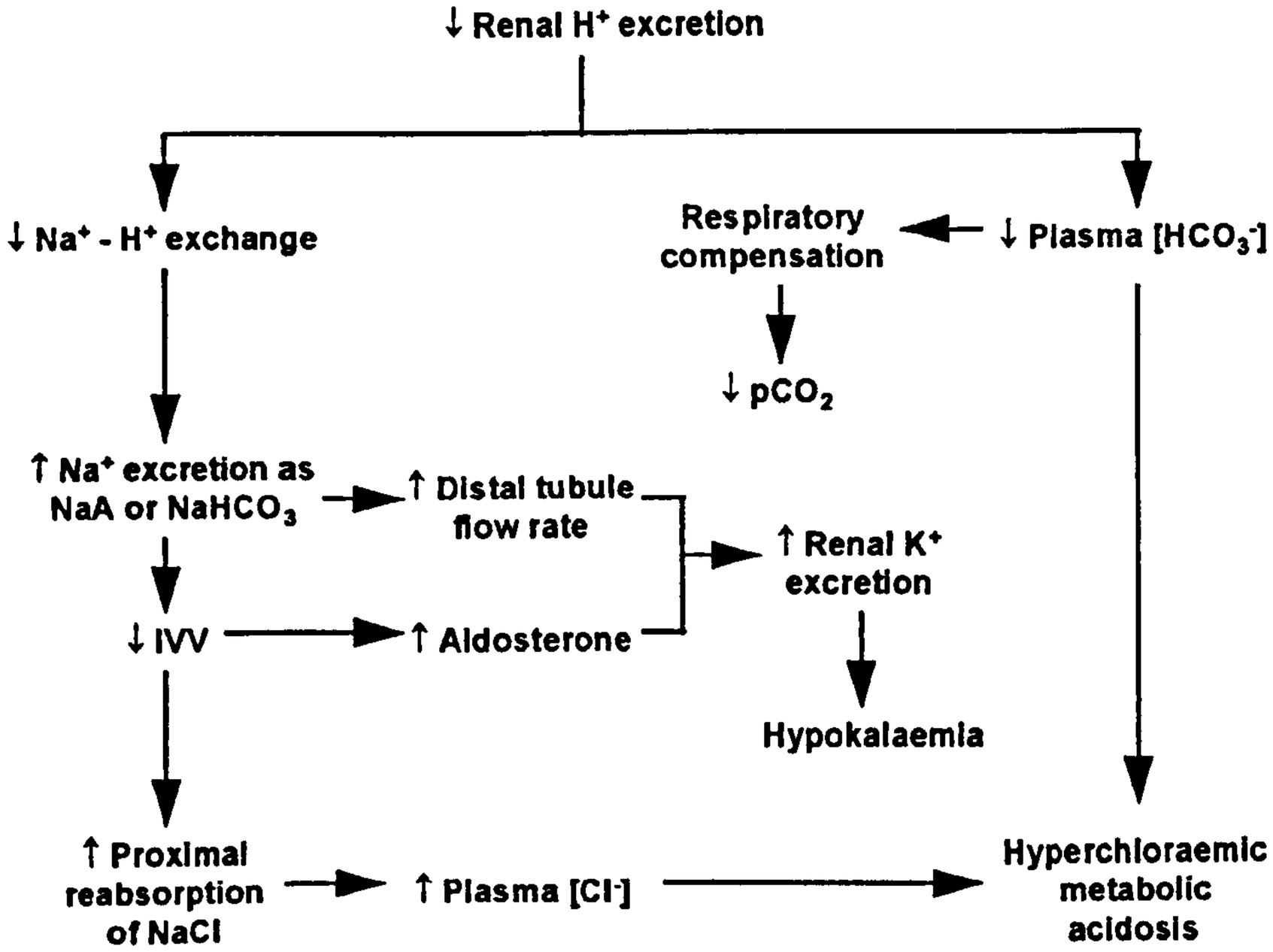


Figure A-2. Mechanism of hyperchloraemic metabolic acidosis in renal tubular acidosis. NaA = sodium salt of acid anions (A). (from Walmsley and White, 1983).

Case 3: ICIMS Interpretation = Partially compensated metabolic acidosis.

These results are in agreement with the source interpretation. The pH value is still considered as pathophysiological (see Cases 1 and 2).

Normal-High Serum Potassium	Low Serum Potassium
Hyperalimination	Gastrointestinal disorders
Posthypocapnia	Diarrhoea
Rapid hydration (dilutional acidosis)	Pancreatic, biliary fistula
Hypoaldosteronism	Ureteral diversions
Hyporeninism	Ureterosigmoidostomy
Selective or diffuse adrenal damage	Ileal bladder (obstructed)
Failure of tubular response to aldosterone	Renal tubular acidosis
NH_4Cl, $CaCl_2$ (oral), lysine or arginine hydrochloride therapy	Proximal RTA
	Distal RTA
Early 'uraemic' acidosis	Lack of buffer

Table A-7. Differential diagnosis of normal anion gap metabolic acidosis (source: ARIEFF, A.I. and DeFRONZO, R.A. Fluid electrolyte and acid-base disorders, Vol.1. New York, NY: Churchill Livingstone).

Case 4: Metabolic acidosis - loss of bicarbonate (diarrhoea)

A 22 year-old man admitted with a history of several days diarrhoea. The laboratory findings for this patient are presented in Table A-8.

Measured parameter	**Physiological range**	**Measurement value**	**Units**	**Finding**
pH	7.35 - 7.45	7.29		acidaemia
pCO_2	4.7 - 6.0	3.45	kPa	hypocapnia
pO_2	10.6 - 13.3	11.0	kPa	normoxia
$[HCO_3^-]$	24 - 32	12	mmol/L	hypobicarbonataemia
Anion gap	8-17	16	mEq/L	normal anion gap
Na	132 - 144	138	mmol/L	Normonatraemia
K^+	3.2 - 4.8	3.1	mmol/L	Hypokalaemia
Cl^-	95 - 110	114	mmol/L	Hyperchloraemia

Table A-8. Laboratory findings for case 4.

Case 4: Pathophysiology

Diarrhoea → loss of $NaHCO_3$ (small intestinal fluid) → Na^+ and HCO_3^- depletion → (1) ↓↓ $[HCO_3^-]$, (2) renal retention of Na^+ (as NaCl) → ↑ plasma $[Cl^-]$ (also any NaCl taken orally will be retained).

$\downarrow\downarrow [HCO_3^-] \rightarrow pH(\downarrow\downarrow) \propto [HCO_3^-] (\downarrow\downarrow) / pCO_2$ (N) - (metabolic acidosis)

acidosis stimulates respiration → increased excretion of $CO_2 \rightarrow \downarrow pCO_2$,

i.e. pH ($\downarrow$) $\propto [HCO_3^-] (\downarrow\downarrow) / pCO_2 (\downarrow)$ - (partially compensated metabolic acidosis).

Case 4: ICIMS Interpretation = Partially compensated metabolic acidosis.

These results are in agreement with the source interpretation. The pH value is still considered as pathophysiological (see Cases 1, 2 and 3).

Case 5: Respiratory alkalosis

Patient with gram-negative septicaemia and hyperventilation (mechanical ventilation). The laboratory findings for this patient are presented in Table A-9. The clinical classification of respiratory alkalosis is listed in Table A-10.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.44		euphaemia
pCO_2	4.7 - 6.0	4.0	kPa	hypocapnia
pO_2	10.6 - 13.3	14.0	kPa	hyperoxia
$[HCO_3^-]$	24 - 32	20	mmol/L	hypobicarbonataemia
Anion gap	8-17		mEq/L	
Na	132 - 144	144	mmol/L	Normonatraemia
K^+	3.2 - 4.8	3.5	mmol/L	Normokalaemia
Cl^-	95 - 110	111	mmol/L	Hyperchloraemia

Table A-9. Laboratory findings for case 5.

Hypoxia	Hypermetabolic states
Lung diseases: pneumonia, asthma atelectasis, fibrosis, etc.	Fever
Pulmonary oedema	Thyrotoxicosis
Cyanotic heart disease	Anaemia
High altitude	
	Salicylate poisoning
Central nervous system disorders	Septicaemia
Cerebral diseases: tumour, encephalities, meningities	Liver cirrhosis
Subarachnoid haemorrhage	Pregnancy
Psychogenic hyperventilation, anxiety	Physical exercise
	Pain

Table A-10. Causes of respiratory alkalosis (source: ARIEFF, A.I. and DeFRONZO, R.A. Fluid electrolyte and acid-base disorders, Vol.1. New York, NY: Churchill Livingstone).

Case 5: Pathophysiology

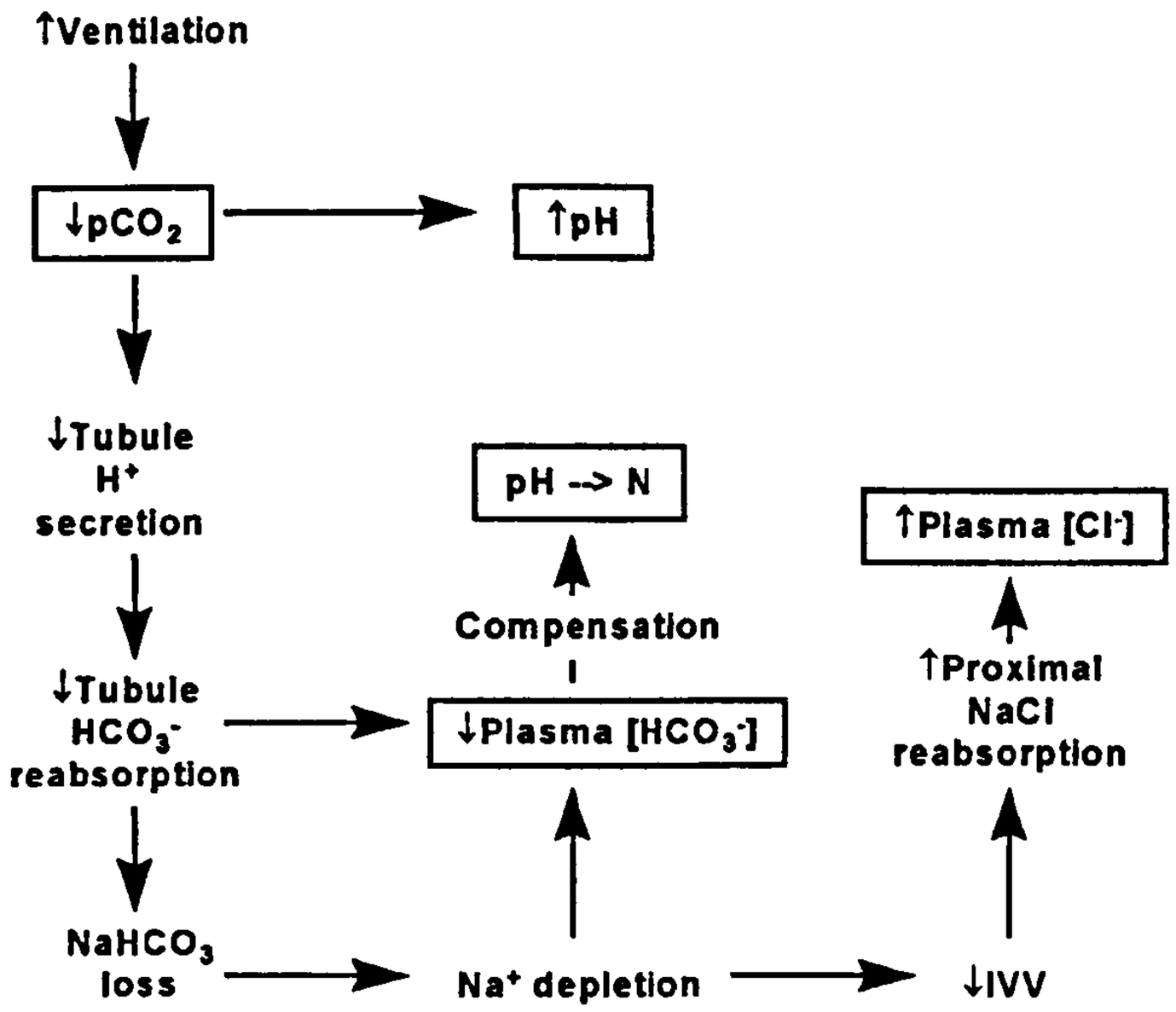


Figure A-3. Mechanism of a compensated respiratory alkalosis (from Walmsley and White, 1983).

Case 5: ICIMS Interpretation = Fully compensated respiratory acidosis.

The result is in agreement with the source interpretation.

Case 6: Metabolic alkalosis

Infant with projectile vomiting (pyloric stenosis). The laboratory findings for this patient are presented in **Table A-11**. The clinical classification of metabolic alkalosis is listed in **Table A-12**.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.58		alkalaemia
pCO_2	4.7 - 6.0	6.0	kPa	normocapnia
pO_2	10.6 - 13.3	6.8	kPa	hypoxia
$[HCO_3^-]$	24 - 32	41	mmol/L	hyperbicarbonataemia
Anion gap	8-17		mEq/L	
Na	132 - 144	131	mmol/L	hyponatraemia
K^+	3.2 - 4.8	2.1	mmol/L	hypokalaemia
Cl^-	95 - 110	76	mmol/L	Hypochloraemia

Table A-11. Laboratory findings for case 6.

Case 6: Pathophysiology

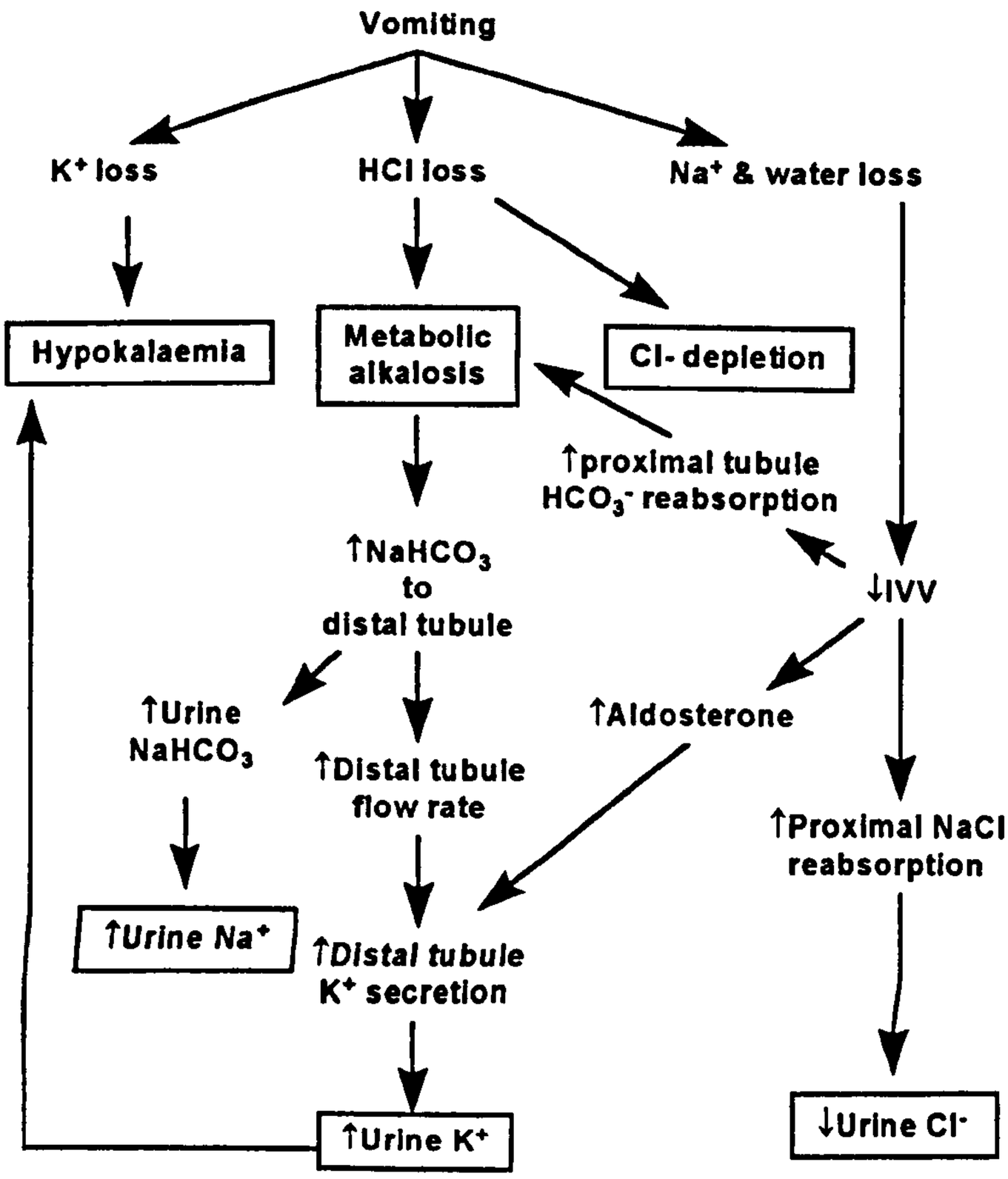


Figure A-4. Mechanism of hypokalaemic alkalosis due to vomiting from above the pylorus (from Walmsley and White, 1983).

Associated with H^+ depletion	Severe potassium depletion
Prolonged vomiting	Diarrhea
Diuretic therapy	Cirrhosis
Post-hypercapnic metabolic alkalosis	
Hypoparathyroidism	Excess alkali intake
	Excess $NaHCO_3$ administration
Associated with mineralcorticoid hyperactivity	Excess administration of some antacids
Cushing's syndrome	
Primary hyperaldosteronism	
Bartter's syndrome	

Table A-12. Causes of metabolic alkalosis (source: ARIEFF, A.I. and DeFRONZO, R.A. Fluid electrolyte and acid-base disorders, Vol.1. New York, NY: Churchill Livingstone).

Case 6: ICIMS Interpretation = Metabolic alkalosis.

The result was in agreement with the source interpretation.

Case 7: Respiratory acidosis

Patient with emphysema. The laboratory findings for this patient are presented in Table A-13. The clinical classification of respiratory acidosis is presented in Table A-14.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.36		euphaemia
pCO_2	4.7 - 6.0	8.4	kPa	hypercapnia
pO_2	10.6 - 13.3	7.0	kPa	hypoxia
$[HCO_3^-]$	24 - 32	35	mmol/L	hyperbicarbonataemia
Anion gap	8-17		mEq/L	
Na	132 - 144	135	mmol/L	Normonatraemia
K^+	3.2 - 4.8	3.8	mmol/L	Normokalaemia
Cl^-	95 - 110	88	mmol/L	Hypochloraemia

Table A-13. Laboratory findings for case 7.

Case 7: Pathphysiology

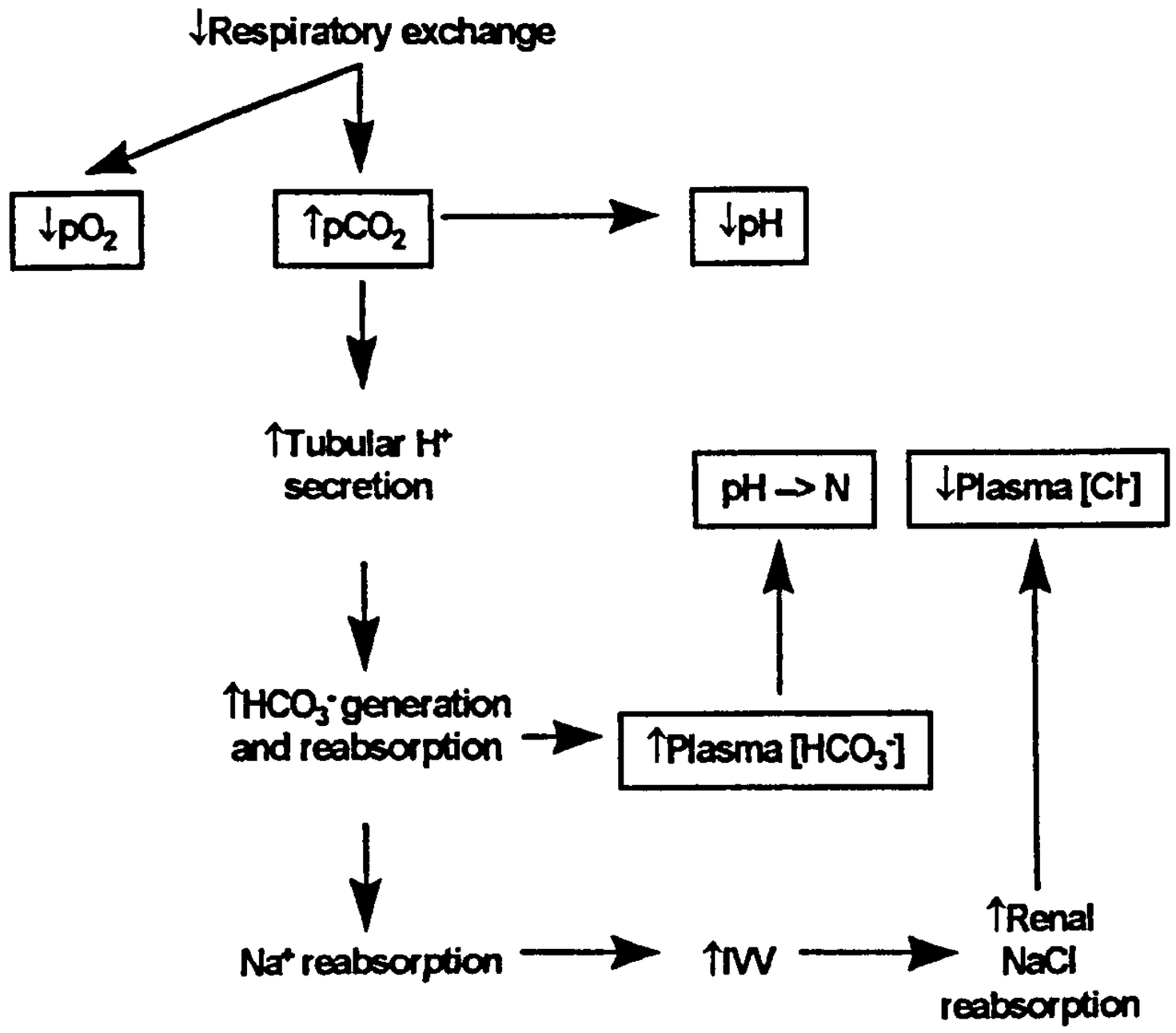


Figure A-5. Mechanism of respiratory acidosis with partial compensation (from Walmsley and White, 1983).

Alterations of alveolocapillary diffusion and perfusion	Inhibition of bulbar respiratory centers
Chronic bronchities	Drugs: opioids, anaesthetics, sedatives
Emphysema	Excess O_2 therapy in chronic hypercapnia
Severe asthma	Lesions of the central nervous system (rare)
Chronic obstructive lung disease	Cardiac arrest
Acute pulmonary oedema (rare)	
Diseases or alterations of repsiratory muscles or the rib cage	
Muscular insufficiency	
Kyphoscoliosis	
Severe obesity (Pickwickian syndrome)	

Table A-14. Causes of respiratory acidosis (source: ARIEFF, A.I. and DeFRONZO, R.A. Fluid electrolyte and acid-base disorders, Vol.1. New York, NY: Churchill Livingstone).

Case 7: ICIMS Interpretation = Fully compensated respiratory acidosis.

The source interpretation was partial compensation.

Case 8: Mixed respiratory and metabolic acidosis

Patient with cardio-pulmonary arrest. The laboratory findings for this patient are presented in Table A-15.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.01		acidaemia
pCO_2	4.7 - 6.0	8.8	kPa	hypercapnia
pO_2	10.6 - 13.3	6.8	kPa	hypoxia
$[HCO_3^-]$	24 - 32	16	mmol/L	hypobicarbonataemia
Anion gap	8-17		mEq/L	
Na	132 - 144		mmol/L	
K^+	3.2 - 4.8		mmol/L	
Cl^-	95 - 110		mmol/L	

Table A-15. Laboratory findings for case 8.

Case 8: Pathophysiology

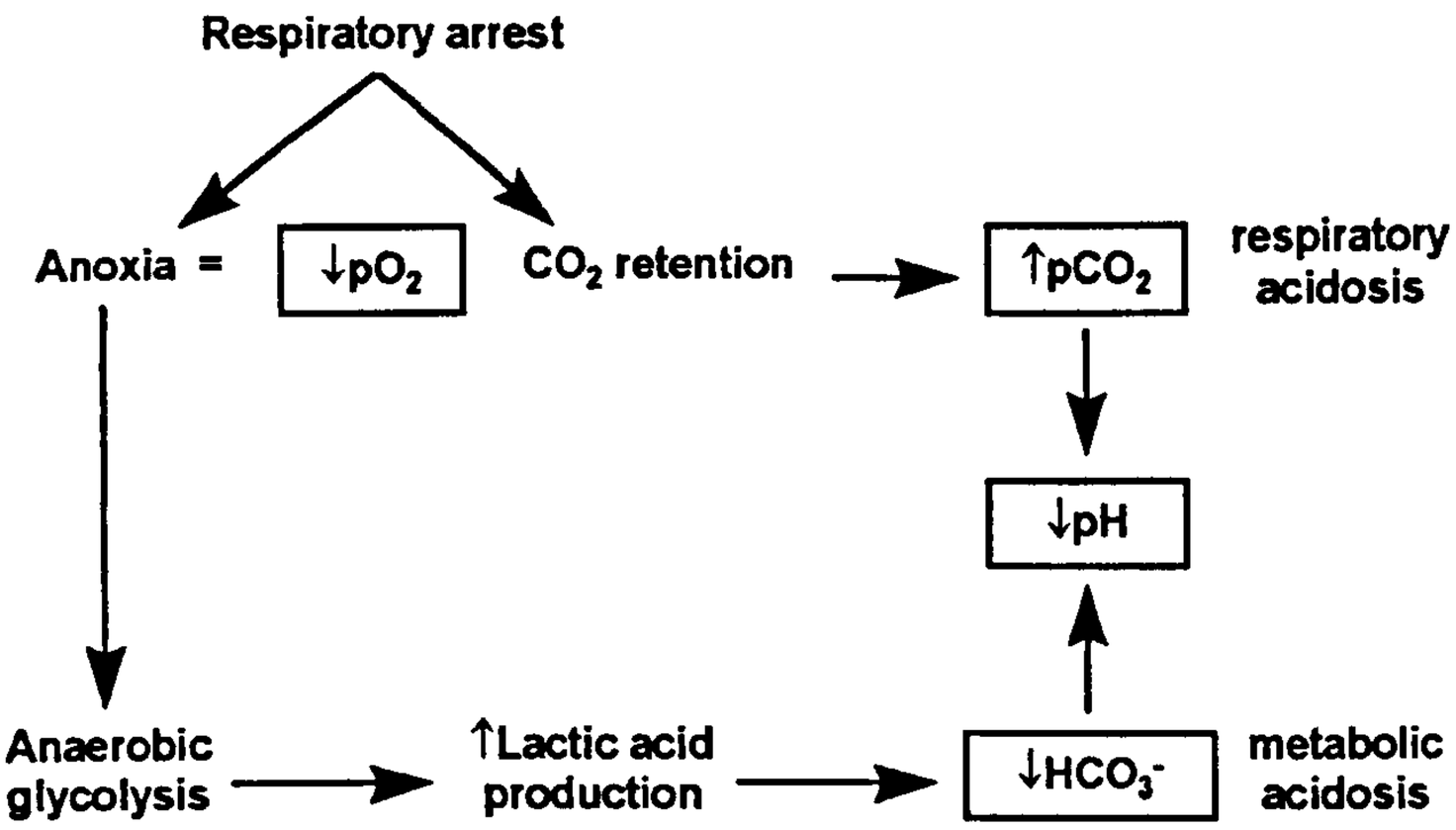


Figure A-6. Mechanism of mixed respiratory and metabolic acidosis in respiratory arrest (from Walmsley and White, 1983).

Case 8: ICIMS Interpretation = *Respiratory and metabolic acidosis.*

The result was in agreement with the source interpretation.

Case 9: Mixed respiratory and metabolic alkalosis

Male patient, aged 70 years, with respiratory failure due to chronic obstructive airway disease. He was admitted to ICU and mechanically ventilated. The laboratory findings listed in Table A-16 refer to 5 hours after ventilation was initiated.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.66		alkalaemia
pCO_2	4.7 - 6.0	4.2	kPa	hypocapnia
pO_2	10.6 - 13.3	9.9	kPa	hypoxia
$[HCO_3^-]$	24 - 32	35	mmol/L	hyperbicarbonataemia
Anion gap	8-17		mEq/L	
Na	132 - 144		mmol/L	
K^+	3.2 - 4.8		mmol/L	
Cl^-	95 - 110		mmol/L	

Table A-16. Laboratory findings for case 9.

Case 9: ICIMS Interpretation = Respiratory and metabolic alkalosis.

Result in agreement with source interpretation.

Case 10: Mixed respiratory acidosis and metabolic alkalosis

Male aged 82 years, with chronic obstructive airway disease and congestive cardiac failure, who has been on long term thiazide therapy. The laboratory findings for this patient are presented in Table A-17.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.43		euphaemia
pCO_2	4.7 - 6.0	9.9	kPa	hypercapnia
pO_2	10.6 - 13.3	8.1	kPa	hypoxia
$[HCO_3^-]$	24 - 32	48	mmol/L	hyperbicarbonataemia
Anion gap	8-17		mEq/L	
Na	132 - 144	131	mmol/L	hyponatraemia
K^+	3.2 - 4.8	2.2	mmol/L	hypokalaemia
Cl^-	95 - 110	72	mmol/L	hypochloraemia

Table A-17. Laboratory findings for case 10.

Case 10: ICIMS Interpretation = Respiratory acidosis and metabolic alkalosis.

The result is in agreement with the source interpretation. The complex disorder was recognised by superimposing the pathophysiological PSM (hyperdynamic compensation) on the clinical picture of the patient. Thiazide therapy indicates primary metabolic alkalosis due to potassium deficiency consequent to the diuretic therapy. Airway disease indicates a primary respiratory acidosis superimposed on the metabolic alkalosis.

Case 11: Mixed respiratory alkalosis and metabolic acidosis

45 year-old female patient who attempted suicide by consuming large amounts of aspirin (approximately 100g, 12 hours prior to admission). On presentation she was comatose and had peripheral cyanosis. The laboratory findings for this patient are presented in Table A-17.

Measured parameter	Physiological range	Measurement value	Units	Finding
pH	7.35 - 7.45	7.38		euphaemia
pCO_2	4.7 - 6.0	3.5	kPa	hypocapnia
pO_2	10.6 - 13.3	8.0	kPa	hypoxia
$[HCO_3^-]$	24 - 32	15	mmol/L	hypobicarbonataemia
Anion gap	8-17		mEq/L	
Na	132 - 144	142	mmol/L	normonatraemia
K^+	3.2 - 4.8	5.5	mmol/L	hyperkalaemia
Cl^-	95 - 110	98	mmol/L	normochloraemia

Table A-18. Laboratory findings for case 11.

Case 11: ICIMS Interpretation = Respiratory alkalosis and metabolic acidosis.

The result is in agreement with the source interpretation. The complex disorder was recognised by superimposing the pathophysiological PSM (hypodynamic compensation) on the clinical picture of the patient. Salicylate poisoning affects both the central nervous system and thus respiration as well as certain metabolic pathways causing overproduction of organic acids, including lactic acid. Thus, salicylate poisoning indicates the presence of two primary disorders: respiratory alkalosis and metabolic acidosis.

Appendix B

Results of a retrospective evaluation against the Siggaard-Andersen Nomogram as a 'Gold data-interpretation Standard'

Case 1: Chronic obstructive pulmonary disease and pickwick syndrome in a 61 year-old man

During air travel the patient developed acute respiratory insufficiency and was admitted to the ICU. He was mechanically ventilated with an oxygen fraction of 0.8 (80%).

1.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.491 (7.373 - 7.433)
2.	Arterial CO2 tension (at 37°C) =	5.69	(4.86 - 6.10) kPa
3.	Plasma [HCO3-] =	32.2	(22.1 - 28.2) mmol/L

1.2. Gold standard

NORMOCAPNIA, moderate ALKALAEMIA, moderate metabolic alkalosis.

1.3. ICIMS

1. Uncompensated metabolic alkalosis (0.715)
2. Partially compensated metabolic alkalosis (0.285)

Agreement. Metabolic alkalosis is the only primary disorder. The qualifier 'moderate' refers to lack of compensation or perhaps slight compensation. Normocapnia excludes the presence of a 'respiratory component'.

Case 2: Case 1 - second sample, spontaneous ventilation

2.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.346 (7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	8.92	(4.86 - 6.10) kPa
3.	Plasma [HCO_3^-] =	36.1	(22.1 - 28.2) mmol/L

2.2. Gold standard

HYPERCAPNIA, slight ACIDAEMIA, MARKED metabolic alkalosis.

2.3. ICIMS

1. Respiratory acidosis and metabolic alkalosis (0.804)
2. Partially compensated respiratory acidosis (0.196)

Agreement. Complex disorder comprises a marked (uncompensated) metabolic alkalosis and a slight acidaemia caused by a partially compensated respiratory acidosis (hypercapnia).

Case 3: Terminal chronic obstructive pulmonary disease in a 72 year-old woman

The patient is in a steady state and breathing spontaneously without supplementary oxygen. She is awake and conscious, she can speak, and is mobile in a wheel chair.

3.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.316 (7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	10.94	(4.86 - 6.10) kPa
3.	Plasma [HCO_3^-] =	41.4	(22.1 - 28.2) mmol/L

3.2. Gold standard

EXTREME HYPERCAPNIA, moderate ACIDAEMIA, MARKED metabolic alkalosis.

3.3. ICIMS

1. Respiratory acidosis and metabolic alkalosis (0.511)
2. Partially compensated respiratory acidosis (0.489)

Comment: **Agreement**. The change in belief value for the two components of the interpretation signifies the increased acidaemia compared with the previous case. In this case, the distribution of evidence over the pathophysiological network is more uniformly spread between the metabolic and respiratory components.

Case 4: Extreme arterial hypoxia in a 25 year-old man (R.Messner) at the top of Mt. Everest without supplementary oxygen

The example shows the limits of tolerance of a normal healthy adult. An arterial blood sample was not drawn at the summit but the oxygen status was reconstructed on the basis of data from the literature recorded at high altitude (8848m).

4.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.743 (7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	1.50	(4.86 - 6.10) kPa
3.	Plasma $[HCO_3^-]$ =	15.2	(22.1 - 28.2) mmol/L

4.2. Gold standard

EXTREME HYPOCAPNIA, EXTREME ALKALAEMIA, slight metabolic acidosis.

4.3. ICIMS

1. Partially compensated respiratory alkalosis (0.554)
2. Uncompensated respiratory alkalosis (0.446)

Comment: **Agreement**. The alkalaemia is due to the extreme hypocapnia. A slight metabolic acidosis is manifested in slight compensation as indicated in the narrow belief margin. However, this is a physiological response and does not represent a second primary disorder. Cases such as this (very high pH with partial compensation) will be further refined by observing state-trajectories (Section 5.4.5).

Case 5: Cardiac incompensation and plural excusation in a 54 year-old man

5.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.564 (7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	4.49	(4.86 - 6.10) kPa
3.	Plasma [HCO_3^-] =	30.1	(22.1 - 28.2) mmol/L

5.2. Gold standard

slight HYPOCAPNIA, MARKED ALKALAEMIA, moderate metabolic alkalosis.

5.3. ICIMS

1. Respiratory and metabolic alkalosis (0.883)
2. Uncompensated metabolic alkalosis (0.117)

Comment: **Agreement**. The marked alkalaemia is caused by a moderate metabolic alkalosis and a slight hypocapnia, i.e. metabolic and respiratory alkalosis.

Case 6: Case 5 - second sample, after eliminating 3000 mL of fluid

6.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.480 (7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	6.14	(4.86 - 6.10) kPa
3.	Plasma [HCO_3^-] =	33.9	(22.1 - 28.2) mmol/L

6.2. Gold standard

slight HYPERCAPNIA, moderate ALKALAEMIA, moderate metabolic alkalosis.

6.3. ICIMS

1. Partially compensated metabolic alkalosis (0.585)
2. Compensated metabolic alkalosis (0.415)

Comment: **Agreement**. Slight hypercapnia compensates the alkalaemia which is now moderate (partially compensated metabolic alkalosis).

Case 7: Severe haemorrhagic anaemia in a 25 year-old woman

The patient was injured in a traffic accident necessitating a splenectomy. She suffered a great deal of blood loss. The haemoglobin concentration fell to 2.2 mmol/L but she refused blood transfusion for religious reasons (Jehove witness). She was breathing spontaneously. An arterial blood sample was drawn after 20 minutes without supplementary oxygen. Patient is hyperventilating. Since blood transfusion was refused, the only way to improve the oxygen status was by administering pure oxygen.

7.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =	7.403	(7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	2.50	(4.86 - 6.10) kPa
3.	Plasma [HCO_3^-] =	11.6	(22.1 - 28.2) mmol/L

7.2. Gold standard

EXTREME HYPOCAPNIA, NEUTRALAEMIA, MARKED metabolic acidosis.

7.3. ICIMS

1. Respiratory alkalosis and metabolic acidosis (0.705)
2. Normal blood gases (0.295)

Comment: **Agreement**. Extreme hypocapnia causes respiratory alkalosis. A superimposed marked metabolic acidosis causes neutralaemia.

Case 8: Accidental hypothermia and carbon monoxide poisoning in a 79 year-old woman

The patient had turned on the gas in her kitchen but forgot to light the burner. She was poisoned by carbon monoxide and lost consciousness and had been lying on the

floor for about 24 hours at an ambient temperature of about 23°C when found. Her body temperature had fallen to 30.4°C.

8.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.004 (7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	4.73	(4.86 - 6.10) kPa
3.	Plasma $[HCO_3^-]$ =	8.7	(22.1 - 28.2) mmol/L

8.2. Gold standard

moderate HYPOCAPNIA, MARKED ACIDAEMIA, MARKED metabolic acidosis.

8.3. ICIMS

1. Uncompensated metabolic acidosis (0.607)
2. Partially compensated metabolic acidosis (0.333)

Comment: **Agreement**. The moderate hypocapnia is a physiological response to compensate for the acidosis. However, the compensation is not enough because of the hypothermia (uncompensated metabolic acidosis).

Case 9: Coronary artery bypass grafting in a 63 year-old man

The operation was performed during cardiopulmonary bypass with haemodilution and hypothermia. The postoperative course was uncomplicated, the patient was breathing spontaneously and cardiac function was stable. Samples of arterial and mixed venous blood were drawn 24 hours after surgery.

9.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.398 (7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	5.58	(4.86 - 6.10) kPa
3.	Plasma $[HCO_3^-]$ =	25.5	(22.1 - 28.2) mmol/L

9.2. Gold standard

NORMOCAPNIA, NEUTRALAEMIA.

9.3. ICIMS

1. NORMAL BLOOD GASES (0.973)
2. Measurement error (0.0268)

Comment: **Agreement**.

Case 10: Coronary artery bypass grafting in a 47 year-old man

The operation was performed during cardiopulmonary bypass with haemodilution and hypothermia. The postoperative course was uncomplicated, the patient was breathing spontaneously and cardiac function was stable. Samples of arterial and mixed venous blood were drawn 24 hours after surgery.

10.1. Laboratory data abstract

1.	Arterial blood pH (at 37°C) =		7.368 (7.373 - 7.433)
2.	Arterial CO_2 tension (at 37°C) =	5.93	(4.86 - 6.10) kPa
3.	Plasma $[HCO_3^-]$ =	25.3	(22.1 - 28.2) mmol/L

10.2. Gold standard

NORMOCAPNIA, slight ACIDAEMIA.

10.3. ICIMS

1. NORMAL BLOOD GASES (0.949)
2. Partially compensated respiratory acidosis (0.0512)

Comment: **Agreement**. The slight acidaemia appears as of a respiratory origin due to the slightly elevated pCO_2 compared with the $[HCO_3^-]$ value.

Appendix C

High-level Patient Summary Interpretative Trend Display Data

AccNo	Date	Time	pO_2 (kPa)	Be(B)	pH	pCO_2 (kPa)	$[HCO_3^-]act$ (mmol/L)
15673	12/2/96	19:47	13.3	-6.6	7.260	6.25	20 6
15677	12/2/96	21:00	14.3	-6.5	7.275	5.95	20.3
15680	12/2/96	22:44	11.1	-6.8	7.301	5.27	19.1
15687	13/2/96	03:27	8.49	-7.4	7.305	5.00	18.2
15692	13/2/96	06:16	8.52	-7.3	7.277	5.61	19.2
15694	13/2/96	08:49	9.64	-6.1	7.290	5.74	20.2
15697	13/2/96	10:26	10.4	-6.5	7.288	5.64	19.8
15705	13/2/96	11:49	9.13	-6.2	7.300	5.50	19.8
15711	13/2/96	14:34	8.08	-7.3	7.285	5.44	19.0
15716	13/2/96	16:00	11.1	-8.7	7.262	5.37	17.8
15721	13/2/96	18:50	12.8	-7.7	7.278	5.44	18.7
15728	13/2/96	23:13	11.3	-8.6	7.234	6.04	18.7
15734	14/2/96	04:23	10.7	-8.5	7.207	6.76	19.7
15743	14/2/96	10:48	13.1	-7.9	7.273	5.46	18.5
15748	14/2/96	11:47	12.5	-8.1	7.272	5.42	18.3
15755	error	16:58	18.1	-2.7	7.204	9.36	27.1
15756	14/2/96	17:00	11.8	-8.6	7.253	5.63	18.2
15761	14/2/96	20:00	9.60	-7.7	7.266	5 69	19.0
15764	14/2/96	23:25	10.2	-7.7	7.281	5.36	18.5
15771	15/2/96	05:32	10.6	-7.9	7.306	4.81	17.6
15777	15/2/96	11:30	9.58	-7.6	7.291	5.20	18.4
15788	15/2/96	18:01	13.8	-6.5	7.354	4.38	17.9
15789	15/2/96	20:11	12.3	-8 3	7.313	4.53	16.8
15793	15/2/96	22:49	13.5	-8 6	7.341	3.93	15.6
15796	16/2/96	02:34	12.7	-7.4	7.340	4.33	17.1
15801	16/2/96	06:10	12.7	-7.4	7.351	4.17	16.9
15803	16/2/96	07:08	13.3	-8.0	7.343	4.11	16.4
15807	16/2/96	10:55	11.6	-6.5	7.368	4.14	17.5
15814	16/2/96	14:04	11.3	-6.4	7.313	5.17	19.2
15820	16/2/96	16:54	12.2	-6.4	7.334	4.77	18.6
15828	17/2/96	00:01	12.0	-6.3	7.314	5.19	19.3
15834	17/2/96	03:32	11.1	-6.2	7.317	5.15	19.3
15838	17/2/96	06:01	15.4	-6.5	7.319	5.07	18 9
15842	error	10:20	14.6	-6.5	7.315	5.11	19.1
15845	17/2/96	11:30	20.1	-6.5	7.326	4.88	18.7
15846	error	11:50	9.74		7.28	8.83	30.4
15850	17/2/96	12:50	17.8	-6.6	7.332	4.73	18.4
15857	17/2/96	16:45	10.2	-6.9	7.304	5.17	18.8
15862	17/2/96	20:00	18.0	-7.2	7.315	4.86	18.2
15865	17/2/96	22:20	8.72	-6.6	7.317	5.04	18.9
15873	18/2/96	03:37	10.5	-6.5	7.328	4.86	18.7
15876	18/2/96	05:38	10.2	-6.8	7.323	4.85	18.4
15881	18/2/96	08:47	11.0	-7.5	7.308	4.90	18.0
15889	18/2/96	13:36	13.9	-8.3	7.28	5.18	17.8
15892	18/2/96	16:50	14.0	-7.7	7.298	5.01	18
15897	18/2/96	20:50	14.0	-6.9	7.308	5.11	18.8
15903	19/2/96	00:59	11.8	-6.1	7.333	4.88	19.0
15908	19/2/96	05:23	11.7	-5.4	7.334	5.1	19.9
15916	19/2/96	08:55	12.6	-6.6	7.324	4.88	18.6
15921	19/2/96	12:10	10.8	-7.5	7.307	5.02	18.2
15927	19/2/96	15:01	16.7	-8.0	7.301	4.85	17.5
15934	19/2/96	18:07	9.18	-5.9	7.313	5.36	19.9
15940	19/2/96	19:39	13.1	-5.8	7.323	5.19	19.7
15949	19/2/96	23:04	10.4	-5.2	7.330	5.24	20.3
15958	20/2/96	03:00	9.52	-5.4	7.328	5.20	20.0
15961	20/2/96	04:30	11.0	-5.0	7.334	5.22	20.4
15965	20/2/96	09:48	11.4	-7.1	7.281	5.59	19.3
15970	20/2/96	11:44	10.9	-6.8	7.294	5.41	19.3
15977	20/2/96	15:52	9.03	-6.9	7.291	5.46	19.3

AccNo	Date	Time	pO_2 (kPa)	Be(B)	pH	pCO_2 (kPa)	$[HCO_3]act$ (mmol/L)
15983	20/2/96	17:50	11.4	-6.1	7.281	5.96	20.6
15990	20/2/96	22:33	10.9	-5.8	7.303	5.58	20.3
15994	21/2/96	01:49	9.78	-6.0	7.329	5.00	19.3
16001	21/2/96	05:49	10.4	-4.6	7.348	5.08	20.5
16004	21/2/96	08:28	10.4	-3.9	7.354	5.18	21.2
	21/2/96	11:40	9.7		7.29	6.0	21.4
	21/2/96	15:22	11.5		7.30	5.99	21.9
	21/2/96	18:00	10.09		7.27	6.56	22.2
	21/2/96	19:15	12.13		7.29	6.09	21.9
	22/2/96	00:20	11.4		7.27	5.7	19.8
	22/2/96	02:41	12.20		7.28	5.75	20.1
	22/2/96	05:27	10.96		7.32	5.50	20.9
	22/2/96	10:20	11.79		7.30	5.20	19.0
	22/2/96	11:33	13.03		7.32	5.07	19.4
	22/2/96	15:29	8.47		7.307	5.18	19.0
	22/2/96	17:49	8.91		7.348	5.16	20 8
	22/2/96	21:18	11.29		7.349	5.18	20.9
	23/2/96	00:12	9.39		7.341	5.41	21.5
	23/2/96	03:16	9.87		7.362	5.06	21.1
	23/2/96	05:52	10.65		7.35	5.16	20.9
	23/2/96	09:13	10.45		7.339	5.17	20.4
	23/2/96	18:21	12.67		7.39	4.50	20.1
	23/2/96	22:36	13.09		7.38	4.76	20.9
	24/2/96	02:31	12.61		7.34	5.22	21.0
	24/2/96	06:04	15.15		7.34	5.15	20 4
	24/2/96	09:27	14.41		7.370	4.85	20.6
	24/2/96	12:20	14.7		7.38	4.91	21.3
	24/2/96	16:56	15.76		7.36	5.05	21.3
	24/2/96	20:59	14.97		7.34	5.42	21.7
	24/2/96	23:17	14.61		7.38	5.06	22.3
	25/2/96	03:06	17.91		7.36	5.25	21.9
	25/2/96	06:49	16.29		7.36	5.23	21.9
	25/2/96	09:15	15.46		7.36	5.28	22.1
	25/2/96	11:21	13.39		7.331	5.44	21.1
	25/2/96	13:25	14.04		7.295	6.04	21.5
	25/2/96	16:14	16.93		7.31	5.27	19.5
	25/2/96	18:29	12.9		7.31	4.91	18.1
	25/2/96	21:42	13.49		7.35	4.42	18.2
	25/2/96	23:02	8.91		7.33	5.10	20.5
	26/2/96	00:48	8.85		7.32	5.25	20.5
	26/2/96	01:52	11.8		7.3	5.7	21.1
	26/2/96	04:53	12.9		7.3	5.6	20.6
	26/2/96	11:32	14.7		7.29	5.36	19.2
	26/2/96	13:59	15.25		7.31	4.97	18.3
	26/2/96	17:01	14.43		7.282	5.23	18.1
	26/2/96	19:53	16.1		7.292	5.62	19.9
	26/2/96	21:31	14.62		7.299	5.59	20.1
	27/2/96	00:45	16.69		7.32	5.29	20.3
	27/2/96	03:38	13.94		7.33	5.21	20.2
	27/2/96	07:03	16.86		7.32	5.30	20.6
	27/2/96	08:30	13.4		7.3	5.3	20
	27/2/96	10:40	9.6		7.32	5.7	20.1
	27/2/96	12:08	11.4		7.32	5.4	20.6
	27/2/96	16:15	7.33		7.31	5.61	21.0
	27/2/96	18:34	14.4		7.3	5.83	21.3
	27/2/96	20:55	12.89		7.31	5.92	21.2
	28/2/96	06:28	21.05		7.32	5.64	21.05
	28/2/96	08:57	16.62		7.37	5.18	21.9
	28/2/96	12:07	15.77		7.33	5.56	21.6
	28/2/96	17:28	15.26		7.32	4.97	18.8
	28/2/96	20:37	14.14		7.26	5.24	17.3

AccNo	Date	Time	pO_2 (kPa)	Be(B)	pH	pCO_2 (kPa)	$[HCO_3]act$ (mmol/L)
	29/2/96	02:52	13.39		7.30	5.86	21.4
	29/2/96	07:09	12.99		7.33	5.34	20.9
	29/2/96	11:35	14.6		7.30	5.4	19.8
	29/2/96	14:03	15.5		7.30	5.4	19.7
	29/2/96	15:57	14.20		7.30	5.54	20.3
	29/2/96	19:31	13.64		7.28	5.80	20.4
	29/2/96	21:09	14.69		7.32	5.60	21.4
	01/3/96	00:02	14.59		7.312	5.45	20.2
	01/3/96	02:00	14.51		7.293	5 44	19 3
	01/3/96	05:50	17.57		7.284	5.39	19.8
	01/3/96	08:15	16.78		7.27	6.09	20.8
	01/3/96	09:39	12.83		7.31	5.67	21.4
	01/3/96	12:12	12.72		7.28	6.07	21.3
	01/3/96	14:08	12.26		7.33	5.62	21.8
	01/3/96	17:11	13.1		7.26	5.99	19.8
	01/3/96	19:49	11.6		7.27	6.24	21.2
	02/3/96	00:16	15.26		7.25	6.5	21.5
	02/3/96	05:59	12.8		7.31	6.1	22.6
	02/3/96	13:12	14.57		7.29	6.07	21.7
	02/3/96	17:40	13.82		7.28	6.26	21.6
	02/3/96	21:40	11.04		7.331	5.96	23.1
	03/3/96	00:13	11.23		7.32	5.84	22.1
	03/3/96	04:00	11.4		7.3	6.04	22
	03/3/96	07:06	11.27		7.31	5.6	21.1
	03/3/96	11.34	10.35		7.29	6.03	21.1
	03/3/96	16.53	10.94		7.31	5.72	21.4
	04/3/96	00:28	9.69		7.31	5.08	19.1
	04/3/96	06:04	11.72		7.33	5.21	20.4
	04/3/96	09:22	14.08		7.305	5.36	19.6
	04/3/96	11:49	12.15		7.301	5.65	20.4
	04/3/96	15:10	11.18		7.35	4.58	18.5
	04/3/96	19:49	11.78		7.42	4.23	20.3
	04/3/96	23:45	14.9		7.4	3.9	20.1
	05/3/96	04:45	17.6		7.4	3.7	20.5
	05/3/96	06:48	17.1		7.4	3.9	21.1
	05/3/96	11:16	17.71		7.47	3.69	21
	05/3/96	14:06	10.38		7.49	4.32	24.5
	05/3/96	17:35	14.69		7.48	4.24	23.3
	05/3/96	21:18	14.3		7.46	4.4	23.4
	05/3/96	23:50	12.61		7.425	4.92	23.7
	06/3/96	03:05	15.4		7.411	4.97	23.2
	06/3/96	06:25	17.25		7.463	4.37	22.9
	06/3/96	09:21	16.14		7.46	4.26	22.2
	06/3/96	10:48	13.10		7.45	4.09	21.1
	06/3/96	12:41	11.96		7.48	4.12	22.7
	06/3/96	14.21	15.1		7.48	4.05	22.1
	06/3/96	16:18	15.54		7.454	4.06	20.9
	06/3/96	19:48	17.58		7.481	3.91	21.4
	06/3/96	21:41	18.93		7.471	3.79	20.3
	07/3/96	01:42	13.03		7.49	3.50	20
	07/3/96	06:48	13.35		7.48	3.57	19.5
	07/3/96	17:41	10.96		7.5	3.65	21.2