A FRAMEWORK FOR MULTI-DIMENSIONAL ONLINE TEMPORAL ABSTRACTION

By

Michael R. Stacey

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
AT
UNIVERSITY OF WESTERN SYDNEY
PENRITH, NSW
JUNE 2009

UNIVERSITY OF WESTERN SYDNEY

Date: **June 2009**

Author: **Michael R. Stacey**

Title: **A Framework for Multi-dimensional Online Temporal Abstraction**

School: **Computing and Mathematics**

Degree: **Ph.D.**

Permission is herewith granted to University of Western Sydney to circulate and to have copied for non-commercial purposes, at its discretion, the above title upon the request of individuals or institutions.

Signature of Author

Table of Contents

List of Tables

List of Figures

Abstract

This dissertation details a framework for providing knowledge-based temporal reasoning and data analysis within a complex and high frequency data environment. The process of Temporal Abstraction (TA) involves a transformation from raw quantitative time-stamped data to a qualitative interval-based representation which can be matched directly to the domain expert's knowledge. Clinical data analysis involves a high degree of both contextual and temporal information. Diagnoses, prognosis and therapy incorporate knowledge of factors such as patient history, present medication and temporal factors like the characteristics of a particular condition with respect to time. Temporal abstraction provides a means to instill such properties into the data analysis process.

Patient monitoring in an intensive care environment is an area where TA can be deployed to detect dangerous temporal trends or shifts in physiological data. The domain of the Neonatal Intensive Care Unit (NICU) is the context in which I demonstrate the proposals in this dissertation. The NICU data environment is both complex and multi-dimensional and represents a context where TA has not been previously applied. The process of Multidimensional Online Temporal Abstraction (MOTA) is the abstraction of physiological data across multiple data streams and data stream-sets where a single patient gives rise to a single data stream-set and data is sourced from monitoring equipment connected to the patient.

Applying TA to a multi-dimensional data environment provides a tool to support recent clinical research which proposes that aberrant behaviours across a number of physiological streams may have an additive effect and certain combinations thereof

may be a sign of critical deterioration that could indicate a high likelihood of death or of brain injury in pre-term infants.

Traditional patient monitoring equipment utilises the threshold alarming model where alerts are triggered upon the breaching of preset values. False alarms are common due to probe displacement or patient movement, temperature changes can adversely affect transducer accuracy and bad sensor positioning can result in low amplitude signals venturing outside prescribed ranges. Short lived excursions beyond thresholds often do not indicate clinically significant problems and could be dealt with by a more intelligent alarming model that incorporates a temporal dimension. Alarms can then be configured to sound if such excursions exist for longer than a pre-determined amount of time.

Correlation between multiple physiological parameters is another desirable aspect of an improved alarming model. For example, the ability to simultaneously detect a decreasing trend in blood oxygen saturation with a decreasing trend in blood pressure would enable an early warning mechanism for a potentially dangerous situation. Cross correlation also offers the potential of reducing alarms through simultaneous measurement of the same parameter via different sources.

Enabling the application of TA to data from multiple patients within the NICU also offers the potential for early detection of conditions such as Sepsis, Pneumothorax and Periventricular Leukomalacia (PVL) which have been shown to exhibit possible early warning characteristics before being diagnosable either through blood analysis in the case of sepsis, chest x-rays for Pneumothorax, or cranial sonography for PVL.

TA is a well studied topic within the field of intelligent clinical data analysis but the research carried out to support this dissertation revealed that TA systems have difficulty coping with both high frequency and multi-dimensional data environments, typical of that found in neonatal intensive care. I present a software system called the MOTA framework which enables TA within such an environment and represents a step forward in the evolution of TA-based systems. The TA mechanisms of the MOTA framework are supported by a formal model of time, the Event Calculus, which I

have adapted in order to facilitate online and incremental TA where abstractions are constructed on a sample by sample basis.

The MOTA framework has undergone evaluation using clinically defined rules describing dangerous physiological patterns for infants in the NICU. These rules were executed on data captured from babies in the NICU, the results demonstrating that MOTA can be applied within a multi-dimensional and high frequency data environment such as Neonatal Intensive Care. Furthermore, potential is offered for improved patient monitoring leading to enhanced clinical management and better patient outcomes.

Acknowledgements

I would like to thank Dr Carolyn McGregor, my supervisor, for her expert guidance which provided a clear path through the complexities which constitute PhD candidature. Her positive attitude and encouragement was always a welcome and supportive factor throughout my studies.

Kim Aubrey, my partner, has endured the trials and tribulations of being closely associated with the almost obsessive behaviour required for thesis writing. She has offered a solid and supportive backbone that I have fallen back on many times when things looked either too confusing or too complex.

My Father, Graham Stacey, has offered much in the way of encouragement and humour, of which the latter has been especially useful in assisting me to 'come back to Earth' every now and then.

My Mother, Ineke McIntosh, over the course of my life, has imparted a large dose of her drive and energy on which I drew on many occasions throughout my candidature. Her attention to detail, spirit, faith and love are heavily invested in this work and will continue to motivate me in projects to come.

This work was funded by an Australian Research Council (ARC) Linkage Project (Project #LP0561518). I would also like to acknowledge and thank the ongoing support of the staff at the NICU at Nepean Hospital, NSW Australia.

Related Publications

M. Stacey and C. McGregor, "Temporal abstraction in intelligent clinical data analysis: A survey," *Artificial Intelligence in Medicine*, vol. 39, pp. 1-24, 2007.

M. Stacey, C. McGregor, and M. Tracy, "An architecture for multi-dimensional temporal abstraction and its application to support neonatal intensive care," presented at Engineering in Medicine and Biology Society, (EMBS 2007), 2007.

C. McGregor and M. Stacey, "High frequency distributed data stream event correlation to improve neonatal clinical management," presented at 2007 inaugural international conference on Distributed Event-based Systems (DEBS07), Toronto, Ontario, Canada, 2007.

M. Stacey, "Knowledge Based Temporal Abstraction within the Neonatal Intensive Care Domain," presented at CSTE Innovation Conference, Penrith, Australia, 2005.

Chapter 1

Introduction

The last decade, in particular the latter 5 year period, has seen a worldwide explosion in the amount of data travelling between computing systems. The increasing trend of distributed computing, largely fueled by the Internet and developing ubiquitous protocols has resulted in heavier volumes of data traffic travelling at escalating speeds across the globe. Data streams are now an inherent component in environments such as: financial markets [1]; network monitoring; telecommunications data management; web applications that monitor site traffic; military battlefield positional systems [2] and medical monitoring environments [3, 4]. Applications that can process streaming data in real-time are becoming a requirement where continuous assessment and analysis of incoming data items is expected.

Within medicine, the environment of the Intensive Care Unit (ICU) has also experienced expansion in terms of both the frequency and volume of data collected from patients. Today's ICUs generate vast quantities of data. Increasingly sophisticated monitoring equipment performs high frequency measurement of multiple physiological parameters that require timely and context sensitive analysis in order to sustain effective decision support. The need for decision support in real-time medical environments is high; clinicians find it problematic to make accurate decisions when overwhelmed with multiple signals and, according to Miller [5], humans have great difficulty developing accurate responses to problems involving more than seven variables. This problem is compounded when the decisions to be made are potentially

life threatening and must be arrived at within the constraints of a real-time situation typical of an ICU.

There is a requirement for Intelligent Decision Support Systems (IDSS), within the field of patient monitoring, that can intelligently analyse multiple data streams to detect adverse medical conditions in real-time. Recent medical research has discovered that conditions such as Sepsis [6] and Periventricular Leukomalacia (PVL) [7], have been shown to exhibit early indicators in physiological data. Such IDSS would provide clinicians with timely information that is impossible for a human to identify, especially considering the escalating trend of data density and frequency within ICU environments.

The process of *Temporal Abstraction* (TA) is a key concept of this research and a component of Intelligent Data Analysis (IDA) [8]. It has been utilised widely within clinical data analysis and clinical decision support systems to provide a transformation in data representation from unrefined numeric form to a higher level qualitative form incorporating the temporal dimension. This allows the "matching" of patient data with medical knowledge [9] to provide clinical decision support and aid clinical management.

This dissertation proposes a framework for online temporal abstraction of data streams within a multi-dimensional data environment. I call this multi-dimensional online temporal abstraction (MOTA). The work is demonstrated in the case-study area of Neonatal Intensive Care where data arrives at high sample frequencies from monitoring equipment connected via sensitive transducers to the patient. The challenge is not only to recognise clinically relevant scenarios as they happen, in real time, but also to apply this analysis process across a multi-dimensional data environment where there exists multiple data sources (patients) each consisting of multiple data parameters. Before introducing the thesis research case in section 1.3, I provide definitions of the key concepts this thesis covers. Section 1.1 defines the process of *Temporal Data Abstraction* and section 1.2 defines *Multi-dimensional temporal data abstraction* in the context of the case-study environment.

1.1 Temporal Data Abstraction

The first decision made by an analyst when confronted with a data analysis problem is how to best represent the data in order to enable the extraction of essential information from it. For example, repeating sinusoidal type patterns are easily represented using Discrete Fourier Transforms (DFTs) or Wavelets because the original data is best described in the frequency domain. A set of harmonics describing the frequency components of the signal can provide a window into the makeup of the signal and allow either further analysis or inferential processes to follow. Many traditional methods from the areas of mathematics, theoretical physics or econometrics assume the process under consideration is either deterministic or stochastic in nature. The theory of dynamical systems [10], typically employed in physics, utilises methods for modelling deterministic processes while system theory [11] and digital signal processing techniques are applied within engineering. All of these methods deal with completely deterministic processes where events proceed in a fixed direction. In contrast, in the context of tracking physiological data in order to detect disease states or other anomalous clinical conditions, physiological time series data rarely exhibit such predictable characteristics.

Statistical time-series approaches have been investigated [12–14] with regard to the demands of intensive care medicine, where data dimensionality is extremely high, but the difficulty of model development requires a high level of statistical understanding and models are difficult to 'tune' to account for the large variation in individual patient characteristics. The use of analytical models that fit raw data requires knowledge of the underlying physical process and, as Dojat [15] explains, this is rarely the case in medicine. Additionally, the computational effort required for techniques involving numerical data analysis is often considerable when faced with the constraints of real-time processing.

In the medical domain, analysis of patient data generally follows a less quantitative approach. For example, features apparent in Electrocardiogram (ECG) waveforms

are often described using informal descriptions depicting the relative 'sharpness' or 'smoothness' of curves and 'straightness' of sections between them [16]. Encoding the clinical knowledge that connects such observations with the health status of the patient into numerical methods is not a viable solution. The limitations of purely mathematical approaches prompts the introduction of knowledge-based techniques which can elevate data to higher abstraction levels allowing domain knowledge to be directly integrated into analysis processes. *Data abstraction* provides such a method.

Abstraction is an increasingly contemporary idea within the computing world. It is a widely used concept where high level views of complex, low level detail is the desirable outcome. In the early days of computing, computer users submitted their 'jobs' to a computer expert for execution on the machine and usage of the computer was restricted to those who possessed a greater degree of knowledge of computers. Nowadays, almost anyone can use a computing machine. The low level detail and complexity has been hidden behind multiple abstraction layers. When it is not strictly necessary to understand the primitive constructs or detail of low level functionality, then abstraction can provide a context sensitive summary which is closer to the world of the user. For example, a computer user does not require knowledge of how the computing machine opens and closes files or allocates memory for their word processing documents. Even more experienced computer users, such as programmers, employ the concept of Abstract Data Types (ADTs) and Object Orientation (OO) to provide 'black box' views of re-usable software components allowing underlying detail to be hidden from the programmer's view. *Data* abstraction can be used to provide context-sensitive summaries of complex data and is especially useful in the field of medicine for providing a data analysis technique which can lead to diagnostic, prognostic or therapeutical decision support.

Medical knowledge is stored in associations of rules, treatment protocols, guidelines and of course in the mind of the professional health care providers where, on the other hand, physiological data collected from a patient consists of various raw

time-stamped numeric measurements. To successfully match patient data with medical knowledge requires a representation of data which elevates it from quantitative to qualitative, as provided by data abstraction.

Reasoning processes in medicine involve time as an integral and essential element. Diagnoses often involve careful assessment of the temporal order of symptoms, prognosis includes prediction of the development of conditions based on periods of time and therapeutical decisions incorporate a time-based ordering of treatment events [17]. Systems which operate on clinical data are required to provide either an implicit or explicit model of time in order to allow accurate diagnostic, prognostic and therapeutical decision support. The process of data abstraction which includes such temporal reasoning is known as *temporal data abstraction*.

Temporal Abstraction (TA) takes either raw or pre-processed data as input and produces context sensitive and qualitative interval based representations. This is required in order for symbolic reasoning strategies to proceed and can be seen as a first point of entry for knowledge into the data analysis system. TAs can then be interpreted by matching against pre-defined templates or guidelines [18, 19], or reasoned with in a higher context [20]. More specifically, Shahar [21], defines the temporal abstraction task as, "a process which, given a set of time-stamped parameters, external events and abstraction goals, produces abstractions of the data that interpret past and present states and trends and that are relevant for the given set of goals."

I provide an example taken from the case-study context of neonatal intensive care: given a hypothetical newborn baby born 16 weeks premature, a fall in mean blood pressure less than 24 mm Hg is clinically relevant. At all gestations, a fall in peripheral oxygen saturation less than 85% for greater than 20 seconds is also clinically relevant [22]. Summarising this domain knowledge we have the following:

> Detect periods of time of 20 seconds or more when mean blood pressure is less than 24 millimeters of mercury (mm/Hg) and oxygen saturation is less than 85%.

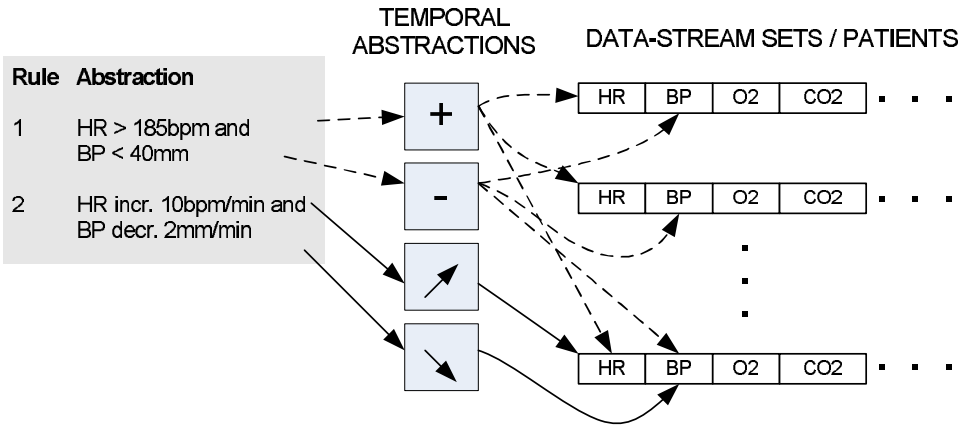

Figure 1.1: Multi-dimensional online temporal abstraction (MOTA). Multiple patterns, abstractions, parameters and data streams/sets. Each data stream-set represents data from a single patient. Temporal abstraction rule 1 requires both positive and negative level shift abstractions of heart rate and blood pressure parameters on all data stream-sets. Temporal abstraction 2 requires increasing and decreasing trend abstractions on a single stream-set. Note: temporal aspects are not included in this figure

The above represents a qualitative input to a TA process responsible for the detection of such 20 second time intervals. Generally, the result would be a set of time-stamped data points comprising the relevant interval or, in some cases, simply the start and end time points are returned. The exact TA process output is of course flexible and depends on the particular implementation requirements but, in all cases, a context sensitive interval-based abstraction is created which summarises what can be overwhelmingly dense time-stamped data.

As revealed by the literature survey, TA has been widely applied to stored singular data-sets or streams but has not been extended to cope with situations where there are multiple abstractions to be performed on multiple parameters within multiple high-frequency data streams in real-time. Applying TA to such a multi-dimensional environment is one of the open research areas addressed by this thesis and represents a logical move forward for IDA frameworks [23].

1.2 Multi-dimensional Online Temporal Abstraction

The inherent complexity of the data environment in areas such as the Neonatal Intensive Care Unit (NICU) gives rise to a number of levels of intricacy as shown in Figure 1.1. *Multi-dimensional online temporal abstraction* refers to the process of applying TA within this complex environment.

Two abstraction rules describing a total of four abstractions are shown in Figure 1.1. Rule 1 describes a TA where it is required to detect a shift in heart rate (HR) over 185 beats per minute (bpm) occurring at the same time as a decrease in blood pressure (BP) to below 40 millimetres of mercury (mm/Hg). Rule 2 describes a trend TA where heart rate increases at a rate of 10 bpm/per minute and blood pressure decreases at a rate of 2mm/Hg/per minute. Each data stream-set represents the physiological data load from a single patient. A single data stream is defined as a single data parameter, such as heart rate. It should be noted that time values for relevant abstraction intervals would normally be defined and these have been omitted to facilitate clarity.

Within this environment I have defined two levels of complexity and dimensionality as follows:

1. *Level One Dimensionality*: The temporal abstraction of data emanating from a *single* patient. TA Rule 2 in Figure 1.1 provides an example of such a situation. Whilst the example implies a single TA on the data stream-set, there could be multiple TAs on the same or different parameters.

2. *Level Two Dimensionality*: TA Rule 1 is an example of the temporal abstraction of data emanating from *multiple* patients. The context includes complex inter-relationships between multiple TAs, patient data streams (parameters) and data-stream sets. Intensivists may be interested in assessing a number of abstractions concurrently across multiple patients, especially for the purpose of

early detection of dangerous clinical conditions such as Sepsis [6] and Periventricular Leukomalacia (PVL) [7]. The tracking of disease spread within the NICU is a further area requiring a system capable of data analysis across multiple patients and parameters. My investigations have not revealed any systems which can perform TA at this level of dimensionality in a streaming environment where sample frequencies are high (in the order of 1Hz and above) [23].

1.3 Research Case

1.3.1 Research Motivation and Contribution

This research is driven by a number of motivating factors which stem from the disciplines of *Health Informatics* and *Computer Science*. The following sections provide detail of these motivating influences categorised accordingly.

1.3.1.1 Health Informatics

From a total of 90,608 babies born in NSW during 2005, 14% (12,320) required admission to a special care ward and 2.5% (2,257) were admitted to neonatal intensive care units (NICUs). These figures are similar to previous years [24, 25]. A high proportion of babies admitted to NICUs are very premature and suffer serious illness. Of those babies, 7.7% (174) will die and 15% will have life long disabilities including cerebral palsy, mental retardation, blindness and deafness. During their stay in the NICU, these babies undergo numerous medical diagnoses and treatments, many of which have long term repercussions for the individual [26]. The following points deal with aspects directly related to the health of newborn infants in a NICU environment.

- **Continuous Patient Monitoring**

  Neonates in intensive care environments are connected to numerous monitoring devices that generate large quantities of data at high frequencies. Devices include ventilators, dedicated pulse oxymetry monitors, dedicated pulmonary mechanics monitor and arterial blood gas and lactate measurements [27]. Physiological data from these machines is displayed on bedside monitors that are

monitored by nursing staff who hand record relevant measurements in 30 to 60 minute intervals. It is not uncommon for babies to exhibit substantial variations in vital signs between the recording times. These fluctuations go unnoticed. This research proposes an architecture for real-time monitoring that can detect such abnormalities to greatly enhance health care.

- **Intelligent Alarming**

  Physiological data sourced from patients in the NICU are subject to simple threshold alarm techniques that indicate an anomalous excursion of a single parameter beyond a predefined level. Threshold alarming models have been shown to produce a high incidence of false alarms where up to 86% of all alarms in Intensive Care Units (ICUs) may be false [28]. Numerous reasons exist for the high rate of false alarms including sensor repositioning, poor sensor-skin contact, bad data format or therapy modification (drug or ventilation) [29].

  A more desirable monitoring situation incorporates the *intelligent* alarming model [19, 28, 30, 31] where the temporal course of data is tracked (rather than in a transient fashion) and multiple parameters are analysed with respect to their relationships with one another. Commonly, medical knowledge pertaining to adverse states is described in terms of an increase, decrease and either the stability or instability of a parameter with respect to time, rather than the instantaneous violation of a threshold [32]. Alarming mechanisms that demonstrate an ability for temporal reasoning and abstraction, such as those proposed by this thesis, carry the potential to fulfil such requirements.

- **Early Detection of Disease**

  Babies admitted to the NICU are extremely fragile and susceptible to many complications during their stay in the ward. Some of these conditions have been shown to exhibit early physiological signs of the impending situation prompting the requirement for patient monitoring which can detect these signs and enable either pre-emptive medication and treatment or simply increased attention from

clinical staff. The multi-dimensional nature of the proposed architecture allows temporal assessment of data emanating from multiple patients within a ward and offers the potential of early detection of conditions such as Sepsis, Pneumothorax and Periventricular Leukomalacia (PVL), which may exhibit early warning characteristics before being diagnosable either through blood analysis in the case of sepsis [6], chest x-rays for pneumothorax [33] or cranial sonography for PVL [7]. This also provides for the tracking of disease outbreaks within a NICU ward as each condition can be monitored for across several patients.

1.3.1.2 Computer Science

- **Multi-dimensional Temporal Abstraction**

  The dimensionality of TA-based IDA frameworks is limited. TA has been widely applied to singular datasets or streams but has not been extended to cope with situations where there are multiple abstractions to be performed on multiple parameters within multiple high-frequency data streams. It is evident from a recent survey paper [23] and further literature review associated with this thesis, that for IDA systems to successfully move forward into the future where clinical environments are becoming increasingly data-intensive, the ability for managing multi-dimensional aspects of data at high observation and sample frequencies must be provided. Continued evolution of such systems involves expansion into larger computing architectures which offer the necessary degree of robustness, control and co-ordination of TA processes required for such data driven worlds. The framework proposed by this thesis offers multi-dimensional online temporal abstraction (MOTA) through distributed processing of TA-based processes and a central domain registry for accumulation of results, and a temporal model for real-time reasoning on incoming data.

- **Temporal Reasoning in Real-time using the Event Calculus**

  The process of temporal abstraction requires a model of time for the support of temporal reasoning facilities. Where temporal abstraction is the primary

outcome, the Event Calculus (EC) provides a *natural* method for modeling time. To date, there are no known existing methods for true real-time execution of the EC for the purpose of creating temporal abstractions. Additionally, temporal interpolation, a necessary component of abstraction, has not been tackled through sole use of EC axioms.

The EC temporal model provided by this thesis enables the specification of a *maximum allowable gap* within an abstraction which supports a TA specification such as the following:

> "Detect periods of time of 10 minutes or more where heart rate is over 185 beats per minute. Heart rate can fall below 185 beats for no more than a total of one minute within the ten minute window".

In this example, one minute is the *maximum allowable gap* and may be the sum of multiple smaller such gaps. Development of axioms which allow EC to model the passing of real time enables temporal interpolation, or concatenation, between contextually similar intervals either side of the gaps. The temporal model also enables true sample by sample reasoning and abstraction of online data.

- **Data-mining and data abstraction**

  The two principle techniques employed in IDA, have developed independently of each other and remain somewhat separate processes. Advances in the usefulness and applicability of IDA can be gained by uniting these concepts [9]. The proposed system will support the definition of temporal abstraction rules by allowing an interface to an Analytical Processor [34] which mines physiological data to reveal new associations between data parameters.

1.3.2 Limitations of Previous Research

A number of limitations have come to light regarding intelligent data analysis systems that employ temporal abstraction mechanisms. These issues will be elucidated in the

literature review and were highlighted in a comprehensive survey paper published in 2007 [23]. Here, I briefly outline the major areas I believe to be crucial for TA-based IDA systems to move successfully into the future where data environments are becoming increasingly complex in terms of the amount of raw data accrued, the rate at which it is acquired and the dimensionality of the data environment itself; as defined in section 1.2.

1. It has become apparent that the dimensionality of TA-based IDA frameworks is limited. The dimensionality of the data environment in which TA mechanisms will be applied has a large influence on the design of the system. Compared with the abstractions required for the environment defined in section 1.2, it is not sufficient to assume the same methods will be adequate. The nature of intensive care domains places constraints on the time taken to reach a diagnosis; this in addition to the computational resources required for temporal reasoning and abstraction within a multi-dimensional and high frequency data environment creates a significantly challenging problem. Only three TA-based systems were found to be able to deal with multiple data-sets [20, 35–44], and only one of these [38] performed TA on streaming data; and then it was only on 2 channels of ECG data. Of all the systems that handled streaming data input [16, 18, 38, 39, 42, 45–56], most were able to apply multiple TAs to multiple data parameters but this occurred within a singular patient data stream-set only.

2. This research employs the Event Calculus for performing temporal reasoning tasks associated with temporal abstraction. The Event Calculus (EC) has been used previously for TA within the area of Health and Medicine [51, 57, 58]. To my knowledge, there have been no implementations of EC capable of abstracting data in real time, sample by sample. EC has no concept of real time hence requires extensions in order to enable real time reasoning. A complete mechanism for TA should also incorporate levels of reasoning capable of temporal interpolation [59]. There are no known axiomatisations of the EC that include this ability in a real-time model.

3. The inherent complexity of the application environment introduces the requirement for a high level, semantic representation capable of specifying and defining relationships between domain entities involved in the temporal abstraction process. These entities are patients, TA rules, TAs, patient data streams (parameters) and data stream-sets; see section 1.2). This is provided for by the Active Domain Registry (ADR), which generates the Event Correlation Specification (ECS). The ECS enables automatic generation of EC-based rule scripts for dissemination to multiple patient monitor modules (PMMs), which execute them on patient data streams. Once TA rules have terminated, results are accumulated and correlated by the ADR to facilitate intelligent and automated alerting. There has been much work on ontologies and similar data structures for formally describing the terms, knowledge and relationships within a domain but none that provides a multi-dimensional method of modeling temporal rules which are to be executed on multiple data streams. Tools have been developed for automatic EC axiom generation [58], but not within a multi-dimensional and distributed environment such as the case-study related to this work.

4. There is a lack of seamless integration between knowledge obtained though data mining processes and data abstraction techniques. Storage of medical data using data warehousing allows the exploration of patient data for possible knowledge, in the form of associations and trends, which may act as early warning signs for impending disease, as in the case of sepsis. To be able to fully exploit this ability, the integration of data mining processes performing clinical research into IDA systems becomes a necessity. Providing this bridge between clinical research and clinical management connects the two, generally independent IDA processes of data mining and data abstraction which, in the year 2000, were reported as having progressed independently of one another [8]; this trend has continued over the last 5 years to the present day.

The continued evolution of Intelligent Data Analysis (IDA) systems within increasingly data-driven worlds is imperative for successful management, analysis and

decision support. The very reason IDA came into being was to provide intelligent analysis of medical data where other techniques fail. A number of authors [8, 9, 60] have expressed concern regarding the widening gap between data gathering and storage and methods needed to interpret data. The ability for managing both multi-dimensional aspects and high sample rates is one important facet which must be catered for by such systems. The research outlined by this thesis, represents a first step in that direction.

1.3.3 Research Hypotheses

The following thesis hypotheses address the afore-mentioned research limitations. This thesis proposes that:

1. A framework can be defined to enable temporal abstraction within a multi-dimensional data environment in real time. This framework is named the Multi-dimensional Online Temporal Abstraction (MOTA) framework .

2. A method to semantically *represent* the complex multi-dimensional relationships between data streams, parameters and TAs can be defined to support TA within such complex environments. This method is catered for by both the Event Correlation Specification (ECS) and the Active Domain Registry (ADR).

3. The Event Calculus can be extended and applied to cater for temporal interpolation and abstraction of real time data.

4. The framework can support the definition of TA rules by allowing an interface to an analytical processor which mines physiological data to reveal new associations between data parameters thus providing a bridge between data *mining* and data *abstraction* processes.

5. Proposals 1 to 4 can be applied within a Health and Medicine context to demonstrate the potential for its use to enrich Clinical Management.

1.3.4 Research Contributions

Here I outline the research contributions made by this thesis by addressing the five research hypotheses from section 1.3.3.

1. The MOTA framework is a distributed architecture providing a backbone on which to address the multiple independent computations to be executed on multiple patient data stream-sets. *Patient Monitor Modules (PMMs)* are responsible for abstraction of an individual patient's data where rule decomposition techniques aid separate execution of conjunctive parts of TA rules. PMMs utilise an extended version of the EC allowing true real-time processing of data values, sample by sample. The incorporation of a central and *active* domain registry (ADR), which includes a Medical Alert Monitor (MAM), facilitates intelligent alarming across multiple patient data stream-sets based on the ADR's accumulation of results from the PMMs.

2. The Active Domain Registry provides a central semantic core of instantiated domain entities which is a run-time realisation of the Event Correlation Specification (ECS). The ECS provides for permanent storage of all TA rules entered into the system and, through its schema, also describes the *relationships* between the entities which constitute TA rules. These relationships underpin and define the multi-dimensional online temporal abstraction (MOTA) required for rule execution.

3. The thesis describes a method for TA, including a temporal interpolation mechanism, through the definition of a temporal model which is based on the Event Calculus. The model has been implemented through a forward chaining rule engine which is embedded into each patient monitor module. Testing and evaluation carried out in this research has shown that patient monitor modules are capable of a 1 second processing/reasoning cycle, meaning they can process and abstract data which is sampled at a rate of 1Hz.

The EC temporal model provided by this thesis enables the specification of a 'maximum allowable gap' within an abstraction which supports a TA specification that includes a temporal gap, as outlined in section 1.3.1.2.

4. Data mining processes are being developed within an *Analytical Processor* [34] to reveal possible alert rules which can drive MOTA framework abstraction processes. As previously mentioned, these rules will offer the potential for early detection of dangerous clinical conditions. An interface to the Analytical Processor, which includes the ECS schema for domain entities, allows The MOTA framework to receive rules generated from mining processes and utilise these in the same way as rules entered by clinical staff.

5. Through ongoing collaboration with the NICU at Nepean Hospital in Western Sydney, the research described here was tested and verified within the domain of Health and Medicine to demonstrate the potential for its use to enrich Clinical Management

1.3.5 Research Method

This research employed a constructive research approach. Constructive research is a stream of interventionist research and includes other notable approaches such as action research, design science and clinical research [61]. The constructive approach was first developed by Finnish researchers Kasanen, Lukka and Jonsson [62–64]. Jonsson et. al. [61] describes this method as follows:

> Through strong intervention, the researcher, jointly with members of the target organisation, develops a new *construction*, tests its usability, and draws theoretical conclusions based on this process.

A *construction* can be considered any human artefact such as models, diagrams, plans, organizational structures, commercial products or information systems architectures. To qualify as constructive research the study must combine problem solving and theoretical knowledge, as illustrated in Figure 1.2.

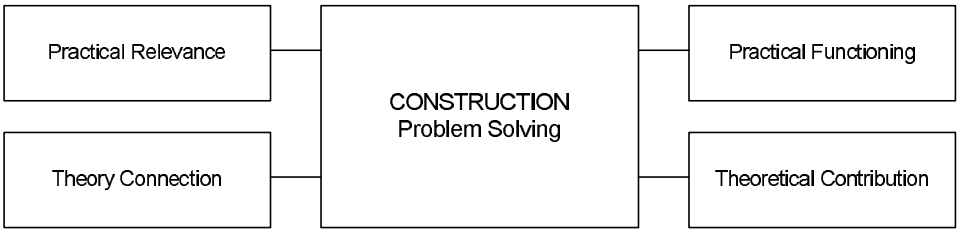

Figure 1.2: Elements of Constructive Research illustrating the combination of problem solving and theoretical knowledge. Adapted from Kasanen et. al. [62]

Kasanen et. al. propose the following phases of the constructive approach which have been augmented with aspects the research involved in this thesis:

1. *Find a practically relevant problem.* The challenge of building IDA systems that are capable of handling multi-dimensional and data-intensive environments is a current area of open research and indicates a way forward for such systems. Deficiencies in former systems have been clearly identified in [23] and these provide much motivation for this research effort.

2. *Obtain a general and comprehensive understanding of the topic.* An extensive literature review has been conducted and a survey of existing IDA systems, within the domain of medicine, has been published in the journal *Artificial Intelligence in Medicine* [23]. Consultation with clinical staff at the Nepean Hospital NICU has also provided invaluable knowledge and awareness regarding the case-study environment.

3. *Innovate. Construct a solution idea.* The design and construction of framework for MOTA, as proposed in this thesis, will provide a platform for future IDA systems that are required to function in multi-dimensional and online data environments.

4. *Demonstrate that the solution works.* The proposed framework is evaluated

against the hypotheses presented in section 1.3.3. During the course of the research it has been necessary to perform a combination of quantitative and qualitative assessment of the theories developed and artifacts constructed. Laboratory testing and experimentation to ascertain the practicality and performance of intended methods has been combined with ongoing discussion with neonatologists at Nepean Hospital NICU.

5. *Show the theoretical connections and the research contribution of the solution concept.* Extensive literature surveying has, and is, demonstrating the connections between solution concepts and the disciplines of computer science and health informatics. The research contribution will be demonstrated within the domain of Neonatal Intensive Care.

6. *Examine the scope of applicability of the solution.* The solution holds the possibility of being extended to other domains. Wherever intelligent data analysis is relevant, the solution will acquire an application domain. The literature survey shows a prevalence of IDA methods within Medicine, primarily due to the qualitative nature of diagnosis, prognosis and therapy, and hence data analysis. However, there has been application to other domains such as gas turbine analysis and meteorology [42, 43, 54], and traffic control analysis [65].

1.4 Thesis Application Context within the e-Baby Architecture

As the MOTA framework has been developed as a component within a larger architecture, which is defined by the e-Baby Project [4, 26, 27, 66, 67], it is necessary to describe the precise context in which MOTA is deployed.

The e-Baby research project provides a distributed architecture where physiological data captured from bedside monitoring equipment in the environment of neonatal intensive care, is integrated and analysed by a central intelligent decision support

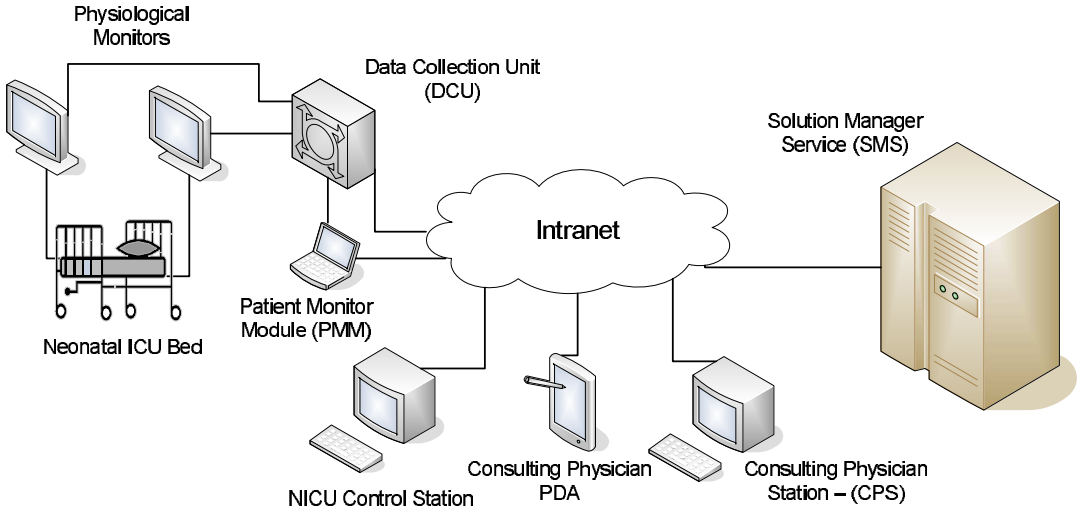

Figure 1.3: e-Baby Architecture

system; the Solution Manager Service (SMS). Figure 1.3 depicts the e-Baby architectural components relevant to this research. The SMS was originally defined within the environment of business performance monitoring [68, 69] for capturing business process information. It allowed authorised parties to monitor the enactment of business processes to ensure performance measures are met. The e-Baby project re-defines the SMS in the domain of Health and Medicine, specifically neonatal intensive care, for the purpose of supporting neonatal clinical management and research. Its primary function, resulting from the first iteration of its application to health and medicine prior to the research detailed in this dissertation, was to capture and integrate physiological data from a variety of disparate sources and provide permanent storage in a data management layer incorporating a near real-time data store and data warehouse. Data mining facilities are also being developed which will provide clinical trial analysis at patient or summary levels from stored data [67].

The Data Collection Unit (DCU) is a small footprint machine located in close proximity to the NICU Bed or humidicrib and supports the collection of data from monitors associated with 2 beds/humidicribs. It packages and transmits physiological data to the SMS. The NICU control station is located in a separate sever room and enables initiation and termination of data collection services and also control over

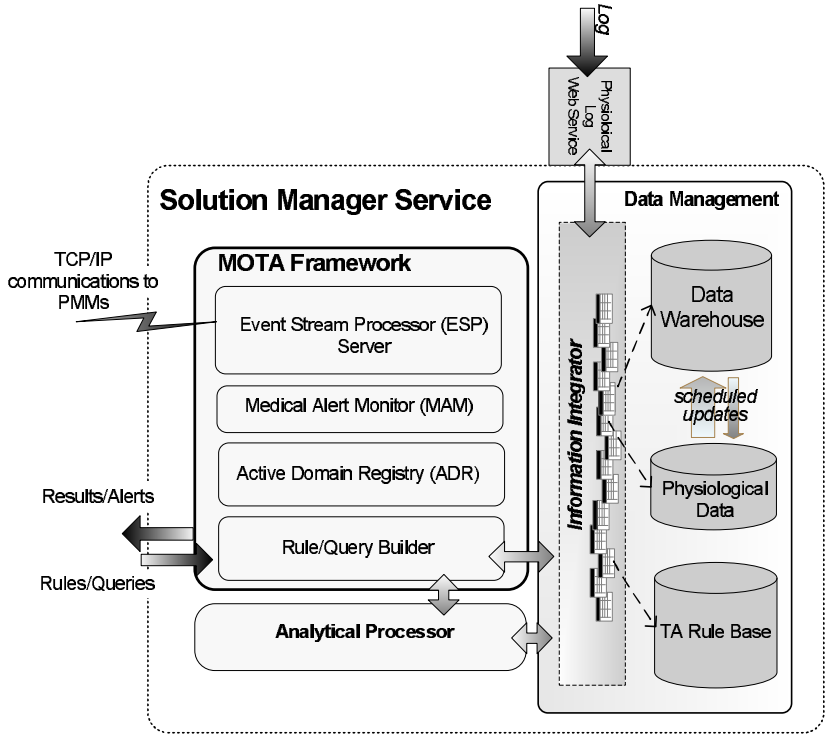

Figure 1.4: The extended Solution Manager Service (SMS) showing the Multi-dimensional Online Temporal Abstraction (MOTA) Framework and associated components.

the size of data packets that the DCUs transmit. The Consulting Physician Station (CPS) is utilised by neonatologists to access physiological data located within the SMS. The architecture is web service-based in that the DCU, CPS, SMS, and NICU control station all have well defined web service interfaces and physiological data is formatted by the DCUs into XML packets before being sent to the SMS [4, 66].

The extended SMS as proposed by this research is illustrated in Figure 1.4 showing the MOTA framework including Event Stream Processor (ESP) Server, Medical Alert Monitor (MAM), Active Domain Registry (ADR), Rule and Query Builder and TA Rule Base components. As mentioned previously, the Analytical Processor provides an interface to patient data residing in the data warehouse and performs temporal data mining tasks searching for associations between physiological parameters which may be used to aid clinical research.

In order to integrate the MOTA framework into the existing e-Baby architecture

depicted in figure 1.3 and enable online monitoring of physiological data, a Patient Monitor Module (PMM) is located alongside each DCU. This is a MOTA framework component, not part of the original e-Baby infrastructure. Each PMM performs real-time temporal abstraction of the data streams from two patients and communicates with the ESP Server via a TCP/IP link. Temporal abstractions defined by TA rules are entered via the Rule and Query Builder interface and after being processed, are distributed to the relevant PMM/s where execution of rules and abstraction of patient data proceeds. After rule termination, results are returned and integrated into the ADR and MAM for the purpose of medical alerting. Querying facilities are also offered via the Rule and Query Builder interface offering such information as rule status, TA interval start and end times and alert information regarding particular patients and rules.

Originally, the SMS defined a set of web services to enable interaction and configuration of its internal components. The web service which is relevant to the MOTA framework is the *Log* web service. Its function is to allow transmission of XML-encoded packets between DCUs and the SMS Data Management Layer. Physiological data captured from the monitors is packaged based on a given time interval and transmitted to the Data Management Layer for storage. This area of functionality has been demonstrated via a pilot study using data captured from a Dragar Ventilator and a Philips Component Monitoring System (CMS). The Data Management Layer is implemented in SQL Server and data is loaded into the real-time data store and data warehouse using the SQL Server bulk load service [67].

1.4.1 The MOTA Framework as an Extension of the SMS

The original SMS was concerned with the execution and monitoring of business processes and utilised mechanisms for detection and correlation of real-time events, such as the start and end times of transactions. In this use case, the notion of an event is one of instantaneous change, a transition from one state to another and as such, there exists no duration for events; rather they happen at a particular time instant. The

MOTA framework also involves detection of significant events, although the events it is concerned with have a duration and extend over some interval of time. This is a temporal abstraction, a *temporal abstraction event*, or *extended event*, if you like. The capture, processing and integration of significant events is provided through the Event Stream Processor (ESP) which extends the previous SMS event monitoring implementation by providing a formal model of time which underpins the processes it executes on data. The temporal model is an extension of the Event Calculus to enable real-time temporal abstraction of streaming time-stamped data.

The initial SMS business-based implementation included a *Performance Monitor* component enabling "organisations to define service-level agreements based on selected performance measures" [69]. Essentially, it tracked service-level agreements, including transactional information, so that notification could be provided to concerned parties when an agreement had been violated. The MOTA framework employs a *Medical Alert Monitor* component which acts in a similar fashion. In partnership with the ADR, it monitors the state of rules that have been entered by clinicians and provides automated alerting when those rules, which define clinically relevant conditions, return *true*.

1.5 Thesis Overview

An architecture for multi-dimensional temporal abstraction (MOTA) of real-time data, named the Multi-dimensional Online Temporal Abstraction (MOTA) framework, is presented in the following chapters of this thesis.

1.5.1 Thesis Structure

I begin in Chapter 2 by reviewing the literature emanating from fields of research that are associated with the development of the afore-mentioned framework. A broad range of research topics were covered, including temporal reasoning, temporal abstraction, time-series analysis, expert systems, decision support systems, real-time clinical and

process monitoring, data stream processing and existing temporal abstractions systems. A survey of existing TA-based systems was carried out which highlighted the areas requiring further development if such systems are to evolve into the future where data and monitoring environments are becoming increasingly complex.

Chapter 3 details the nature of Neonatal Intensive Care Unit (NICU) which forms the case-study domain for the dissertation. I begin by revealing statistical information regarding births in NSW and ACT which serves to illuminate the importance of neonatal intensive care. Typical care protocols and guidelines for the first hour of life are provided which then leads into patient monitoring; which is where this thesis finds its application context. The complexity of the data and monitoring environments are discussed with relevance to their multi-dimensional nature and current monitoring methods are examined revealing areas which this research proposes to improve through the application of hypotheses 1 to 4.

Temporal reasoning is covered in chapter 4, specifically the EC, which I use as the formal logic underpinning the creation of temporal abstractions from online time-stamped data. In order to permit real-time reasoning, Hypothesis 3 proposes adaptions that include a method for counting the passing of time itself. These variations to the EC are discussed along with the axioms defining the temporal model.

Thesis hypothesis 1 proposes a framework can be defined for performing Multi-dimensional Online Temporal Abstraction (MOTA); this is presented in Chapter 5. The MOTA framework is a distributed framework where processing is dispensed amongst multiple patient monitor modules, each responsible for the abstraction of data emanating from a single patient. Synthesis of results returned from patient monitor modules enables multi-dimensional TA which is performed by the Active Domain Registry (ADR). Patient monitor modules utilise the Event Calculus (EC) as their method for temporal reasoning on patient data and execute EC-based rule scripts which are translated by a *Transformer* service from the original TA rule entered by a clinician.

The implementation and evaluation of artifacts which have been created in order to support and test the theories proposed by the thesis hypotheses 1 to 4 are discussed in Chapter 6. I provide an evaluation procedure where the MOTA framework is tested in its ability to perform MOTA, both at the single patient and multiple patient levels. Clinically defined rules were used to define abstractions and a total of 17 patient datasets were made available from the Nepean Hospital NICU on which to execute the rules. Limitations of the proposed methods were brought to light and are discussed along with suggested solutions although the initial results are encouraging regarding the validation of the proposed ideas for MOTA.

A conclusion and discussion of the thesis is provided in Chapter 7. The significance of the work, in terms of its contribution to the fields of computer science and health informatics, and pointers for future research are offered along with the limitations of the proposed prototype framework. I also address the thesis hypotheses and discuss how these have been achieved.

Chapter 2

Temporal Abstraction within Intelligent Clinical Data Analysis

This chapter reviews major contributions made in the development of Intelligent Data Analysis (IDA) systems [8] utilised within medicine, that incorporate temporal abstraction mechanisms. Information gathered enables a clear view of what has been achieved to date, where current research is heading and challenges for future research. The chapter highlights aspects that are either poorly addressed or non-existent in previous research efforts and those I believe will assume a greater level of importance for future IDA systems. These aspects were summarised in section 1.3.2. My aim is the development of a framework which will enable detection of *multiple* temporal patterns across *multiple* high-frequency patient data streams for *multiple* patients within the domain of Neonatal Intensive Care. This encompasses the five hypotheses stated in section 1.3.3 of the previous chapter and extends the process of temporal abstraction into a streaming multi-dimensional data environment, as defined in section 1.2. The system forms a vital component within the e-Baby architecture that is under development as part of the e-Baby collaborative research project [4, 26, 66, 67, 70, 71].

Data abstraction has been an integral part of knowledge-based systems in medicine since the early 1980s when Clancey proposed his method of heuristic classification [72], which utilised abstraction as a critical part of the reasoning process. More recently, data abstraction methods have been primarily concerned with the temporal dimension of data, hence the term *temporal abstraction*. Seen in the evolution of knowledge-based

systems, temporal abstraction is a crucial process which endeavours to bridge the gap between raw data and high level domain dependent descriptions of the data. This practice became especially pertinent during the 1990s, which were characterised by the increasing gap between voluminous quantities of data and the understanding of the data requiring new data analysis techniques. It is not surprising that most of the research reviewed in this chapter was developed in the nineties.

This period led to an explosion in challenging research areas such as data mining and intelligent data analysis [8], with an emphasis on machine learning techniques aimed at extracting knowledge from large databases. Temporal abstraction played a role in this active phase of development in a number of ways. In a data mining sense it can be seen to provide a pre-processing phase where data is 'summarised' according to domain knowledge and then further processed using machine learning methods or data mining processes. Temporal abstraction (TA) has also been employed in the real-time, or near real-time, analysis of data as is common in a patient monitoring scenario [16,47,49,51,52,55] and it is this context that the work described by this thesis is involved with. The example of smart alarming is such an area where assimilation of domain knowledge and data analysis can enable better health care. TA provides the means of integrating domain knowledge into the data analysis process, however the constraints of real-time environments, such as ICUs, place computational demands on knowledge-based frameworks that have proven difficult to overcome [60].

Section 2.1 explains the research methods used to perform the review of relevant literature and also the reasons behind the selection of topics. Section 2.2 then provides an introduction to the criteria and reasoning used for performing the review and section 2.3 presents the review itself. Section 2.4 represents reviewed literature in a summarised tabular format for easier analysis and cross referencing followed by section 2.5 which discusses the findings in light of present challenges and problematic areas. Finally, section 2.6 provides a conclusion which sketches the principal areas important for further development of IDA systems employing temporal abstraction within the clinical domain.

2.1 Review Methods

Papers were located through the use of a number of keywords such as "temporal abstraction", "patient monitoring", "real-time monitoring", "data-stream mining", "knowledge-based monitoring", "clinical data analysis", "intelligent monitoring", "data-stream processing", "time series analysis", "event monitoring", "temporal data mining" and combinations of these terms. Many research efforts were sourced from ScienceDirect and Artificial Intelligence in Medicine in particular, although ACM, IEEE and PUBMED databases also provided relevant material. Emphasis was given to papers published after 2000 although many of the papers that provide the foundations of research into intelligent clinical data analysis and knowledge-based temporal abstraction (KBTA) were written in the mid-nineties. Supplementary knowledge was also obtained through discussion with neonatologists at the Nepean Hospital NICU. Over two hundred papers were read in the preparation for this literature review. Many provided relevant background knowledge but have not been cited here as their content is reflected and represented more strongly in other works.

This chapter is not designed to represent a complete coverage of the topic of temporal abstraction in IDA as the primary focus are research directions that will enable IDA systems to deal with the escalating density and frequency of patient data hence the survey encompasses areas such as data-stream mining and processing and event monitoring where high speed data analysis is commonplace. KBTA may not be a component of such systems but the constraints of the real-time environment are shared between all these fields. Other researchers have published review papers addressing differing aspects of temporal reasoning within medicine [17, 73, 74], and inevitably there will be some small overlap but my primary focus, as explained above, is unique.

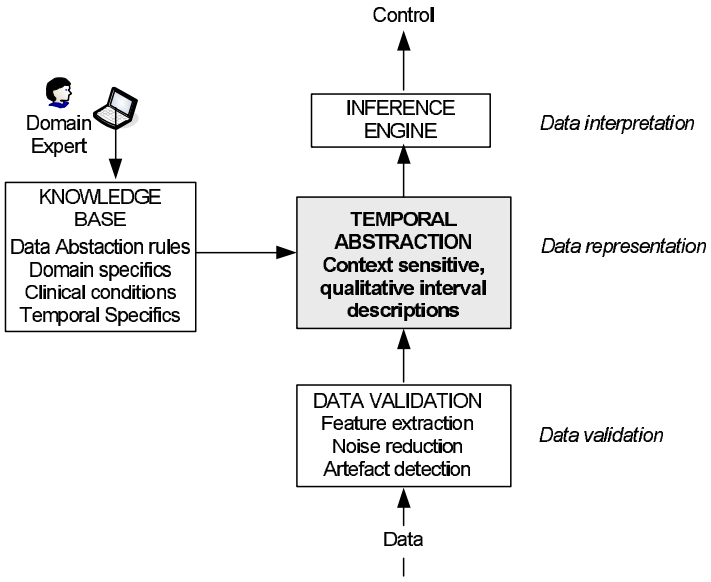

Figure 2.1: Typical Intelligent Clinical Data Analysis schema. Patient data is validated to remove artifacts and noise and is then subject to temporal abstraction (TA) mechanisms. A knowledge base stores domain knowledge that underpins TA processes. TA outputs are symbolically tagged and passed to an inference engine where abnormal states or conditions are detected. Control actions such as intelligent alarms or clinical guidelines are then initiated. Note: typical ICUs include multiple data input streams.

2.2 Review Criteria

This study reviews the technique of temporal abstraction which is an integral component within Intelligent Data Analysis (IDA). Lavrac et. al. [8], defines intelligent IDA as: "encompassing statistical, pattern recognition, machine learning, data abstraction and visualization tools to support the analysis of data and discovery of principles that are encoded within the data." They state that the principal differences between IDA and Knowledge Discovery in Databases (KDD) is that the techniques used are those of artificial intelligence rather than pure traditional statistical methods. Also KDD is primarily concerned with learning new knowledge whereas IDA is directed toward application of knowledge for data interpretation. Figure 2.1 illustrates a typical IDA system within the context of patient monitoring, where the system performs a number of distinct tasks which extend from processing raw physiological data to diagnosis and intervention. These tasks have been defined previously as: data validation, data

representation, data interpretation and control tasks [75]. Data representation and interpretation, being the areas where temporal abstraction (TA) is applied to data, are the principal areas I am concerned with in this review. Control tasks represent therapeutic actions to be taken upon discovery of abnormal data phenomena and are not discussed as they are beyond the scope of the survey.

Many IDA systems reviewed were not specifically designed for use in an intensive care environment, but instead, operated solely with stored clinical data. However, these frameworks still exhibit the core features of an intelligent data analysis system such as temporal abstraction, knowledge bases and inferencing capabilities and thus are relevant to this thesis. The data sources for the case-study patient monitoring environment are electrical transducers (sensors) attached to ICU patients whereas the data source for many of the reviewed systems are clinical time-series data bases, the latter often containing the exact same data that is measured in an online fashion within the NICU.

The key areas reviewed, as illustrated in Figure 2.2, are those concerning the task of TA within a clinical IDA framework such as: aspects of the data that are abstracted, complexity available within temporally abstracted data, dimensionality of the TA and data framework and the knowledge and reasoning underpinning TA processes. I propose these aspects are those that will become increasingly important for future IDA frameworks in clinical environments where data is being accrued at a greater rate than ever before and intelligent systems must be able to reason efficiently within tighter constraints imposed by the nature of these environments. Originally, knowledge-based systems in medicine endeavoured to integrate as much domain knowledge as possible. Today, as Horn [60] states, to cope with the changing nature of clinical environments there is a move toward more data-driven systems, but in medicine, knowledge is crucial in order to reach accurate diagnoses and to realise the evolution of a patient's condition. This notion is explored using the following criterion in relation to the needs of modern and future clinical IDA frameworks employing temporal abstraction of patient data.

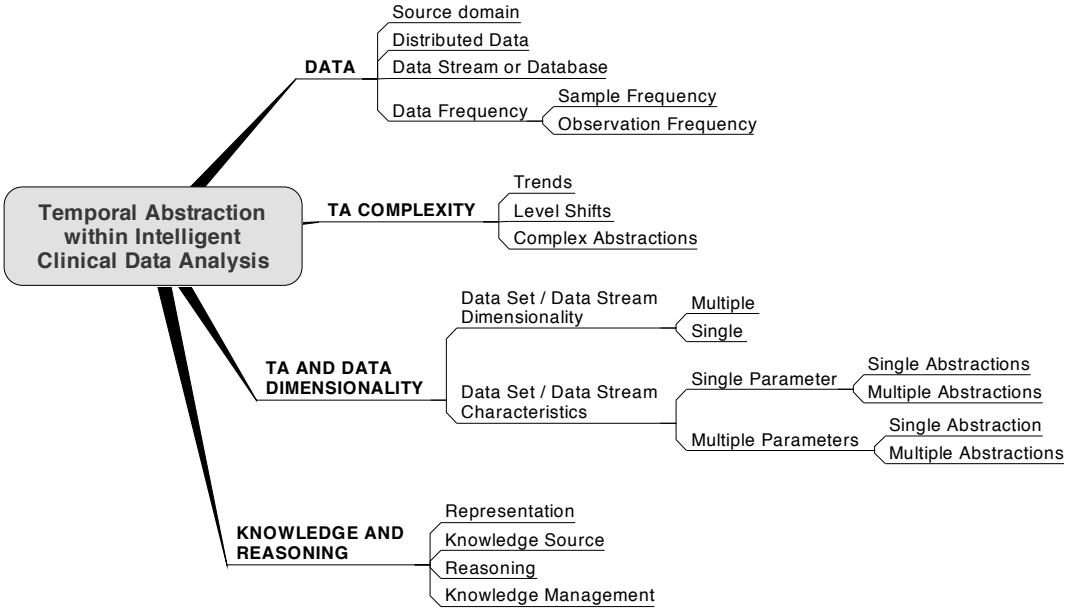

Figure 2.2: Categories utilized for review of intelligent clinical data analysis systems that incorporate temporal data abstraction. These categories form the structure of the 4 tables presented in section 5 for data, TA complexity, TA and data dimensionality and knowledge.

- **Data**: IDA frameworks incorporating TA have been used within a number of medical domains from Diabetes Mellitus [18, 20, 21, 35, 36, 45, 46, 76–79] to artificial ventilation of ICU patients [47–49, 80]. Characteristics of the source domain determine the type of data, for example, systems dealing with Diabetes are concerned with stored database data whereas ventilation monitoring produces a stream of physiological time-series data. The source domain also influences the types and frequencies of important observations and conditions. The data stream model presents differently from the traditional stored relational model. Generally, it is agreed that a data stream is a sequence of unbounded, real time data items with a very high data rate that can only be read once by the application [81]. The data rate imposes constraints upon processing in terms of response times, computational overhead and storage requirements [82] and these factors will affect the design of TA mechanisms. Hence, I am interested in what has been achieved regarding TA within various data environments where data characteristics have an influencing effect upon TA processes.

The e-Baby project, of which this research is a part, has enabled the real-time transmission of physiological data between remote hospital wards and referral hospitals [4,66,70,71] and I believe this is an area of health care that will assume a greater level of importance for clinical IDA systems of the future. Distributed data is included as a topic within the review for this reason.

- **TA Complexity**: Temporal abstraction mechanisms have been designed previously to depict relatively simple aspects of time-series data such as level shifts, periods of stability and trends. Although these are the most commonly used criteria clinicians use when making diagnoses from patient data, the extension of TA to include the representation of complex phenomena such as rapid heart rate decelerations and baseline variability is a feature worthy of exploration. It has been proposed that these particular heart rate characteristics exhibit as early indicators in neonates who are in danger of developing sepsis [6, 83]. An important aspect for future research is the early detection of such diseases hence the inclusion of TA complexity in the review criteria.

- **TA and Data Dimensionality**: Based on my investigations, TA mechanisms have not been applied to a multi-dimensional data analysis environment where multiple temporal patterns are abstracted from multiple data streams concurrently. Some work has been undertaken concerning distributed TA [36, 37, 84–86], where a level of parallelism is available within abstraction processes, but not within the real-time data streaming environment of an ICU ward. Languages have been developed that specify temporal patterns [77–79, 87, 88] and at a higher level, clinical guidelines [18, 45, 46, 89], but these do not express relationships between patterns and data on a multi-dimensional level. Integrating the specifications of TAs, which facilitate efficient storage, retrieval and maintenance, with a global specification of their coordination and control within a complex data streaming environment is an interesting area of development.

- **TA Knowledge and Reasoning**: The rules and knowledge that define the

> TAs to be processed are usually sourced by a clinician in cooperation with a knowledge worker. The knowledge that drives abstraction rules is crucial to create meaningful and contextual data abstractions and furthermore to interpret these abstractions correctly for successful diagnosis. I believe there is merit in providing the benefits of direct interaction with domain experts as well as gathering knowledge from automated data mining mechanisms. As Foster et. al. [90] discovered, there is a distinct gap between the concepts of clinical research and clinical management; which map, in computing terms, to the gap between data mining and data abstraction, the two primary methods involved in IDA [8]. Providing a bridge between the two, enabling the results of clinical research from data mining processes to be applied directly to data abstraction processes within a clinical management system presents a worthy direction for clinical decision support systems to take.

Successful management of knowledge within IDA frameworks in medicine is vital for arriving at accurate and timely data interpretation leading to diagnosis, prognosis or therapeutic decisions. The line which divides expressiveness and computational efficiency in a data environment such as Intensive Care, that is both high frequency and multi-dimensional, is an important and crucial consideration for ensuring timely response and accurate diagnosis. Aside from computational requirements, features such as the ease at which knowledge can be edited and understood by experts are aspects that will impact upon both the usefulness and adaptability of IDA systems to the dynamic nature of clinical environments.

2.3 Temporal Abstraction

Temporal abstraction has become a topic of much interest and research in clinical IDA systems [16,18–20,36–39,47,50,76,78,91–96] where high dimensionality and large volume of time stamped data is the norm. Its goal is to transform patient data from a low level quantitative form to high level qualitative descriptions which are closer to the language of clinicians [97]. As Miksch et. al. [98] state: “where the underlying

structure-function models are poorly understood, as in the case of medicine, it is almost impossible to build quantitative models for analysis purposes." Additionally, the context of analysis changes due to altering medications and progression of disease states, meaning data values that were considered normal at one time, or in a particular context, may be dangerously abnormal at another.

The process of TA takes either raw or pre-processed data as input and produces context sensitive and qualitative interval based representations. This is required in order for symbolic reasoning strategies to proceed and can be seen as a first point of entry for knowledge into the data analysis system. TAs can then be interpreted by matching against pre-defined templates or guidelines [18, 19, 99], or reasoned with in a higher context [20]. More specifically, Shahar [76], defines the temporal abstraction task as a process which, given a set of time-stamped parameters, external events and abstraction goals, produces abstractions of the data that interpret past and present states and trends and that are relevant for the given set of goals.

2.3.1 Data

For applications that process data-streams, the data rate and dimensionality make it mandatory to be able to perform computations on present data items before the next set of items arrives [82]. In other words, processing rate must be greater than arrival rate. This also enforces the requirement that a data-stream can only be read once by the application [82, 100]. Not only is data rate and dimensionality important, but the domain of interest has an impact on computational overhead, for example, the trends and patterns typical of the diabetes domain [46, 76] exhibit lower frequencies than those of artificial ventilation [47–49, 80]. A number of systems reviewed did not process data streams, but rather stored database data. Where these data repositories are large and it becomes infeasible to read the complete dataset into memory and process it as a whole, similar constraints are imposed upon these systems as stream processing frameworks. In an environment such as intensive care, data dimensionality and observational pattern frequency are high and computational requirements must

be taken into careful consideration to facilitate reasonable response times. As Lin et. al. [101] correctly assert, the choice of data representation has a large impact on efficiency and ease of data analysis tasks, hence I am interested in the efficiency of TA mechanisms at dealing with data intensive environments and also where the abstractions to be performed occur at high frequencies. Table 2.3.1 summarises the findings regarding data environments relating to TA.

Monitoring equipment within ICUs, such as the electro-cardiogram (ECG), sample physiological phenomena at very high rates (512Hz in the case-study environment). Many other devices are connected to the patient such as mechanical ventilators, dedicated pulse oxymetry monitors, dedicated pulmonary mechanics monitors and arterial blood gas and lactate measurements, to name a few. The amount of data streaming from a single patient is substantial and presents a challenging environment for intelligent data analysis. The techniques used by many data-stream processing frameworks are somewhat different to those of intelligent clinical data analysis. Stream processing is divided into either the application of continuous querying using the relational data model and a streaming version of an SQL type query language such as CQL [102], or stream data mining using statistical and quantitative techniques [1, 81, 102–105]. Whilst IDA is more concerned with obtaining context sensitive results from data mining activities, the algorithms and concepts underpinning stream processing could be highly valuable if integrated into the TA processes of IDA systems dealing with high frequency ICU data.

Datar [103] studied the method of sliding windows over single streams to compute basic statistics such as sums and averages. The idea of data reduction using sketches [106] and histograms [107] has been researched in order to provide an approximate answer when the exact answer would occupy more memory than may be available. Synopsis data structures are provided by the STREAM system [108] so that queries can be answered using batch techniques rather than processing each element individually. The synopsis is updated incrementally as new items arrive and supports two operations, an update and a query. This concept is similar to that employed

by Chittaro et. al. [51, 107, 109] where information to be queried (maximum validity intervals) are cached and updated as new data arrives.

RÉSUMÉ [21, 76] is a pivotal work in knowledge-based temporal abstraction and provides a framework of deep knowledge representation to perform temporal abstractions of patient data. Four types of knowledge are defined as being necessary to support the abstractions; structural knowledge, classification knowledge, temporal-semantic knowledge and temporal dynamic knowledge. These are instantiated by three domain ontologies. RÉSUMÉ's use of extensive domain knowledge is impressive but creates a computational overhead where memory and CPU requirements increase exponentially with the size of the data set [37]. However, RÉSUMÉ has been embedded into a number of frameworks for intelligent medical data analysis and has proven extremely useful for summarizing clinical data sets. Combined with the temporal pattern representation language CAPSUL [77, 79] and the therapy planning unit EON [113], RÉSUMÉ is used to abstract repeating temporal patterns in stored data. TZOLKIN [114] represented yet another development that incorporated the RÉSUMÉ data abstraction module into an architecture for integrating temporal reasoning with temporal databases; the so-called temporal mediation process. TZOLKIN itself has been utilized with a number of frameworks such as the Asgaard project [89] for clinical guideline execution and KNAVE [115] which enables temporal querying and visual exploration of abstracted data. IDAN [84, 112], is a more recent temporal mediator which when combined with KNAVE-11 [86, 116] and a modern version of RÉSUMÉ called ALMA [112] features a distributed approach to medical knowledge management and heterogeneous database access. Interestingly, the temporal abstraction process used within IDAN [85] has been designed for parallel implementation allowing abstractions to be applied to multiple patients concurrently. The process, called Probabilistic Temporal Abstraction (PTA), represents a move towards TA architectures that can deal with large amounts of data and knowledge and provide a level of computational tractability.

System, authors, year	Data source domain	Data stream or database	Data sample frequency	Observation frequency	Distributed data
RÉSUMÉ [21, 76], 1994	Diabetes Mellitus, Bone Marrow Transplant [79], Children's growth, protocol-based care	Database	Not stated, assumed low	Low	No
TRENDX [19, 99], 1995	Paediatric growth monitoring and ICU	Data stream and database	Not stated, assumed low	Very low	No
VIE-VENT [47, 55], 1996	Neonatal Intensive Care: artificial ventilation	Data stream	0.1Hz	High	No
Cached Event Calculus (CEC) [51], 1996	Artificial ventilation	Data stream	0.1Hz	High	No
NEO-GANESH [49], 1997	Adult Intensive Care: artificial ventilation	Data stream	0.1Hz	High	No
Larizza et. al. [36], Bellazzi et. al. [20, 35, 40], 1997 - 2000	Diabetes Mellitus (Blood-glucose data)	Database	Not stated, assumed low	Low	Yes
SUMTIME [39, 41, 42, 53, 54], 1999 - 2003	Neonatal intensive care, Gas turbines, Meteorology	Database	1Hz	High	No
EHCO [56], 1999	Open heart surgery: perfusion	Data stream	Not stated, assumed high	High	No
CAPSUL [77–79, 110], 1999 - 2000	Diabetes and Oncology	Database	Not stated, assumed low	Low	No
Miksch et. al [16, 52], 1999	ECG	Tested on database data	200Hz	High	No
RASTA [37, 111], 2001	Tested within ATHENA [111] for managing hypertension	Database	Not stated, assumed low	Low	Yes
Asgaard [18, 45, 46], 2001 - 2004	Neonatal Intensive Care: artificial ventilation. Diabetes	Data stream and database	1kHz or higher	High	No

Carrault et. al. [38], 2003	Electro-cardiogram (ECG): arrhythmia recognition	Data stream	Not stated, assumed high	Very high	No
Silvent et. al. [50], 2004	Intensive care: artificial ventilation	Data stream	100Hz and 1Hz	High	No
Belal et. al. [48], 2005	Neonatal Intensive Care: artificial ventilation	Data stream	128Hz for waveform data	High	No
IDAN-KNAVE [84–86, 112]	Bone marrow transplantation - designed for heterogeneous time-oriented databases	Database	Not stated, assumed low	Low	Yes

Table 2.1: Data environment characteristics relevant to temporal abstraction tasks.

Another temporal database mediator is that of Momentum [117, 118] which incorporates active database technology [119] to enable continuous evaluation of TA rules whenever new data is asserted to the database. TAs are calculated, stored and incrementally updated to enable more efficient query processing. This idea is very much inline with that proposed in [109, 120, 121] where a cache of continually updated TAs is used for faster querying. A dynamic processing environment is provided with data structures and algorithms which allow faster temporal reasoning, such as interpolation [59], and also separation of individual patient aspects to facilitate multiple queries on the same patient but with differing contexts. Although Momentum was not tested with high frequency data, an architecture incorporating dynamic data structures and streamlined TA algorithms provides a platform from which research into online MDTA could proceed.

Miksch et. al. [94] have provided a comprehensive method of TA which takes into account the context of time-point and time-interval values and can be applied to the higher frequency observations within data streams (up to 0.1Hz sample rate) evident within the field of artificial ventilation for neonates. Their system VIE-VENT extends Shahar's work to a more complete model of domain dynamics together with the ability for higher frequency reasoning. In the domain of protocol-based care, Seyfang and

Miksch [18, 45] eventually replaced the data abstraction methods in the Asgaard framework [89] with an adaption of their own work [47, 52, 80, 94] which was founded in VIE-VENT. This enabled Asgaard to operate in real-time on high frequency data streams. The emphases are on data validation and correct interpretation of values within given contexts and also on therapy planning using clinical guidelines. Their scheme is an elaborate one which combines statistical methods for data validation and a knowledge-based schema where the notion of a state spread allows both the position and uncertainty of data points to be realised.

Bellazzi and Larizza [20, 35, 40], employ similar methods of TA to the RÉSUMÉ system but they also utilise post processing of abstracted data where machine learning and statistical methods are used to classify data collected from diabetes home monitoring situations. Their analysis shows that abstracted data may reveal new knowledge more easily than raw numeric data but, similar to RÉSUMÉ, abstractions were based on low frequency observations from stored data. The marriage of temporal abstraction and machine learning is one of interest to this research as the ESP is to be designed to operate from rules either sourced from a clinician or extracted through data mining processes which can search for predictive associations between parameters in large clinical datasets.

Belal et. al. [48] extended the work performed by Miksch on the VIE-VENT framework to develop another system for open loop control of artificially ventilated babies within the domain of neonatal intensive care. They fuzzified the qualitative trend function developed by Horn et. al. [92] to allow for minor deviation around qualitative categories and to make the system less sensitive when small deviations are tolerable. This work is interesting as it is applied to the domain of real time and high frequency monitoring in a NICU and demonstrates that complex data analysis on multiple parameters (12) is possible with modern machinery (2005). Like many developments, this system is application driven and involves the pre-definition of fuzzy templates; systems that perform TAs that are defined at run-time are worthy of further investigation.

There has been a move towards incorporating TA into larger computing architectures where distributed processing is possible [36,37]. This represents an extension of TA from standalone data analysis into frameworks which can apply TA at a higher, more global level within an IDA framework. I would argue that this move is a necessary and inevitable one if IDA systems are to cope successfully with the ever increasing loads and rates of data collected from clinical environments across numerous institutions. Larizza and Bellazzi [36], have provided an HTTP-based server for carrying out temporal abstractions on behalf of clients using an internet based infrastructure based on TCP. Using Server to Server protocol (STSP), clients request and specify a particular TA together with relevant time periods for the abstraction process. The data to be analysed is sent to the server which then performs the required TAs and returns the results in terms of the number of intervals and associated time values that were discovered. The system is used within the Telematic Management of Insulin Dependent Diabetes Mellitus (T-IDDM) project where patients submit home monitored glucose data for analysis. The post-processing of abstracted data is also performed by another server within the framework [20]. Whilst the system provides a step forward regarding distributed temporal abstraction and the use of TA within client-server architecture where it could be possible to distribute the TA tasks between multiple processes concurrently, their focus was not on real time data analysis of multi-dimensional data and only single parameters and low frequency observations are analysed.

O'Connor et. al. [37], address the computational burden that results from temporal reasoning tasks, especially interpolation between intervals. Their system, RASTA, distributes the temporal abstraction processes using a configurable distribution scheme where a number of options are available. Each abstraction process can become a single thread or they can be configured as separate processes running on either the same or different parallel machines. The algorithm used ensures minimum resources are consumed by each abstraction task and hence reduces communication overhead between processes. Abstraction knowledge, which resides in a knowledge base, occupies

a hierarchical structure and is entered by a domain expert. RASTA was designed for the temporal abstraction of stored database data as opposed to real time data streams but represents a significant development in terms of expanding the application of TA to larger datasets and represents one of the few systems which can abstract multiple parameters in multiple data streams.

Much of the reviewed literature is based upon frameworks that operate within domains such as Diabetes or paediatric growth [19,20,20,21,35,37,76–79,99,111] where clinical aspects of the data evolve at a slow rate relative the high frequency nature of a patient monitoring domain. Only two systems provide facilities for distributed data processing [36, 37] and although these are aimed toward abstracting data from the low frequency Diabetes domain, they represent a step toward integrating TA into larger computing frameworks; this, I argue, will be necessary for future clinical IDA systems.

2.3.2 TA Complexity

I refer to complex data abstractions as those other than simple trends, level shifts or stable states. Using the temporal reasoning mechanism of temporal interpolation formally described by Allen [122], complex abstractions can be constructed from shorter intervals but this approach has generally been limited to the concatenation of simple trends or quiescent states. When data is oscillating at high frequencies and/or frequencies are irregular (as in the case of ECG), the process of data abstraction becomes increasingly complex and when compounded with processing constraints imposed by data intensive environments such as intensive care, many standalone systems reviewed would prove insufficient. Keravnou [123] provides a definition for complex abstractions that is used in this research and one that is also agreed upon by numerous researchers in the field of TA [20, 21, 35, 36, 40]. She depicts complex TAs as compound time objects, each with a repetition element, repetition pattern and progression pattern where the repetition element itself could be a periodic occurrence. Table 2.2 depicts the aspects of TA complexity chosen as relevant for this review.

Keogh et. al. [124] surveyed 3 major approaches of segmentation of time-series; sliding windows, top-down and bottom-up algorithms. Their experiments revealed that whilst sliding windows can be performed online, it lacks an overall view of the data that the batch methods, such as top-down and bottom-up, have. Batch methods suffer from the fact that they must read the entire dataset which is unreasonable in a streaming environment or where the volume of data is large. They proposed the Sliding Window and Bottom Up (SWAB) method which keeps a small buffer and enables what the authors call a "semi-global" view of the data whilst maintaining the ability to operate with online data. These methods do not integrate domain knowledge but rely on purely numerical methods for processing raw data. Once contextual information is added to the process and data has is merged using this information a complex abstraction or super interval is formed. This complex abstraction is really what is performed as a pre-processing step in many TA systems before data is partitioned on a qualitative basis. Larizza et. al [36] provide a definition for complex abstraction; "Complex TAs allow detecting patterns of complex shapes in uni-dimensional or multi-dimensional time series. Complex TAs represent higher-level abstractions not directly derived from the data, but from other TAs. The inputs of a complex TA are two series of intervals and the result is given by a set of intervals that verify a certain temporal relationship between the inputs." Their web-based TA server creates complex abstractions in the domain of diabetes monitoring and can detect two overlapping interval abstractions and also the successive increase and decrease of blood glucose levels [40]. These complex TAs require the use of Allen's [122] overlaps and meets operators however there is no frequency limit provided and the recurrence rate of these observations is low compared to what may be encountered within an intensive care unit using online data.

Chakravarty and Shahar have developed a language for the specification of periodic patterns called CAPSUL [77, 79]. It works in conjunction with the RÉSUMÉ system which performs the abstractions constituting patterns. The original RÉSUMÉ knowledge base was extended to accommodate the information required to define complex

System, authors, year	TA complexity	
	Comments	**Complex TAs**
RÉSUMÉ [21, 76], 1994	Sophisticated, domain independent management of temporal abstraction knowledge	Temporal interpolation of lower level abstractions to form low-frequency patterns
TRENDX [19, 99], 1995	Non-linear (quadratic) trends possible	Non-linear (quadratic) trends
VIE-VENT [47, 55], 1996	Expectation guided abstractions and smoothing of oscillating data near qualitative thresholds	No
Cached Event Calculus (CEC) [51], 1996	Computation of maximum validity intervals for mechanical ventilation properties and patient classification using Event Calculus formalism	Aggregation of intervals possessing the same property
NEO-GANESH [49], 1997	Aggregation of similar situations, 'forgetting' of non-relevant information	Yes, simple interpolation between intervals using rules
Larizza et. al. [36], Bellazzi et. al. [20, 35, 40], 1997 - 2000	Trend and cyclic component separation	Yes, aggregation of abstracted intervals to form oscillations
SUMTIME [39, 41, 42, 53, 54], 1999 - 2003	Produces textual summaries of time-series data	Yes, high level abstractions of patterns in different contexts
EHCO [56], 1999	Produces temporal sequence of state descriptions during course of monitoring, which may be disease states	No
CAPSUL [77–79, 110], 1999 - 2000	Addition of periodic pattern language CAPSUL to RÉSUMÉ	Yes. Linear periodic patterns. Cannot express patterns where repeating elements have differing frequencies
Miksch et. al [16, 52], 1999	Ability to abstract curves and bends in ECG trace	Yes. Irregular frequency of cycles is possible
RASTA [37, 111], 2001	Similar abstraction capabilities as RÉSUMÉ	Yes
Asgaard [18, 45,46], 2001 - 2004	Flexible description of complex repetitive patterns possible	Yes
Carrault et. al. [38], 2003	Abstraction of complex features of ECG: P and QRS complex waves	Yes
Silvent et. al. [50], 2004	Artificial ventilation specific: 'desaturation' and Sp02 disconnection	Yes
Belal et. al. [48], 2005	Fuzzy trend templates for expected normal conditions	Yes
IDAN-KNAVE [84–86, 112]	Incorporates the Probabilistic TA method [85] and TAR language [112]	Yes. Linear and repeating patterns, temporal interpolation between similar intervals

Table 2.2: The abstraction complexity available within temporal abstraction processes.

patterns and was tested in the domain of platelet transfusion in bone marrow transplant patients. The pattern frequencies in this context are low and CAPSUL cannot express patterns where the frequency of parameters within each repeating pattern differs. This creates a specification that is very regular and unsuited to the abstraction of complex and varying temporal patterns. CAPSUL provides a specification which has been based upon a formal representation of time and incorporates work done by Cukierman and Delgrande [87, 125], where repeating events can be represented in a time loop. It also draws on research performed by Terenziani [88] on the distinction between global and local time frames and constraints.

Miksch et. al. [16] approached the problem of abstracting ECG data using a less formal, more application directed approach, by comparing two different techniques aimed at creating a representation that was closer to the way a clinician analyses such data. Their method used the change in the derivative and the length of a regression line to detect bends in the trace. They found that the derivative method worked best for smooth data and the regression-based technique was more accurate for noisy data. Repeating patterns within the ECG are important for analysis so an approach was developed that calculated a reference point's position relative with the start of the oscillation and then compared this with the average of all preceding reference points. This revealed trends or deviations from the average along a series of repeating cycles. Carrault et. al. furthers the work on ECG analysis by providing a complex system of data abstractions using a combination of techniques including artificial neural networks (ANN) and wavelet decomposition for P wave detection and classical numerical algorithms for QRS complex detection. Their framework [38] translates rules from machine learning into temporal constraint networks [15, 41, 126–130] which are matched against abstracted data in the input stream.

Shashar et. al [131] utilise the work of Calvelo et. al [132] for decomposing the data stream into trend and stability for each data point. The maximum length of each segment which gives the best approximation according to the overall piece-wise accuracy and also localised linear accuracy is called the characteristic span. Rules are

then applied to facilitate symbolic conversion of the signal into transient, decrease, increase or constant categories. This symbolic description is elevated to another level of abstraction using words where individual characters, defining the state of the signal, are concatenated to form words giving rise to complex abstractions. The next phase of their research endeavours to design a system that can recognise certain sequences of words as being clinically relevant. This will involve a degree of temporal reasoning in order to relate 2 or more sequences of meaningful words. The natural extension to Calvelo et. al's work is transformation into the symbolic form, which is where this work makes it's greatest contribution.

The use of characters and words as symbols for describing the evolving state of a time-series signal has also been investigated by Lin et. al. [101] who state that the wealth of data structures and algorithms for dealing with discrete data (characters in this case), provides the strength of the method. Kahn et. al. [133] provided early work in the area of textual summaries for time-varying patient data with the TOPAZ system. TOPAZ uses symbolic and numerical techniques to interpret and abstract clinical features from raw data which are then converted to a textual output which can be understood by clinicians. At the heart of TOPAZ is an object-oriented temporal network called an ETNET [133, 134] which associates important time intervals with observations from the patient's medical record. For the retrieval of data from the ETNET a temporal query language called TQuery [133] was developed which constrains queries to only the contextual properties of interest. The SUMTIME project [39, 41, 42, 53, 54] also produces natural language summaries of time-series data and is currently being investigated within the domains of neonatal intensive care, meteorology and gas turbines. SUMTIME requires an extensive knowledge base and uses TA to create high level summaries which are then transformed into complex sentences. The system processes archived data where up to 600 values per second from more than 250 channels are stored and then sampled at a rate of 1Hz. Correlations between multiple channels can be performed by matching data to stored libraries of chronicles that are modelled using the CAPSUL [110] language. The volume of data

processed is impressive and the sample rate of 1Hz makes online application feasible.

The creation of complex TAs where there are varying frequencies within single pattern periods is a non-trivial task. The reviewed literature proposes numerous methods for creating complex TAs but, as Table 2.2 illustrates, none are aimed towards accomplishing this within online high frequency environments. High frequency and multi-dimensional data exacerbates the computational aspects of the problem to a degree which necessitates extensions to existing methods and further integration into parallel or threaded execution models which can cope with these demands.

2.3.3 TA and Data Dimensionality

I use the term TA dimensionality to refer to the process of obtaining multiple TAs from multiple patient data streams or datasets, each of which contains multiple data parameters. Intensivists may be interested in assessing a number of trends, level shifts or more complex patterns concurrently across multiple patients. Figure 2.3 was originally introduced in section 1.2; I include it here for its relevance to this section of work and to enhance readability. It shows two such patterns requiring a total of four abstractions; positive and negative level shifts and increasing and decreasing trends. Each data stream-set represents the physiological data from a single patient. My investigations have not revealed any systems which can perform this task in a data streaming environment. These aspects of TA dimensionality are used for review criteria in this section of the paper.

Several levels of dimensionality, regarding the application of TA to data, became apparent whilst reviewing the literature. The investigation, summarised in Table 2.3, revealed whether the data source consisted of either single or multiple parameters and furthermore whether single or multiple abstractions were applied. In many instances, multiple TAs were applied to multiple parameters. The next level of complexity relates to the number of data-sets or data streams. In other words, I am interested in systems that can apply TA mechanisms to multiple data-sets or data stream-sets each of which contains multiple parameters.

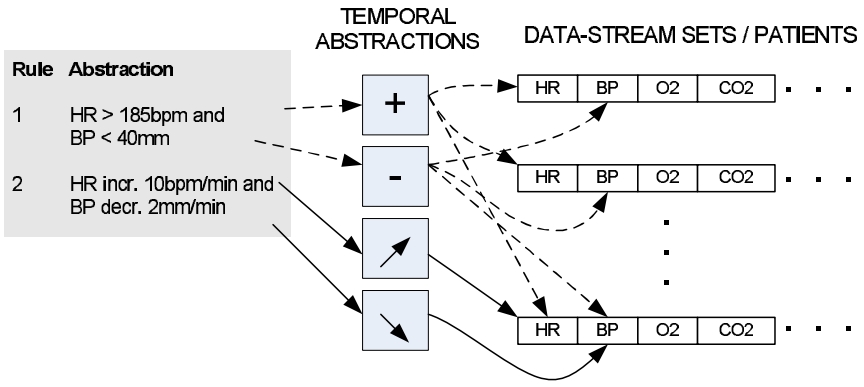

Figure 2.3: Temporal pattern 1 requires both positive and negative level shift abstractions of heart rate and blood pressure on all data streams. Temporal pattern 2 requires increasing and decreasing trend abstractions on a single stream. Note: temporal aspects of patterns are not included in this figure.

The research into TA within medicine has focused on knowledge representation, temporal reasoning and control tasks such as visualisation and clinical guideline execution. The application of these paradigms within larger scale architectures has not been as extensively investigated. This facet of TA within IDA is one of prime interest and motivation to further research enabling the detection of abnormal events within numerous patients' data streams in the domain of intensive care.

RÉSUMÉ [21, 76] was designed as a singular TA module and hence does not include the ability for abstracting multiple data streams. As previously mentioned, it has provided the inspiration for many subsequent research efforts into TA, including my own. It forms part of the EON framework [113] which has recently been further developed by O'Connor et. al [37] into a client-server framework for decision support in medicine. The framework has been tested on the implementation of ATHENA [111] for management of hypertension. Their emphasis was on the ability to integrate DSSs into existing legacy architectures whilst minimising the use of resources by clients. Whilst the clinical guidelines written in the Asbru language [18, 45, 46, 89] and utilised in Asgaard serve as a specification for sequential invocation of abstraction modules [135], neither VIE-VENT nor Asgaard considers the parallel application of multiple abstractions to multiple data streams. Larizza et. al's [36] HTTP-based

System, authors, year	Multiple or single datasets or data stream-sets abstracted	Single dataset or data stream-set dimensionality	
		Multiple or single data parameters	Multiple abstractions
RÉSUMÉ [21, 76], 1994	Single	Multiple (number not stated)	Yes
TRENDX [19, 99], 1995	Single	Multiple	Yes
VIE-VENT [47, 55], 1996	Single	Multiple	Yes
Cached Event Calculus (CEC) [51], 1996	Single	Multiple	Yes
NEO-GANESH [49], 1997	Single	Multiple	Yes
Larizza et. al. [36], Bellazzi et. al. [20, 35, 40], 1997 - 2000	Multiple	Multiple	Yes
SUMTIME [39, 41, 42, 53, 54], 1999 - 2003	Multiple (250)	Single	Yes
EHCO [56], 1999	Single	Multiple (max number not stated; 2 provided by example)	Yes
CAPSUL [77–79, 110], 1999 - 2000	Single	Multiple	Yes
Miksch et. al [16, 52], 1999	Single	Single	Yes
RASTA [37, 111], 2001	Multiple	Multiple	Yes
Asgaard [18, 45, 46], 2001 - 2004	Single	Multiple	Yes
Carrault et. al. [38], 2003	Multiple (2 x ECG channels)	Multiple	Yes
Silvent et. al. [50], 2004	Single	Multiple	Yes
Belal et. al. [48], 2005	Single	Multiple	Yes
IDAN-KNAVE [84–86, 112]	Multiple	Multiple	Yes

Table 2.3: TA and data dimensionality of temporal abstraction mechanisms. A data stream-set may consist of either multiple or single data parameters.

server can potentially perform TA on multiple datasets but not on multiple parameters simultaneously. Also, the abstractions performed are relatively simple and have been designed for low frequency observations within the diabetes domain.

For IDA frameworks, operating within the field of real-time patient monitoring, to successfully move into the future where ICU environments will become increasingly data-intensive, and decisions bounded by tighter time constraints, there is the potential for the gap between data generation and data comprehension to widen [8]. In order to integrate TA into a multi-dimensional data analysis environment it could be necessary to incorporate concepts from the fields of parallel computing, multi-agent systems or artificial intelligence real-time control and planning into the coordination of complex relationships between multiple TAs, data parameters and streams. TA mechanisms must be carefully specified and integrated into computing architectures that allow simultaneous processing of large amounts of data and integration of an appropriate level of domain knowledge.

2.3.4 Knowledge and Reasoning

Intelligent clinical data analysis systems require representation of knowledge to create meaningful and contextual data abstractions and consequently to enable correct interpretation of these abstractions leading to successful diagnosis [97, 136]. Over the past two and a half decades researchers into clinical expert systems have expressed the knowledge required to drive such systems at various levels from deep to shallow. Temporal abstraction requires enough knowledge to efficiently represent and summarise data where the level of abstraction relates to the level of knowledge. Various approaches have been proposed ranging from elaborate ontologies [21, 76, 96] to more simple data point schemata [18, 46, 55, 94, 98, 135] for assigning qualitative categories to data points. Table summarises the review findings for this section.

The computational effort required for reasoning with complex and deep level knowledge is substantially higher than for more shallow schemes. Within real-time domains such as intensive care, the time taken for a diagnosis becomes a constraint

that must be balanced against the representation of knowledge [56,137]. Today's data intensive medical environments are placing demands on IDA frameworks that require careful assessment of knowledge representation [60]; and TA, being a knowledge-based technique, requires that knowledge representation not become a hindrance to its beneficial outcomes. Ideally, knowledge structures for temporal abstraction should conform to the general requirements for knowledge representation such as expressiveness, consistency, ease of verification, formally well defined and easily understood by domain experts [60]. However, these qualities have been found to induce an intolerable overhead with some researchers realising that less formal approaches employing concise domain knowledge and structure are more efficient and useable for real-time patient monitoring systems [49, 138]. The balance between depth of knowledge and time constraints placed on reasoning is one that has been approached through application driven approaches, such as those of Miksch et. al. [16, 20, 47, 55, 80, 98], where succinct knowledge has allowed a degree of computational efficiency in real-time environments such as neonatal intensive care. Chakravarty and Shahar [78] reinforce this notion by using a more "goal directed approach" to the matching of patterns by only allowing relevant abstractions to be applied to data rather than allowing a more comprehensive set of abstractions to be instantiated. This concept of goal directed or application orientated is exemplified with the current trend toward the use of clinical guidelines [18,45,135,139]; which provide an efficient means of matching patient data to clinical knowledge and protocols.

Capturing domain knowledge has proven one of the largest challenges for builders of expert systems [140, 141] and it has become evident, through research into Intelligent Decision Support Systems (IDSS), that expert knowledge needs to be supplemented with facts gleaned from machine learning processes in order to solve more difficult problems [8]. Most reviewed systems capture domain knowledge from experts and encode this into rules or ontologies and often the process is guided in order to restrict and control the final outcome. For example, Achour et. al. [142] provides a user interface that leads the expert through a hierarchy of medical terms and relationships

from the Unified Medical Language System (UMLS). A more innovative approach was taken by Yu. et. al. [41], where experts are provided with a visualisation of actual data using a tool called Time-Series Workbench. Scenarios of normal or abnormal events are depicted, identified and then constructed into a temporal constraint network that can be used for pattern matching against real data. Silvent et. al. [50], suggest a form of closed loop knowledge management system where data mining is use to acquire new knowledge which is then used to drive new data abstractions. In a similar vein, the e-Baby project aims to direct the output of data mining processes, which may give rise to new knowledge, into data abstraction within real-time patient monitoring.

RÉSUMÉ serves as a benchmark in terms of knowledge management and enables the TA tasks to be defined at the knowledge level for any given domain as opposed to other systems where application level rules and functions hinder both the use of knowledge outside the current domain of interest and also the re-usability of knowledge to similar TA tasks within the same domain [96]. In terms of knowledge maintenance, the provision of AsbruView, [45, 143], a graphical interface for creating, editing and visualising clinical guidelines in the Asbru language, and PIXEE, an advanced plan editor, allows domain experts to access and modify treatment protocols. Guidelines and abstraction definitions are stored in a hierarchical plan library which facilitates efficient retrieval and maintenance and an XML like syntax is used to allow general interoperability with existing editing tools.

Architects of IDA systems within data intensive environments are required to carefully assess the level of knowledge representation required for the task at hand in order to ensure timely decision support. As previously mentioned, some researchers have proposed more application oriented designs where knowledge is succinct and efficiently queried [49, 138]. This notion is in contrast with earlier efforts, such as RÉSUMÉ [21, 76], where knowledge bases were designed to be a complete source of knowledge for the domain in question in order for higher order inferencing to take place. RÉSUMÉ represents a system that can be used within any domain as long as

the knowledge base for that domain is complete, and in this way, it represents the desirable property of re-usability within differing domains. Future knowledge based IDA systems will need to satisfy the software engineering paradigms of re-usability and component-based design and also be specific enough to ensure a high degree of efficiency in data intensive domains.

One of the key outcomes of temporal abstraction is a change in data representation from time-point to time-interval based [20], consequently most systems reviewed employ an interval based method of temporal reasoning that agrees with (or is a subset of) Allen's [122] definition of 13 possible interval relationships. This allows the inference engine to determine if, for example, an episode of elevated heart rate preceded an episode of decreasing blood pressure. Temporal reasoning has been a major area of focus in intelligent medical systems and many ideas have been proposed [17, 51, 57, 73, 95, 109, 123, 138, 144–147]. Explicit representation of time is important in medicine as the temporal order and duration of clinical events can make a substantial difference to the diagnosis [17]. Many systems developed for reasoning with time in medicine have been based on temporal ontologies and formalisms such as the work by Allen [122], Kowalski [148, 149], Shoham [150] and McDermott [151]. Clinical DSSs operating on stored patient information require the ability to reason with data that is either out of order or associated with inexact time points [147, 152]. A popular method has been that of constraint propagation although this has been shown to be NP-complete when using interval algebra [153, 154]. In the domain of patient monitoring, it can be assumed that all events can be considered time-stamped and totally ordered which removes the complexity associated with dealing with partially ordered events [155]. This aspect has been exploited in a number of patient monitoring systems [51, 95, 138] and results in a more efficient reasoning process.

Chittaro et. al. [51, 91, 109], have extended the Event Calculus [148] in order to make it suitable for application within the area of patient monitoring. The Event Calculus (EC) represents and reasons about events and the properties they either initiate or terminate and employs the concept of a Maximum Validity Interval (MVI)

which is the interval of time for which some property holds for. In the original implementation of EC [156], temporal reasoning involved with determining MVIs was performed at query time and after determination were not stored for later use, resulting in a complete re-computation of MVIs for each query. This results in a level of inefficiency not suited to high frequency monitoring domains. The Cached Event Calculus (CEC) generates the set of MVIs as incoming events are being updated to the database and then caches them to enable more efficient query processing at a later time. Domain knowledge is integrated into this process by either shortening or extending the MVIs dynamically. In fact, this method follows closely that employed by Sripada [121] who used the idea of a *Digest* to store all the conclusions that are deducible from a temporal database at a given time. Queries are then answered using only the Digest thus improving performance. The use of EC for more complex TAs has been researched [58] and a framework proposed which is based on an object-oriented data model for modelling both the domain knowledge and EC ontology. The authors claim a greater level of computational tractability over previous implementations due to the use of a procedural language for performing the MVI computations although, to the author's knowledge, there have been no experiments to verify this. Many extensions or variations to the original EC have been proposed in the literature [57, 120, 157–160] establishing it as an important and popular way of reasoning about events and the properties they either initiate or terminate.

When data is dense, one of the most expensive techniques is the concatenation of adjacent intervals, or temporal interpolation and inference [37, 59]. This is a type of complex abstraction (section 4.2) [36, 40] where lower level interval abstractions are combined into another, longer interval that satisfies given constraints or contextual information. Salatian and Hunter [161] have applied a technique utilising a median filter [162] and rules for temporal inference which create context sensitive intervals that can be merged into longer super intervals. Their algorithm has been applied to real-time heart rate data and is based on the work of Shahar [59] whose methods for

interpolation make up an integral part of the RÉSUMÉ system. Within the temporal database community, the practice of coalescing seeks to merge value-equivalent tuples with adjacent intervals. This is a form of complex abstraction which is also computationally expensive and is investigated by Böhlen et. al [163] who propose three different methods for performing coalescing using SQL expressions.

The EHCO system [56], incorporates decision trees to reason with abstracted data from patients undergoing cardiac surgery. Each parameter traverses at least one tree where the leaf level represents a disease state with corresponding treatment. Node tests are cached to provide explanations on how a decision was made and intermediate level knowledge [137, 164] is used to prune ambiguous hypotheses. Multiple parameters are reasoned with although frequency limits are not discussed and the detection of abnormal states requires careful definition beforehand. Tsien et. al. [165, 166], also employed decision trees to classify multiple parameters from a NICU by abstracting the data using linear regression, moving mean and maximum and minimum values, then using an expert to annotate artefacts in heart rate, blood pressure, carbon dioxide and oxygen data. Annotated data was used to create a decision tree for each parameter which were then applied to data collected form the ward with the aim being to address the problem of false alarms [28, 167, 168]. Each tree contained information relating to the other measured parameters hence a degree of signal integration is offered, however, results demonstrated that best performing tree was the O2 tree as it contained only O2 data; prompting the authors to query whether the method of signal integration could be improved. The window lengths of 3, 5 and 10 minutes that were used for the derivation of values are too wide for real-time monitoring scenarios and the performance of decision trees, under multiple signal conditions with small (1 second) windows, was not

The TrenDx system creates temporal templates that are used to match process data in order to arrive at a hypothesis [19, 99]. Trend templates are temporal data abstractions that are pre-defined, low-order polynomial regression based predictors of expected trends in patient data sets. Introduction of limited domain knowledge is via

qualitative or quantitative value constraints placed on the trend templates and the grouping of templates into monitor sets that are related to a particular context. The development of trend templates is a time consuming task that must be performed prior to the matching of patient data with the expected template. Of interest is a more dynamic method of pattern detection where standard abstraction modules, performing trend and level shift detection, can be invoked at run time to process incoming data. Furthermore, TrenDx does not take into account the possibility of shifting thresholds in different contexts, such as where medication is administered and expected behaviours are substantially above or below 'normal'. During testing of TrenDx on stored clinical data representing paediatric growth, Le [169] found it took several hours to fully process some patient's data however, using more modern hardware, De Souza's experiments [170] yielded results within a few minutes. De Souza's conclusion was that TrenDx requires further improvements before being useful in a clinical setting due to lack of sensitivity when small amounts of data are available. Tests on streaming data were not carried out but as Li [171] states, the computational complexity of TrenDx is prohibitive for high frequency clinical data.

The creation of temporal scenarios or chronicles has been popular with a number of researchers [15, 38, 41, 126–130] to model a succession of events constrained by time. The Déjà Vu patient monitoring system [15], abstracts events from the raw data stream and constructs a session, which is a temporal constraint network consisting of vertices (events) connected by edges (time constraints). Events are important occurrences within the data such as the disconnection of suctioning or beginning of a normal ventilation period. Sessions are matched in real time against pre-defined scenarios, which are also temporal constraint networks. Limitations of this approach are the computational overhead and the difficulty involved in building scenarios from expert knowledge.

Rules are one of the oldest schemes for representing knowledge and have been utilised extensively within clinical data analysis systems. They have the advantages

System, authors, year	TA knowledge representation	TA knowledge editing	TA Knowledge source	Reasoning
RÉSUMÉ [21, 76], 1994	Extensive frame-based parameter, event and context ontologies	Easy (User Interface)	Expert	Rule-based (CLIPS) temporal reasoning
TRENDX [19, 99], 1995	Low-order polynomial trend templates represented by Temporal Utility Package (TUP)	Not stated, assumed difficult	Expert	Data assigned to intervals within templates using TUP [147] creating numerous hypothesis. Mean absolute percentage error (MAPE) used to score hypothesis
VIE-VENT [47, 55], 1996	Rule-based	Easy	Expert	Rule-based (CLIPS)
Cached Event Calculus (CEC) [51], 1996	Rule-based on Event Calculus	Not stated, assumed difficult	Expert	Rule-based (Prolog)
NEO-GANESH [49], 1997	Object orientated rule bases	Easy	Expert	Rule-based (SMALLTALK-NEOPUS). Event calculus [148] basis for temporal reasoning
Larizza et. al. [36], Bellazzi et. al. [20, 35, 40], 1997 - 2000	Not stated. Appears integrated into pre-processing and also TA mechanisms	Not stated, assumed difficult	Expert	Statistical analysis of abstracted data
SUMTIME [39, 41, 42, 53, 54], 1999 - 2003	Rules [39], Domain ontology, Output ontology (concepts for communicating results to experts)	Not stated	Expert	Unclear. Numerical methods combined with assumed rule-based method
EHCO [56], 1999	Frame-based decision trees	Not stated	Expert	Decision trees with frames
CAPSUL [77–79, 110], 1999 - 2000	As for RÉSUMÉ	Easy (User Interface)	Expert	Rule-based. More goal directed than RÉSUMÉ
Miksch et. al [16, 52], 1999	Rule-based	Not stated, assumed easy	Expert	Rule-based and numerical techniques for abstracting curves - regression and 2nd order derivative
RASTA [37, 111], 2001	Frame-based ontologies	Easy (User Interface)	Expert	Not stated
Asgaard [18, 45, 46], 2001 - 2004	Clinical Guidelines - XML-based schemata	Easy (User Interface)	Expert	Rule-based and numerical

Carrault et. al. [38], 2003	Rules, Quantitative algorithms, Neural Network, Chronicles	Not stated, assumed difficult due to distributed and differing nature of knowledge representation	Machine learning (ILP)	Chronicle recognition
Silvent et. al. [50], 2004	Rule-based	Planned but not implemented	Expert and machine learning	Rule-based
Belal et. al. [48], 2005	Rule-based	Easy (User Interface)	Expert	Numerical, rule-based and fuzzy trend templates
IDAN-KNAVE [84–86, 112]	Prolog rules, XML formatted knowledge bases	Easy (User Interface)	Expert	Rule-based

Table 2.4: Knowledge and reasoning components of temporal abstraction systems.

of representing a clinician's knowledge in a declarative way that is simple, easily understood and offers an easy inference mechanism [8]. One of the earliest rule-based systems for reasoning with time-oriented clinical data was VM [172]. It used context-sensitive production style rules to create TAs, such as hemodynamic stability, which could then be reasoned with to aid in therapy management. Similarly, TOPAZ [133] creates context awareness by placing rules within the nodes of a temporal network that are activated when the associated event is active. Chakravarty and Shahar [77–79,110] apply rules to detect patterns in data by automatically accepting definitions of patterns from a knowledge base and translating these into rules that generate an instance of the desired pattern. The instantiated pattern can be re-integrated into the TA mechanisms of RÉSUMÉ serve as the basis of more complex abstractions. Miksch et. al. [47] also use rules to adjust the qualitative value (abstracted value) of parameters in the domain of artificial ventilation control of newborn infants in neonatal intensive care. Rule-based Systems (RBS) are an effective method of reasoning not only for the specification of TAs but for the detection of abnormal states within data. They are easily integrated with TA mechanisms which produce symbolic descriptors relating to patient state. Descriptors form the antecedent and the consequent of a rule can

be configured to activate alarms or initiate control actions, as described in [16, 47].

2.4 Comparison of Intelligent Clinical Data Analysis Systems that incorporate Temporal Abstraction

Tables 2.3.1 - 2.3.4 provide visual summation of the characteristics used to perform comparison between the reviewed research. The criteria used are those that are immediately important to the development of the framework for MDTA and have been used in the preceding literature review as discussion points.

2.4.1 Data

Characteristics of the data environment which have been reviewed as being relevant to TA processes are presented in table 2.3.1. The data source domain determines the nature of observations to be abstracted. Some domains encompass low frequency observations, such as blood-glucose levels in the domain of Diabetes Mellitus, while others entail high frequency characteristics, for instance heart rate fibrillation in the domain of neonatal intensive care. Reviewed literature either abstracted data from a database or from online data streams. In either case, the sample frequency of the data is an important factor to the design of the TA mechanisms and has an impact upon the granularity of the time measure which determines whether TA will be employed to bridge gaps in data or to summarise more dense data [8]. I believe the integration of TA into larger architectures, for dealing with distributed data or larger amounts of data, is an interesting and timely development for possible future systems, hence the inclusion of a distributed data category.

2.4.2 TA Complexity

Reviewed literature applied TA in varying degrees of complexity from relatively simple level shifts and trends to compound abstractions based on combinations of more primitive abstractions. Here, complex TAs are defined as being either interpolated

simpler TAs or depictions of complicated and sometimes non-linear aspects of data, such as ECG characteristics. Generally, more complex abstractions create a computational burden related to the temporal reasoning mechanisms involved in interpolation between shorter intervals. In cases where data abstraction is performed using numerical or model based approaches, issues of computational complexity are still of vital importance especially where real-time data environments constrain the time and resources available for TA processes.

2.4.3 TA and Data Dimensionality

Reviewed literature utilises data in the form of either stored datasets or in data stream format sourced from physiological monitors. In either case, for IDA systems that employ temporal abstraction, it is relevant to review the dimensionality of this data and ability of TA techniques to be applied within multi-dimensional data environments. Figure 2.2 in section 3 depicts how I have defined the dimensionality of the data environments reviewed and Figure 2.3 section 4 provides further explanation of what is meant by TA dimensionality. Whether data is sourced from databases or physiological monitors, within each dataset or stream there may be multiple parameters. Within the scope of an ICU (data streams) or clinical data environment (databases), there may be multiple datasets or streams. Within the TA environment itself, there will be multiple abstractions to apply, some of these could apply to a single dataset or stream and some may be employed on multiple datasets or streams.

2.4.4 Knowledge and Reasoning

Temporal abstraction creates qualitative abstractions of raw numerical data and lifts the level of description to a level that is closer to the language of clinicians so that knowledge-based reasoning can attempt to summarise a patient's state and provide a path towards diagnosis [9, 97]. TA is the point of entry for knowledge into the IDA process and as such, the representation and management of knowledge is an important aspect of IDA frameworks. Features such as the ease at which knowledge

can be edited and understood by experts and the reasoning involved in application of knowledge to data are aspects that will impact upon both the usefulness and efficiency of IDA systems. Knowledge in IDA systems has traditionally been sourced from domain experts and this has been recognised as being one of the most difficult tasks involved in building knowledge-based systems [173–175]; I am interested in applying the results of clinical research to real-time patient monitoring hence creating a bridge between the two primary methods used in IDA systems supporting decision making in medicine; data abstraction and data mining [8].

2.5 Discussion and Review Summary

The range of medical data domains represented is wide, ranging from Diabetes Mellitus to intensive care unit ECG monitoring and artificial ventilation. The differing domains give rise to varied data environments which extend from stored data to online and high frequency streaming data (Table 2.3.1). The domain of interest also governs the features in the data that will be relevant for diagnosis of abnormal states which vary from low frequency observations, such as weekly patterns of anaemia in the domain of oncology [78], to the 120ms duration of a QRS complex in an ECG trace [38]. Approximately half of the reviewed literature dealt with low frequency observations and data was sourced from a database [19–21, 35–37, 40, 78, 79, 84, 85, 99, 110, 112]. The other half analysed streaming data and performed high frequency data abstraction [16, 18, 38, 39, 42, 45–55]. The analysis delved deeper into the complexity of abstractions and dimensionality of the data environment to ascertain where possible problems may arise due to increasing computational requirements (Tables 2.2 and 2.3).

Complex TAs (Table 2.2) are a requirement in clinical IDA systems for determining the evolving state of the patient as they represent a relationship between time intervals in patient data and provide facility for reasoning about recurring events [74]. Many researchers have developed complex TAs, where shorter intervals are connected to form larger super-intervals. Shorter sub-intervals are often stable states, level shifts

or increasing or decreasing trends which can then be connected using Allen's [122] meets and overlaps operators for convex intervals, forming a so called, complex TA. This process of temporal reasoning can consume significant computational resources depending on the complexity of the final abstracted pattern. The reviewed literature created complex abstractions ranging from simple connections of linear sequences, where there are no periodic constraints [39], to the depiction of complex aspects of ECG data [16,38]. The area of most difficulty lies in the abstraction of rhythmic data where repeating elements differ, thus requiring definition of increasing numbers of periodic elements. The specification of periodicity in the reviewed literature included Chakravarty and Shahar's provision of the CAPSUL language for periodic pattern definition [79,110], and the less formal approach of Miksch et. al. [16], that is able to represent patterns with elements that have differing and high frequency characteristics, in the domain of ECG monitoring. Carrault et. al. [38] and Dojat [49] employed temporal constraint networks for the detection of temporally related events within the fields of ECG monitoring and artificial ventilation respectively.

The data environment where TA is to be applied (Table 2.3) is of paramount importance to the design and efficiency of TA mechanisms. Reviewed TA frameworks cover the issue of data dimensionality to varying degrees, but in general, TA has been designed for standalone processing of singular datasets or streams. Of the 3 systems that were able to handle multiple datasets [20,35–44], only one performed TA on data streams [38]; and then it was only 2 streams (2 x ECG channels). Of all the systems that handled streaming data input [16, 18, 38, 39, 42, 45–56], most were able to apply multiple TAs to multiple data parameters but this occurred within a singular patient data stream only.

It is apparent from the review that a major step forward can be made in the evolution of TA-based IDA frameworks in the ability to abstract and analyse multiple patient data streams in real-time (as depicted in Figure 2.3). In all cases except for two [16,38,39,42,52–54], it was possible to apply multiple TAs to multiple parameters within a single stream or dataset however only two frameworks could apply multiple

TAs to multiple patient datasets concurrently [36,37]. In these cases, patient data was sourced from databases and observation frequencies were very low. The reason for these results may stem from the fact that TA has traditionally been a technology for "generating useful information about a single patient" [8]. Moving the TA paradigm from the standalone processing of singular patient datasets into a more distributed role where multiple data streams are abstracted concurrently, would represent a major step forward for clinical data analysis systems.

From the 14 systems reviewed only two [38,50] utilised machine learning techniques for the gathering of knowledge to aid in the data analysis process however they did not provide the ability for automatic translation and integration of that knowledge into TA mechanisms. This would enable knowledge from clinical research to be applied directly into clinical management and real-time patient monitoring and represents a worthy direction for future research.

2.6 Conclusions and Correlation with Thesis Hypotheses

This chapter has presented an overview of some of the most significant research efforts in the field of temporal abstraction. The survey was based on factors that will assume an increasing level of importance for future clinical IDA systems and are also of importance to my research into temporal abstraction within the domain of neonatal intensive care. These topics were: aspects of the data that is abstracted, complexity available within TA mechanisms, dimensionality of the TA and data environment and the knowledge and reasoning underpinning TA processes. Based on the review criteria, a number of limitations of existing systems have come to light. Here, those findings are summarised.

The dimensionality of TA-based IDA frameworks is limited. TA has been widely applied to singular data-sets or streams but has not been extended to cope with situations where there are multiple abstractions to be performed on multiple parameters

within multiple high-frequency data stream-sets. It is evident from the review that for IDA systems to successfully move forward into the future where clinical environments are becoming increasingly data-intensive, the ability for managing multi-dimensional aspects of data at high observation and sample frequencies must be provided. In recognition of these facts, this dissertation proposes hypothesis 1 as follows:

> **Hypothesis 1**: A framework can be defined to enable temporal abstraction within a multi-dimensional data environment in real time. This framework is named the Multi-dimensional Online Temporal Abstraction (MOTA) framework.

Standalone systems will be required to expand into larger computing architectures which offer the necessary degree of robustness, control and co-ordination of TA processes required for such data driven worlds. Hypothesis 2 proposes the development of mechanisms necessary to ensure complex inter-relations between entities within the multi-dimensional environment are explicitly represented to provide organisation and co-ordination of MOTA framework processes.

> **Hypothesis 2**: A method to semantically *represent* the complex multi-dimensional relationships between data streams, parameters and TAs can be defined to support TA within such complex environments. This method is catered for by both the Event Correlation Specification (ECS) and the Active Domain Registry (ADR).

The knowledge and reasoning which underpins TA processes has been primarily developed for temporal reasoning over large and static data-sets. In this application, the reasoning involved is different to that required for a real-time, online scenario. Operations such as temporal interpolation, which carries the potential for significant overhead in terms of computing resources, require a different approach for the online application context. The ability to offer a formal and well-studied method for temporal reasoning, in real-time, is proposed by hypothesis 3.

> **Hypothesis 3**: The Event Calculus can be extended and applied to cater for temporal interpolation and abstraction of real time data.

There is a lack of seamless integration between knowledge obtained though data mining processes and data abstraction techniques. Storage of medical data using data warehousing allows the exploration of patient data for possible knowledge, in the form of associations and trends, which may act as early warning signs for impending disease, as in the case of sepsis. To be able to fully exploit this ability, the integration of data mining processes performing clinical research into IDA systems becomes a necessity. Providing this bridge between clinical research and clinical management connects the two, generally independent IDA processes of data mining and data abstraction which in the year 2000, were reported as having progressed independently of one another [8]; this trend has continued over the last 5 years to the present day. Hypothesis 4 addresses this requirement through development of an interface to an analytical processor which mines physiological data for potential detection of 'early indicator correlations' between parameters leading to the onset of disease.

> **Hypothesis 4**: The framework can support the definition of TA rules by allowing an interface to an analytical processor which mines physiological data to reveal new associations between data parameters thus providing a bridge between data *mining* and data *abstraction* processes.

Through the application of hypotheses 1 to 4, which tackle the above limitations, hypothesis 5 proposes that clinical management will be enriched.

> **Hypothesis 5**: Proposals 1 to 4 can be applied within a Health and Medicine context to enrich Clinical Management.

Future directions for TA research include the abstraction of complex data features, the application of TA to multi-dimensional data and the use of TA in data-streaming environments. Improving the computational efficiency of TA processes would aid in all of these endeavours. Research has approached this aspect from differing angles;

from the improvement of initial raw data processing tasks, such as linear regression and segmentation, through to the caching mechanisms employed in conjunction with the event calculus. Extending IDA research with the ability to perform online MDTA will endow clinicians with more powerful methods for the early detection of disease and the capability to monitor *multiple* patients for *multiple* conditions concurrently. This is especially important in neonatal intensive care where babies can exhibit abrupt changes in health status that often go unnoticed.

Chapter 3

Case Study Background

This chapter describes the case study environment of neonatal intensive care with particular emphasis placed on aspects relating to this thesis. In doing so, I address hypothesis 5 stated in section 1.3.3, which proposes the application of hypotheses 1 to 4 within a health and medicine context to enrich clinical management.

As revealed in section 1.3.1.1, during 2005 there were 90,608 babies born in NSW; of these there were 14% (12,320) requiring admission to special care wards and 2.5% (2,257) requiring admission to neonatal intensive care units (NICUs). Studies performed over previous years expose similar statistics [24, 25]. Commonly, admission to an NICU indicates serious illness and/or a high level of prematurity leading to the death of 7.7% (174) babies. Life long disabilities including cerebral palsy, mental retardation, blindness and deafness are evident for a further 15% of the NICU babies. Numerous medical diagnoses, treatments and tests, many of which have long term repercussions for the individual, are carried out within the NICU [26]. Nepean Hospital is located in the western area of Sydney and is one of ten level three neonatal referral centres in the states of NSW and the ACT. Typically, approximately 700 babies are cared for annually in the Nepean NICU.

Some babies may spend up to three or four months in the NICU and during their stay are connected to numerous electronic monitoring instruments which are responsible for measuring and displaying the patient's vital signs to clinical staff. Present

monitoring equipment creates major limitations in clinical management in that patient physiological parameters are not measured with respect to time. This has ramifications in both the frequency of alarms and rate of false positives [28,167]. Additionally, it is not common for such equipment to be able to deal with a multi-dimensional data environment. Present machines have difficulty measuring and analysing more than one data parameter, such as heart rate, and there are no such monitoring environments which are able to monitor multiple patients concurrently in real time [23]. The monitoring of multiple patients at the same time offers the possibility of tracking disease spread within a NICU ward. Finally, recent medical research proposes that conditions such as Sepsis [6], Periventricular Leukomalacia (PVL) [7] and Pneumothorax [33] may exhibit early warning characteristics in physiological data before being diagnosable using traditional means such as blood cultures (sepsis), chest x-rays (pneumothorax) or cranial sonography (PVL). These limitations to clinical management imposed by inadequate monitoring equipment are addressed by the thesis hypotheses 1 to 4 in section 1.3.3.

3.1 Levels of Care

In NSW, neonatal and obstetrics care is classified into the following categories, as defined by the NSW Department of Health [176]:

3.1.1 Neonatal Care

1. Tertiary

   (a) *Level 3*: Neonatal Intensive Care Unit. A unit providing high dependency specialist nursing and medical care for newborns including sustained life support such as mechanical ventilation. Includes staff neonatologists and neonatal registrars.

2. Non-tertiary

(a) *Level 2a*: a unit that can provide high-level oxygen, can start mechanical ventilation and had paediatric house staff

(b) *Level 2b*: a unit that can give low-level oxygen and has a paediatrician on call.

3.1.2 Obstetrics Hospitals

1. *Level 1*: local hospital, no births, postnatal only.

2. *Level 2*: small isolated hospitals, low-risk births only. Staffed by general practitioners and midwives.

3. *Level 3*: country district and smaller metropolitan hospitals, care for mothers and infants at lowmoderate risk. Full resuscitation and theatre facilities available. Has Level 2b neonatal care.

4. *Level 4*: country basemetropolitan district hospitals. Delivery and care for mothers and/or babies with moderate risk factors. Has Level 2b neonatal care.

5. *Level 5*: country basemetropolitan district hospitals, care for mothers and infants known to be at high risk. Has Level 2a neonatal care.

6. *Level 6*: (tertiary) specialist obstetric hospitals (supra regional). All functions: low, moderate and high-risk births. Has Level 3 neonatal intensive care.

3.2 The Transition Period

I begin the discussion of neonatal intensive care with the typical events and actions constituting the first few hours of life of a baby admitted to the NICU. The *transition period* is considered to be the first six to ten hours of life and is a time when the newborn moves from intrauterine placental support to extrauterine self-maintenance [176].

Abbr.	Measurement	0	1	2
A	Appearance (colour)	Pale or blue	Body pink but extremities blue	Pink
P	Pulse rate (measured by stethoscope)	No pulse	$<$ 100 bpm	$>$ 100bpm
G	Grimace (response to stimulation such as mild pinch)	No reaction	Grimace	Grimacing and cough, sneeze or vigourous cry
A	Activity (Muscle tone)	Muscles loose and floppy	Some muscle tone	Active motion
R	Respiratory Effort	Not breathing	Breathing slow and irregular	Cries well

Table 3.1: The APGAR score.

3.2.1 The First Hour of Life

Babies that are born at a gestational age of less than 37 weeks are classified as premature. Premature babies are usually born by caesarean section (58.5%), although a number are born through normal vaginal delivery (33.7%) [176]. Resuscitation is sometimes necessary to initialise breathing activity and during the first minute of life cardiac and respiratory problems are normally identified through visual examination [177]. At one and five minutes after birth the newborn is assessed using the Apgar score. The one minute score determines how well the baby handled the birthing process while the 5 minute score assesses how the newborn is adapting to its new environment. A rating is given from one to ten with ten being the healthiest score possible. Scores below five indicate that the infant requires immediate attention and assistance in adjusting to its new environment. The Apgar score is given according to Table 3.1. A physical examination proceeds to discover common variations of normal defects and to quickly initiate any interventions that may be required. Generally the examination is carried out whilst the baby is relatively free of stress, then a data base can be established for further observations and comparisons. [177].

Most premature babies suffer from breathing problems due to the immaturity of their lungs at birth. These problems can range from severe respiratory failure to apnoea [178]. If the clinician suspects a baby will be born prematurely, often he/she will be administered a corticosteroid medication to help speed up lung development

and prevent bleeding into the baby's brain after birth. The infant will then be placed on a ventilator and, if the birth was within the same tertiary centre as the NICU, transported to the NICU in a warmed transport incubator with associated monitoring equipment such as a pulse oximeter for measuring heart rate and blood oxygen levels.

Not all premature births are carried out within immediate vicinity of a level three NICU. During 2005, across all NICUs in NSW, 33% of NICU registrants were born following a maternal transfer, 29% were transferred following birth (4.4% of these were transferred during the first day of life) and 36% were born following a booked tertiary centre birth [25]. Babies born premature in regional centres, which could include level two or three obstetrics hospitals, are isolated from the expert level of care that a neonatologist can provide. Critical decisions often have to be made pertaining to the treatment of the newborn and a geographically isolated neonatologist must assess the often critical and rapidly evolving situation via the telephone. The neonatologist cannot view the baby's physiological data and relies on a verbal description by the attending physician. Transferral of the baby to a tertiary centre with level two or three neonatal support can be life threatening given the vital requirement to preserve a consistent environment [26]. These aspects of geographical isolation are especially pertinent in Australia where many areas of the country are remote, both geographically and through lack of supporting medical infrastructure. The Bush Babies [71] and e-Baby [4, 26, 27] research initiatives addressed these issues through the provision of a framework for remote transmission of physiological data thereby, in some cases, elimiating the need for moving the baby.

Generally, upon arrival at the NICU, the infant is weighed, measured and connected to numerous monitoring instruments which measure and display the baby's vital signs. If the baby is receiving assistance from a mechanical ventilator it will remain connected throughout these procedures. Physiological data to be measured includes blood pressure, blood oxygen saturation (SaO2), blood carbon dioxide (CO2), heart rate, respiratory rate and numerous ventilator settings such as Peak Inspired Pressure (PIP) and Positive End Expiratory Pressure (PEEP). A cerebral ultrasound

is usually conducted to check for Intraventricular Haemorrhage (IVL) and Periventricular Leukomalacia (PVL), both of which are high risk conditions for premature babies. IVL is characterised by bleeding into the fluid-filled areas surrounding the brain and PVL involves the death of small areas of brain tissue around the fluid-filled areas. This damage creates "holes" in the brain. [179, 180] and can lead to serious health issues such as cerebral palsy and blindness. It is unclear as to the cause of these conditions but they are commonly thought to be resulting complications of respiratory distress syndrome [7, 181, 182].

This early stage in the baby's life is a most important period for determining what interventions are required (if any) and assessing the overall health status of the newborn. Once the initial examination and testing phase concludes, which normally takes approximately an hour, the baby moves into the monitoring phase of the transition period.

3.2.2 NICU Monitoring

Following the initial examination and assessment period, the baby is connected to a number of monitors for continuous reading of vital physiological parameters. An electrocardiograph (ECG) is attached to the baby's chest via sticky electrode pads and used to ascertain cardiac function and determine if abnormality exists. If the baby is receiving mechanical ventilation, which is used in neonates to correct abnormalities in oxygenation, alveolar ventilation, or respiratory effort [177], there are numerous measurements to be monitored by nursing staff. These include set and delivered Peak Inspiratory Pressure (PIP), set and delivered mean airway pressure (MAP), set and delivered Positive Expiratory Pressure (PEP), inspiratory time, rate, mode of ventilation (trigger or high frequency ventilation (HFV), amplitude (HFV only), Hertz (HFV only), Fractional Inspired Oxygen (FiO2) concentration, tidal and minute volumes, derived parameters of dynamic compliance and resistance, and pressure/volume and pressure/flow loops [27]. Other parameters monitored include blood oxygen saturation (SaO2), blood carbon dioxide concentration (PaCO2), blood

pressure, respiratory rate and temperature.

3.2.2.1 NICU Flow Charts

Presently, the raft of technology that is employed to measure the newborn's physiological parameters, merely displays instantaneous values for inspection by clinical staff. Some machines also provide a trend plot of the data being measured over the preceding few hours so that clinical staff can identify particular developments or drifts in values. Most of the monitors do not store data, this is performed by hand on a large sheet of paper, called the NICU flow chart, which resides beside the crib or bed. Nursing staff record instantaneous monitor values, every 30 or 60 minutes, onto this sheet in a tabular format for daily inspection by a neonatologist who "eyeballs" the sheet and may recognise trends or correlations between parameters which could be indicative of serious complications in the baby's health. It is not uncommon for ill babies to exhibit large variations in their physiological parameters between the 30 to 60 minutes recording times. These variations go unnoticed and could be symptomatic of an impending serious condition or be of importance to the survival, or quality of survival free of disability [27].

Data is recorded without respect to the temporal dimension and temporal inferences are made through visual scanning the paper chart. Detection of subtle trends or level shifts with respect to time are difficult to detect using this method. Correlations between multiple parameters is even more problematical using the "eyeballing" technique. The requirement for improvement is a method for automatically incorporating temporal reasoning into monitoring equipment. Temporal abstraction provides a method for performing automated temporal reasoning on patient data where level shifts and trends can be specified by the clinician beforehand and programmed into monitoring devices to allow for automated detection of such phenomena as opposed to the visual scanning technique. It is also possible to provide for correlations of abstractions across multiple parameters [22, 23].

3.2.2.2 NICU Alarms

Alarms are triggered within most NICU monitoring machines when present thresholds are breached, either in a positive or negative direction. False alarms are common due to probe displacement or patient movement, temperature changes can adversely affect transducer accuracy and bad sensor positioning can result in low amplitude signals venturing outside prescribed ranges. As indicated in section 1.3.1.1, research studies have shown that up to 86% of these alarms can indeed be false [28, 167]. This model for clinical alerting also reflects the absence of temporal analysis in commonly used NICU monitoring equipment. Transient, short lived excursions in data can be ignored if the monitor can reason temporally with incoming data.

3.2.2.3 Newborn Frailty and Early Detection of Disease

Babies admitted to the NICU are extremely fragile and susceptible to many complications during their stay in the ward. Premature babies can be up to 17 weeks early and may only weigh 450gms; they can spend 3 or 4 months in intensive care and have dozens of specific diseases before discharge [26]. There are numerous clinical conditions that are prevalent throughout neonatal intensive care units; the following list is provided by Ward [183]: catheter complications, chronic lung disease, developmental delay, growth reduction, hearing impairment, intraventricular haemorrhage (IVH), necrotising enterocolitis F perforation, neonatal abstinence, nosocomial infections, patent ductus arteriosus, periventricular leukomalacia (PVL), pulmonary barotraumas, respiratory distress syndrome (RDS), retinopathy of prematurity and sepsis.

Of the above conditions, recent research has found that PVL and sepsis have been shown to exhibit early indicators in physiological data [6,7,178] which, at present, goes undetected due to the inadequacies of current monitoring equipment. McIntosh [33], also highlights issues with the late diagnosis of pneumothorax. Pneumothorax is a life-threatening complication of ventilatory assistance in the neonatal unit and has high mortality [184, 185] and morbidity rates [186]. The possibility for early diagnosis

of these serious complications, through analysis of physiological data, shows promise at greatly improving mortality and morbidity rates.

Clinicians at the Nepean NICU are conducting research supporting the hypothesis that while large single deviations of a physiological measure, such as blood pressure, are recognised as serious and require immediate action, frequent smaller deviations maybe also be as significant when the accumulated effect is quantified over a given time interval [187]. Currently, systems do not present accumulated events over time, rather, monitors in an intensive care environment graphically display trends to allow clinical staff to visually identify such accumulated events. Further research is investigating whether multiple aberrant behaviours across a number of physiological streams may have an additive effect and certain combinations thereof may be a sign of critical deterioration that could indicate a high likelihood of death or of brain injury. The requirement for a patient monitoring system to support this research is to be able to abstract multiple data streams into relevant temporal episodes and trigger clinical alerts if necessary.

An example of a relevant case decision of an extremely preterm baby born 16 weeks premature:

> (i) "Detect episodes of 20 seconds or more where blood oxygen saturation (SaO2) is less than 90% and blood pressure is less than 24 mm/Hg."
>
> (ii) "Detect episodes of 60 seconds or more where heart rate is less than 70 bpm and blood pressure is less than 24 mm/Hg."

Each parameter Sa02, blood pressure and heart rate moving into the abnormal range as indicated are indicators separately of critical illness and are correlated with increased likelihood of death. Multiple concurrent parameters in the abnormal range are clinically suspected of indicating greater likelihood of adverse outcome than each parameter taken alone [187].

Figure 3.1: Neonatal Intensive Care Unit crib and monitoring equipment [188].

3.3 NICU Data Environment

The NICU is an environment where technology performs a principal and vital role in the survival of newborn premature babies. Figure 3.1 depicts a typical neonatal intensive care crib surrounded by associated monitoring equipment. The baby is contained inside the humidicrib (or incubator), which can be seen on the left covered by the towel. The crib itself maintains controlled environmental conditions, such as temperature and humidity. On the right is the control panel for the mechanical ventilator; the ventilator pump and associated hoses can be seen leading into the crib. The monitor screen above the ventilator is displaying parameters relating to the ventilation process and the screen to the left of the crib presents the baby's vital parameters such as heart rate, and blood pressure. As previously mentioned, many other measurements may be routinely taken, such as arterial and blood gas measurements which are retrieved using a subcutaneous or even intravenous probe.

Figure 3.2: Neonate showing connections to monitoring equipment. Cuffs on the wrist and ankle are connected to blood pressure monitoring devices, pads on the chest for heart rate monitoring and nasal connection to mechanical ventilator (CPAP) [188].

3.3.1 Frequency and Volume

Figure 3.2 shows the source of physiological data in the NICU; a baby who is connected to numerous machines that measure the vital signs of life. Voluminous quantities of data are generated within the NICU on a daily basis. During experiments in XML data compression using equipment and data collected from Nepean NICU, McGregor et. al. discovered three primary types of physiological data were generated from the monitoring equipment: *numeric*, *slow wave* and *fast wave* data [66].

All data were retrieved from an Agilent Component Monitoring System (CMS), which is a unit accepting various plug-in modules which are connected to the patient monitors, which are in turn, each connected to the baby via sensitive electrical transducers such as wrist cuffs, subcutaneous probes or chest pads. The *numeric* data stream includes measurements of heart rate and respiratory rate. These values are obtained at a sample frequency of 1Hz each (1 value per second). Along with the raw numeric value for the physiological parameter is a time stamp and a signal identification string. For example, the following shows a numeric data stream in tab delimited format:

```
14:05:04   RESP    45
14:05:04   HR     144
14:05:05   RESP    45
14:05:05   HR     143
```

Slow wave data consisted of reading associated with the mechanical ventilator. Sample rate for slow wave data is 128Hz (128 values per second). The following shows 12 data values which are sampled during the same second, obviously a further 116 will be accrued before that particular second ends.

```
14:05:02   RESP -CW  1244  1253  1266  1282
14:05:02   RESP -CW  1302  1326  1354  1386
14:05:02   RESP -CW  1419  1452  1485  1517
```

The volume of slow wave data amounts to approximately 1.3kB per second per baby.

Fast wave data represented numerical values related to channel one from the Electrocardiograph (ECG). Here the sample frequency was very high at 512 Hz, or 512 data readings every second, amounting to a volume of 3.3kB per second per baby when time stamps and signal identifiers are taken into account.

3.3.2 Dimensionality

In the context of the NICU, there are two areas which are relevant when addressing the notion of dimensionality. These are *data* and *monitoring* dimensionality. Dimensionality was also discussed and defined in section 1.2 and is again presented here, although with an emphasis on the case-study environment in terms of data production and associated monitoring processes.

3.3.2.1 Data Dimensionality

The NICU at Nepean hospital houses numerous sick newborns who are being continuously monitored by the above mentioned equipment. The amount of monitoring depends on the original and ongoing examinations and assessments by clinical staff.

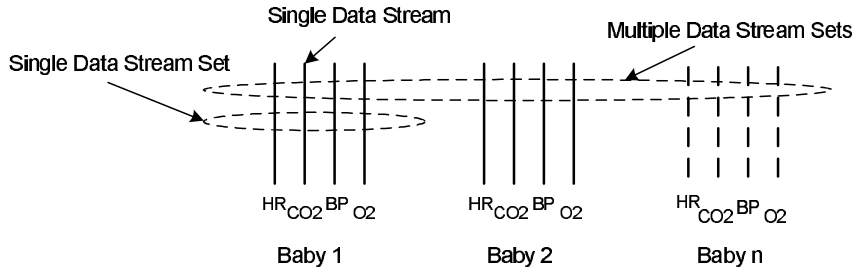

Figure 3.3: The multi-dimensional *data* environment of the NICU.

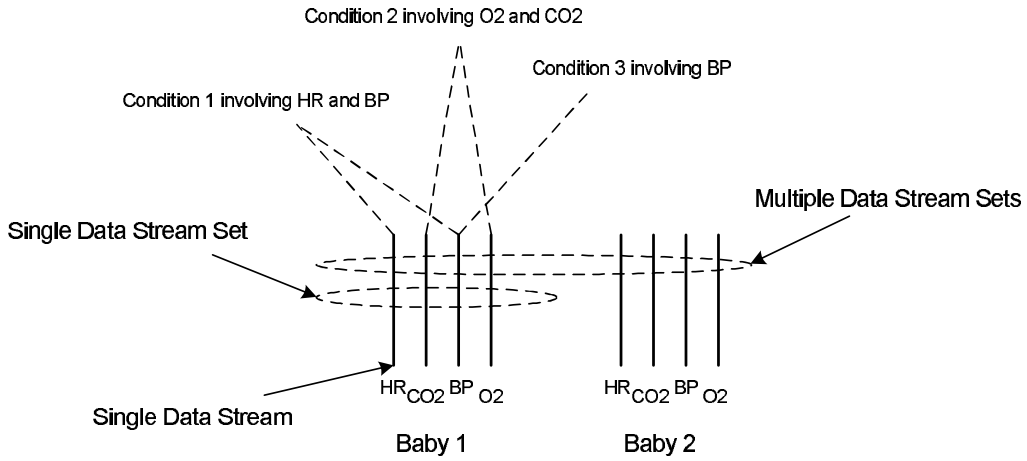

Figure 3.4: The multi-dimensional *monitoring* environment of the NICU. Clinical conditions to be monitored for are shown relating to the data streams involved.

This represents a level of dimensionality in the data environment which is illustrated in figure 3.3 and summarised as follows:

- *Multiple data parameters emanating from each baby*: I define a single data parameter, such as heart rate, as a *single data stream*. I define the multiple data streams emanating from a single baby as a *single data stream set*.
- *Multiple babies within the NICU*: Hence there are multiple data stream sets for analysis and monitoring.

3.3.2.2 Monitoring Dimensionality

The environment, from the point of view of monitoring, has an extra "layer" of dimensionality provided by the clinical conditions that are being monitored within the data. The extra layer is as follows:

- *Multiple possible clinical scenarios to monitor within each data stream-sets*: This was defined in section 1.2 as "Level One Dimensionality". At present, monitoring equipment is not designed to monitor multiple conditions, including interrelationships between individual data parameters, within a single data stream-set.

- *Multiple possible clinical scenarios to monitor within multiple data stream-sets*: This was defined in section 1.2 as "Level Two Dimensionality". The framework proposed by this thesis allows the clinician to enter a rule defining a particular condition and then execute it across *multiple* data stream-sets. There are three example conditions depicted in figure 3.4 which are relevant to a single baby. Obviously, other newborns will have their own, or in some cases the same, set of conditions to be monitored for.

When considered collectively, figures 3.3 and 3.4 present a challenging environment for monitoring equipment. As stated in chapter 2, current monitoring systems do not analyse data to this degree of dimensionality. Chapter 1 section 1.3.1 explains the thesis contribution in this area.

3.4 Clinical Alarming

Most patient monitors in the NICU can be configured with preset alert thresholds. For instance, if the baby's heart rate exceeds 185 bpm then an alarm will sound to alert clinical staff of the situation. As stated in chapter 1 section 1.3.1.1, the so called *threshold alarm* model has its pitfalls. According to Tsien [28], up to 86% of alarms may actually be false and in addition to the ramifications of a false alert, which include wasted time and effort on behalf of nursing staff, the end result is that alarm mechanisms are often switched off. Short lived excursions beyond thresholds often do not indicate clinically significant problems and could be dealt with by a more intelligent alarming model that incorporates a temporal dimension. This can alleviate some of the problems associated with the threshold model. In chapter 4 I

detail the temporal model for the alarm mechanisms employed in this research and in doing so address the problems of transient data 'spikes' associated with the threshold alarming model.

3.5 Pertinent NICU Matters

From the preceding discussion there are a number of NICU matters which are particularly significant for the development of a framework providing online decision support in patient monitoring. These are summarised as follows:

1. Present patient monitoring equipment exhibits several qualities that limit clinical management. These are:

   (a) No ability to reason with data using the temporal dimension. This can alleviate some of the problems associated with the threshold alarming model.

   (b) No capacity to correlate temporal conditions across two or more data streams to support recent medical research hypothesising that multiple aberrant behaviours may have a multiplicative effect.

   (c) No capacity for accruing temporal events with the passage of time to support recent medical research hypothesising that multiple 'out of range' excursions may be of equal importance to one extended interval of time where either single or multiple parameters are out of normal range.

   (d) In order to detect the early onset of potentially life threatening conditions such as sepsis, PVL and pneumothorax, monitoring equipment is required to apply points a, b and c above.

2. Frequency and volume characteristics of the NICU data environment are both very high. Potential frameworks for analysing data in real time must be able to cope with the load imposed.

3. Current NICU practice involving the sporadic recording of the newborn's instantaneous vital parameters onto the NICU flow chart can lead to the neglect of

important transient falls or rises in the baby's data which may happen between recording times.

3.6 Conclusion

In this chapter I have described the NICU environment in reasonably general terms and have deliberately included the human aspect to this research through the introductory discussion which included mortality statistics.

Neonatal intensive care is an extremely 'high tech' setting which generate large quantities of data on an hourly basis. Advances in the last twenty years has enabled equipment to capture increasing amounts of data although the actual analysis and usage of that data is falling behind. Presently there exists a gap between the amount of data produced and the level of analysis which can be applied to the data [60].

As Horn states [60], data-intensive systems have entered the health care domain yet it is the development of *knowledge-based* systems that is required to make use of data and enable better health care. In this chapter I provided a number of areas where monitoring equipment was found to be lacking and even impeding clinical management. One of the largest principal advances in monitoring equipment is to integrate a degree of temporal reasoning which has the effect of instilling the domain expert's knowledge into analysis processes. In this thesis this is achieved through the process of temporal abstraction (TA) and applied via a framework which allows TA within a multi-dimensional and high frequency data environment. Healthcare issues which are dealth with through the application of such a framework are outlined in section 3.5 and were also addressed in chapter 1 section 1.3.

Chapter 4

Temporal Reasoning for Online Temporal Abstraction

Previously, in section 2.3.4, I reviewed a selection of temporal reasoning approaches with the goal of revealing differences, similarities and importantly, deficiencies in regard to the application to high frequency and multi-dimensional environments. That review was situated in the context of a general literature survey exposing areas which are considered as being worthy of exploration or are impeding the evolution of temporal abstraction systems in general. The presentation in this chapter differs in that I discuss and review temporal reasoning approaches with reference to the temporal model which is exhibited in the latter half of the chapter. The topics discussed include temporal reasoning in clinical data analysis, common theories of time relevant to clinical contexts, event monitoring and chronicles and scenario recognition. The discussion introduces major approaches to temporal reasoning which are relevant to this thesis. I discuss germane aspects of the various theories and then present the method utilised in this dissertation for reasoning with clinical data. In doing so, hypothesis 3 in section 1.3.3 stating that, "The Event Calculus can be extended and applied to cater for temporal interpolation and abstraction of real time data", is addressed directly as I introduce an adaption of the Event Calculus which enables real-time temporal reasoning and incremental temporal abstraction across multiple data streams.

Time is a central concept to human reasoning which is integrated into our language and thinking to a degree that somewhat belies both its complexity and value. Reasoning about events that happen in time requires a large amount of knowledge about the world and the ability to utilise that knowledge [189]. Domain knowledge of the world includes a multitude of what we may call 'simple' notions, such as if an object is inside a car then the object moves with the car, a person cannot be in two places at once and a dropped ball will take a finite amount of time to hit the ground. This knowledge has been termed *commonsense knowledge* and allows us to create inferences and make decisions and predictions.

Intrinsic to our commonsense knowledge and associated reasoning ability is the notion of time. Events happen at a particular time and may inflict some degree of change upon the world. The order of events may be important just as the amount of elapsed time between events might be crucial for a correct deduction or supposition to be made. For instance, a person opening a door may be represented as an event, as could the person entering the room. We know, through our commonsense knowledge, that if the door is initially closed, then the *open door* event must precede the *enter room* event otherwise the person will not be in the room. In this case, the elapsed time between the 2 events is of no consequence. Imagine though, if the door had an automatic closing mechanism that requires a time of 10 seconds to close the door after it is opened.

Machines that reason about the world and its inherent dynamism are required to represent, either implicity or explicitly, an ontology of time. As implied above, it is a non-trivial task to endow a machine with adequate knowledge and temporal theory to enable it to reason temporally about events and facts. We humans *represent* temporal scenarios using language defined by a formal syntax; this plus suitable data structures for 'physical representation' are what a machine requires to facilitate a degree of temporal reasoning ability. Numerous researchers have proposed theories and logics for temporal reasoning and there has been much interest in the application and development of such theory in the context of medicine [21, 51, 58, 99, 138, 146],

natural language understanding [122, 148, 190, 191], temporal databases [192, 193], planning [151,194] contracts [195,196], policy and and protocol specification [197–199].

Although medicine has received much attention in the area of temporal reasoning, primarily through the close relationship between diagnosis, prognosis and therapy and the specific timing of clinical events, the field of event monitoring for tracking business processes or network intrusion, is another requiring a representation of time and change. Event monitoring places the emphasis on detection of events, or combinations thereof, where the notion of an *event* is that of an instantaneous change of state and possesses no significant duration. Herein lies the principal disparity between event monitoring and the reasoning required for making a medical diagnosis for example. Clinicians are more interested in the *trajectory* of a physiological parameter or symptom over time rather than instantaneous values or events [17]. Hence we have temporal abstraction; an interval-based and qualitative representation of raw numeric data. Typical event monitoring systems do not permit the detection of events with duration, or intervals of time where some property holds.

As mentioned previously in section 1.4, this thesis extends a pre-existing framework called the Solution Manager Service (SMS) (more specifically, the Event Stream Processor) which was designed for business event monitoring and did not address the hypotheses proposed in this work. I detailed the fundamental differences between the MOTA framework and the SMS in section 1.4 and highlighted areas where this thesis extends the SMS. In the context of this chapter, emphasis is placed upon the mechanisms, and theory thereof, which interact with and process the data rather than system architecture issues which were covered in the previous chapter.

This research employs the Event Calculus (EC) to model the temporal domain of interest and to create Temporal Abstractions (TAs) of patient data streams. Abstractions are performed using time as a fundamental dimension involved in the change of representation space from quantitative to qualitative. TA is well researched but a recent study [23] revealed that TA systems have difficulty coping both with high frequency and multi-dimensional data environments, such as that found within neonatal

intensive care environments.

4.1 Temporal Reasoning in Clinical Data Analysis

In medicine, time plays a fundamental role in decision making where diagnosis, prognosis and therapy all include the temporal dimension [17]. Diseases and clinical conditions proceed along a time line where there are significant events, such as drug administration, giving rise to patient responses which are essentially 'temporal states' or properties that persist for various periods of time. Such temporal states are at the heart of temporal abstraction, which has been widely utilised in medicine to provide time-based 'summaries' of patient data and to aid in decreasing the widening gap between large amounts of quantitative data and qualitative medical knowledge [60]. The temporal abstraction process itself is based on some model of time and uses this to determine the required abstractions. Many TA systems also offer extensive querying capabilities [21, 58, 93], intelligent visualisation of abstracted data [86, 115], and guideline-based care protocols [18, 46, 80, 89, 111]; tasks which also entail reasoning with time-based data and information.

Temporal Abstraction has been a significant area of research in clinical domains primarily because the knowledge associated with diagnosis, prognosis or therapy is largely qualitative in nature. Raw patient data is strictly quantitative and TA attempts to bridge the gap by providing a technique of temporal inference where data can be matched against clinical descriptions. Vital to the task of TA is the ability to reason with time-oriented data so that inferences leading to either diagnosis, prognosis or therapy can be offered. In reference to the work described in this dissertation, the method of temporal reasoning employed must be able to reason with data in a real-time sense, as it arrives into the system, and evaluate, process and classify each data item before the next one arrives.

Temporal representation requirements for systems which model clinical contexts

have been stated as being: time-points, time-intervals, absolute and relative occurrences, different forms of vagueness, temporal causality and other constraints, multiple granularities, compound occurrences and temporal abduction, deduction and induction [200–202]. However, the domain over which the temporal reasoning system functions is of utmost importance in determining the reasoning and representation requirements.

4.1.1 Clinical Environments and Temporal Reasoning

Many temporal reasoning systems are employed in the field of historical temporal databases where it is necessary to provide a model that expresses what is true and when it is true. In order to maintain the validity of truth propositions within the database, updates must cause a revision of previous truths and an explicit representation of temporal facts is required [203]. The requirement for reasoning with temporal database data combined with the desire for a *general* and re-useable theory of time, as opposed to re-implementation of disparate and ad-hoc application-driven program functions, prompted the development of two significant theories of time, those of Kowalski and Sergot's event calculus [148] and Dean and McDermott's 'time map manager' [152]. Such theories have been shown to be lacking when it comes to problem solving in medical domains [202] and yet have also proven useful when utilised in a particular environment or context [51,58]. I re-iterate that the area of application dictates the exact reasoning requirements and general theories of time can be extended or *massaged*, to enable satisfactory useability within a medical context while maintaining the soundness of a formally defined and well researched principle. Many medical temporal reasoning frameworks are testimony to this [21, 51, 58, 93, 110, 138, 201].

Medical expert systems are an extension of the temporal database paradigm where the reasoning capability of the system acquires an equal level of importance to the data. For example sound diagnosis requires that the order of events and facts in the patient's history are known precisely, and often at a higher level than that of the raw data. The process of temporal abstraction can provide 'views' of the data at a higher

level of abstraction which can 'match' qualitative clinical descriptions or queries on the data. In this scenario the ability to create TAs is required, as is the capability to update or maintain historical abstraction propositions. For instance, if a test result arrives some time after it was carried out it may alter the view of the past and if so, TAs will require updating. Maintaining historical truths is sometimes referred to as *truth maintenance* [117] and is a task that requires temporal reasoning, as does the process of TA. Often, these types of medical expert systems will be primarily employed in answering queries regarding the historical nature of clinical episodes. Spokoiny and Shahar [117] provide an example from the domain of oncology, "locate all patients who have had, within the past 9 months, more than two episodes of grade II or higher bone marrow toxicity (which is an abstraction defined by the clinical trial's protocol), each lasting at least 2 weeks."

Another area of application for temporal reasoning is that of real-time patient monitoring, as performed within intensive care units. These systems require the ability to respond in real-time to incoming data and, if necessary, to create clinically relevant abstractions that can be unified with qualitative clinical knowledge. This necessitates a direct mapping of representational time to domain clock time and also a method of efficient computation so that reasoning and processing 'keeps pace' with data sample frequency.

In a patient monitoring scenario data is accrued in a temporally ordered sequence and abstractions are constructed incrementally. Each new data point may either validate or invalidate the current abstraction hence it is not necessary to include the ability to revise past conclusions; as is common in a temporal database. Multiple time granularities are generally not required (as specified by Keravnou [146]), as derived abstractions are based solely on clock time and time line granularity is that of the data sample rate; which in the case-study area of neonatal intensive care is 1Hz. Exceptions to this could be when abstractions are interrelated to form a *compound occurrence*, or temporal pattern, conforming to a new temporal context.

4.2 Common Temporal Theories and their Application to Clinical Temporal Reasoning Tasks

The temporal theories I discuss in this section were born of the assumption that time exists independently of change. Galton discusses this assumption and distinguishes two separate models of time; an *independent-time model* and a *dependent-time model* [203]. A dependent-time model treats time as an entity of itself which can be described in a temporal framework upon which events can be assigned relevant times. The order of events is then established by using the precedence relation on the time values themselves. The independent-time model has a weaker coupling between events and time hence *change* is usually inferred rather than explicitly represented in the model. The subsequent discussion will concentrate on the dependent-time model.

Dependent-time models proceed initially from a set of events, and the relations between them, which describe the modelled world. Event times are then established from the domain event model rather than from time directly. The nature of medical reasoning is much closer to a dependent-time model as it is the events, and temporal states that are caused by events, which dominate the reasoning process. Also, sometimes it is not known at what time an event either has occurred, or will occur.

4.2.1 The Event Calculus

One of the earlier temporal reasoning formalisms, the Event Calculus (EC) [148] was initially developed for reasoning with data in temporal databases and natural language understanding. The EC is essentially a model for time and change with the underlying principle being that events happen which change the state of the world in some way and the resulting states persist unless another event terminates them; this complying with the commonsense law of inertia. The usual terminology incorporates the term *fluent* to indicate a time-varying property (state) of the world. Time is not represented explicitly in the EC, although it is possible to associate an event with a particular time point and a fluent has a specific truth value over an interval of time.

The basic EC uses a set of predicates as follows: *Happens*, *Initiates*, *Terminates*, *HoldsAt* and *Clipped*. Fluents are initiated thus:

$$Initiates(e, f, t) \Leftarrow Happens(e, t) \ \wedge HoldsAt(f_1, t) \wedge \cdots \wedge HoldsAt(f_N, t)$$

The above is interpreted thus: If event e happens at time t and fluent f_1 holds at time t and fluent f_N holds at time t, then fluent f is initiated at time t. Similarly axioms can be defined to *terminate* the fluent at a later time point. Events can occur concurrently in the EC and relationships are defined with respect to time points although fluents occupy intervals of time between 2 time points. *Linear time* is used where time is considered to be a line, rather than a tree, as in the *branching time* of the Situational Calculus [204].

Causal constraints are an important requirement for temporal reasoning in that changes in the world are brought about by occurrences of some description. The event calculus models causality using the notion of fluents which give rise to *trigger* events. If a fluent's value or state is dependent on the compound value of other fluents then an appropriate *trigger* event is generated by the influencing fluents to ensure the initiation or termination of the dependent fluent. The trigger events described here are not the same as the *ghost* events, as discussed by Chittaro and Dojat [51]. They describe *ghost* events as those that are not directly recognisable in the domain of interest and whose sole purpose is to cause ether the initiation or termination of a fluent. The same authors utilised the EC within the domain of ventilator management and found that certain properties in their domain had no definable terminating event but were rather self terminating; for instance, clinical symptoms or disease states. Hence they introduced *ghost* events to give such fluents their desired value.

Farrell and Sergot [196] (Sergot was one of the original developers of the event calculus), proposed the notion of *timer events* for describing the expiration of time relating to fulfilment of an obligation within a contract. This enabled their xml-based reasoner to realise when the time representing the maximum time for fulfilment of an obligation had expired. In a sense, these are also *ghost* events in that they have

no natural occurrence in the application domain, although with the introduction of the timer event, it is possible to describe *delays* and also model the termination of real-life situations where some properties terminate based on the time they have been active.

Chittaro and Dojat [51] noted that there is no way of modelling delayed effects in the EC and that it is only possible to model the immediate effect of events. Miller and Shanahan [205] demonstrate that it is in fact possible to model delays using simple *Happens* clauses. They provide an example where setting an alarm clock causes it to ring 8 hours later and, as stated by the authors, is a simplistic way to represent delayed effects which is not without problems.

$$Happens(StartRing, t + 8) \leftarrow Happens(Set, t)$$
$$Initiates(StartRing, Ringing, t)$$

They offer a "more flexible" approach which incorporates both the notion of a *trajectory* and a new sort P of *parameters* that is not subject to default persistence. Along with the new sort P, the function $ValueAt$ is added which may be used to represent the time remaining before the alarm rings: $ValueAt(CountDown, 6) = 2$, where *Countdown* is a parameter not subject to default persistence. Finally, the *Trajectory* predicate allows the association of fluents to parameters and enables representation of continuous change, delayed effects and causality. For example:

$$Trajectory(SwitchedOn, t_1, CountDown, t_2, 8 - t_2)$$

can be translated as, "if fluent *SwitchedOn* holds at time t_1 and continues to hold until time t_2, then parameter *CountDown* will have the value $8 - t_2$." When the *CountDown* parameter reaches zero, the alarm sounds:

$$Happens(StartRing, t) \leftarrow ValueAt(CountDown, t) = 0$$

The final axioms state that if the timer is not already switched on, then switching on the alarm activates the timer :

$$Initiates(Set, SwitchedOn, t) \leftarrow \neg HoldsAt(SwitchedOn, t)$$

The *Ringing* event terminates the timer,

$$Terminates(StartRing, SwitchedOn, t)$$

and when the timer is terminated, *CountDown* is fixed at a value of 8, the alarm is initially switched off and then it is switched on at time $t = 2$.

$$\begin{aligned} ValueAt(CountDown, t) = 8 \leftarrow & \neg HoldsAt(SwitchedOn, t) \\ & \neg HoldsAt(SwitchedOn, 0) \\ & Happens(Set, 2) \end{aligned}$$

Previous works by Shanahan defined a model for continuous change without the introduction of the sort P [160] and refined the model further in [206] through the use of the *Releases* clause. *Releases* effectively releases a fluent from the notion of default persistence and the fluent is free to fluctuate until it is ether initiated or terminated by another event. $Releases(e, f, t)$ states that if an event e happens at time t, then fluent f is released from inertia.

Providing a temporal model for the domain of online temporal abstraction of time-stamped data streams imposes the requirement for a counting mechanism not dissimilar to the *CountDown* fluent just described. Typical abstractions in the case study domain of neonatal intensive care will specify an interval of time for which the abstraction is valid, hence the model must be able to *track* time as data is accrued and abstractions constructed. The process of temporal interpolation, a necessary component of temporal abstraction which is described in section 4.4.1, also entails the capacity for counting the passage of time.

Whilst it is true, as Keravnou and Shahar state, the *pure* or *simple* event calculus cannot adequately model continuous change [202], the extensions offered by Shanahan and Miller [160, 205, 206] enable the EC to be used for real time temporal abstraction. The incorporation of these additions to the EC into a temporal model for online TA is described in section 4.6.1.1 and directly addresses hypothesis 3 from section 1.3.3.

4.2.2 Allen's Interval Algebra

Allen's Interval Algebra (IA) [122] employs the time interval as the temporal primitive and defines 13 possible relations between any 2 intervals. These being *before* ($<$), *equal* ($=$), *meets* (m), *overlaps* (o), *during* (d), *starts* (s) and *finishes* (f). Each relation, with the exception of *equal*, has an inverse: ($>$, $<$, mi, oi, di, si, fi).

Allen's work does not acknowledge the existence of the time-point and the mapping of time to such points on the real number line. The reasoning behind this relates to the fact that it is always possible to decompose seemingly instantaneous events into more detailed explanations. The Interval Algebra allows three types of propositions to hold over an interval, these being static properties and dynamic events and processes. These can be defined using the predicates *holds*, *occurring* and *occur*. Prior to introducing such axioms it is useful to define a predicate which summarises how one interval can be wholly contained within another. The *in* predicate is defined as follows:

$$in(t_1, t_2) \leftrightarrow (during(t_1, t_2) \vee starts(t_1, t_2) \vee finishes(t_1, t_2))$$

where *during*, *starts* and *finishes* are further defined thus:

- $during(t_1, t_2)$: time interval t_1 is fully contained within t_2.
- $starts(t_1, t_2)$: time interval t_1 shares the same start time as interval t_2, but ends before t_2 ends.
- $finishes(t_1, t_2$: time interval t_1 shares the same end as t_2), but begins after t_2 begins.

We can now observe the relationship between an interval (T) and its sub-intervals (t).

$$holds(p, t) \leftrightarrow (\forall t\, in(t, T) \rightarrow holds(p, t))$$

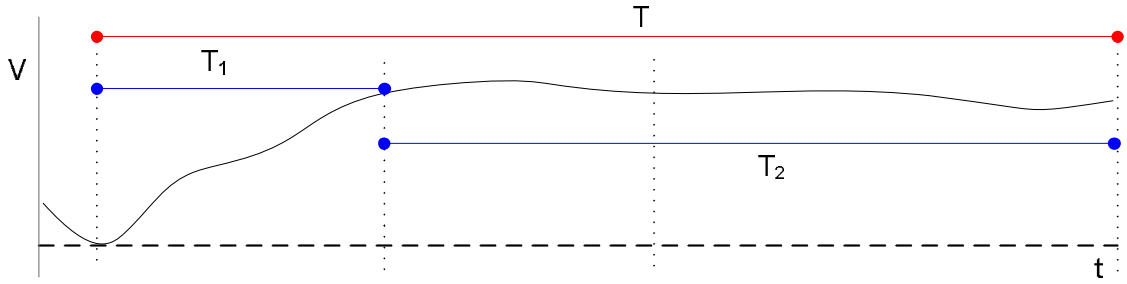

Figure 4.1: A complex temporal abstraction. Abstraction T_1 is an increasing trend abstraction which *MEETS* the stationary abstraction T_2 to form the complex abstraction T.

The above illustrates if a property holds over an interval T, then it also holds over all sub-intervals of T.

$$occur(e,t) \wedge in(t',t) \rightarrow \neg occur(e,t')$$

As defined by the previous axiom, an event occurs over a particular interval t as long as there is no sub-interval of t at which the event occurred. The *occurring* predicate defines that if a process occurs over an interval t, then it also occurs over at least one sub-interval of t.

$$occurring(p,t) \rightarrow \exists t' in(t',t) \wedge occurring(p,t')$$

Interval algebra, or at least the interval operators and their associated axioms, have been used widely in the medical context [20,21,93,138,207] in relation to the process of temporal abstraction, especially with regard to *complex* temporal abstractions where a sequential relationship may exist between two or more propositions of interest. This is shown in figure 4.1. These abstractions are built on abstractions themselves rather than on the raw data. For instance, Larizza [208], defines the complex abstraction *increase(V) MEETS stationery (V)* where V is the parameter of interest and *increase* and *stationery* are simple temporal abstractions.

One of the more challenging temporal reasoning tasks is solving temporal constraint problems which involves searching the dataset for the existence of variables that satisfy a given set of temporal constraints. Typically, this is performed using

a temporal constraint network and algorithms for constraint satisfaction [209–213]. In the interval algebra, constraints are specified by the relative location of paired intervals and Vilain et. al. showed that determining the closure of such assertions is NP-complete [153, 154] and proposed a less expressive point-based algebra which provided a computationally sound solution in polynomial time. The nature of real-time temporal abstraction does not require reasoning backward over time and data arrives in a temporal order so qualitative constraints are evaluated at the time they are sampled by the system. *Computational* constraints are imposed primarily by data sample frequency enforcing the need for the system to be able to "keep pace" with the sample rate and perform its reasoning before the next data sample arrives.

The very nature of a temporal abstraction per se, being a proposition which holds over an interval, implies a certain adherence to the philosophy of Allen's calculus, which places the interval above all else. For instance, Shahar's method of temporal reasoning [21, 76] employs the time point as its temporal primitive although propositions are only ever interpreted over intervals, as is the case with most temporal abstraction frameworks.

4.2.3 Shoham's Temporal Logic

The semantics of Allen's, and also McDermott's [151][1] , temporal logic was questioned by Shoham [150] who proposed that the meaning of propositions in these works was ambiguous and unclear. He recommended ten propositional types in order to provide clear semantics for when and how propositions hold.

Shoham defined the propositional type *downward-hereditary* as meaning that if a proposition holds over an interval then it holds over all of its sub-intervals and *upward-hereditary*, where if a proposition holds for all sub-intervals of an interval then it also holds for the interval itself. He noted that propositions made in Allen's interval algebra were one of these two types. The remaining eight propositional types

[1]McDermott's temporal logic uses the time point as a primitive instead of the time interval which Allen [122] uses; hence facts are interpreted over points.

add a higher level of expressivity than is possible in either Allen's or McDermott's logics and carry implications for the process of temporal interpolation, as required by temporal abstraction tasks.

4.2.4 Shahar's Knowledge-Based Temporal Abstraction

The RÉSUMÉ system for knowledge-based temporal abstraction of stored data [21,76] was introduced in chapter 2 and represents a pivotal work in the developmental evolution of such systems. Shahar's method for temporal reasoning includes influences from both McDermott and Allen while extending the propositional types defined by Shoham.

Shahar employed the time point as a primitive, although propositions can only be interpreted over intervals. In the application domain of medicine, propositions are qualitative descriptions of parameter values within a certain clinical context. Different propositions can have various temporal inference properties, again depending upon the clinical context. A domain knowledge base includes this temporal knowledge and is used to create inferences over medical data. Relevant to my own temporal model is Shahar's method for dealing with temporal "gaps" within abstractions and his use of Shoham's propositional type *concatenable* to enable temporal interpolation.

The technique of temporal interpolation allows concatenation of contextually similar, temporally disjoint intervals [59]. Shahar [59] provides a solution to the temporal interpolation task using what is termed *primary* and *secondary* temporal interpolation. Primary interpolation aims to join two parameter points into an abstraction interval and secondary interpolation receives two intervals and returns an abstraction *super* interval. My use of the event calculus includes a default notion of persistence. That is properties (or abstractions), once initiated, are assumed to hold until an event terminates them hence we do not require a *primary* temporal interpolation method. My method is similar to that of Shahar's secondary temporal interpolation method in that the inclusion of domain knowledge can allow specification of a maximum allowable gap. As long as the temporal distance between two intervals belonging to an

abstraction is less than the allowable gap, then the two intervals are concatenated to form a single continuous TA.

In order to determine whether a pair of intervals is concatenable, Shahar utilises a *maximal gap function* [59, 214] which incorporates a measure of the rate of change of the parameter before and after the time gap and returns the length of the maximum temporal gap. The design goals of RÉSUMÉ and the case-study domain of neonatal intensive care create very dissimilar requirements. I do not store temporal domain knowledge nor do I apply a maximal gap function, instead, the maximum gap is specified at run time by the clinician who is responsible for inputting the initial abstraction rule. Temporal interpolation is then resolved through application of the event calculus-based temporal model.

As revealed in chapter 2, many TA-based systems operate on either stored data or reason backward over data in a semi real-time sense. The model for TA proposed in this chapter performs incremental TA on data as it arrives to the system and in this way is similar to event monitoring systems which detect events in real time and chronicle-based systems that propagate constraints in chronicles on the fly.

4.3 Event Monitoring

The field of event monitoring covers a vast range of differing application contexts including business transaction monitoring [68,215], network monitoring and intrusion detection [43, 216, 217], stock monitoring [218] and process control [219]. It is applicable wherever an appropriate response is required from the system upon detection of a particular event, or sequence of events and is therefore a candidate for review and discussion within the context of this dissertation.

Within the event monitoring domain, an event is commonly defined as a happening of interest which is atomic and bound to a specific point in time [220]. There are generally two types of events: *simple* and *complex*. A simple event is a single state transition, as just described, whereas a complex event is constructed by combining

multiple simple events which may occur sequentially, concurrently or a combination thereof depending on the expressivity of the language used [221–223]. A linearly ordered discrete point-based time line is the model of time used in such frameworks and many implement an event algebra, for describing when events occur and also for representing complex events [218, 224]. Typically *disjunction*, *conjunction*, *negation* and *sequence* operators are employed. A *disjunction* of two events mean either or both of the events has occurred where *conjunction* refers to the fact that both events have occurred, although possibly not simultaneously. *Negation* occurs when there is an instance of one event during which there is no instance of the other. Sequence is used to denote that a particular event has occurred after another [222].

Event detection is typically either performed in a real-time sense, as data arrives to the system, or within an active database. Early event monitoring systems were primarily based around an active database employing Event-Condition-Action (ECA) rules. The semantics being that if an event has happened (event) within a particular context (condition), then the action is carried out [225]. SAMOS [226, 226, 227], Snoop [228, 229], Ode [220, 230, 231], Liu et. al. [232] and Pietzuch et. al [221] are examples of systems employing the ECA paradigm and although they are similar in this regard, they utilise differing detection mechanisms.

The Ode database system incorporates regular expression-based definitions for event representation and finite state automata (FSA) are used for detection. Complex event detection requires successive, and possible concurrent, detection of primitive events until the terminating event is identified. The inclusion of event parameters, such as object or transaction identifiers, is not possible using such an algebra and Ode overcomes this through provision of data structures for recording the time of primitive event occurrences, called the event history, leading to the formation of the complex event. An implication of the language used in Ode is that primitive events cannot occur simultaneously [223].

SAMOS utilises coloured Petri Nets and an object-oriented database for detection of complex events, and although able to represent concurrent events, they can become

complex even for relatively simple expressions [233]. Event parameters are included in event expressions and, interestingly, intervals are also incorporated in the event model. An interval is defined by a *start* time and an *end* time, in a way similar to that described by Allen [122], although SAMOS only includes one of Allen's 13 possible relations, *overlaps*, and one additional operator; *extends*.

In SNOOP, event trees are to represent complex events where there is a single tree for each composite event and multiple trees are then combined into an event graph to detect a *set* of composite events. Each node of the tree represents an algebraic operator in the event language and detection propagates upward via each node, which is an individual primitive event detector. Although Chakravarty et. al [228] formalised SNOOP by clarifying the semantics of each operator, it has been said that it is difficult to formalise and reason about the behaviour of the tree due to the Turing nature of the nodes [233] and semantics can be ambiguous [223]. Similar to Ode, a total event order is assumed making simultaneous events impossible.

4.3.1 Interval-based Event Detection

One feature in common to the previously described event detection systems is that events are assumed to be instantaneous and associated with a single time point. The same model applies to complex events; the time of the last occurring instance of a primitive event, which constitutes part of a composite event, is recorded as the occurrence time of the composite event. In other words, time relating to other (previous) composite event components is not taken into account. The reason for this is primarily due to the requirements placed on event detection systems; their principal motivation is the detection of events rather than reasoning with temporal information relating to event occurrences. In the domain of medicine, the temporal aspect is of utmost importance. Clinicians need to know at what time a particular combination of events was *initiated* and *terminated*. Also, Galton and Augusto [234] note that so called detection-based semantics [235], can result in ambiguous semantics for some event compositions.

This ambiguity is demonstrated by Carlson, who uses the following complex event description [222]. The example uses the sequence (;) operator denoting event $A; B$ occurs whenever A occurs, and then B occurs. Using the single point method, the event description $A; (B; C)$, will be detected if B occurs first, followed by A and then C. This combination of event occurrences causes detection of $B; C$ hence, as A happens before $B; C$, the complex event $A; (B; C)$ is detected.

The solution to this problem is to associate the occurrence time of events with intervals rather than time points. Using the previous example, if $A; B$ was defined to only occur if the interval time of A does not overlap the interval of B, then the sequence $A; (B; C)$ would not be detected as interval A happens within the interval of $B; C$.

The interval-based method of event definition has become a popular topic within the event monitoring community with a number of researchers developing such methods [222, 233, 236, 237], or adapting previous efforts to comply with the interval paradigm [235]. Of course, interval-based models of time and events are commonplace in the knowledge representation community, as revealed in the earlier part of this chapter. The work described in this chapter is applied within the broader context of artificial intelligence and knowledge representation although, the proposal of Galton and Augusto [234] to attempt to bring together the fields of knowledge representation and event detection in the database community, is in accord with my event calculus method for real-time temporal abstraction. I believe the temporal model for events and fluents provided herein is one which could be applied within an event monitoring environment and would enable the definition and detection of interval-based, domain dependent phenomena (temporal abstractions) in that context.

4.3.2 Chronicles

Chronicles provide a method for modeling dynamic systems and include the temporal dimension as a fundamental element [238]. They are sets of temporally constrained events representing the evolution of a complex scenario [239] and have been applied in

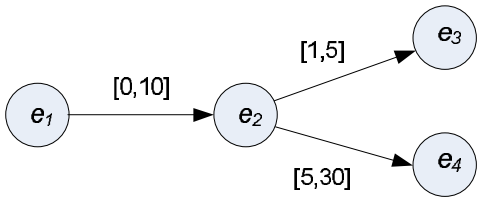

Figure 4.2: A chronicle.

a wide variety of areas such as clinical real-time monitoring [15, 128, 240], gas turbine monitoring [241] and alarm analysis in telecommunications networks [242] and voltage distribution networks [243].

Time is represented using a point-based algebra and is considered as being a linearly ordered set of instants whose resolution is sufficient for the environment being monitored. Figure 4.2 shows a typical chronicle where the set of events e_1 to e_4 are bounded by timing constraints pictured inside square brackets. For example, event e_2 happens at a minimum time of zero time units after e_1 and a maximum time of 10 time units after e_1. Similarly, e_2 precedes e_3 by between 1 and 5 time units and so on. This leads to the following chronicle definition:

$$\begin{aligned} event(e_1, t_1) \wedge event(e_2, t_2) \wedge event(e_3, t_3) \wedge event(e_4, t_4) \wedge \\ t_2 - t_1 \in [0, 10] \wedge t_3 - t_2 \in [1, 5] \wedge t_4 - t_2 \in [5, 30] \wedge \\ t_1 < t_2 < t_3 \leq t_4 \end{aligned}$$

Chronicle detection has been performed in a variety of ways and numerous algorithms proposed for processing incoming events and matching them with the time constrained chronicle definition. Generally, events are received by the system and incrementally matched against the model providing a partial instance of the chronicle model. When a complete match is detected, the chronicle is said to be recognised and associated actions can then be executed; such as triggering an alarm or sending a message to a controlling agent.

The use of *scenarios* and *sessions* for chronicle detection has been one of the more popular detection mechanisms. A *scenario* models the expected temporal evolution

of the process whereas a *session* models the real evolution of a process and is derived from the monitored domain [240]. The method used in [15, 240] represents both session and scenario as temporal constraint networks and utilises standard constraint network algorithms, such as the path consistency algorithm [210], for matching the session against the scenario. Interestingly, the same authors incorporate temporal abstraction processes into the chronicle models through the definition of *states*, which are essentially TAs and are derived from a separate TA module. This approach exploits the ability of chronicles to model intervals and as such, demonstrates the suitability of chronicles for representing the interval-based temporal propositions described in [96, 150]. The work described in [15] was integrated into the *Déjà Vu* system and evaluated in the domain of ventilator management. It showed promise for the use of such techniques in the area of real-time patient monitoring.

Chronicles, as opposed to event monitoring, offer the ability for temporal reasoning and have been readily adopted by the knowledge representation community. Ghallab [127] adopts the common method of using the reified predicates *event* and *hold* to express both change and persistence. This provides a propositional reified logic, as in [150], for asserting the truth value of a set of multi-valued domain attributes. The clause $hold(p: v, (t, t^{'}))$ expresses the persistence value v of a domain attribute p during the interval $[t, t^{'}]$. The event clause $event(p : (v_1, v_2), t)$ states that the value v of an event changes from v_1 to v_2 at time t and is defined through the predicate hold thus:

$$event(p : (v_1, v_2), t) \equiv \exists \tau < t < \tau^{'} \mid hold(p\colon v_1, (\tau, t) \wedge hold(p\colon v_2, (t, \tau^{'})) \wedge v_1 \neq v_2$$

A conjunction of event predicates forms a chronicle model. This logic is similar to that used in the Interval Calculus [122] and also the Event Calculus [148]. For example, an event calculus fluent is initiated thus:

$$Initiates(e, f, t) \wedge happens(e, t) \leftarrow holds(f, t)$$

Ghallab also demonstrates the ability to make predictions through the use of partially recognised chronicle instances. A partially recognise chronicle instance represents the complete temporal evolution of a chronicle and as such, at each instant throughout the detection process it is possible to predict future events given the expected evolution. Similarly, in the Event Calculus, one can predict future conditions given the notion of default persistence and a set of *holds* clauses combined with a set of events.

The dissimilarity between temporal reasoning formalisms such as the Event Calculus and chronicles lies in the fact that the event calculus is mainly concerned with the inferences that can be made from events that have occurred in the past [234]. It has also been noted that the Event Calculus does not address the recognition of evolving temporal states on the fly [127] but has rather been used to reason backwards over stored data in order to perform its reasoning. The method proposed in this chapter remedies these issues through provision of an Event Calculus-based model which performs incremental deduction on incoming data to ascertain the validity of current temporal abstractions.

The EC-based model for TA combines the processing model of a typical chronicle recognition system with the temporal awareness of a well-known and formal theory of time. TAs are detected on the fly and existing abstractions are either in incremented or discarded using an Event Calculus model for online TA. The detection mechanism is rule-based as opposed to the finite state automata or active database triggers of event detection environments and the time constraint networks of chronicle recognition systems.

4.4 Temporal Abstraction Classes

To clarify some of the more essential terms used in relation to temporal abstraction and provide an introduction to the EC-based method for online TA, I define the three primary TA types. Common classes of TAs are *stationary*, *state* and *trend* [21,36,58,208] and although exact definitions may vary slightly, these are well accepted

and known throughout the artificial intelligence in medicine community.

- *Stationary:* The value of a variable is below an upper threshold and above a lower threshold value, ie., within a given range.
- *State:* The value of a variable is classified into a pre-defined range such as *high, low, normal*. A *level shift* is a type of *state* TA where values are classified as being either above or below a particular threshold.
  - *Positive level shift:* values are greater than a threshold value.
  - *Negative level shift:* values are less than a threshold value.
- *Trend:* The value of a variable either increases or decreases over time at a rate greater than a specified one.

For example, a negative level shift TA may be described by the following TA specification *TA1*, where a clinician is interested in discovering periods of time where a parameter is under a particular threshold value.

> *TA1:* "Detect periods of time of 20 seconds or more where SaO2 is less than 90%. ".

4.4.1 Temporal Interpolation

The process of TA must incorporate a mechanism for concatenating contextually similar concepts which belong to temporally disjoint intervals in order to create longer intervals [21, 59]. This allows the specification of a maximum allowable gap as illustrated in the following TA specification *TA2*:

> *TA2:* "Detect periods of time of 20 seconds or more where SaO2 is less than 90%. Oxygen saturation may exceed 90% for no more than a total of 5 seconds within the 20 second interval".

Assuming $t_6 - t_1$ in Figure 4.3 represents an elapsed time of 20 seconds, as long as the time $5 - (t_5 - t_4) - (t_3 - t_2)$ is greater than zero then intervals $[t_1, t_2]$, $[t_3, t_4]$ and

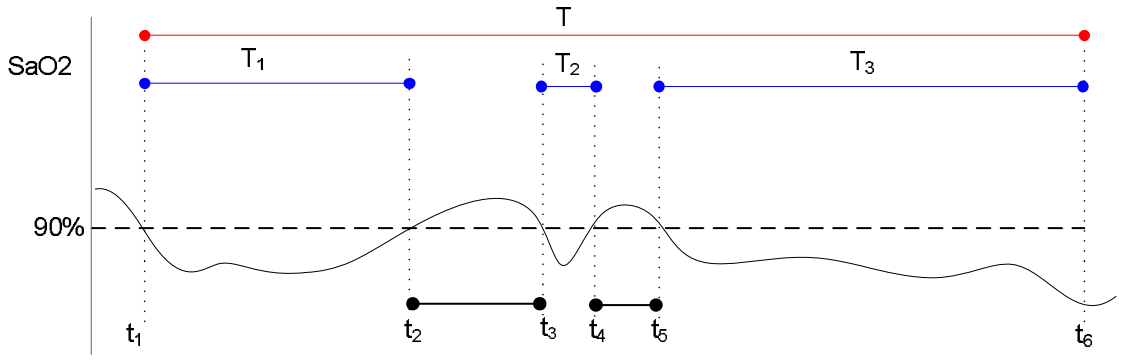

Figure 4.3: A time-series of blood oxygen (SaO2) samples subjected to temporal abstraction. Time intervals where SaO2 is below a threshold (90%) form a *negative level shift* temporal abstraction T. Time intervals where the abstraction is invalid (SaO2 $>$ 90%) can be ignored if a *maximum allowable gap* is defined and out of range excursions are measured ($T_{gap} = (t_2 + t_3) + (t_4 + t_5)$). Disjoint intervals (T_1, T_2, T_3) can then be concatenated to form the longer interval T if the maximum allowable gap is less than T_{gap}.

$[t_5, t_6]$ are joined resulting in a valid temporal abstraction for rule *TA2*. Of course, if the total time spent above 90% is greater than 5 seconds then the disjoint intervals are not joined and the TA cannot be said to exist within the data. There is also the possibility that the data samples never exceed 90%, in which case the abstraction should be terminated based solely on the time $t_6 - t_1$ being greater than or equal to 20 seconds.

My aim is the design of a framework that is flexible enough to enable sound temporal reasoning over a range of differing types of TAs on real time data. The example above abstracts time-stamped blood oxygen (SaO2) samples but my theory enables temporal abstraction of any stream of time-stamped values, whether they originate directly from physiological transducers or are derived from an intermediate agent that calculates other numerical quantities, such as mean value or Area Under the Curve (AUC). Enabling this TA process across multiple data streams (see Hypothesis 1 section 1.3.3) is also a primary goal within this research project and one that is explained in section 4.6.1.2.

4.5 Temporal Abstraction of Multiple Online Data Streams

Here I detail the Event Calculus-based temporal model used to perform online TA to a single patient's data stream set (multiple parameters, single data stream set). This has been defined in section 1.2 as "Level One Dimensionality". One of the primary contributions of this research is proposed in hypothesis 3 which states that the Event Calculus can be extended and applied to cater for temporal interpolation and abstraction of real-time data.

As discussed in section 4.2.1, the EC was not originally designed to enable temporal reasoning with data in a real-time sense. That is to say, there are no inherent mechanisms for performing temporal abstraction in an incremental way, as there are with chronicles for instance. This research proposes an adaption of the EC which caters for this kind of temporal abstraction being based on work previously done by Shanahan [160, 206] where he modelled continuous change in the EC.

The model has been implemented using a forward chaining rule engine environment which is embedded inside multiple Patient Monitor Modules (PMMs), each assigned the task of temporally abstracting a single patient's data and themselves forming part of the larger research framework, named the Multi-dimensional Online Temporal Abstraction (MOTA) Framework. The MOTA framework permits TA and intelligent alerting within a multi-patient context (multiple data-stream sets) [22,244]. Moreover, the ESP itself represents an element contained by the *e-Baby Project* [4,26], which aims to provide local and remote intelligent decision support to clinicians within the domain of neonatal intensive care.

4.5.1 The Temporal Domain - An Example

In section 4.2.4 I discussed the process of temporal interpolation with regard to the temporal reasoning involved. In this section, I introduce an example scenario taken from the case-study domain of neonatal intensive care, depicting a clinical situation

where physiological data streams are abstracted in real-time. It involves an aspect of multi-dimensionality since there are two data streams to be processed. The TA required is the determination of a period of time where SaO2 samples are less than 90% *and* blood pressure is less than 24 mm/Hg as specified in specification TA3.

> *TA3:* "Detect periods of time of 20 seconds or more where SaO2 is less than 90% and blood pressure is less than 24 millimeters of mercury (mm/Hg). SaO2 may exceed 90% for no more than a total of 5 seconds within the 20 second interval".

The TA specification contains two conjunctive parts, each associated with a separate data stream for a particular patient. My approach decomposes the *parent* specification *TA3* into its *child* constituents $TA3_a$ and $TA3_b$ and abstracts the data accordingly.

> $TA3_a$: Detect periods of time of 20 seconds or more where SaO2 is less than 90%. SaO2 exceed 90% for no more than a total of 5 seconds within the 20 second interval.
>
> $TA3_b$: Detect periods of time of 20 seconds or more where blood pressure is less than 24 mm/Hg.
>
> $Warning \leftarrow TA3_a \wedge TA3_b$

The following possible scenarios are noted:

- For the temporal abstraction specified by *TA3* to be valid then both parts $TA3_a$ and $TA3_b$ must be valid for the stated interval of 20 seconds.
- Assuming both $TA3_a$ and $TA3_b$ hold, then if either or both $TA3_a$ or $TA3_b$ contain 'out of range' excursions in the data above their respective threshold values, temporal interpolation mechanisms must keep track of the time spent 'out of range'. If either or both of these 'gap timers' exceed their pre-defined maximum temporal gap (5 seconds for $TA3_a$), the abstraction *TA3* is no longer valid and should be terminated.

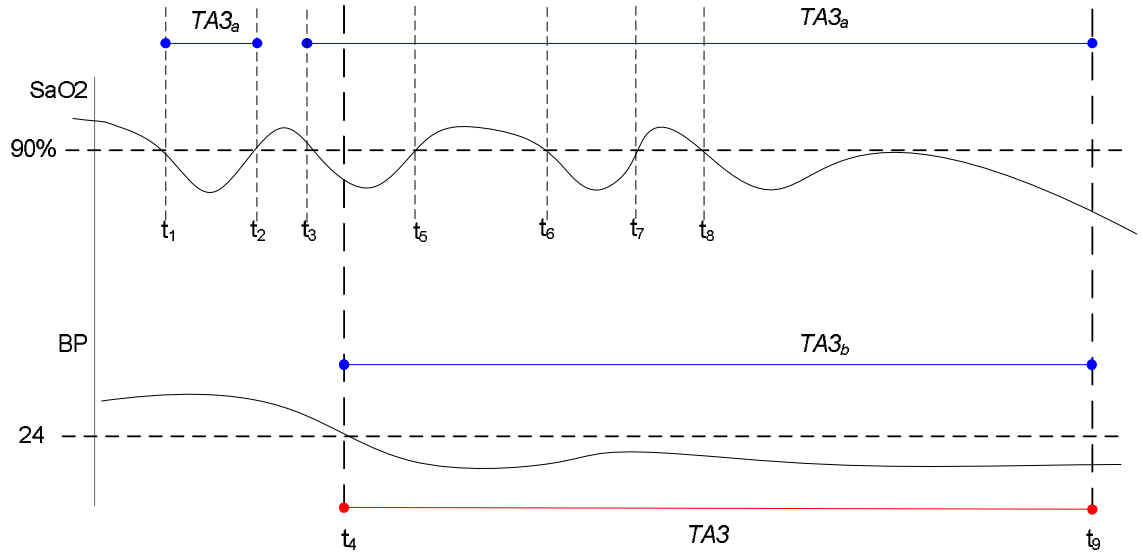

Figure 4.4: Example SaO2 and blood pressure data streams showing child temporal abstractions $TA3_a$ and $TA3_b$, and parent abstraction *TA3*.

Figure 4.4 illustrates an example data stream-set and TA scenario involving abstractions on SaO2 and blood pressure. The TA specification *TA3* is applied to this data and the following results observed:

Specification $TA3_a$ becomes valid at time point t_1 as SaO2 falls below 90%. Unless the parent specification $Warning$ holds, we do not track time spent within temporal gaps hence at point t_2, $TA3_a$ terminates and the time $t_3 - t_2$ is not counted. Again, at t_3 the threshold is breached and $TA3_a$ initiates. At point t_4 blood pressure has breached 24 mm/Hg in a negative direction hence $TA3_b$ is valid. When all child TA specifications ($TA3_a$ and $TA3_b$) hold then parent specification $TA3$ becomes valid and we are now beginning to monitor the 20 second interval originally specified. At t_5, temporal interpolation mechanisms begin tracking the time spent above the threshold value for $TA3_a$. At t_6 the value $t_6 - t_5$ is less than the specified value of 5 seconds hence abstractions $TA3_a$ and *TA3* are still valid and the total time for the abstraction *TA3* is now $t_6 - t_4$. At t_7 the temporal interpolation mechanism again begins to count the time spent above the threshold and, at t_8, the total time $(t_8 - t_7) + (t_6 - t_5)$ is still less than the specified maximal temporal gap of 5 seconds. At this time, the abstraction $TA3_a$ remains valid, and as $TA3_b$ still holds true, *TA3* also holds. At t_9

the total time elapsed for parent specification *TA3* is 20 seconds meaning the predefined abstraction has now been found to exist in the data and alerting mechanisms can be activated.

The example only depicts one of many possible sets of circumstances but includes the essential aspect of temporal interpolation. A variation on the above could include the possibility of temporal gaps in $TA3_a$ exceeding 5 seconds. At that time $TA3_a$ would be invalid leading to termination of *TA3* and no alarm deployment.

4.6 The Event Calculus for Multidimensional Online Temporal Abstraction

Using the EC we can instantiate a model for a temporal world through the use of axioms defining which events affect which fluents. Given a narrative of events from that world we can then reason about which fluents hold and when they hold. The problem with the simple version of the EC for modelling this kind of temporal domain is that there are no axioms for representing continuously changing quantities. As we have seen in section 4.2.1, Shanahan and Miller introduced axioms that enable continuously changing variables [160, 205], such as the height of a falling ball or the level of liquid in a filling vessel, to be modelled. The temporal model I describe in the following sections incorporate these axioms and apply them to the domain of temporal reasoning over time-stamped clinical data. In doing so I am extending the capabilities of the EC over previous implementations in this domain where the EC was found to be lacking expressive power.

The following description sets out the method of abstracting data according to the previously discussed specification *TA3* . From this point onward, we rename the TA3 fluent to the *Warning* fluent to better describe the importance of its existence. Previous to this point, it was important to emphasis that it is a TA, just like its child components $TA3_a$ and $TA3_b$. Its truth value signifies that all child TAs currently hold, the TA rule as a whole is true and the timer is counting. In section 4.6.1,

we demonstrate how we model the temporal world of both child specification $TA3_a$ and parent specification *Warning*. We have omitted representation of $TA3_b$ due to its similarity with $TA3_a$ and assume the reader can extrapolate from the explanation given for $TA3_a$. We model changing properties of the temporal domain as fluents in EC. Fluents are then essentially temporal abstractions of the data, being initiated by a definable domain event and persisting over some time interval until being terminated by another domain event.

Section 4.6.1.1 details how we have represented continuous change in our temporal model through integration of the *Trajectory* and *Releases* predicates. This is directly connected to thesis hypothesis 3 and enables the EC to be used as a basis for online and incremental temporal abstraction.

Section 4.6.1.2 details how the temporal model is utilised in abstracting data from multiple data streams. This is an areas which has not been incorporated into previous temporal models for TA, most having been concerned with reasoning based upon a single parameter. In terms of the types, or classes, of temporal abstractions available we demonstrate the model for a level shift type TA in this chapter although, as discussed in section 4.6.2, other TAs are possible such as those requiring more complex numerical calculations and TAs where the relationships between fluents is sequential rather concurrent, as in the proposed model.

4.6.1 Modeling the Temporal Domain using the Event Calculus

Table 4.1 provides a description of EC predicates relevant to the model provided by this research.

Predicate	Definition
Initially$_P f$	Fluent f holds from time $t = 0$
Initially$_N f$	Fluent f does not hold from time $t = 0$
Happens(e, t)	Event e happens at time point t
Initiates(e, f, t)	Fluent f is initiated by event e at time t and holds until terminated
Terminates(e, f, t)	Fluent f is terminated by event e at time t
Clipped(f, $[t_1, t_2]$)	Fluent f is broken between times t_1 and t_2
HoldsAt(f, t)	Fluent f holds at time t
Releases(e, f, t)	Fluent f is no longer subject to inertia if event e happens at time t
Trajectory(f_1, t_1, f_2, t_2)	If fluent f_1 is initiated at time t_1 then fluent f_2 is true at time $t_1 + t_2$
$t_1 < t_2$	Time point t_1 occurs before time point t_2

Table 4.1: A subset of the extended event calculus predicates and their definitions

$$Initially_N(TA3_a) \quad \text{(EC 1)}$$

$$Initially_P(Threshold\,(90)) \quad \text{(EC 2)}$$

$$Initially_N(TA3_aOR) \quad \text{(EC 3)}$$

$$Initially_N(GapTime(0)) \quad \text{(EC 4)}$$

$$Initially_P(MaxGapTime(5)) \quad \text{(EC 5)}$$

$$Initially_N(Warning) \quad \text{(EC 6)}$$

$$Initially_N(FinalAlarm) \quad \text{(EC 7)}$$

$$Initially_N(WarningTime(0)) \quad \text{(EC 8)}$$

To establish an initial state the *Initially* predicate is used to apply values to domain fluents. EC 1 simply states that the $TA3_a$ fluent is not true at time $t = 0$. A property named *Threshold* is used to state the maximum SaO2 value for which the abstraction holds. It is initialised to a value of 90% at time $t = 0$ using equation EC 2. $TA3_aOR$ ($TA3_a$ Out of Range) models intervals where SaO2 is greater than the threshold value of 90% and, as asserted by EC 3, it is not true at time $t = 0$. *GapTime* tracks the time spent within the $TA3_aOR$ interval and EC 4 sets its value to zero at time $t = 0$. Clause EC 5 says that the maximum time for *GapTime* is 5 seconds. This is modeled by fluent *MaxGapTime*; as long as *GapTime* is less than this value after $TA3_a$ has

been initiated, then fluent $TA3_a$ holds true. EC 6 states that fluent *Warning* is not true at $t = 0$. *Warning* models the conjunction of $TA3_a$ and $TA3_b$, and if it holds true, then the *FinalAlarm* fluent will result in an alarm state being generated. *FinalAlarm* is also initialised to false using EC 7. EC 8 sets fluent *WarningTime* to zero which is used to track the time spent within the parent abstraction interval *Warning*.

We model the acquisition of new patient data values by means of a *Receive* event in a way similar to [58]. For example, the event *Receive(SaO2)* means that at a certain time a value was received by the system from the SaO2 sensor. The *Receive* event has the potential to initiate the $TA3_a$ property as shown in the following axiom:

$$Initiates(Receive(SaO2), TA3_a, t) \Leftarrow Happens(Receive(SaO2), t) \wedge HoldsAt(Threshold(90), t) \wedge SaO2 < 90 \quad \text{(EC 9)}$$

EC 9 states that if a receive event happens at time t, $Threshold(90)$ holds at time t and the value of received SaO2 is less than 90 then the $TA3_a$ property initiates. Through the common sense law of inertia once the $TA3_a$ property is initiated it is assumed to persist until some event terminates it. The above axiom would be true after time t_1 in Figure 4.5. We assume the *weak* interpretation of Initiates, as defined in [51], where a fluent is only initiated by an event if it has not been previously initiated and is not holding at that time.

If SaO2 rises above 90% and any of the other other child TA specifications ($TA3_b$) are false, then we need to terminate $TA3_a$. This happens at time point t_2 in Figure 4.5. Negation of the *Warning* fluent is used to represent the fact that not all child TAs hold. EC 10 defines the termination of $TA3_a$:

$$Terminates(Receive(SaO2), TA3_a, t) \Leftarrow Happens(Receive(SaO2), t) \wedge HoldsAt(Threshold(90), t) \wedge SaO2 > 90 \ \wedge \neg HoldsAt(Warning, t) \quad \text{(EC 10)}$$

As soon as all other child TAs begin to hold, the only thing which can terminate $TA3_a$ is the value of *GapTime* reaching the maximum allowable temporal gap quantity, *MaxGapTime(5)*. Until all other TAs hold, we don't bother monitoring temporal

gaps. We only want to know if $TA3_a$ is initiated (SaO2 values less than 90%), then as soon as all other child TAs hold, we begin the monitoring of temporal gaps for each child TA. This will also coincide with the time that *Warning* holds. Figure 4.5 shows that, at t_4, $TA3_b$ begins to hold hence *Warning* also holds (see EC 20-21)).

Assuming all child TAs hold, then if a receive event happens at time t and contains an SaO2 value greater than 90% and $Threshold(90)$ holds at time t then the $TA3_aOR$ property initiates as shown in axiom EC 11. This would become true at points t_5 and t_7 in Figure 4.5.

$$Initiates(Receive(SaO2), TA3_aOR, t) \Leftarrow Happens(Receive(SaO2), t) \wedge HoldsAt(Threshold\,(90), t) \wedge SaO2 > 90 \quad \text{(EC 11)}$$

The $TA3_aOR$ property terminates when the received SaO2 value is less than 90% corresponding to time points t_6 and t_8 as follows in EC 12:

$$Terminates(Receive(SaO2), TA3_aOR, t) \Leftarrow Happens(Receive(SaO2), t) \wedge HoldsAt(Threshold\,(90), t) \wedge\ SaO2 < 90 \quad \text{(EC 12)}$$

Tracking the time spent within $TA3_aOR$ effectively monitors the temporal gap quantity *GapTime*, and allows temporal interpolation described as follows.

4.6.1.1 Temporal Interpolation using *Trajectory*

GapTime is a continuously changing variable that tracks the period of time where the $TA3_aOR$ fluent holds. It is similar to the *CountDown* parameter defined by Miller and Shanahan [205], where the notion of default persistence does not apply and the fluent is free to continuously change state. This aspect of dynamism applied to the EC is the one that is exploited in order to extend the EC into the application environment of online, sample by sample, temporal abstraction; as proposed by Hypothesis 3.

The introduction of *GapTime* essentially establishes a *timer fluent* and it is this which is crucial for performing online and incremental temporal abstraction. We do

not know how long a particular fluent will persist before termination but to perform interpolation between temporal properties, and also to force termination based on interval time, a timer is required.

If the value of *GapTime* reaches the pre-defined value for maximum allowable 'gap time' of 5 seconds, then the abstraction is invalid and is terminated. The *Trajectory* clause is used to define the continuously changing nature of *GapTime*:

$$Trajectory(TA3_aOR, t, GapTime(x_2), t_2) \Leftarrow HoldsAt(GapTime(x_1), t) \wedge x_2 = x_1 + (t_2 - t) \quad \text{(EC 13)}$$

Axiom EC 13 states that if the $TA3_aOR$ fluent holds at time t then $GapTime(x_2)$ will attain the value $x_1 + (t_2 - t)$ at time t_2 which equates to a value equal to the time elapsed. Rather than using a newly introduced sort, as Miller and Shanahan did with their *parameter* sort, my axiomatisation incorporates the *Releases* clause which essentially 'frees' the *GapTime* fluent from inertia and allows it to change dynamically. *Releases* was introduced by Shanahan in an earlier work on modelling continuous change in the EC [206]; as discussed in section 4.2.1.

$$Releases(Receive(SaO2), GapTime(x), t) \Leftarrow Happens(Receive(SaO2), t) \wedge HoldsAt(GapTime(x), t) \wedge HoldsAt(Warning, t) \wedge HoldsAt(Threshold\,(90), t) \wedge SaO2 > 90 \quad \text{(EC 14)}$$

Note that unless the parent fluent *Warning* holds, we don't bother counting *GapTime*. *Warning* holds when all constituent child TAs hold. So *GapTime(x)* will continuously increment from the time $TA3_aOR$ is initiated (t_5 and t_7) until it is terminated (t_6 and t_8). Once $TA3_aOR$ is terminated we need to set *GapTime* to its current value as

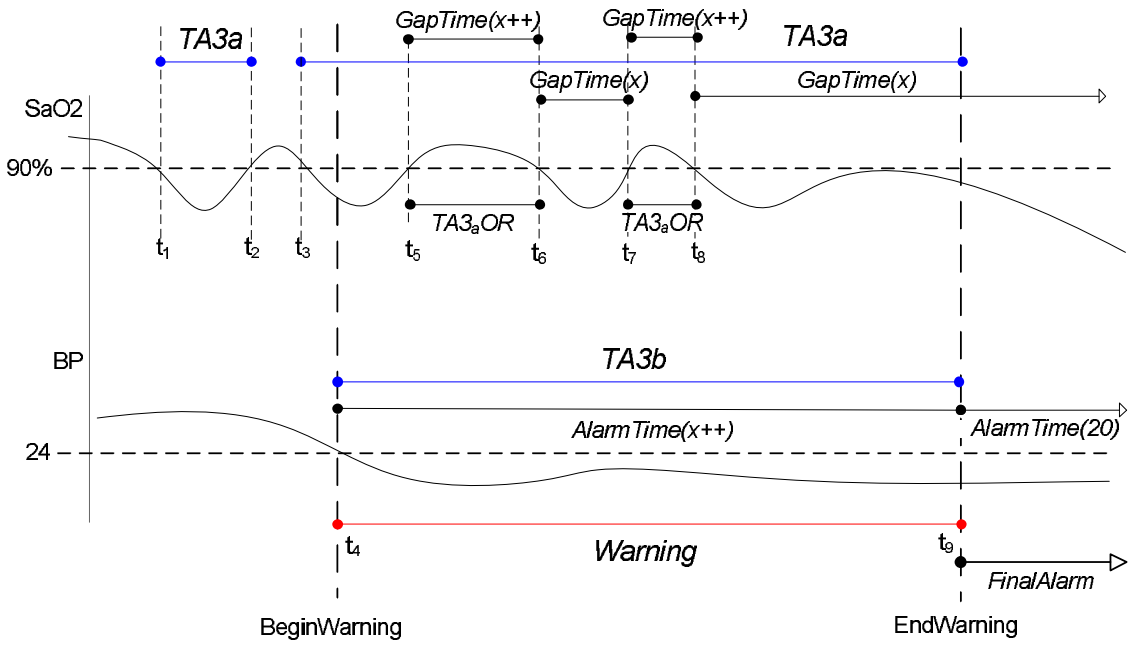

Figure 4.5: Event Calculus based temporal abstraction of multiple data streams showing primary fluents with initiating and terminating time points. x^{++} denotes a fluent which has been released from inertia and in a state of continuous change.

follows in EC 15:

$$Initiates(Receive(SaO2), GapTime(x), t) \Leftarrow Happens(Receive(SaO2), t) \wedge HoldsAt(GapTime(x), t) \wedge HoldsAt(Warning, t) \wedge HoldsAt(Threshold\,(90), t) \wedge SaO2 < 90 \quad \text{(EC 15)}$$

As soon as *GapTime(x)* is initiated with its current value it retains its state until being released from inertia again. This is shown in Figure 4.5 between t_6 and t_7. In this way we keep track of the time spent only within $TA3_aOR$ intervals. EC 16 demonstrates that trigger event *GapTimeUp* is fired when *GapTime* reaches a value of '5' and $TA3_aOR$ holds:

$$Happens(GapTimeUp, t) \Leftarrow HoldsAt(GapTime(5), t) \wedge HoldsAt(TA3_aOR, t) \quad \text{(EC 16)}$$

GapTimeUp terminates the $TA3_a$ fluent when it reaches its maximum defined value of '5', as shown in EC 17. As soon as any child TA specification becomes invalid, the

parent should also be terminated. Hence *Warning* will also terminate (see EC 29). Just as *GapTime* tracks time spent within the $TA3_aOR$ fluent, *WarningTime* monitors the time that *Warning* holds. EC 18 - 19 reset both to zero so that we may begin counting if *Warning* again becomes valid.

$$Terminates(GapTimeUp, TA3_a, t) \Leftarrow Happens(GapTimeUp, t) \quad \text{(EC 17)}$$

$$Initiates(GapTimeUp, GapTime(0), t) \Leftarrow Happens(GapTimeUp, t) \quad \text{(EC 18)}$$

$$Initiates(GapTimeUp, WarningTime(0), t) \Leftarrow Happens(GapTimeUp, t) \quad \text{(EC 19)}$$

With reference to Figure 4.5, *GapTime* does not reach the value '5' hence there are no terminations or initiations based on the occurence of *GapTimeUp*.

4.6.1.2 Temporal Abstraction across Multiple Data Streams

The *BeginWarning* event is fired when all conjunctive TAs begin to hold:

$$Happens(BeginWarning, t) \Leftarrow HoldsAt(TA3_a, t) \wedge HoldsAt(TA3_b, t) \quad \text{(EC 20)}$$

The *BeginWarning* event causes initiation of the *Warning* fluent as depicted in Figure 4.5 at time point t_4:

$$Initiates(BeginWarning, Warning, t) \Leftarrow Happens(BeginWarning, t) \wedge HoldsAt(TA3_a, t) \wedge HoldsAt(TA3_b, t) \quad \text{(EC 21)}$$

The continuously changing *WarningTime(x)* fluent tracks time spent within the *Warning* fluent in a way similar to *GapTime* tracking time spent within $TA3_aOR$. It is released from inertia using the *Releases* predicate in EC 22, so that it may begin counting, by the *BeginWarning* event:

$$Releases(BeginWarning, WarningTime(x), t) \Leftarrow Happens(BeginWarning, t) \wedge HoldsAt(WarningTime(x), t) \quad \text{(EC 22)}$$

The trajectory of *WarningTime* is defined by EC 23:

$$Trajectory(Warning, t, WarningTime(x_2), t_2) \Leftarrow HoldsAt(WarningTime(x_1), t) \wedge x_2 = x_1 + (t_2 - t) \quad \text{(EC 23)}$$

EC 24 shows that as soon as *WarningTime(x)* reaches the value '20', indicating 20 seconds have expired, the trigger event *EndWarning* is fired:

$$Happens(EndWarning, t) \Leftarrow HoldsAt(WarningTime(20), t) \wedge HoldsAt(Warning, t) \quad \text{(EC 24)}$$

EndWarning initiates *FinalAlarm*, *WarningTime* to the value '20' and terminates the temporal abstraction abstraction *Warning*. See EC 25 - 27

$$Initiates(EndWarning, WarningTime(20), t) \Leftarrow Happens(EndWarning, t) \quad \text{(EC 25)}$$

$$Initiates(EndWarning, FinalAlarm, t) \Leftarrow Happens(EndWarning, t) \quad \text{(EC 26)}$$

$$Terminates(EndWarning, Warning, t) \Leftarrow Happens(EndWarning, t) \quad \text{(EC 27)}$$

When *Warning* terminates due to an *EndWarning*event we have a Maximum Validity Interval (MVI) for fluent *Warning* which represents the time over which *Warning* maximally holds. In other words, the TA specification *Warning* (originally defined as *TA3*) has been found to exist in the data and the MVI is the interval $[t_1, t_2]$ as follows in EC 28:

$$MVI(Warning, [t_1, t_2]) \Leftarrow Initiates(BeginWarning, Warning, t_1) \wedge Terminates(EndWarning, Warning, t_2) \wedge \neg Clipped(Warning, [t_1, t_2]) \quad \text{(EC 28)}$$

This is illustrated as the interval $[t_4, t_9]$ in Figure 4.5. If either or both constituent TAs has terminated due to its *GapTime* fluent reaching the maximum defined value then the *Warning* fluent is clipped as shown in EC 29:

$$Clipped(Warning, [t_1, t_2]) \Leftarrow Terminates(GapTimeUp, TA3_a, t) \vee Terminates(GapTimeUp, TA3_b, t) \quad \text{(EC 29)}$$

The final outcome of TA processes is the state of the *FinalAlarm* fluent (EC 26). If it holds, then all child TA specifications have been found to hold simultaneously for the pre-defined period of time and alarm mechanisms can be deployed to alert clinicians of a potentially dangerous situation.

4.6.2 Further Abstraction Types

The example provided describes how an individual *negative level shift* abstraction ($TA3_a$) and the conjunction of multiple abstractions (*Warning*) has been modelled in the EC. The model used is not solely limited to determining periods of time a parameter value, such as SaO2, has fallen below a threshold value. The EC model for online TA will operate on any stream of successive time-stamped values; for example, trend or Area Under the Curve (AUC) values. This would necessitate a separate agent for calculating the required parameter and supplying it, along with a time stamp, to the EC framework. In this way, the TA mechanism is isolated from possibly complex numerical analysis of time-stamped data and provides a component that performs temporal reasoning and abstraction upon the data it receives.

Sequential relationships between fluents, and hence *delayed* effects, can also be modelled using the EC. In the context of the case-study monitoring environment, this may manifest itself as in the following example:

> "Detect periods of 30 seconds where SaO2 falls below 90% and *then* (from time t = 30 seconds) arterial blood pressure falls below 24 mm/Hg for a further 20 seconds from that point."

We may write:

$$Trajectory(LowSaO2, t_1, CountDown, t_2, 30 - t_2)$$

which can be translated as, "if fluent *LowSaO2* holds at time t_1 and continues to hold until time t_2, then parameter *CountDown* will have the value $30 - t_2$." When the *CountDown* parameter reaches zero, the trigger event *BeginLowBP* is fired:

$$Happens(BeginLowBP, t) \leftarrow ValueAt(CountDown, t) = 0$$

If *LowSaO2* is not already valid, then receiving an SaO2 value which is less than 90% will initiate it:

$$Initiates(Receive(SaO2), LowSaO2, t) \leftarrow \neg HoldsAt(LowSaO2, t) \\ \wedge Happens(Receive(SaO2), t) \wedge SaO2 < 90$$

If *BeginLowBP* is fired and a value for BP is received with a value less than 24, then fluent *LowBP* will initiate:

$$Initiates(Receive(BP), LowBP, t) \leftarrow \neg HoldsAt(LowBP, t) \\ \wedge Happens(BeginLowBP), t) \\ \wedge Happens(Receive(BP), t) \\ \wedge BP < 24$$

Another trajectory predicate is required for counting the time of the *LowBP* fluent:

$$Trajectory(LowBP, t_1, CountDown, t_2, 20 - t_2)$$

When the timer expires at zero, the trigger event *BeginAlarm* is fired:

$$Happens(BeginAlarm, t) \leftarrow ValueAt(CountDown, t) = 0$$

BeginAlarm can be used to indicate the abstraction pattern consisting of a period of low SaO2 for 20 seconds followed by a period of low BP for a further 30 seconds has been detected. The timer requires resetting after *LowSaO2* has been detected in order for it to begin counting for the *LowBP* fluent; this is not shown above.

4.7 Discussion

This chapter has described the Event Calculus-based temporal model developed for performing TA in an online fashion to multiple physiological data streams. The model is implemented as a forward chaining rule engine and executed by a bank of Patient Monitor Modules (PMMs); one PMM per patient. Each PMM addresses "Level One

Dimensionality", as defined in section 1.2. Chapter 6 describes the implementation and evaluation of the model.

In in first half of the chapter I introduced influential and relevant approaches to temporal reasoning and to both the representation and detection of events. The two areas have existed and developed separately which has led to a divergence in terminologies and notations [234]. The EC-based model for online TA provides a bridge between the two areas in that it incorporates a detection engine akin to that found within an event monitoring system which is driven by a formal theory for temporal reasoning. Events of interest are temporal properties having some duration as opposed to event definition which relates to a particular time instant. This extends and enriches existing work concerned with interval-based event definition [222, 233, 236,237] while simultaneously broadening the application context of the event calculus from a backward reasoning model to a forward reasoning one which is able to perform incremental assessment of data on the fly.

The EC is a natural candidate for creating temporal abstractions. In the EC, events initiate fluents which are assumed to persist until another event terminates the fluent. The concept of a temporal abstraction is very similar and domain events can be defined which initiate and terminate intervals of time where some property holds. Shanahan's work on the EC [206] provided inspiration for the development of a model for real-time reasoning with his introduction of a model for continuous change. Adaption of the 'water tank' model [160], where a vessel is filled with water and the *trajectory* predicate used to model the changing level of water in the tank, allowed the representation of 'out of range' fluents (or properties) as continuously changing variables. This is required in order to track the time spent both within abstraction intervals and within the 'gaps' between abstractions to facilitate real-time temporal interpolation. Causal constraints are also a feature of the temporal model described in this paper where a fluent may be dependent on the state of other fluents. When the influencing fluents attain the correct value, *trigger events* are fired which initiate or terminate the dependent fluent [206].

Most clinical temporal abstraction frameworks have operated on a single patient's data [19, 21, 47, 49] and multiple concurrent high frequency abstractions have proven problematic. It is evident from [23], that for IDA systems to successfully move forward into the future where clinical environments are becoming increasingly data-intensive, the ability for managing multi-dimensional aspects of data at high observation and sample frequencies must be provided. This work addresses the issues of dimensionality and frequency of TA mechanisms through the union of a temporal model capable of real-time reasoning and an implementation based on a forward chaining rule engine enabling abstraction of multiple data streams.

A challenging aspect involved in the design of systems that must process high frequency data in real time is the ability to cope with a multi-dimensional data environment. At a first level of dimensionality, the system must be able to concurrently process multiple data parameters emanating from a single patient and at a second level, it is desirable to enable data analysis across multiple patients within a ward. The former level of dimensionality is addressed by the theories proposed in this chapter while multiple patient TA is discussed in the following chapter where numerous PMMs are interconnected and their TA process results integrated for the purpose of intelligent alerting.

The implementation is able to create complex abstractions where the TAs for each stream involved are combined through conjunction into a single axiom (rule). Temporal interpolation allows the clinician to specify a 'maximum gap', where the parameter involved may venture 'out of range' and the TA still remain valid. The model presented in this chapter does not allow for the creation of sequential temporal patterns such as I mentioned in section 4.6.2 although I have stated how a model could be built using the EC in that section.

In my model, multiple fluents are allowed to exhibit a causal dependency where that dependency is concurrent, not sequential. I plan to extend the temporal model to incorporate a sequential dependency relation which would allow modelling the above example as described in section 4.6.2. Previously, this has been accomplished

through the use of one or more of Allen's interval operators [122], such as *before* or *meets*. The work of Combi and Chittaro [58] is an example where *complex* TAs can be built consisting of numerous sequential *simple* TAs.

Another limitation of the work is a lack of noise and transient filtering as I have assumed the data is 'clean'. I realise this is an unrealistic postulation to make in an environment such as neonatal intensive care where data is measured though electrical transducers which are inherently prone to noise and spurious interference. The reader may also have noted there is also no inclusion in the temporal model for calculating gradients or other such numerical calculations which may be required to enable more complex abstractions. I have provided a model which will accept a series of time-stamped data values and temporally abstract them into context-sensitive intervals of interest indicating whether interval values are over or under a threshold (positive and negative level shifts). The time-stamped quantities received by the system could be trend values, baseline variation or Area Under the Curve (AUC); the TAs thus created then determine intervals of time where a particular trend, baseline variation or AUC value has either been exceeded. This obviously requires the use of separate agents for performing such calculations and supplying the data to the EC-based TA framework. In summary, my framework accepts clean time-stamped data and creates level shift abstractions of that data.

The EC has been shown to be a useful method for performing temporal projection or prediction [189,196], where we start with an initial state and some events and then reason about the state that results from the events. A somewhat obvious example in the domain of patient monitoring would be, to provide suitable warning if a parameter, at its current rate of increase, will attain a certain dangerous level. For example, we can detect that heart rate is increasing and at what rate it is increasing at. We can then predict (and warn) how long it will take for heart rate to reach a certain value.

I am also interested in accruing the total time spent *within* 'in range' abstraction intervals. Presently, the temporal model tracks the total time that the TA fluent holds, which may or may not include 'out of range' temporal gaps. Clinical research

is presently interested in the total time spent above or below a certain threshold as an early indicator of impending deterioration in the baby's health. If the temporal gap quantity (*MaxGap*) is set to a large value relative to some clinically interesting monitoring time, then the temporal model requires only slight modification in order to allow this to happen.

The work described in this chapter provides part of the Event Stream Processor (ESP) [22]; a framework to support recent clinical research which hypothesises that there may be an additive effect when multiple aberrant behaviours exist across a number of physiological data streams. Moreover, the work addresses the issues of dimensionality and data frequency that are seen to be lacking in current clinical TA-based systems and which must be confronted for the continued evolution of such systems.

Chapter 5

An Architecture for Multidimensional Online Temporal Abstraction

The literature survey performed for this research exposed several limitations of TA-based systems performing Intelligent Data Analysis (IDA) [8] which were summarised both in section 1.3.2 and also in section 2.6. This chapter substantiates hypotheses 1, stated in section 1.3.3, which proposes that a framework can be defined to provide for multi-dimensional online temporal abstraction (MOTA) to occur. In the context of the case-study domain of the neonatal intensive care unit (NICU), the framework provides a platform for continuous and concurrent monitoring of multiple patients where it is possible to execute multiple TA rules, each consisting of numerous abstractions; this was outlined in section 1.2 and defined as *multi-dimensional online temporal abstraction*.

The problem domain is one of high data dimensionality where a single patient gives rise to multiple streams of physiological data and there may be numerous TAs created for each patient data stream set. The reader is referred to Figure 1.1 for a graphical representation of the dimensionality of the problem domain. This chapter presents a high level view of the primary components, their interrelationships and the concepts underpinning their operation. Initially, I introduce the multi-dimensional online temporal abstraction (MOTA) framework through a high level description of

the primary components and steer the reader through a typical sequence of operations resulting from the input of a TA rule followed by initiation of TA processes and terminating with the accumulation and exploration of results. Following this, I elaborate on the computational components of MOTA, beginning with the Active Domain Registry (ADR), which enables a level of multi-dimensionality for TA processes through the fusion of results from multiple Patient Monitor Modules (PMMs). The ADR includes a Medical Alert Monitor (MAM), allowing the automatic generation of clinical alerts, and produces the Event Correlation Specification (ECS); which defines the relationships between domain entities and facilitates orchestration of MOTA processes. Subsequently, definition of the Event Stream Processor (ESP), the temporal abstraction engine of the MOTA framework, is revealed.

5.1 Architectural Overview

The architecture presented here is a component of the e-Baby architecture that is under development as part of the e-Baby collaborative research project. [4, 26, 27, 66]. The e-Baby project defines a data management layer providing permanent storage of physiological data collected from bedside monitors which is written to a data warehouse via regular updates. In addition, data mining processes are being developed within the Analytical Processor [34] to reveal new associations between historical physiological data; the goal being the derivation of TA rules which act as early warning indicators of potentially dangerous clinical conditions.

Integrated into the e-Baby framework is the Solution Manager Service (SMS), which was originally developed for business event monitoring and performance measurement [68, 69]. The SMS is an intelligent decision support system that allows organisations to collect information pertaining to business processes and share this with authorised parties. Integral to to the SMS, and most significant to the context of this thesis, is the *Event Stream Processor* (ESP). The ESP supports the integration of a large number of near real-time events and subsequent transformation of those events to business performance data which is then stored in a data warehouse. An

Analytical Processor then mines stored data enabling users to obtain information such as process times and costs and service level violations [69]. The e-Baby research collaboration is applying this research within the domain of health and medicine.

In this thesis, the ESP is redefined to enable temporal processing of streaming data, as opposed to the monitoring of instantaneous events. Figure 5.1 depicts the ESP and associated components described in this chapter, which are housed within the MOTA framework, which itself is contained inside the SMS as depicted in section 1.4. Similar to its predecessor, the ESP defined here supports the processing of a large number of events in real-time, although its function is different to that originally specified. Clinical diagnosis incorporates the temporal dimension, often explicitly, and reasoning about the order of a set of occurrences or the evolution of patient indicators over time is imperative for an accurate diagnosis [17]. Realistic decision support systems in a patient monitoring environment should therefore encompass, and be able to model, the temporal dimension in order to facilitate alarming mechanisms which more accurately resemble the clinical reasoning which underpins a diagnosis. Hence, the ESP includes a model of time exploiting time itself as an explicit concept and is able to reason with time-stamped data to provide a temporal abstraction service for use within a real-time data streaming environment. The Event Calculus [148] provides the formal reasoning which underpins the temporal abstraction method employed within the ESP environment.

The architecture described here, and illustrated in Figure 5.1, addresses the little researched area of MOTA by distributing processing to multiple Patient Monitor Modules (PMMs), each of which is responsible for temporally abstracting the data stream-set from a single patient. This has been defined in section 1.2 as "Level One Dimensionality". Rules are initially input by a clinician via the Rule and Query Builder and written to both the Active Domain Registry (ADR) and the TA and Physiological Database. Although not implemented in this work, there is also the possibility for interfacing with the Analytical Processor in order to obtain rules discovered through data mining processes and apply these to patient data in a real-time

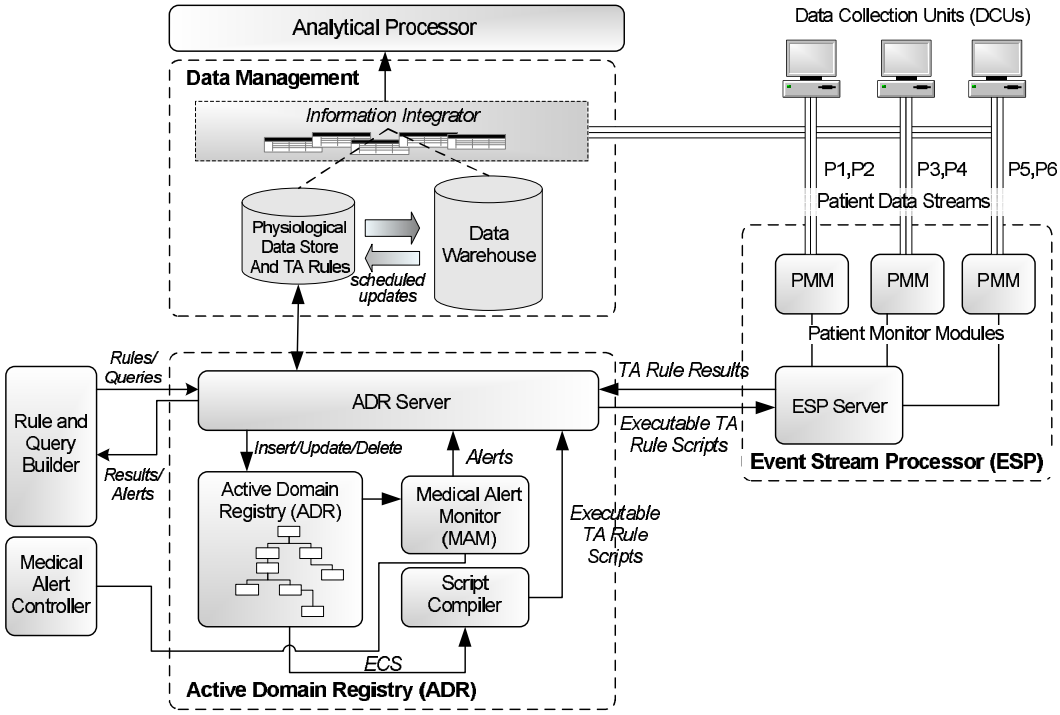

Figure 5.1: High level architecture of the MOTA framework and associated components.

sense for early detection of possibly life threatening conditions. Validation, editing and activation of rules, together with query functionality, is also provided for via the Rule and Query Builder interface to the ADR. The ADR provides a central conceptual knowledge model of domain concepts while the Medical Alert Monitor (MAM) allows a degree of reactive behaviour from where alerts can be triggered upon changes of state within the registry. The distribution, execution and collation of TA rules to and from multiple PMMs, which implements the concept of MOTA and represents "Level Two Dimensionality", is defined within the Event Correlation Specification (ECS), which is a permanent XML-based instantiation record of the contents of the the ADR. The ADR, in collaboration with the Medical Alert Monitor (MAM), accumulates results from PMMs and facilitates automated alarming mechanisms.

5.1.1 Process Walkthrough

The process of MOTA begins when a clinician enters a TA rule into the system. An example high level rule description is as follows:

> Detect periods of time of minimum length twenty seconds where SaO2 is below 85% and blood pressure is below 24mm/Hg. Run this analysis on patients with identification codes 1A1 and 1A2. Begin the monitoring phase at 12.00 hours and terminate the monitoring phase at 18.00 hours. The name of this rule is "Low O2 and low BP".

The Rule and Query Builder decomposes the rule into *child* and *parent* abstractions. There is one *child* abstraction for SaO2 and one for blood pressure. These are combined conjunctively to form the *parent* abstraction, which is essentially the alert condition that only exists if both child TAs are valid.

When the new TA rule is entered, necessary parameters such as threshold values, dates, times, interval lengths and patient identification codes are parsed from the input information and relevant objects are instantiated which constitute the core of the ADR, forming a runtime model of the relationships between patients, rules, TAs and data parameters, such as blood oxygen and blood pressure. Parameters are also inserted into the TA Rule and Physiological Database for permanent storage in the Data Management Layer. Relational links exist between the TA rules and stored physiological data from the bedside monitors to enable querying and retrieval of data that is pertinent to the abstraction rule. The ADR then generates the ECS and notifies the ESP Server of the presence of a new rule and the related patient identification numbers/s. The ESP server queries an availability table to determine the names and locations of available PMMs. In the case of the example rule above, there would be two PMMs required; one for patient 1A1 and one for patient 1A2. Once these are contacted, the *Script Compiler* renders the ECS to a temporal abstraction rule script that is directly executable by a PMM. The ADR Server then forwards the script to the ESP Server which conveys it to the relevant PMMs. Using a threaded execution model, PMMs execute the TA rule script on the patient data stream set which they are responsible for monitoring. Each PMM can execute multiple TA rules, such as the one provided in the example, on a single patient data stream set, where the stream set may consist of multiple physiological parameters such as heart rate, blood

pressure and respiratory rate. The reader is referred to section 1.2 for clarification of the multi-dimensional nature of the rule and data environment.

After rules have been executed by PMMs, results are returned to the ESP Server which then relays them back to the ADR. If a rule has returned a status result of *true*, meaning the TA was found to exist in the patient data, then the status of relevant ADR objects is modified. The MAM will instantly recognise such changes of state, representing anomalous clinical conditions, and facilitates alert generation through its internal rule engine environment. During the monitoring period when TA rules are being executed, it is possible for a clinician to perform queries which reveal the state of the executing rules and also rule details to allow editing or termination of current executions. I refer to this style of querying as *Immediate Querying*. *Historic Querying*, on the other hand, is possible after a rule has completed its monitoring period and returns the start and end times for the interval constituting the temporal abstraction, if the rule returned *true*. Querying is explained further in section 5.2.

In this way, results from multiple TA processes within a 'bank' of PMMs can be cross-correlated and evaluated thereby allowing TA across multiple patient's data stream-sets. This is the process of multi-dimensional online temporal abstraction (MOTA).

5.1.2 Physical Organisation of MOTA Framework Components

The framework contains all the components required to perform online temporal abstraction of multiple patient data streams. The ADR, ESP Server and Data Management Layers reside within the SMS which is located in an NICU server room, whereas each PMM is located at the patient's bedside. The ESP Server, although logically part of the same area of functionality as the PMMs, namely the Event Stream Processor (ESP), are located in the NICU server room and connected via TCP/IP connections to the PMMs. Figure 5.2 depicts the physical locations of PMMs and associated MOTA framework components.

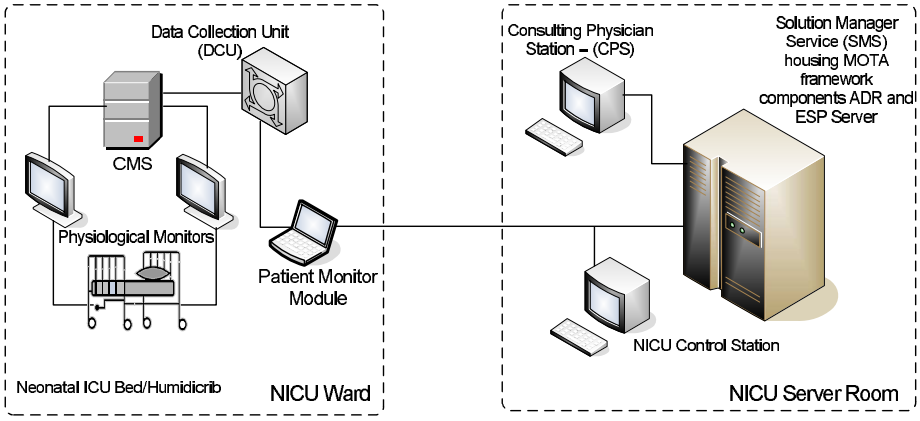

Figure 5.2: Physical organisation of Patient Monitor Modules (PMMs) and MOTA framework within the context of the Neonatal Intensive Care Unit.

During our work on the e-Baby project, a method was devised to collect data from bedside monitors and transmit it in XML-encoded format across a network for storage in a data warehouse [66, 67]. This was briefly described in section 1.4. There is one Data Collection Unit (DCU) for every two patient cribs; these are located within the NICU wards, each stationed between two cribs. They connect to bedside monitoring equipment via a Philips Component Monitoring System (CMS), which is a chassis system accepting multiple plug-in modules that connect to the physiological monitors. The NICU Control Station provides the mechanism to control the establishment and cessation of the collection of data from the medical monitors and also allow control of the size of the data packets that are transmitted. The NICU Control Station is a part of the e-Baby project, as are the Data Collection Units and the web service-based method of communication between the SMS and the DCUs. This research situates the PMMs at a point local to the bedside where data can be analysed and processed in real time and it is envisaged to provide a graphical interface for each PMM which will supply vital feedback to clinical staff on the floor when anomalous physiological conditions are detected. The Consulting Physician Station (CPS) is also an e-Baby project development providing clinicians with access to stored physiological data via a set of web services providing data mining capabilities [34]. I incorporate the CPS in this research as an interface to the MOTA framework ADR component as a means to

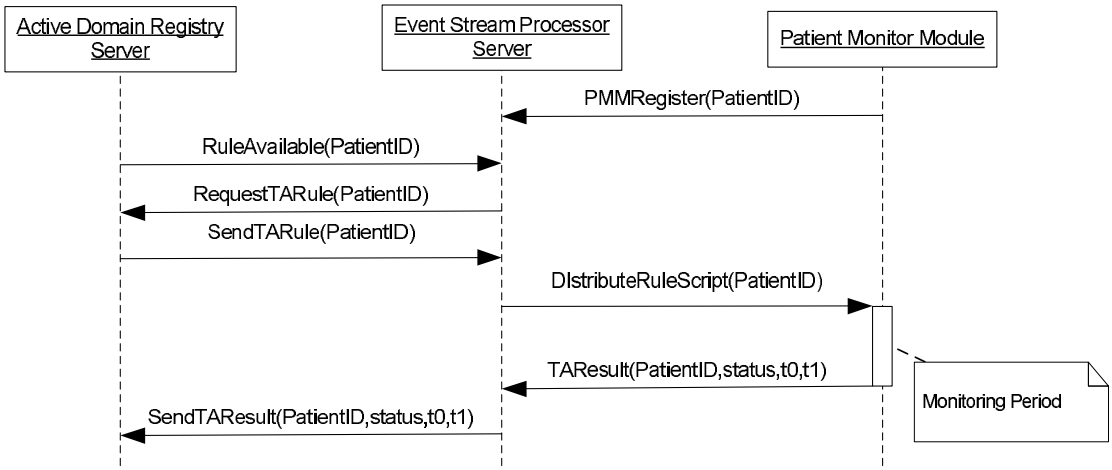

Figure 5.3: Primary message sequence between computational modules within the MOTA framework. Note: not all messages are shown.

access functionality which permits addition of TA rules, querying of results, inspection of current rule status and alert configuration.

PMMs are connected via a TCP/IP connection to the ESP Server and utilise HTTP for the transport protocol. Messages are XML-encoded and semantics are described through each component's interface. Figure 5.3 illustrates a high level summary of the sequence of messages involved in the insertion of a TA rule, execution of the rule and finally the return of results from the ESP back to the ADR where alerting rules can be triggered by the MAM. Section 5.3 describes communication issues and message semantics. The choice of distributed computational components is driven by the nature of the application area. Applying TA to this complex data environment requires the ability to perform multiple independent computations and to provide a task-centred environment where clinicians may enter rules that drive abstraction mechanisms and monitor their progress from a single and central location.

5.1.3 Active Domain Registry (ADR)

The ADR is the conceptual core of the MOTA framework where domain entities are represented and instantiated using an object-oriented paradigm. It generates and

stores the ECS which provides a permanent semantic representation of the relationships between patients, TA rules, data streams, data parameters and alerts. The ADR also acts as an 'accumulator', where the results from multiple PMMs can be integrated and reasoned with thereby facilitating MOTA and intelligent alerting in a multi-dimensional data environment. The ADR and the ECS are therefore intrinsically linked to both hypotheses 1 and 2 (section 1.3.3), particularly hypothesis 2 which states, "that a method to semantically represent the complex multi-dimensional relationships between data streams, parameters and TAs can be defined to support TA within such complex environments."

The conceptual model contained within the ADR is a high level representation of important objects in the domain and as such, its main purposes are to:

1. Accurately reflect the relationships between domain entities supporting:
    - a mapping to and from the Event Correlation Specification (ECS)
    - a layer of querying mechanisms for inspection and browsing of TA rules within a multi-dimensional context
    - MOTA though the linking and accumulation of TA results from multiple independent PMMs
    - run-time storage of TA rules
2. Provide reactive behaviour for facilitation of alarming mechanisms.

The ECS provides a specification which is used for orchestration of PMMs hence providing the specification for MOTA within the system. Whilst temporal concepts are modelled therein, I emphasise that the ADR is not required to perform temporal reasoning on patient data; this is accomplished by Patient Monitor Modules (PMMs) described both in section 5.1.5.1 and chapter 4.

Initially, various ontology building methods were investigated and experimented with although the characteristics of this component dictated a different approach.

Firstly, it is a relatively small data structure; the TA Rule Base exists for the permanent storage of rules and to establish the relationship with patient specific information while the ADR houses only presently executing rules and domain objects. Secondly, the design requires a degree of reactive behaviour, primarily for the generation of clinical alerts but also for asynchronous communication with the ESP Server. A small and powerful production rule engine is easily connected into the OO-based design of the ADR, making it a hybrid knowledge representation scheme similar to [49]. This *active* behaviour gives rise to the nomenclature: *Active* Domain Registry, which I define as a domain model that includes the reasoning ability to enable both instantiation and modification of concepts in a way analogous to that of an active database [119] where rules can be triggered upon pre-defined changes of domain attributes. A principal advantage of this type of in-memory data structure over the active database is its speed of execution, which is especially pertinent in the real-time world of patient monitoring.

In terms of its similarity with an active database (ADB), the ADR provides a repository for domain information including a schema for representing the relationships between elements and, through the MAM, offers a degree of reactive behaviour. An active database differs from a traditional relational database management system (DBMS) in that tasks which were previously executed by external applications are integrated into the database itself [225], allowing the modeling of both structural and *behavioral* aspects. Active databases normally specify their reactive behaviour using rules having three components, an event, a condition and an action (ECA). An *event* is an instantaneous happening to which the rule will respond. The *condition* part of the rule is checked when the event occurs and if it holds then the *action* part is executed. The ADR also includes rules for reactive behaviour in order to facilitate automatic alerting although these are condition-action rules or *production* rules. The ADR is also dissimilar in that it is an in-memory data structure which only stores the most recent and pertinent information. It delegates permanent storage of domain

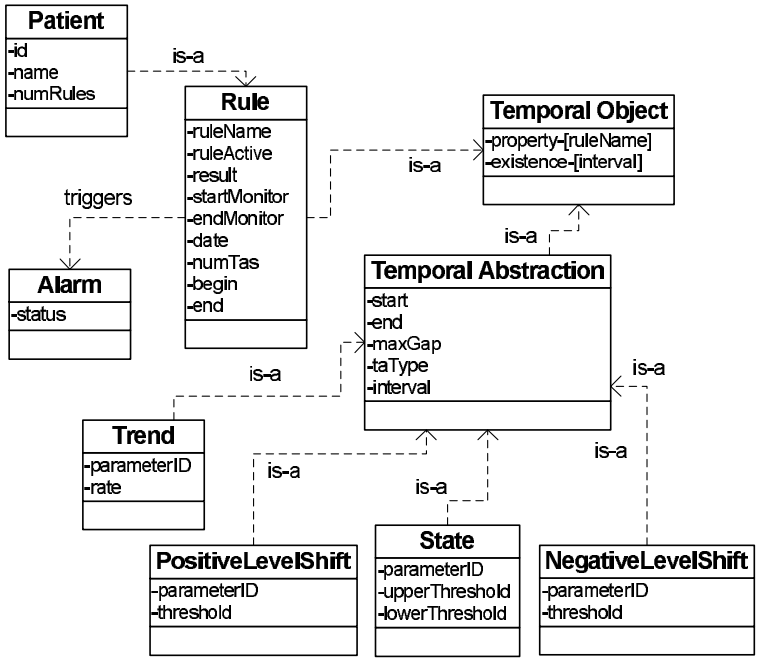

Figure 5.4: Active Domain Registry (ADR) conceptual schema illustrating how different temporal abstractions types are constructed from an interval having *start*, *end* and *maxGap* parameters. TAs inherit the associated patient and rule details for which the TA is relevant.

concepts to the ECS and the TA Rule Base. The ECS is comparable to the meta-information contained in the ADB catalogue, which defines ECA rules in terms of the data model underpinning the ADB.

The Momentum system [117] utilises an extension of the ADB concept in which temporal reasoning and temporal data maintenance are both catered for. The authors call this an active time-oriented database. The Momentum system philosophy is similar to that of the MOTA framework as its database supports incremental temporal abstraction and querying of complex concepts for multiple patients, although the application context is different. Momentum performs its TAs on static data using the database architecture and can mine potentially large data-sets whereas the MOTA framework operates in real-time on incoming data and delegates the responsibility of TA to individual PMMs. The ADR is optimised for efficient amalgamation of PMM result sets and automated alerting and there are no disc accesses by the ADR during the TA process.

Figure 5.4 depicts the conceptual ontology used in the ADR to model domain entities. The *TemporalObject* entity, which is similar to that defined by Keravnou [245] and Dojat [49] is defined by Keravnou as a possessing a "tight coupling between a *property* and its *existence*, where its existence can be expressed with respect to different time-axes." The property is that which is modeled by the temporal abstraction. For example a clinician may be interested in periods of 20 seconds or more where blood oxygen saturation is less than 90%, simply called "low O_2". The *temporal* property is then "low O_2". I define only a single time-axis in the ADR model and this is an abstraction on the real-time axis which is infinitely dense and extends from $-\infty$ to $+\infty$. The single time-axis is based on the time unit of seconds, which is the finest time unit required in this particular domain and largely governed by the fact that data is sampled at 1Hz intervals. As in [245], it is possible to specify events at coarser granularities than the second, for example, minutes are often practically relevant when clinicians specify periods of time for which certain properties hold.

For each rule entered into the ADR a number of object instances are created. A *Patient* instance contains individual patient details such as identification number and name. The *Rule* entity inherits a patient's characteristics enabling a rule to be associated with a particular patient. Within the Rule object there are attributes for holding the number of TAs for the rule and the name of the rule, which will often be a clinical descriptor for the abstraction itself. Both the *Temporal Abstraction* and *Alarm* objects inherit from the Rule entity enabling TAs and alarms to be connected with a particular rule and patient. The TA instance contains all information required to perform the abstraction, for example, the physiological parameter ID, the length of the TA interval, maximum specified gap, TA type and the relevant threshold values. The *Alarm* instance contains a single field called *status* representing the resulting state of a rule's execution and is instantiated inside the Medical Alert Monitor for automated clinical alerting. All ADR entities are represented by the ECS and described more explicitly in the section 5.1.3.2.

As shown in Figure 5.4, a *temporal abstraction* concept inherits all the attributes of

a *rule*, which in turn inherits from a *patient*. In this way, data is organised in the ADR to allow for easy querying. Object-oriented technology exhibits characteristics which were seen as advantageous in the design of the ADR; namely inheritance. This allows modelling the fact that a physiological parameter is associated with a TA *belonging* to a rule which *belongs* to a specific patient. TAs, rules and patients are endowed with attributes and querying proceeds using expressions which denote objects. Both Dojat [138] and Combi [58] employ object-orientation in the modelling of domain concepts although their models are directly involved in the process of TA whereas the MOTA ADR models TA specifications only and is not used for the abstraction of patient data; that being catered for by separate Patient Monitor Modules.

5.1.3.1 Medical Alert Monitor (MAM)

The reactive capabilities of the ADR are provided by the Medical Alert Monitor (MAM), which is shown in Figure 5.5. When a new TA rule is inserted into the ADR, an Alert object is automatically created and passed to the MAM. This instance contains information regarding the identity of the rule and the patient for whom the rule is relevant. Within the MAM, an alert rule is constructed by the Alert Rule Generator and a Rete engine thread spawned which 'watches' the *status* field of the Alarm object. The *status* field becomes one of the rule antecedent parameters, initially set to 'false' and updated to 'true' by the ADR if a PMM determines a positive finding for the TA described by the rule.

A separate Alert monitoring area exists for each generated Alert instance. I define Alert objects as being in a passive state when they enter the MAM; meaning their *status* field is set to 'false' through the initial instantiation of the object. If and when a PMM determines a positive finding, the *status* field becomes 'true' and an alert rule is fired. The object then becomes *active* and alert control actions are triggered.

Alert control actions can be configured via the Alert Controller. Presently, alerts are simple messages displayed on the Rule and Query Builder interface but it is envisaged to offer the functionality for sending SMS and pager alert messages to

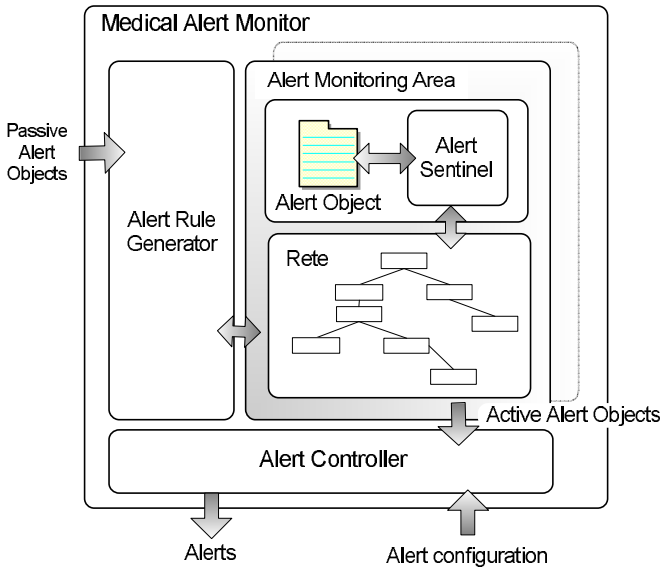

Figure 5.5: The Medical Alert Monitor (MAM). Passive alert objects are propagated from the ADR and enter the Alert Monitoring Area where the object is 'watched' by an alert rule within the Rete engine. An alert is triggered upon detection of a 'true' result in the alert object.

the clinician responsible for initial rule activation. Using a third party toolkit, an HTTP-POST compliant SMS provider, and an internet connection, it is a relatively simple procedure to set up such functionality. Of course, email or visual alerts are simple to implement. The Alert Controller implements an interface to enable such configuration although the primary responsibility of the MAM, at this stage, is for the detection of anomalous clinical conditions through the use of Alert objects and the subsequent generation of *active* Alert objects.

5.1.3.2 The Event Correlation Specification (ECS)

The The Event Correlation Specification (ECS) is a passive component within the MOTA architecture. It is an XML-based representation defining the complex inter-relationships between the primary domain entities of TAs, rules, patients and their respective attributes. Whilst the ADR acts as a runtime storage structure, the ECS provides a permanent record of TA rules that have either been, or are about to be, executed on patient data streams. In this way, it materialises research hypothesis 2

(section 1.3.3).

I partition the ECS into a hierarchy of 2 parts; the *domain* specific and the *patient* specific. The domain specific ECS provides the information required to orchestrate all computational modules required to perform MOTA on patient data stream-sets. In this sense, it is a global specification containing the full set of rules, patients and TAs presently active in the system, whereas the patient specific ECS is a subset including only the rules and TAs relevant to a single patient. This allows a greater degree of modularity within the ADR, organising concepts according to the most relevant entity in this particular NICU domain, the patient. Computationally, it saves time searching and parsing a potentially large domain specific ECS in order to determine details of a rule for a particular patient. The ECS stored and housed within the ADR. Figure 5.6 illustrates a patient specific ECS where a single TA rule is specified; the rule is taken from the application context of neonatal intensive care and is almost identical to the example rule used in section 5.1.1. The rule is named *LowO2_BP* and consists of 2 TAs, one being a negative level shift (*NegLShift*) abstraction on blood oxygen saturation (SaO2) and the other a negative level shift on blood pressure. Further rules for this patient can be specified within multiple <*rule*> tags. The rule is described in English as follows:

> If mean blood pressure falls below 24 mm/mercury (mm/Hg) for a minimum period of time of 20 seconds and peripheral oxygen saturation is less than 85% for 20 seconds, then alarm. Each parameter is allowed to venture outside the prescribed range for a maximum total time of 5 seconds within the 20 second interval. The monitoring period begins at 12:00 hours and terminates at 18:00 and is to be executed on blood oxygen saturation (SaO2) and blood pressure (BP) streams belonging to patient Tom Brown who has a patient ID of "2A-3".

Important ECS tag definitions are as follows:

```
<patient_ecs>
   <patient id="1a1">
      <name>Tom Brown</name>
      <numRules>1</numRules>
      <rule id="TP-1">
         <ruleName>LowO2_BP</ruleName>
         <numTAs>2</numTAs>
         <ruleActive>T</ruleActive>
         <startMonitor>13:00</startMonitor>
         <endMonitor>20:00</endMonitor>
         <date>21/11/2007</date>
         <begin>NULL</begin>
         <end>NULL</end>
         <ta id="TP-1-TA1">
            <parameterID>02</parameterID>
            <interval>20.0</interval>
            <maxGap>5.0</maxGap>
            <taType>NegLShift</taType>
            <threshold>85</threshold>
         </ta>
         <ta id="TP-1-TA2">
            <parameterID>BP</parameterID>
            <interval>20.0</interval>
            <maxGap>5.0</maxGap>
            <taType>NegLShift</taType>
            <threshold>24</threshold>
         </ta>
         <alarm>
            <status>F</status>
         </alarm>
      </rule>
   </patient>
</patient_ecs>
```

Figure 5.6: Patient specific Event Correlation Specification (ECS), illustrating a single patient who is associated with 1 *parent* temporal abstraction rule (*LowO2_BP*) containing 2 *child* temporal abstractions.

- *Patient*: The *patient* tag contains an identification number attribute (<*name id* = “”>) making the patient a unique domain entity. The patient's name is also included (<*name*>), as are the number of rules (<*numRules*>) currently logged for this patient.

- *Rule*: The *rule* tag contains information which pertains to an individual TA rule. The identification number “TP-1” implies temporal pattern number one. Further rules are identified by “TP-2”... “TP-N”. Tags exist for the name of the rule (<*ruleName*>), the relevant monitoring period and day during which the rule can be active (<*startMonitor*>), (<*endMonitor*>) and (<*date*>). The number of constituent child temporal abstractions is specified by (<*numTas*>) and the boolean tag (<*ruleActive*>) relates to whether the rule is currently executing. The (<*timeout*>) tag is either true or false depending on whether the rule has been executed for the entire monitoring period.

- *TA*: The *TA* tag contains information relevant to a temporal abstraction. This information is required by PMMs to enable them to configure their temporal model for the abstraction process. The physiological parameter on which the TA acts is specified by (<*parameterID*>). Abstraction interval time is given by (<*interval*>), threshold value by (<*threshold*>) and TA type by (<*type*>). The maximum gap allowable within the interval is specified inside the (< *max_gap* >) tag. Maximum gap is a property which temporal interpolation mechanisms within the PMM require in order to bridge contextually similar intervals together. Section 4.4.1 contains a detailed description of the temporal model utilised by PMMs for performing TA.

- *Alarm*: The *Alert* tag contains the single child tag (<*status*>). This a boolean type and indicates the final result returned from the PMM responsible for TA rule execution.

5.1.3.3 Rule and Query Builder

The Rule Builder provides the necessary functionality for obtaining TA rule parameters from the user and supplying them to both the ADR and the TA Rule Base enabling construction and instantiation of temporal abstraction rules and their associated instances within the ADR. It also offers an interface to a range of queries, described in section 5.2, which return the status of TA rules in the system.

Essentially, the Rule Builder obtains data from the user which is then inserted into ADR object-based data structures representing each of the ECS tags. The rule, with associated parameters, is also added to the TA Rule Base for permanent storage and querying of TA Rule results; this is explained in section 5.2.2. With reference to Figure 5.6, the only fields which are not supplied by the user are *rule id*, *ruleActive*, *date*, *begin*, *end*, *ta id* and *alarm result*. The *rule id*, *date* and *ta id* are automatically generated by the system. *RuleActive*, *begin*, *end* and *alarm result* are initially set to *T*, *NULL*, *NULL* and *F* respectively and are modified by the ADR upon the return of results from the ESP Server; after the rule has been executed.

5.1.3.4 Rule and Query Builder Interface to ADR

The Rule and Query Builder provides an interface for access to the functionality of the ADR. The following five points summarise the principle functionality which is exposed:

1. **Queries**: An explanation of available queries is provided in section 5.2 under the heading of "Querying the MOTA Framework".
2. **Rules**:
   (a) *EnterPatient*: Enters a new patient given the patient identification number.
   (b) *EnterRule*: Enters a new TA rule for a patient given the patient identification number.
   (c) *DeletePatient*: Deletes a patient given the patient identification number.
   (d) *DeletePatientAndRules*: Deletes a patient and their associated rules given the patient identification number.
   (e) *TerminateRule*: Terminates a rule that is currently executing given the rule identification number and patient identification number.

The following six points of functionality are automatically invoked once a patient and a rule is entered:

1. Enter the temporal abstraction *child* parts of a rule.
2. Instantiates a new MAM Rete engine for each entered rule.
3. Generates an alert rule for each entered rule and adds it to the MAM Rete engine.
4. Adds new rule/patient information to global ECS and creates a new patient specific ECS.

5. Inserts TA rule into the TA Rule Base.

6. Establishes communication with the ESP Server and sends rule to Transformer for compilation into an executable script for use by PMMs.

7. Receives results from PMMs at termination of monitoring phase and modifies the *status* field of the Alarm object to true if the TA was found to exist by a PMM. This will automatically trigger an alert within the MAM.

5.1.4 Interface to Analytical Processor

As mentioned previously, the Analytical Processor mines stored physiological data captured from bedside monitors searching for associations between parameters which may be indicative of serious impending clinical situations. This component is currently being researched in parallel with the work proposed by this thesis. Providing a link between the Analytical Processor and the MOTA framework facilitates the integration of data mining and data abstraction processes which have previously existed as separate fields of research [9].

The Analytical Processor will mine stored patient data for rules of a form similar to those that drive the TA mechanisms in the MOTA framework. Rules will conform to the general form of "If *condition_1* and *condition_2* then *clinical result*", where *condition_1* and *condition_2* are temporal abstractions on the data and *clinical result* is the likely consequence of the precedents holding true. The representation of this type of rule in the MOTA framework is multifaceted; the ADR provides a object-based runtime model and the ECS furnishes a machine readable permanent document, both of which are depictions of the rule. In order for the Analytical Processor to interface to the MOTA framework, it need only map its rule output to the structure of the ECS and the ADR can then utilise the Rule and Query Builder interface to read the rule in order to create its runtime model. Once instantiated inside the ADR, other processes proceed as for a 'clinician-entered' rule. This requires a small module for parsing the XML format of the rule and is a simple addition to the existing framework.

5.1.5 Event Stream Processor (ESP)

The Event Stream Processor (ESP) provides an environment for integrating and temporally abstracting high frequency time-stamped data streams. It is employed in the process of Intelligent Data Analysis (IDA) [8] and presents a link between raw data and domain knowledge where qualitative descriptions can be "matched" with underlying temporal elements within the data. It provides the results of this process of temporal abstraction to the ADR which then facilitates MDTA and intelligent alerting via the MAM. The ESP offers a platform for the execution of research hypothesis 3 (section 1.3.3) through the provision of a set of PMMs, each endowed with an executable temporal model based on the Event Calculus.

The distributed environment of the ESP allows dissemination of TA processing to individual PMMs, one assigned to each data stream-set (patient). As PMMs run as separate processes it is desirable to have each one running on a standalone small footprint machine at the bedside so as to provide immediate feedback to clinical staff on the floor. However, if so desired, it is also feasible to locate PMMs in the central NICU server room along with other components of the MOTA framework. This provides a different operating scenario, where PMMs are not providing alert notification to clinical staff at the bedside but rather analysing physiological patient data which could be streaming in from remote locations, as in the case of the e-Baby project where data is sourced from isolated hospitals and analysis is performed in the referral centre by an expert neonatalogist. An ESP Server acts as a central mediator between PMMs and the ADR and also serves to store a table of available PMMs and their associated identification codes. These codes correspond to the patient identification code which they service.

5.1.5.1 Patient Monitor Module

Each patient data stream set is monitored by a PMM, which has the responsibility of temporally abstracting an individual patient's data according to TA rules previously defined by a clinician or obtained through data mining processes.

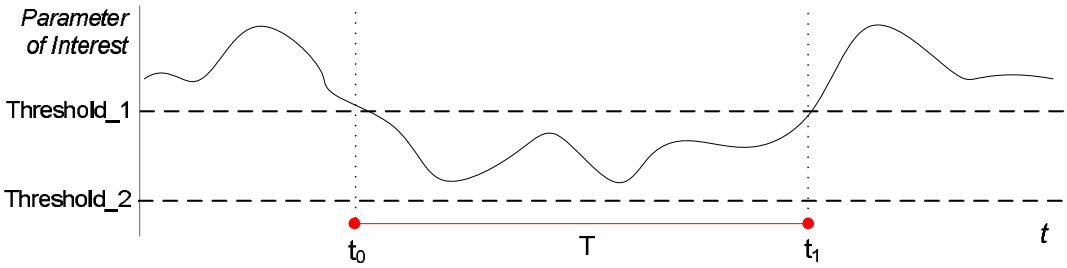

Figure 5.7: A *Stationary* abstraction. Interval T defines a period of time where a parameter is in between pre-defined upper and lower thresholds.

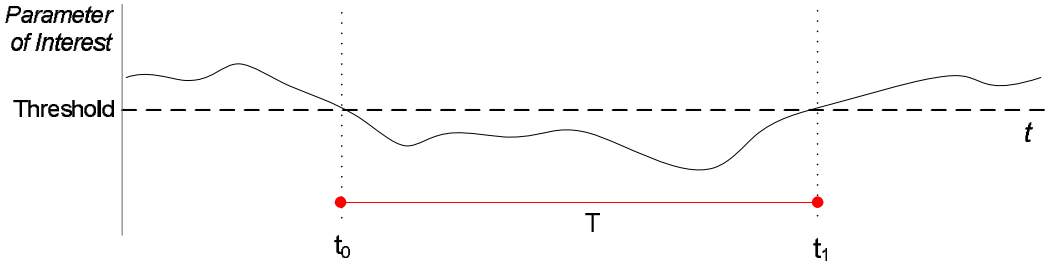

Figure 5.8: A *Negative Level Shift* abstraction. Interval T defines a period of time where a parameter is less than a pre-defined threshold.

PMMs currently have the ability to abstract data according to two primary types: 1) *Stationary:* The value of a variable is below an upper threshold and above a lower threshold value, ie; within a given range. (See Figure 5.7) 2. *Level Shift:* Values are classified as being either above or below a particular threshold (positive or negative level shifts respectively) - See Figure 5.8. PMMs intelligently abstract data according to a temporal model designed around the Event Calculus (EC) [22, 148]. The model incorporates axioms for temporal interpolation allowing specification of reasonably complex TAs and can operate on any stream of time-stamped data values. Commonly these will be physiological parameters received from monitoring equipment connected to the patient but could also be derived or calculated values that are output from intermediate processing agents that calculate heart rate variability (HRV), or area under the curve (AUC) for example. In this way, it is possible to abstract intervals of time where HRV or AUC is less than or greater than a particular threshold. One of the possible early indicators of sepsis is reduced HRV [6] and Hypocarbia and Hyperoxia, both measured using AUC techniques, have been shown to be risk factors

for Periventricular Leukomalacia (PVL) [7].

5.1.5.2 Rule division and evaluation

PMMs receive TA rule scripts from the ESP Server which are directly executable within their rule engine environment. Using the example TA rule previously described in section 5.1.3.2, the following describes the processes internal to a PMM.

The Event Calculus is a formalism for reasoning with *events*, *fluents* (properties) and *time points*. A fluent is simply a temporal property of the domain being modelled and is initiated by an event which occurs at a particular time. The fluent is assumed to hold until another event terminates it. Typically, a set of first order clauses are used to define a temporal model, which is used to describe some temporal world. I employ the EC to model online temporal abstractions where a fluent is *representative* of the TA and domain events are responsible for initiating and terminating the fluent. A complete formal coverage of the temporal model used by this research was provided in chapter 4.

The TA rule is decomposed into conjunctive clauses as follows:

> 1a. If mean blood pressure (BP) is less than 24mm for a period of 20 seconds or more and
> 1b. If blood oxygen saturation (SaO2) is less than 85% for a period of 20 seconds or more then alarm.

Each of the above clauses constitutes a temporal abstraction on the data. Using the model proposed in chapter 4, I model each conjunctive TA as a fluent in the EC, see *TA*_1 and *TA*_2 in Figure 5.10. For the TA rule to hold true, I introduce another fluent, the *Warning* fluent, to model the time when both constituent fluents hold. The final fluent, *Alarm*, begins to hold when the *Warning* fluent has held true for 20 seconds.

Upon reception of a TA rule script the PMM creates a *Stream Monitor* for each physiological parameter, see Figure 5.9. The *Stream Monitor* contains functionality

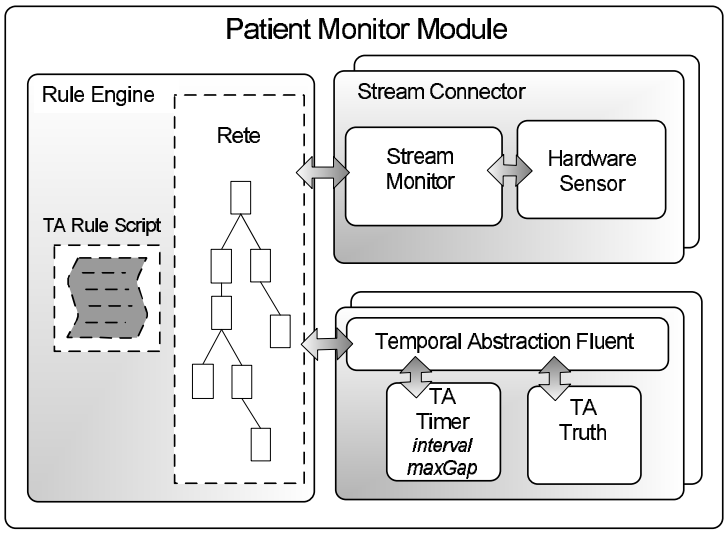

Figure 5.9: Patient Monitor Module.

for connecting to the required data stream and also for reading stream data values and time-stamps at a pre-determined rate. There will be two *Stream Monitors* for the example rule, one for connecting to the blood pressure stream and one for the SaO2 stream . Presently the sample rate is set at 1Hz, which corresponds to the rate at which the actual readings are accrued at the sensor-skin interface. The *Stream Monitor's* job is to supply these values to a single Rete engine thread which is spawned within the PMM for each multi-part TA rule. In addition to the *Stream Connector*, a *Temporal Abstraction Fluent* object is instantiated for each part of the TA rule plus one for the *Warning* fluent, within which there are classes for modelling the time relevant to abstraction intervals and the "truth" values of TA fluents; whether they presently hold or do not hold.

As soon as data values become available and all components are up and running, the *TA Rule Script* is executed inside the single Rete engine and monitoring of data samples and updating of fluent truth and time values begins. The TA rule script is executed on the data for a predefined period of time given by the *startMonitor* and *endMonitor* parameters of the ECS.

As previously described, as soon as the interval timer for the *Warning* fluent reaches the time of 20 seconds, the *Alarm* fluent begins to hold and the PMM will

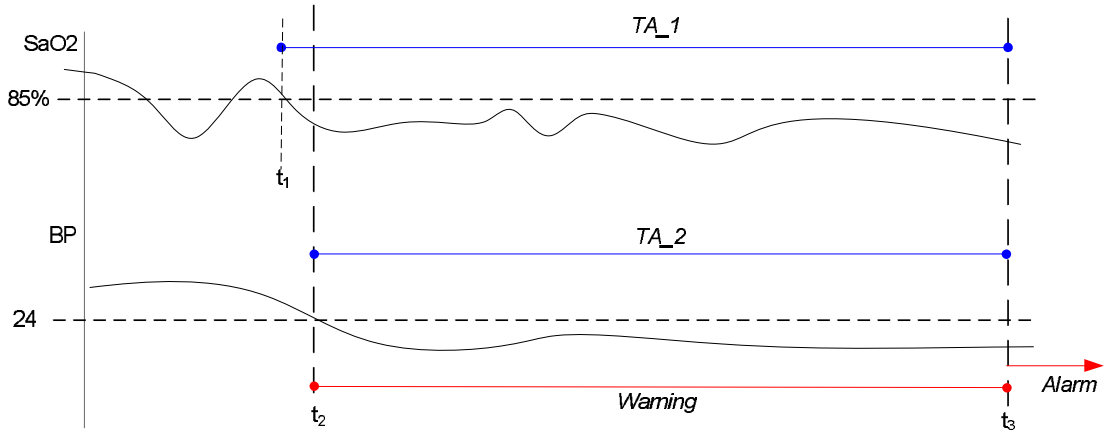

Figure 5.10: Temporal Abstraction example. When SaO2 and BP abstractions, *TA_1* and *TA_2* hold, the *Warning* fluent also holds. When time $t_3 - t_2$ reaches 20 seconds, the Alarm fluent is initiated and alerting mechanisms are activated.

conclude a positive finding, that is to say the abstraction has be found to exist in the data. The relevant information is then communicated to the ADR. The ADR fields *ruleActive*, *begin* and *end* attributes of the *Rule* object and the *status* field of the *Alert* object will be updated to *true*, t_2, t_3, T respectively to reflect the rule's completion status. The MAM's internal rule engine instantly detects the F condition for the *result* field and the alert process is activated via the MAM. The ECS is also modified to reflect the final rule status.

In the case of one of the constituent fluents terminating before 20 seconds has expired, then the *Warning* fluent also terminates and the PMM returns a negative finding, writing an *false* to the *result* field and *NULL* to *begin* and *end* fields. I emphasise this description is only a summary of the logic utilised to derive the temporal abstraction, a complete formal definition was provided in chapter 4.

5.2 Querying the MOTA Framework

TA rules which are originally input by clinical staff are essentially *temporal queries* upon patient data sets. These are entered using functionality within the *Rule and Query Builder* and, after a series of transformation phases, are executed by PMMs on the data. As described in section 5.1.5.2, a rule may return either *true* or *false*,

indicating whether the abstraction was found to exist, or not. At present, functionality of the ADR is exposed through the Rule and Query Builder interface without graphical aids. However, the addition of a graphical interface would simply call the methods defined in the interface and utilise their return values.

Using the example TA rule provided in section 5.1.3.2, which defines a general morbidity predictor of a decline in blood pressure below 24mm/Hg and a drop in SaO2 below 85% for a minimum period of 20 seconds, I describe the possible resulting states within the ADR.

1. *The TA has been detected*: The ADR fields *ruleActive*, *begin* and *end* attributes of the *Rule* object and the *status* field of the *Alarm* object will be updated to $false$, t_2, t_3, $true$ respectively to reflect the rule's completion status. t_2 and t_3 will reflect the actual start and end times of the abstracted interval where both blood pressure and SaO2 parameters match the above TA rule. In this case, an alert will be generated by the MAM.

2. *The TA has not been detected*: The ADR fields *ruleActive*, *begin*, *end* and *timeout* attributes of the *Rule* object and the *status* field of the *Alarm* object will be updated to $false$, $NULL$, $NULL$, T and $false$ respectively. There is no alert generated by the MAM.

The MOTA framework provides for two types of querying defined as *Immediate* and *Historic* queries. *Immediate* queries are those pertaining to TA rules presently being executed whereas *historic* queries are based on rules that have concluded.

Queries are executed on objects in a similar way to queries on a object-oriented database although temporal information is not explicitly represented in the ADR hence queries are not in the same vein as OODAPLEX [246] or Combi and Chittaro's work on modelling the Event Calculus using an object-oriented approach [58].

5.2.1 Immediate Querying

Immediate queries are executed upon the ADR which, at any instant, contains an internal representation of all active rules, TAs and patients within the domain. The ADR schema (Figure 5.4) allows queries to proceed based on the *patient* as the highest order entity. The following queries are available:

- Select all patients and rules, order by patient
- Select all rules for a particular patient
- Select all rules where alarm start time or end time is greater than, is equal to or is less than some value
- Select all rules where alarm result is true or false
- Select all TAs

Immediate querying provides a fast and efficient 'snapshot' view of current rule status across multiple data stream-sets (patients), multiple TAs, rules and data parameters.

5.2.2 Historic Querying

Historic queries are executed upon the TA Rule Base, which is a set of tables co-existing with tables for storage of physiological patient data within the Data Management Layer (Figure 5.11). Currently, historic querying has not been implemented although the following defines necessary design considerations to enable this type of query.

The TA Rule Base provides permanent storage of all rules and associated details that have been entered. It also contains clinical data related to patients such as gestational age and date of birth. These details are entered separately from the MOTA framework by nursing staff on admittance to the NICU ward. TA rules are related to patients through a relational schema shown in Figure 5.11.

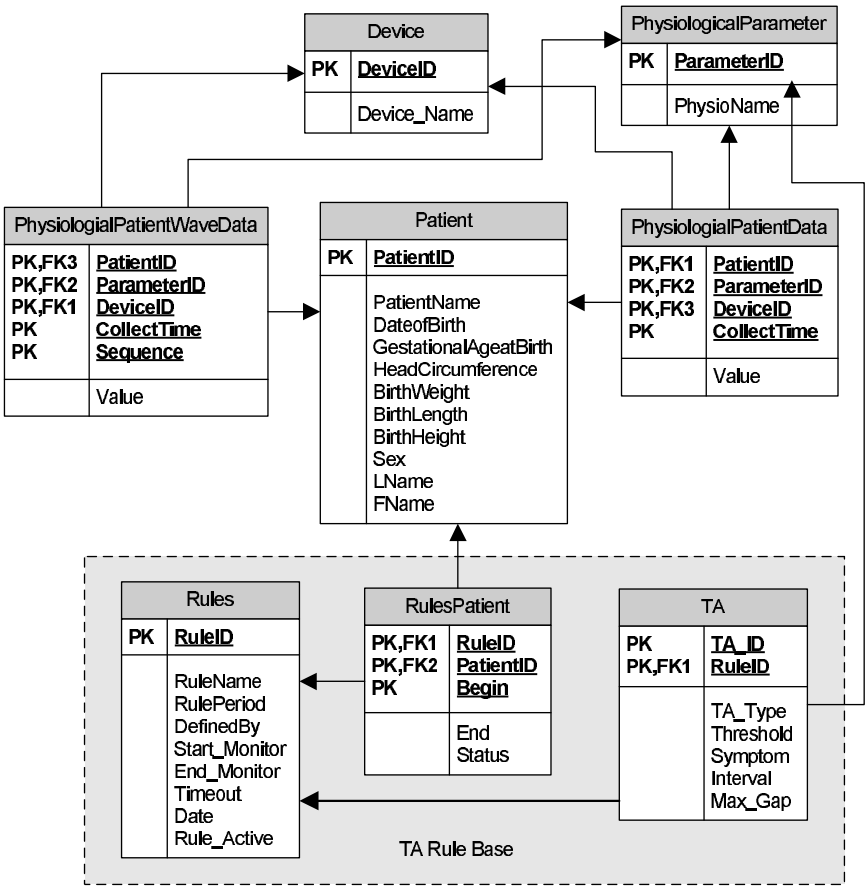

Figure 5.11: TA Rule and Physiological Database Schema.

As previously mentioned, the Data Management layer including the tables for storage of data, and not including the TA Rule Base, has been implemented within the e-Baby project [4, 27, 66] and enables storage of physiological data from bedside monitoring devices. Combined with clinical data, a profile for each baby can be constructed and summaries generated to support research and clinical initiatives [26]. This thesis extends the e-Baby data management layer to incorporate the TA Rule Base tables for the permanent storage of TA rules and to enable querying of patient physiological data where a TA has returned a *true* result.

Using the *patient id*, *rule id*, *start* and *end* values it is possible to retrieve the data sets constituting the temporal abstraction matching the original TA rule. Using the example rule defined in section 5.1.5.2, the following demonstrates an historic query upon patient data residing in the data management layer.

```
Select PatientID, ParameterID, DeviceID, CollectTime, Value
    From PhysiologicalPatientData, Rules, TA, RulesPatient
    Where PhysiologicalPatientData.PatientID = RulesPatient.PatientID
    And PatientID = "<patient id>"
    And CollectTime >= Begin
    And CollectTime <= End
    And Status = "true"
    And PhysiologicalPatientData.ParameterID = TA.ParameterID
    And Rules.RuleID = RulesPatient.RuleID
    And Rules.RuleID = TA.RuleID
    And Rules.RuleID = "<name of rule>"
```

Subsequent development of graphical tools could enable plotting and visual analysis of abstracted intervals by clinical staff.

5.3 Communication

Communication within the MOTA framework is carried out between the ADR Server, ESP Server and PMMs. All involved entities hold a communication protocol specifying the semantics of incoming messages and also the required responses. The ADR Server, ESP Server and PMM Servers can exist in a number of different protocol

states which coordinate the communications between these entities. Figure 5.12 illustrates the sequence of messages and protocol states existing within the ADR, ESP and PMM servers.

A separate communication area for each rule/patient pair is provided to cater for the dimensionality of the rules and data environment. For example, the following rule/patient pairs may exist:

Patient ID: 1A-1, RuleID: TP-1

Patient ID: 1A-1, RuleID: TP-2

Patient ID: 1A-2, RuleID: TP-1

Patient ID: 1A-3, RuleID: TP-3

The above denotes there are two rules to be executed on the data stream-set from patient 1A-1 (TP-1 and TP-2), one rule on patient 1A-2 (TP-2) and one rule for the data from patient 1A-3 (TP-3). This example ruleset will cause four protocol service threads to be spawned by each communicating entity, within which, communications are carried out by corresponding threads so that the state of execution relating to each rule/patient pair can be tracked. All messages contain the rule and patient identification codes relevant to the rule.

In the following, I define the communication between the ADR, ESP and PMM servers as protocol states and messages. Messages control the state which has been reached by that entity during the process of MOTA and states define discrete and specific periods of time between messages. There exists a number of points in the protocol where a mutually exclusive choice between two message sequences follows. I have abbreviated the description to concentrate on the successful completion of a particular patient/rule activation although the alternative case is catered for and includes a sequence of resulting states leading to erroneous termination of the process.

5.3.1 ADR Server Protocol States

The ADR Server receives input from clinical staff in the form of TA rules to be run over patient data stream-sets. In summary, upon reception of such rules, the ADR

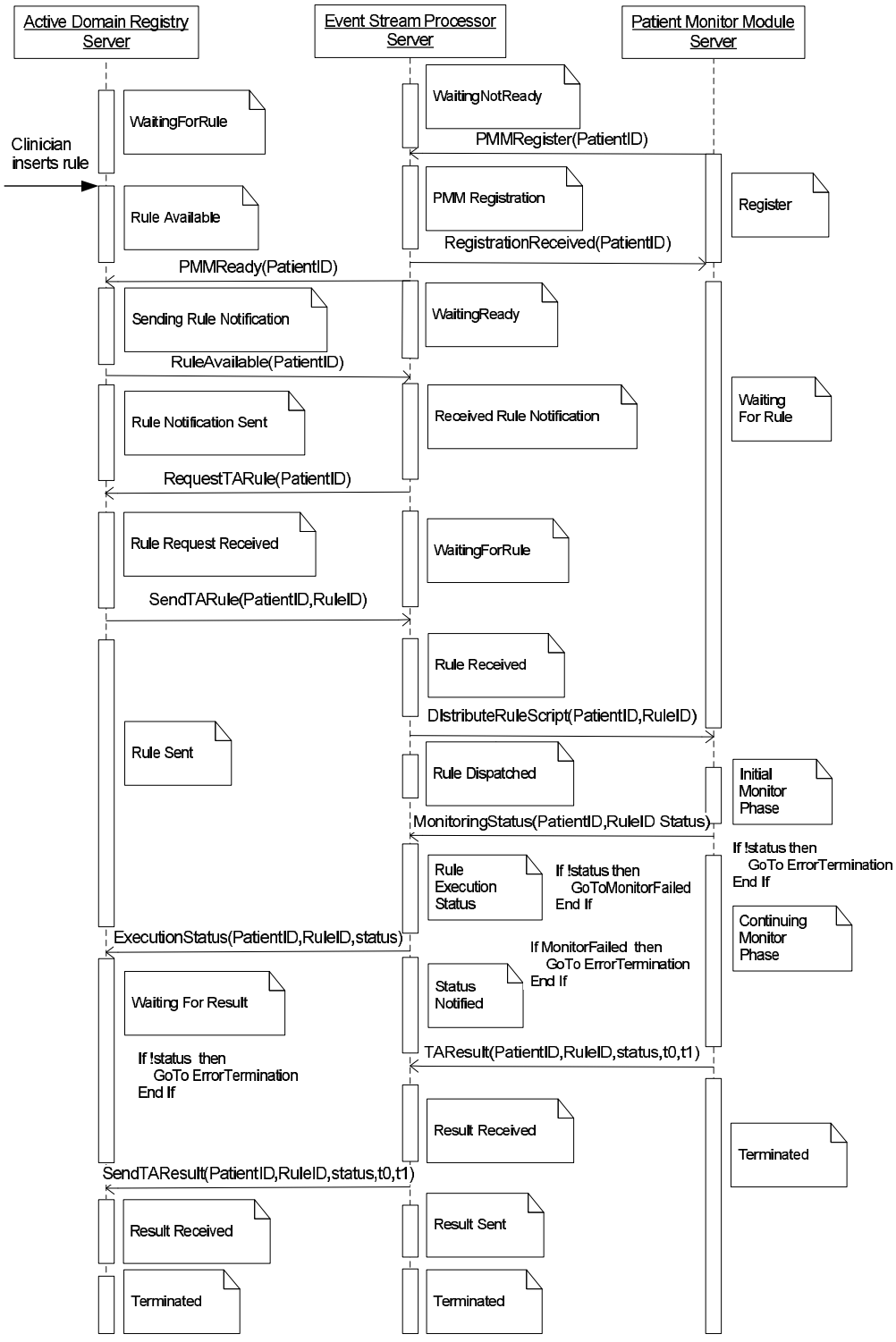

Figure 5.12: Message sequence and protocol states for the Active Domain Registry, Event Stream Processor and Patient Monitor Module Servers. Each server thread services a single rule/patient pair.

must notify the ESP Server of a pending rule request, wait for the acknowledgement from the ESP, send the rule to the ESP and finally wait until the ESP returns the result from the execution of the TA rule. The following protocol states have been defined:

- *Waiting For Rule*: The ADR is in a waiting state until a rule is entered by a clinician.

- *Rule Available*: ADR has received a TA input rule and now waits for notification from the ESP that a PMM is ready.

- *Sending Rule Notification*: The ADR now enters the protocol state where it sends a rule notification to the ESP, stating the relevant patient identification code of the PMM required for rule execution.

- *Rule Notification Sent*: Once the rule has been validated and stored in the ADR runtime object model and the TA Rule Base, the ADR is essentially ready to request the services of a PMM for execution of the rule on patient data. The ADR has notified the ESP of the presence of a new TA rule for a given patient or set of patients and waits for a request from the ESP to send the rule.

- *Rule Request Received*: The ESP has responded to the notification and requested the ADR send the rule.

- *Rule Sent*: The ADR has now dispatched the rule to the ESP and now waits until it receives the monitoring status of the PMM concerned.

- *Waiting For Result*: Monitoring status is received and ADR enters the *Waiting For Result* state. If the monitoring status returned is false, then the ADR will terminate execution of the rule. The ADR will wait for the length if time specified by the TA rule fields *startMonitor* and *endMonitor*.

- *Results Received from ESP*: The ADR has now received the results of rule execution from the ESP and can now integrate these into relevant ADR objects

for alert generation or querying. It also writes to the TA Rule Base and the ECS. If the status result is false then the ADR communication process will terminate in the state *Error Termination*.

- *Terminated*: Finally, the ADR enters the *Terminated* phase indicating the TA process has ended.

5.3.2 ESP Server Protocol States

The ESP Server is responsible for distribution of executable rule scripts to PMMs and also for storing a table of available PMMs. It incoporates the following protocol states:

- *Waiting Not Ready*: The ESP is waiting for a PMM to register, none have registered and it is not ready to receive rule notifications from the ADR.
- *PMM Registration*: At least one PMM has registered its patient identification code with the ESP indicating it is available for the execution of a TA rule.
- *Waiting Ready*: The ESP Server is waiting for rule notifications from the ADR.
- *Received Rule Notification*: The ADR has notified the ESP of a rule that is ready to be sent and the ESP has acknowledged with a request to send the rule. It is now ready to accept the rule.
- *Waiting For Rule*: The ESP has notified the ADR that it received the rule notification and has requested a rule be sent.
- *Rule Received*: The rule has been received from the ADR.
- *Rule Dispatched*: The rule has been dispatched to the PMM/s responsible for executing the rule script.
- *Rule Execution Status*: The ESP has received notification from the PMM/s that rule execution has either begun or failed. If it has failed then the ESP enters the *Monitor Failed* state which is then followed by the *Error Termination* state.

- *Status Notified*: The execution status has been sent to the ADR.
- *Result Received*: A result has been received from the PMM concerned with TA rule execution.
- *Result Sent*: The result has been sent to the ADR.
- *Terminated*: Finally, the ESP enters the *Terminated* phase indicating the TA process has ended.

5.3.3 PMM Server Protocol States

Patient Monitor Modules execute TA rules scripts on patient data stream-sets. They first register their availability with the ESP Server, wait to receive an executable rule script and then enter a monitoring phase where the script is executed according to the temporal model defined in chapter 4. The protocol states are as follows:

- *Register*: PMM has registered it's availability to execute TA rule scripts for its assigned data stream-set.
- *Waiting for Rule*: PMM has registered with the ESP Server and has received an acknowledgement from the ESP. It now waits to receive the TA Rule script.
- *Initial Monitor Phase*: The PMM has received the TA Rule Script and has attempted to enter the execution phase. If intialisation of the monitoring process is successful it enters the *Continuing Monitor Phase* and sends the execution status back to the ESP. If execution of the rule fails to initialise, it enters the *Error Termination* phase.
- *Continue Monitor Phase*: At the end of the monitoring period, the PMM send the results back to the ESP. These include the result status (*true* or *false*) and the start and end times of the temporal abstraction interval which was found in the data.

```
<message id="1A1-TP1">
    <name>RuleAvailable</name>
    <content>
        <patientID>1A1</patient>
    </content>
</message>
```

Figure 5.13: Message structure.

- *Terminated*: Finally, the PMM enters the *Terminated* phase indicating the TA process has ended.

5.3.4 Messages

Messages are sent using TCP/IP over HTTP connections between ADR Server and ESP Server and between ASP Server and each PMM Server. Each message contains a *message* field including the message identification attribute which consists of the concatenated patient and rule identification codes. This effectively identifies the message as belonging to a particular patient/rule activation. The *name* field depicts the name of the message. Also present is the *content* field, which contains the name of the message itself, for instance *RuleAvailable*, plus any additional parameters relevant to the message such as *patient ID*. Figure 5.13 shows the structure of a *RuleAvailable* message.

Messages are stored locally to each ADR, ESP and PMM server process to allow for a permanent transcript of the progress of each temporal abstraction process. During message exchange between communicating entities, the ADR prints a display of the current message engagement so that the user can visually track the process of rule execution.

5.4 Discussion and Summary

This chapter has introduced the MOTA framework whose goal is multi-dimensional online TA across multiple patient's data in real-time. The process of MOTA allows a ward of patients to be monitored for anomalous clinical states and the results amalgamated to generate clinical alerts as required. This has been accomplished

through the distribution of processes responsible for performing temporal abstraction on patient data streams and amalgamation of TA specifications and results using the concept of an Active Domain Registry. Communication between primary entities within the framework is based on a separate thread for each patient/rule pair allowing tracking of the progress of such TA activations. A TA Rule Base is part of a permanent physiological data storage area which facilitates *historical* querying of the results that TA rules return from monitoring processes. *Immediate* queries are also possible providing instantaneous feedback on the status of TA rules in the system.

I have addressed the following areas found to be requiring further research in the domain of TA within clinical data analysis, as outlined in [23] and detailed in chapter 2 of this dissertation. 1: data sample and abstraction frequency, 2: complexity available within abstracted patterns and 3: dimensionality of the TA and data environment. The fourth area identified as lacking is that the knowledge underpinning TA processes is currently not derived from sources other than a domain expert. This leads to the concept of mining stored physiological data for associations between parameters which may be used to predict the early onset of some clinical conditions. Although, not implemented in this work, there is the possibility of interfacing the output of the Analytical Processor, depicted in figure 5.1, with the ADR thereby providing a bridge between data mining processes and the real time abstraction mechanisms of the MOTA framework.

Chapter 6

Implementation and Evaluation

The architecture I have described in the preceding Chapter presents a general framework for performing temporal abstraction across a multi-dimensional data environment in an online fashion. I have used the case-study environment of neonatal intensive care to demonstrate my proposal however environments that give rise to streaming time-stamped data, such as industrial process control and financial market reporting services, represent possible application domains for multi-dimensional online temporal abstraction.

This chapter describes the implementation and evaluation of the MOTA framework. Architectural issues and design were discussed at a higher level in the preceding chapter; the presentation in this chapter is pitched at a lower level and deals with actual code design and arrangement of software objects. Implementation of the Patient Monitor Module (PMM), including aspects such as threading, the rule engine and its integration within the object-oriented organisation of a PMM, and the function of objects that model Event Calculus (EC) fluents, are detailed in section 6.1.1. The PMM is the MOTA framework component related to hypothesis 3 (section 1.3.3) which proposes that the EC can be used for online temporal interpolation and abstraction of time-stamped data. It is also linked to hypothesis 1 stating that a framework can be defined to enable multi-dimensional and online TA. The PMM is the component which performs single patient - single data stream-set TA defined as *Level One dimensionality* in section 1.2.

Description of the Active Domain Registry follows in section 6.1.2. The ADR provides for TA at *Level Two dimensionality*, that is multi-patient - multiple data stream-sets and in so doing supports hypothesis 1 and also hypothesis 2 which offers a means for semantic representation of the complex multi-dimensional relationships between data streams, parameters and TAs.

Communication mechanisms including the protocol and message format are presented in section 6.2. Threading details are also announced which allows for tracking of individual rule status from rule entry to rule termination. The format of messages is XML-based and I describe their semantics using an example; the *SendTAResult* message which contains information pertaining to the termination status of a rule. I show how the rule ID and patient ID are concatenated to form the message ID thereby uniquely identifying that particular rule/patient pair.

The evaluation of the MOTA framework prototype is offered in section 6.4. I set out to test that the proposals I have presented in this thesis do in fact work and do what they claim to do. The prototype which has been constructed is certainly not 'industrial strength' software but was robust enough to allow the gathering of sound evidence validating my proposals. I obtained physiological data which was recorded at the Nepean Neonatal Intensive Care Unit in order to expose the prototype to the most realistic scenarios for the NICU environment. The rules used for testing were acquired from leading Clinicians in the NICU and are indicative of serious degeneration in health status. Tests were executed and offered encouraging results as to the viability of the MOTA framework in performing TA in a multidimensional and high frequency context.

6.1 Implementation

This section describes framework implementation in terms of architectural issues, programming environments, class organisation and execution models.

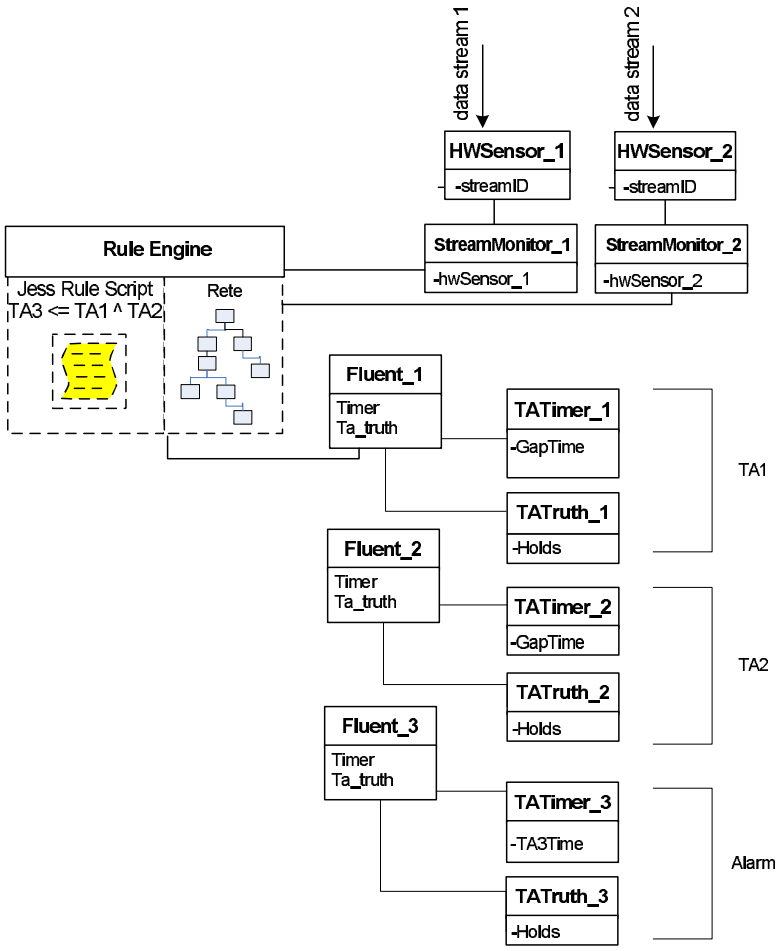

Figure 6.1: Patient monitor schema.

6.1.1 Patient Monitor Module (PMM)

The Patient Monitor Module (PMM) is a part of the Event Stream Processor (ESP). The responsibility of a PMM is to temporally abstract incoming time-stamped data according to pre-defined TA rules thereby addressing Level One Dimensionality. It must also provide a connection interface to the nominated hardware sensor which supplies the stream of data and communicate with the Event Stream Processor (ESP) Server. A high level overview of this component can be found in section 5.1.5.1 and a discussion of the message sequences which control the PMM in section 5.1.

In Chapter 4, I demonstrated how the Event Calculus can been used to temporally abstract a stream of time-stamped data values and perform temporal interpolation

between contextually similar intervals. The EC axioms have been implemented in a prototype forward chaining rule-based system that evaluates data points in an online fashion. PMMs have been programmed using the Java 5 programming environment and the Java Expert System Shell (JESS) [247] is used to implement the EC-based temporal model.

Figure 6.1 illustrates the classes which are instantiated for a TA rule specified as follows:

> TA1: Blood oxygen saturation (SaO2) less than 90% for a minimum time of 20 seconds. SaO2 may exceed 90% for a total of no more than 5 seconds during the 20 second interval
> TA2: Blood pressure less than 24mm/Hg for a minimum time of 20 seconds
> Warning $\leftarrow$ TA1 $\wedge$ TA2

This is the same example rule used in Chapter 4 to demonstrate the EC-based temporal model. There are three temporal abstractions in the above specification; *TA1* and *TA2*, which are constituent, or *child* TAs, of the parent TA called *Warning*. For each abstraction, there are 5 objects created; a *Fluent* object, a *timer* object (*TATimer*), a *truth* object (*TATruth*) and two objects for connection to the relevant data stream and provision of data values to the rule engine; the *HWSensor* and *StreamMonitor* objects. The *timer* object is responsible for tracking the time that the abstraction ventures out of range to enable temporal interpolation, whereas the *truth* object is a binary valued fluent set to either *true* or *false*, depending on whether the abstraction is currently valid. In the case of the *Warning* abstraction, the *timer* object tracks the total time that the *Warning* abstraction is valid; out of range excursions in the *Warning* fluent are not possible as it does not represent the actual trajectory of any data parameter, its existence is solely to provide a conjunction of *child* TAs. For this reason, provision of hardware sensor and stream monitor facilities is also not necessary.

A single *level shift* TA, such as *TA1* or *TA2*, involves a rulebase consisting of 4 rules for the level shift fluent itself and 2 for the final *Warning* abstraction. The rules react to dynamically modified facts in working memory which are instantiated through the use of the *timer* and *truth* objects. These Java objects are actually created as *JavaBeans* to allow their values to be inserted into the working memory of the JESS rule engine. This allows rules to react to the state of Java objects in the same way as facts created inside the rule engine environment. The rulebase, in the case of the example rule above, will consist of 10 rules; 4 each for TA1 and TA2 and 1 for Alarm.

The processes described here are executed on a single patient's data by a PMM. The design of the PMM is based on a threaded model and contains Java classes for connecting to the relevant data stream (*HWSensor*), sampling the data stream (*StreamMonitor*) and for storing the aforementioned dynamic truth and time values of fluents. A separate rete engine thread is spawned for each parent TA rule within which further threads are created for the classes just mentioned. Figure 6.2 depicts the spawning of threads for a TA rule with N conjunctive child TAs. Note, there will always be one extra thread for the *Warning* fluent.

If a maximum validity interval (MVI) has been determined for a TA rule, meaning the TA has been found within the data (See EC 28 section 4.6.1.2), the PMM stores relevant information such as: time-stamps relating to the start and end of the MVI (see t_4 and t_9 in Figure 4.5), the truth value of the *FinalAlarm* fluent; which will be *true* in this case, and the identification codes of the patient and rule. These are returned to the Active Domain Registry via the *TAResult* message allowing the ADR to then activate alarms and also amalgamate results from multiple PMMs thereby providing multi-patient temporal abstraction (MOTA).

6.1.2 Active Domain Registry (ADR)

The Active Domain Registry is the central core of the MOTA framework and provides a representation of principle domain entities and their relationships. It allows Level

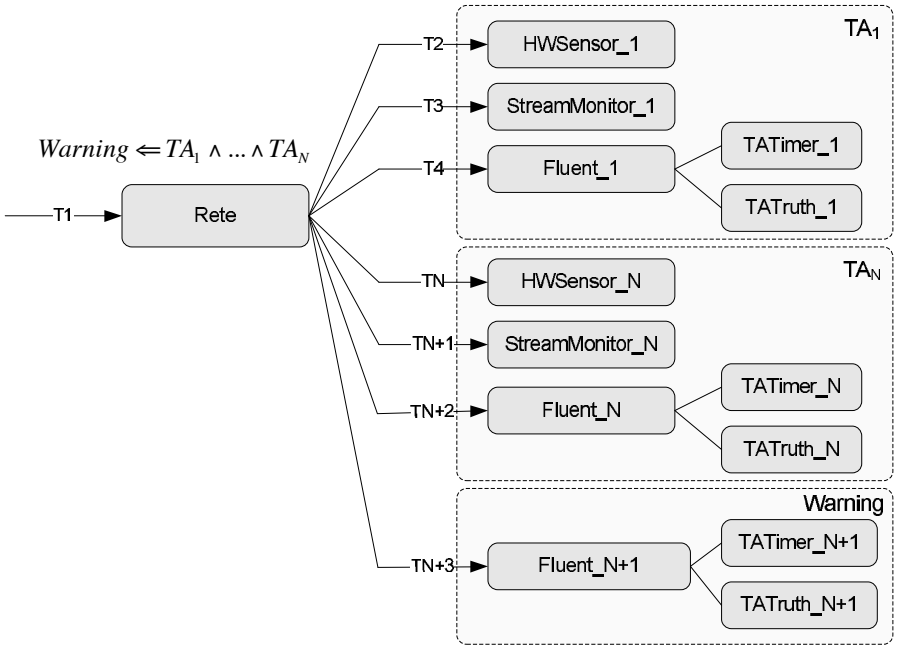

Figure 6.2: Patient Monitor Threads. One Rete engine thread is created for each parent TA rule consisting of N conjunctive parts. The Rete thread spawns further threads for connecting to the data source and monitoring time and truth values of abstraction fluents.

Two Dimensionality within TA processes and contains the Medical Alert Monitor (MAM) and Event Correlation Specification (ECS). As described in section 5.1.3, the ADR exhibits reactive behaviour through the MAM which triggers clinical alerts when a *true* result is returned from a PMM for a TA rule.

As illustrated by figure 6.3, the ADR models patients, rules, alarms and TAs. These are instantiated as vectored objects allowing the existence of multiple patients and numerous rules per patient, the rules themselves consisting of multiple TAs and alarms. The ADR incorporates the Medical Alert Monitor component (MAM), which employs a JESS rule engine for alarm activation based on the value of the *status* attribute in the *Alarm* object. When a TA rule is entered into the ADR, an alarm rule is automatically created by the MAM and added to the MAM internal Rete engine. There will be one JESS alarm rule for each TA rule entered to the system. This alarm rule will include a patient ID, TA rule ID, alarm ID and a status flag, the latter initially set to *false*. After the TA rule has been passed to a PMM and executed

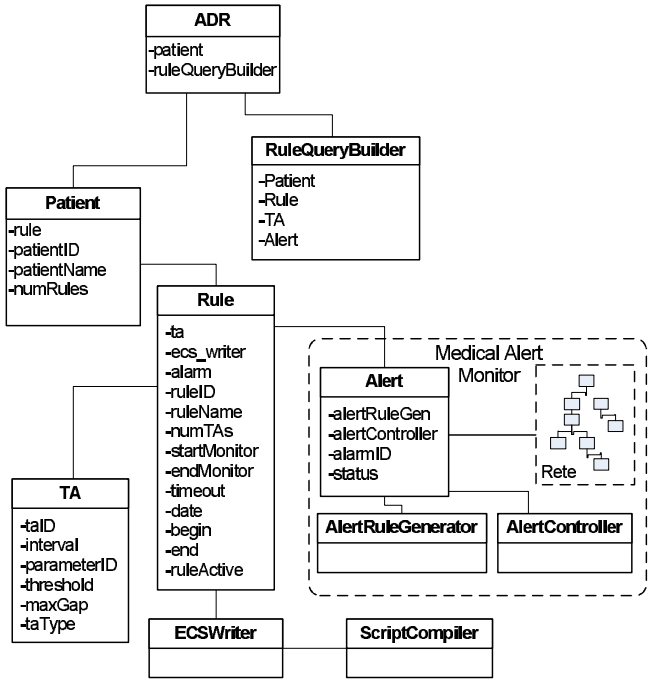

Figure 6.3: Principle Active Domain Registry objects.

on patient data streams, the status flag will be modified by the result returned from the PMM. If the abstraction was found in the data, the status flag will be set to *true* and as soon as this is detected by the ADR rule engine, an alert will be triggered. Using the same programming environment as the PMM, the ADR incorporates an object oriented design working together with the JESS rule engine, where facts are added to the Rete engine as *JavaBeans*.

As depicted in figure 6.3, the principle classes implemented in the ADR are *Patient*, *Rule*, *TA*, *RuleQueryBuilder*, *Alert*, *ECSWriter*, *AlertRuleGenerator* and *AlertController*. The Rule and Query builder provides an interface to the functionality of the ADR. This functionality was described in sections 5.1.3.4 and 5.2 and comprises querying capabilities, both *immediate* and *historical*, and rule addition, deletion and editing.

Rule related queries and functions, such as addition and deletion, are implemented as method calls to the ADR. Parameters are parsed from the input and methods are invoked allowing rules to be entered into the object-based runtime environment of the

ADR. The classes seen in figure 6.3 give rise to the instantiation of domain objects and their associated relationships.

The *ECSWriter* and *Script Compiler* work in conjunction with one another to provide both the ECS itself and executable TA rule scripts which are distributed to PMMs. The *ECSWriter* translates the ADR object-based runtime model of domain entities into an XML-based representation which is used for creating executable rule scripts and maintaining a permanent record of patients, TA rules, data streams, data parameters and alerts that have been entered into the system. The *ScriptCompiler* receives the ECS as its input and generates an executable rule script which is passed to the ESP which then forwards it on to the PMM/s where it is executed in the JESS rule engine environment. The *ScriptCompiler* is a parser that utilises the Document Object Model (DOM) to create a tree model of the input ECS. After iterating through the nodes of the tree, rule parameters are stored and written into the output script.

The Medical Alert Monitor utilises a JESS rule engine which receives alert objects from the ADR. As described in section 5.1.3.1, the MAM recognises and reacts to the *status* field within each rule entered into the system. The JESS rule engine operates in the same way as that described for the PMMs where object attributes are instantiated as JavaBeans which are then added to the working memory of the rule engine; the Rete network. Once existent in memory, rules are configured to "watch" the status attribute of alert objects and are triggered when it transitions from *false* to *true*. At the present time, a message is displayed on the ADR console indicating that a TA rule has been detected in the data of a specified patient although the output of the triggered rule is available for further alert applications such as SMS and pager alerts. This would require the addition of a third party toolkit and an HTTP-POST compliant SMS provider.

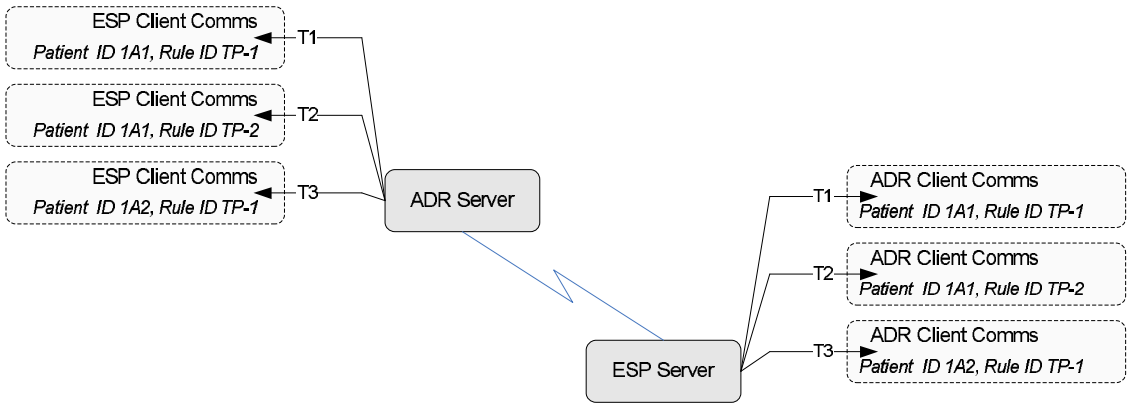

Figure 6.4: The threaded communication model. A separate thread is allocated for communications relevant to each rule/patient pair.

6.2 Communication

Within the MOTA framework, there are three principal communicating entities; the ADR Server, ESP Server and PMMs. Each has its own communication sub-system consisting of a class for establishing socket-based connections and spawning threads for the handling of requests, an XML parser which separates out message content and a protocol class for defining the current state of the communication process. Protocol states, message sequences and a high level view of communication mechanisms were defined in section 5.3.

All communicating entities maintain a separate thread of execution for carrying out communications relevant to a single rule/patient pair facilitating tracking of the process of a TA rule within the system. Given the following example rules, which constitute a total of three rule/patient pairs, figure 6.4 shows the threaded model of communication within the MOTA framework.

Patient ID: 1A-1, RuleID: TP-1

Patient ID: 1A-1, RuleID: TP-2

Patient ID: 1A-2, RuleID: TP-1

The diagram illustrates communications between the ADR server and ESP Server for two rules; TP-1 and TP-2. TP-1 is executing on the data from patients 1A-1 and

```
<message id="1A1-TP1">
    <name>SendTAResult</name>
    <content>
        <status>true</status>
        <t0>13.21.00</t0>
        <t1>13.21.20</t1>
    </content>
</message>
```

Figure 6.5: XML-based message structure.

1A-2 while TP-2 is only relevant to patient 1A-2. Communications between the ESP Server and the PMMs is similar to that shown in figure 6.4, where separate threads are responsible for each rule/patient pair. In this way, it is possible to maintain a record of the progressive status of rule execution, from rule entry through to the final results being returned from PMMs.

The discrete protocol states defined for the ADR, ESP and PMM Servers were stated in section 5.3. Each entity progresses through this sequence of states as messages arrive, eventually culminating in the return of TA rule results from PMMs.

6.2.1 Messages and Protocol Sequencing

Messages are XML-based and are structured according to figure 6.5 which illustrates the *SendTAResult* message. This message is sent to the ADR by the ESP server after it receives the result from the PMM concerned with the execution of rule ID TP-1. As described in section 5.3.4, the *message* tag contains a concatenation of the rule and patient identification codes so as to uniquely identify this particular process activation. The *name* tag specifies the name of the message itself while the content tag encloses the message content, which will vary from message to message. In this case, the content of the message are the start and end times of the temporal abstraction interval, t_0 and t_1, and the status field indicating a *true* finding. As a further example, the *SendTARule* message, which is sent by the ADR to the ESP, will consist of the same *message* tag, the *name* tag will be *SendTARule* and the content tag will contain the executable TA rule script itself.

Upon reception, messages are processed by a separate class which employs an XML

parser to extract information between the tags. Another class controls protocol transitions, which depend on the *name* of the message and the preceding protocol state. A simple algorithm utilising repetition and selection controls protocol sequencing for each communicating entity.

6.3 Installation and Initialisation

The MOTA framework is initialised through the use of a number of scripts. The ADR, ESP and PMM components are separate processes and hence are initialised separately. The only software requirement is that the host machine must incorporate the Java runtime environment. Testing and evaluation was carried out on a Windows machine running the XP Professional operating system with Service Pack 2, Pentium 4, 2.8GHz CPU and 1GB of RAM, so the recommendation would be to install on a similar or more recent machine.

Physical location of the software components is flexible, as mentioned in section 5.1.2. The suggested configuration is for the PMMs to be installed on 'Notebook' style computers at the bedside, where one machine will run two PMM processes; one for each data stream-set coming in from the Data Collection Unit (see figures 5.1 and 5.2. The ESP is located together with the ADR in the recommended configuration, although it can exist in a separate location as communication between itself, the ADR, and PMMs, is TCP/IP-based.

There are initialisation scripts for the ADR, ESP and PMM processes. These are run on the particular machine that hosts the process. It is advisable to initialise the ESP before the PMMs, as the PMMs will immediately search for the ESP upon start up. The ADR, once started, will display a command line console to the user where a menu allows entry and querying of TA rules.

6.4 Evaluation

The series of tests constituting the evaluation have been divided into three categories:

1. Multi- dimensional temporal abstraction of data from a single patient (multiple TA rules, *single* data stream set - Level One Dimensionality)

2. Real-time temporal interpolation.

3. Multi-dimensional temporal abstraction of data from multiple patients (multiple TA rules, *multiple* data stream sets - Level Two Dimensionality). This was depicted graphically in figure 3.3

The first of the above levels of TA dimensionality is assessed by providing a test routine for a single PMM, since the PMM is responsible for applying TA rules to the data from a single patient. Tests 1a and 1b are designed for this purpose. Test 2 examines the process of temporal interpolation and the ability for a PMM to successfully 'bridge' concatenable temporal gaps, or discard them, depending on the specified *MaxGap* parameter. To investigate the capability of the prototype to perform the higher level of TA dimensionality (multiple patients), it is necessary to involve the Rule and Query Builder, ADR, ESP Server and associated communications sub-system. Test 3 addresses these issues.

6.4.1 Single Patient Temporal Abstraction - The Patient Monitor Module

Our case study environment is the NICU at Nepean Hospital in Penrith Australia. Clinicians here are conducting research supporting the hypothesis that multiple aberrant behaviours across a number of physiological streams may have an additive effect and certain combinations thereof may indicate critical deterioration with a high likelihood of death or of brain injury. Aberrant behaviours are modelled as temporal abstractions by the system and defined by clinicians in the form of the example provided in section 6.1.1. Here, I state the complete rule description using plain English and include another rule involving heart rate and blood pressure, both provided by clinicians at the Nepean NICU.

TAR1: "Detect episodes of 20 seconds or more where blood oxygen saturation (SaO2) is less than 90% and blood pressure is less than 24 mm/Hg."
TAR2: "Detect episodes of 60 seconds or more where heart rate is less than 70 bpm and blood pressure is less than 24 mm/Hg."

Both of these rules, TA Rule 1 and TA Rule 2, define possible indicators of a grave degeneration in the baby's health. Defining rules with a greater number of parts than two is difficult at this stage as medical research is only just beginning to investigate the viability of the additive effect that multiple aberrant behaviours occurring concurrently may have on patient outcome. Each of the above rules can be separated into its child components as follows:

TAR1a: "Detect episodes of 20 seconds or more where blood oxygen saturation (SaO2) is less than 90%
TAR1b: "Detect episodes of 20 seconds or more where blood pressure is less than 24 mm/Hg."
TAR2a: "Detect episodes of 60 seconds or more where heart rate is less than 70 bpm"
TAR2b: Detect episodes of 60 seconds or more where blood pressure is less than 24 mm/Hg."

Rules TAR1a, TAR1b, TAR2a and TAR2b also indicate a deterioration in the health of the patient and these will be used in the test routine where it is necessary to execute *single part* rules. I have used this rule-set of 6 rules, which includes TAR1 and TAR2 and is defined as rule-set R, as a basis for performing the evaluation described in this and the following sections.

6.4.1.1 Method

A total of 17 de-identified data sets (one per patient) were obtained from the NICU at Nepean Hospital NSW containing time-stamped samples of the relevant parameters; oxygen saturation, blood pressure and heart rate. Data was originally sampled at a frequency of 1Hz and recorded for between three and five hours.

TA Rule	Valid Episodes	Invalid Episodes
TAR1a	6	2
TAR1b	3	0
TAR2a	3	0
TAR2b	0	2
TAR1	1	0
TAR2	0	0

Table 6.1: The total number of valid and invalid episodes, across all 17 data-sets, for each of the temporal abstraction test rules in rule-set R.

In all tests, the relevant data stream files (SaO2, blood pressure and heart rate) from each of the 17 data stream-sets were read by the PMM and each datum reasoned with and evaluated as belonging to the defined temporal abstraction or not. If the abstraction defined by the test rule was found to exist in the data, the PMM returned a *true* result for the rule along with the start and end times for the interval and the relevant rule and patient IDs. After returning, the PMM terminated. This method of TA has been termed *incremental abstraction* [117], where the abstraction is built incrementally, sample by sample. Results from the PMM under test were output to a console window in simple textual format for visual analysis.

Test 1a

Firstly, it was necessary to ascertain the exact whereabouts of the anomalous conditions; where they started and where they ended or in fact, if they existed at all. ADI Instruments *Chart 5* data analysis software was utilised for this part of the evaluation procedure. Table 6.1 shows the results of the visual analysis phase. In addition to the data contained in table 6.1, the start and end times of the interval and the patient ID (data stream-set ID) were recorded. Rules were then run individually on each data stream-set and results checked against recorded values. The gold standard for the tests was visual recognition and annotation of the intervals by a clinician as shown in Figure 6.6 .

Of particular note is that two of the rules, TAR1a and TAR2b, gave rise to intervals which were marked as invalid due to a transducer disconnection and/or patient movement causing disruption to the signal. The data streams were read by

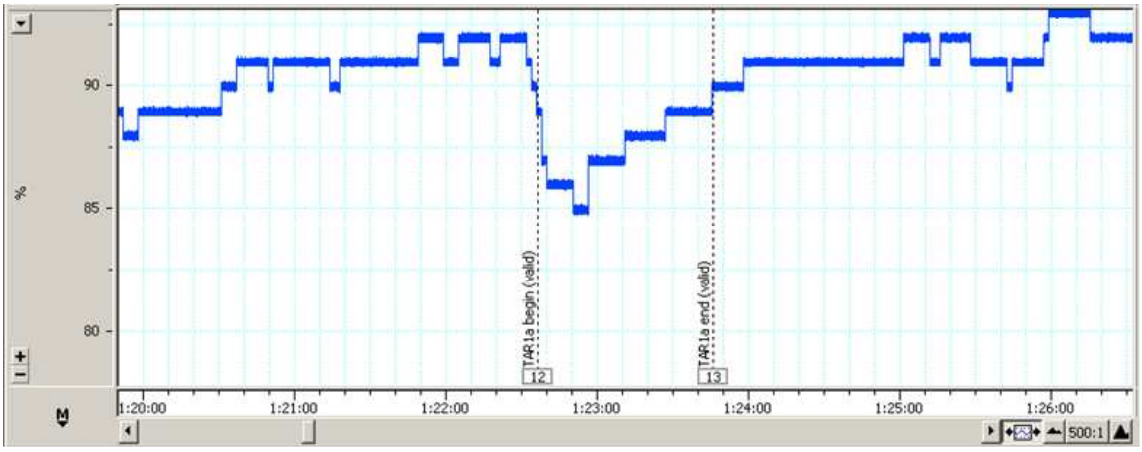

Figure 6.6: Screen capture of annotated patient data as seen in *Chart 5* data analysis software. An instance of rule TAR1a can be seen during its excursion below the threshold of 90%. A total of 6 valid TAR1a episodes were annotated by clinicians across the 17 patient data sets.

the PMM under test at a rate of 1Hz, approximating an online monitoring scenario hence each test required varying amounts of time to complete due to the differing termination times of the PMMs under test. PMM TA processes are designed to terminate when the abstraction is discovered, hence the staggered terminating times. Table 6.2 illustrates the TA rules to be used in the first test and also the method and expected results.

Test 1b

Originally having only the TA rules described above, more rules were required in order to enable satisfactory testing of the system under load (higher levels of TA and data dimensionality). It would not have been clinically acceptable to simply execute multiple duplicate copies of the original rule as there is no reason for more than one instance of the same rule to be executed on the same patient's data stream-set at the same time. Instead, I have defined new rules which are similar to the originals except that they exhibit slightly different threshold values. These new TA test rules have not been validated as clinically relevant but are a practical way to test the system under increasing load whilst adhering to the 'general format' of the clinically defined rules. For the additional rules, I observed the range of values for the parameters involved

Test	**TA Rule**	**PMM TA Process**	**Method**	**Expected Result**
1.1	TAR1a	SaO2 $< 90\%$, T > 20s	Rule executed individually on SaO2 stream from each of the 17 data stream sets	Detection of the first instance of the interval with correct start and end times. Verified through visual inspection of data sets.
1.2	TAR1b	BP < 24 mm/Hg, T > 20s	Rule executed individually on BP stream from each of the 17 data stream sets	As above
1.3	TAR2a	BP < 24 mm/Hg, T > 60s	As above	As above
1.4	TAR2b	HR < 70bpm, T > 60s	Rule executed individually on HR stream from each of the 17 data stream sets	As above
1.5	TAR1	SaO2 $< 90\% \wedge$ BP < 24 mm/Hg, T > 20s	Rule executed on SaO2 and BP streams from each of the 17 data stream sets	As above
1.6	TAR2	HR < 70bpm $\wedge$ BP < 24 mm/Hg, T > 60s	Rule executed on HR and BP streams from each of the 17 data stream sets	As above

Table 6.2: Test 1a method using the original clinical rule-set and expected results. The rule-set constitutes rule-set R.

Rule-set	Parameter		
	SaO2	BP	HR
R	90	24	70
R^1	88	26	80
R^2	92	28	90
R^3	94	30	100
R^4	96	32	110

Table 6.3: Extended rule-sets to be used for Test 1b. R is the original clinically defined rule-set. The threshold value for extended rule-sets were modified according to the table.

and altered the threshold values of the rules according to the following:

> *Blood Oxygen Saturation (SaO2)*: Data-sets exhibited a range of values from 85% to 98%. Hence, to extend rule TAR1a, I defined a further 4 rules with thresholds values of 88, 92, 94 and 96% in addition to the original rule which had the threshold set at 90%
>
> *Blood Pressure (BP)*: Data-sets exhibited a range of 22mm/Hg to 40mm/Hg. TAR1b and TAR2b were extended to produce a total of 8 new rules having thresholds of 26, 28, 30 and 32mm/Hg.
>
> *Heart Rate (HR)*: The range in HR values varied between 65 and 150bpm. Rule TAR2a was extended to include thresholds of 80, 90, 100 and 110bpm.

The above adjustments yielded a further 4 rule-sets to be used in Test 1b for evaluating performance under higher load conditions. I define the original rule-set as R and further extended rule-sets as R^1, R^2 and so on. The rule-sets are defined in table 6.3. Table 6.4 illustrates the TA rules to be used and also the method and expected results.

During this test, all 5 rule-sets were executed concurrently on a single data stream-set comprising the relevant individual data streams SaO2, BP and HR. The test was then repeated on the next data stream-set, and so on. As for the original rule-set and Test 1a, prior visual analysis of all data-sets was carried out using the extended rule-sets to ascertain start and end times of valid intervals. *Disconnect* events and patient movement events invalidated a total of five intervals although these did not

Test	PMM TA Process	Method	Expected Result
2.1	$R + R^1 + R^2 + R^3 + R^4$	Multiple rule-sets (total rules = 30) executed on each of the 17 data stream-sets, one data stream-set at a time.	Correct detection of the first instance of the relevant interval with correct start and end times. Verified through visual inspection of data sets.

Table 6.4: Test 1b method using extended rule-sets R^1, R^2, R^3 and R^4 for multi-dimensional temporal abstraction and expected results.

occur as the first interval encountered in the data-set concerned which meant that they would go unnoticed during the test.

The returned values for start and end times were checked against the initial recorded results to determine accuracy. The object of the tests was to establish if and when the 1 second time available for processing would be breached as the dimensionality of the rule and data environment is increased, both in terms of the number of data streams (data dimensionality) and the quantity of temporal abstractions (TA dimensionality).

Test 2

Finally, it was necessary to test temporal interpolation mechanisms to ensure temporal gaps were either successfully 'bridged' or, when the maximum gap quantity was exceeded, the abstraction terminated and a *false* result returned. These mechanisms were tested through the use of a single-part TA rule with varying values for the maximum allowable 'out of range' time *MaxGap*. For this test, the original NICU data set was modified to include varying sizes of temporal gap quantities and allowed a more rigorous testing of the interpolation method as opposed to the original data. The rule employed for the test was as follows:

> TAR3: SaO2 $< 90\%$ for 20 seconds and $MaxGap = 2$ seconds. The value of *MaxGap* was increased to 4 seconds, 8 seconds and 15 seconds.

It was assumed that potential pitfalls involving the temporal interpolation method

Test	TA Rule	PMM TA Process	Method	Expected Result
3.1	TAR3a	SaO2 < 90%, T > 20s MaxGap = 2s	Rule executed individually on each of the 17 single data streams	Intervals either side of temporal gaps are concatenated and the rule returns a valid abstraction.
3.2	TAR3b	SaO2 < 90%, T > 20s MaxGap = 4s	As above	As above
3.3	TAR3c	SaO2 < 90%, T > 20s MaxGap = 8s	As above	As above
3.4	TAR3d	SaO2 < 90%, T > 20s MaxGap = 15s	As above	As above
3.5	TAR3e	SaO2 < 90%, T > 20s MaxGap = 15s	Data modified. 15s gap = 3 × 5s gaps	As above

Table 6.5: Test 2 method for temporal interpolation and expected results.

would occur when there are numerous small gaps that require individual counting at the same time as a progressive sum being stored. Also, the initialisation and termination of EC fluents each time a gap is encountered may have had a negative effect on processing time. To ascertain whether such problems existed, data was fabricated where the total temporal gap time within a valid abstraction interval varied from one long interval of the same value as *MaxGap*, to numerous shorter intervals which totalled the value of *MaxGap*. To test for misinterpretation of gap times, the dataset was modified again so that the total temporal gap quantity was slightly less than and also slightly larger than the *MaxGap* time. This tested the interpolation mechanisms around the boundaries of the specified gap time. Table 6.5 summarises the procedure used for testing temporal interpolation processes. The other aspect requiring evaluation is the ability of a PMM to abstract data with the inclusion of *temporal gap* quantities. The mechanisms for temporal interpolation around such gaps are potential areas for failure and hence require testing.

Tests were carried out on a stand-alone desktop machine running Windows XP professional operating system and housing a 2.8 GHz Pentium 1V processor with 1GB

of memory.

6.4.1.2 Results

Test 1a

Test 1.1 to 1.6 all resulted in the return of correct start and end times for the first occurring valid interval in each of the 17 data-sets. There were no particular computational problems envisaged for this test, it being more of a low level test to ensure TA mechanisms operated at the simplest level. The results demonstrate that the PMM can detect the temporal abstractions specified by the TA rules and do so within the one second processing window which also establishes its viability for an online monitoring scenario where data is sampled at a rate of 1Hz.

PMMs terminate as soon as they detect an interval corresponding to the TA rule and the intervals which were originally marked as invalid, indicating a disconnection event, occurred *after* the first valid interval. This was the case in all data-sets hence the test routine did not directly reveal problems associated with the lack of a method for detection of *disconnect* events or invalid regions of data (false positives) related with patient movement.

Test 1b

Test 1b involved some sharing of data streams between multiple TA process threads. Each temporal abstraction instances a *HWSensor* and *StreamMonitor* class for interfacing with the data stream being analysed. Rules TAR1, TAR2, TAR1b and TAR2a all require access to the same blood pressure stream and must read and process data within the allocated one second window. It was hypothesised that there would be a point where the one second processing window would be insufficient for all threads to access the same data stream and reliability of results would decrease. For the 30 rules that were run during this test, there was no indication of data points being missed as the returned start and end times for detected intervals was correct.

A relevant complexity measure in rete-based rule engines such as JESS, is the effect that the size of memory occupied by rules has on the time taken for one rule

to fire. It is stated to be $O(P)$ worst case and $O(\log_2 P)$ best case [248], where P is the number of productions, or rules, in production memory. I employ a separate Rete engine thread for each parent TA rule. Assuming many rules will either be single-part or two-part rules, there will be at most ten rules in working memory at any one time (see figure 6.2) which imposes no appreciable constraints on the one second reasoning cycle.

JESS utilises an enhanced version of the Rete algorithm to match rule patterns against fact objects in working memory. The worst case time complexity of the original Rete algorithm for a rule firing is $O(W^{2C-1})$ and best case $O(1)$, where W is the number of fact elements in working memory and C is the number of patterns in the rule [248]. In practice, Rete never performs this badly and depends heavily on how rules are written. In JESS, most pattern matching is done in linear time or better with respect to the size of working memory and absolute worst case complexity is OW^C, however optimising the structuring of patterns in the LHS of rules will generally eliminate the exponent C [247].

Test 2

The method for temporal interpolation requires a timer (*TATimer*) fluent which attains a value equating to the length of time where data is 'out of range'. The fluent 'stops' and holds it current value whenever data is not 'out of range'. The reasoning behind this was explained in section 4.6.1.1 where the *Trajectory* and *Releases* predicate are involved in allowing the timer fluent to exhibit a changing value over time. These extra fact elements increase the size of working memory but not to the point where rules are delayed from firing so that the 1Hz reasoning cycle is compromised. Tests 3.1 to 3.5 operated successfully and temporal gaps, or 'out of range fluents', were interpolated according to the TA rule specification. For example, TAR3e specified an allowable gap of 15 seconds within the 20 second interval which was made up of 3 smaller gaps of 5 seconds each. Although unrealistic in clinical terms, the rule tested interpolation mechanisms could concatenate the 4 separate sections where the abstraction was valid. Interpolation was also verified as being accurate so far as

correct interpretation of gap times. For example, where *MaxGap* was set to equal 4 seconds, the data set 'gaps' were modified to 3 seconds and 5 seconds. This tested the counting mechanisms and rules responsible for temporal interpolation remained accurate in spite of only small changes in measured 'gaps'.

6.4.2 Multiple Patient Temporal Abstraction - The MOTA Framework

The previous test routines for PMMs provides an evaluation of temporal abstraction processes which are executed on the data from a single patient (multiple rules, *single* patient data stream-set). The level of TA dimensionality is increased to "Level Two" when TA rules are executed on multiple patients (multiple rules, *multiple* patient data stream-sets). This is, in fact, what the MOTA framework does; multi-dimensional online temporal abstraction, where the data from numerous patients is abstracted by multiple PMMs and results returned to the Active Domain Registry (ADR) from where alerts may be generated. The next set of tests are designed to assess this higher level of dimensionality.

6.4.2.1 Method

In these tests I invoke all processes which are required for this level of TA including the input of new rule into the MOTA system, creation of domain objects inside the ADR and finally, automatic triggering of alerts. From the point in time of the rule being received by the PMM, through to the rule result being determined, was evaluated in the preceding section. In summary, the processes involved are as follows:

1. TA Rule entered into Rule and Query Builder interface by clinician
2. Rule Builder parses TA rule parameters and instantiates domain objects which constitute the ADR runtime model
3. *Immediate* query functionality is now available

4. Alert rules and objects are created and passed to the Medical Alert Monitor (MAM)

5. The ADR generates the Event Correlation Specification (ECS)

6. The Script Compiler uses the patient specific ECS to generate the executable TA rule script

7. Communication processes begin between the ADR server and the ESP Server. As soon as a suitable PMM is located to execute the TA rule/s, the ADR sends the TA rule script to the ESP Server who relays it to the PMM concerned.

8. Alerts are triggered by the MAM when *true* results are returned from the PMMs.

A more complete description of the above processes can be found in section 5.1.1.

The preceding processes give rise to a number of areas requiring testing. Firstly, and most importantly, there is multi-patient temporal abstraction including creation of relevant ADR domain objects, alert rules and objects, and the triggering of alerts on reception of TA rule results. This constitutes the heart of the MOTA framework and encompasses steps 1, 2, 3, 4, 7 and 8 above. These were tested using Test 3.1. Secondly, the generation of patient specific and global ECSs by the ADR *ECSWriter* and executable rule scripts by the Script Compiler necessitates an evaluation procedure; this is provided for by Test 3.2.

Test 3.1: Rule and Query Builder, ADR, MAM

Functionality included in process steps 1, 2, 3 and 6 were tested extensively. The TA rules used for Test 1b were entered into the ADR and alert rules, alert objects, patient specific ECSs, the global ECS and executable rule scripts created. Each of the 30 rules in the 5 rule-sets, $R + R^1 + R^2 + R^3 + R^4$, were assigned to a different patient ID and entered via a simple command line interface to the Rule and Query builder. Existence of rules and correct relationships between rules and patients was verified through execution of ADR functions which return a list of all patients and rules currently in the system. The system then entered a wait state until results were

returned from relevant PMMs and alerts can then be triggered inside the MAM. The return of results was simulated through the artificial creation of the *SendTAResult* message. This message is sent to the ADR by the ESP Server when a result has been determined for a particular TA rule. The rule parameters are *patientID*, *ruleID*, *status*, t_0 and t_1. On reception of this message, the ADR modifies the state of the relevant alert object's *status* field from *false* to *true* (if the rule returned true). As soon as the status field is switched to *true*, the alert rule fires and signals an alert condition.

Test 3.2: ECS, Script Compiler

The object of this test was to verify that the executable rule scripts generated by the Script Compiler and the ECSs produced by the *ECSWriter* were correct. Output scripts were collected from the results of test 3.1 and these were executed individually using a PMM on the test data. The same rules and data were used as for Test 1b so that correct operation could be verified against the number of valid intervals discovered in the data.

Figure 6.7 shows the ECS for rule TAR1 which was automatically generated by the *ECSWriter* module inside the ADR.

6.4.2.2 Results

Test 3.1

The five rule-sets utilised in Test 1b were entered into the ADR during this test and *SendTAResult* messages randomly sent to the ADR server from the ESP Server. Once the ADR receives messages where the *status* field is set to true, an alert is generated. For rules where the *status* field was set to true, a 100% success rate was obtained and no false alerts were generated if it is assumed that the *true* rule status indicates a valid abstraction interval. This, we know to be an unsafe assumption due to the lack of ability for a PMM to detect *disconnect* and other such events. Still, this test was to evaluate the ADR, Rule and Query Builder and MAM, not the PMM. For rules where the status field was unchanged from its default value of *false*, there were also

```
<patient_ecs>
   <patient id="1a1">
      <name>Tom Brown</name>
      <numRules>1</numRules>
      <rule id="TP-1">
         <ruleName>Mort-Pred-1</ruleName>
         <numTAs>2</numTAs>
         <ruleActive>T</ruleActive>
         <startMonitor>13:00</startMonitor>
         <endMonitor>20:00</endMonitor>
         <date>12/6/2008</date>
         <begin>NULL</begin>
         <end>NULL</end>
         <ta id="TP-1-1">
            <parameterID>02</parameterID>
            <interval>20.0</interval>
            <maxGap>0.0</maxGap>
            <taType>-</taType>
            <threshold>90</threshold>
         </ta>
         <ta id="TP-1-2">
            <parameterID>BP</parameterID>
            <interval>20.0</interval>
            <maxGap>0.0</maxGap>
            <taType>-</taType>
            <threshold>24</threshold>
         </ta>
         <alarm>
            <status>F</status>
         </alarm>
      </rule>
   </patient>
</patient_ecs>
```

Figure 6.7: Event Correlation Specification for Rule TAR1.

no false alerts. Alerts were channelled to the Rule and Query builder interface as a simple text message and visually checked.

Test 3.2

There were a total of 30 patient specific ECSs and one global ECS generated as a result of Test 3.1. These were visually checked to be correct. The 30 resulting TA rule scripts were also visually checked before being executed on a selection of the test data-sets to verify their accuracy. Results of rule executions checked against those of the visual analysis carried out for Test 1 were correct. This test verified correct operation of the *Script Compiler* and also the *ECSWriter* function of the ADR.

6.5 Discussion and Summary

In this Chapter I have presented the implementation and evaluation of the prototype MOTA framework. I have demonstrated that the MOTA framework performs online multi-dimensional temporal abstraction and addresses the issues raised in Chapter 2, which exposed the requirement for such a system within the area of intelligent clinical data analysis. More specifically, the evaluation procedure has corroborated thesis hypotheses one to five stated in section 1.3.3. A more detailed look at how the hypotheses are validated and addressed is contained in section 7.2.

I have described the physical realisation of the temporal model presented in Chapter 4 and evaluated it against the original motivation for its conception; that being to i) demonstrate the Event Calculus can be extended and used for online temporal reasoning and ii) to provide a method for multi-dimensional online temporal abstraction of time-stamped data. In providing the implementation prototype and evaluation procedure, the above two points have been verified. This section of the MOTA framework is the Patient Monitor Module, where a single patient data stream-set is analysed and abstracted according to clinically defined rules entered by a clinician. This provides "Level One" of dimensionality where multiple data streams within the same data stream-set are abstracted by the one software module due to both the

design of the EC-based temporal model and the threaded implementation. The temporal model allows multiple conjunctive associations between *child* abstractions to form the final Alert fluent and the threaded execution model allows numerous *child* abstractions to run concurrently; these two factors work collectively to permit this degree of multi-dimensionality within TA processing.

Also provided in the event calculus-based temporal model is a method for temporal interpolation, which is a necessary prerequisite for any system performing temporal abstraction. As described in section 4.6.1.1, this is enabled through use of *Trajectory* and *Releases* predicates. Test 2 verified that the forward chaining implementation of the temporal model was capable of performing temporal interpolation of data, sample by sample, or incrementally.

The evaluation of this level of multi-dimensional TA proceeded using clinically defined rules as input. The MOTA framework is designed for ad-hoc rule definition as opposed to the detection of pre-defined clinical conditions, as in other TA-based systems [18, 47–49, 51, 96]. The kinds of TAs we are interested in are those that may provide early indication of a deterioration in health and consist of a conjunction of multiple parameters. Clinical research into the effects of aberrant behaviour across multiple parameters such as this is currently proceeding in parallel to this research, hence the present lack of background knowledge to establish rules further to the two provided for the evaluation. To increase the test rule-set, I divided the two rules that were provided into their *child* components and utilised these to create another four rules making an initial six rule test set. In addition to this, 4 extra rule-sets of 6 rules each were created by varying the threshold values from the original rules to slightly higher or lower values. These new rule-sets have not undergone clinical evaluation however they served the purpose of creating 'similar' abstractions to demonstrate the viability of the PMM to perform a *low level* of multi-dimensional TA.

At the current time, TAs are limited to level shifts above or below a given threshold value. I have concentrated on the provision of a temporal abstraction platform which can be extended to permit more complex TAs such as trends, sequentially dependent

fluents or Area Under the Curve (AUC), as described in section 4.6.2. Necessary modifications required for performing AUC calculations are minor. Essentially, the PMM process must continue to monitor after an abstraction is found to exist, rather than terminating at the end of the abstraction interval. The PMM will then monitor the data for the length of the monitoring period. If the maximum temporal gap is set to the length of the monitoring period, then a final count of the time spent within the abstraction can be obtained.

As mentioned in the results of Test 1a, the lack of noise and artifact detection is a disadvantage and false alerts could be triggered due to *disconnect* events occurring during data measurement. The use of transient filtering or knowledge-based methods for detecting such events would be a worthy addition to the PMM's design.

The evaluation routine was based on a processing window of one second as ICU data is commonly sampled at a rate of 1Hz. In implementation terms, this is internally set as the 'wait' time of PMM threads and could theoretically be decreased to say, 0.5 second meaning that data could be sampled at a rate of 2Hz. However, this has not been tested, although judging from the number of rules executing in the test routines and the fact that there were no signs of the one second window being breached, I would predict that it would be quite possible. Certainly, it is quite possible to increase the 'wait' time; this would have no adverse effects on TA processing.

The *high* level of multi-dimensionality defined within this dissertation is that of multi-patient temporal abstraction, encompassing "Level Two Dimensionality". This was introduced in section 1.2 and also explained within section 3.3.2. This is catered for by the rest of the MOTA framework; namely the ADR, MAM, Rule and Query Builder and communications sub-system. The design and architectural arrangement of these components was covered by Chapter 5 and their prototype implementation discussed in this Chapter. Here, data emanating from multiple patients (multiple data stream-sets) is abstracted and alerts automatically generated if a particular rule returns *true*. Tests 3.1 and 3.2 evaluated the ability of the above components to perform their function of MDTA.

The prototype that has been constructed for this research was primarily to verify the practicality of the proposed solutions, and as such offers no graphical interface for the user to interact with. The addition of such an interface would not be difficult as I have provided the necessary functionality through a Java interface to the Rule and Query Builder. The interface can then be designed and constructed separately and without the need to implement the underlying functionality.

Automatic alerts are currently sent to the Rule and Query Builder command line interface as text messages. I have provided the core mechanisms for alerting within the MAM and in a way similar to the addition of a more sophisticated interface, alerts could be relayed to a different output such as SMS or pager, though the use of a third party toolkit implemented on top of the existing infrastructure.

The results obtained from the evaluation are encouraging and verify that the MOTA framework can perform temporal abstraction within a multi-dimensional environment in real-time. Carrying out TA at a dimensionality of *Level One* is solved through the threaded design of the PMM in conjunction with the temporal model, both permitting multiple TAs to be concurrently obtained from a single data stream-set. The abstraction of data within a *Level Two* environment is solved through the implementation of the ADR, where results from a bank of PMMs are amalgamated and correlated. The additional framework features, such as the ECS and Rule and Query Builder, support the complexity involved in multi-dimensional TA by offering a semantic representation of all domain entities and an interface which includes functionality for TA rule specification and editing and also the querying of results obtained from executed rules.

Chapter 7

Discussion and Conclusion

This thesis has presented a framework for performing multi-dimensional temporal abstraction of streaming time-stamped data. The case-study area and application context of neonatal intensive care was used to demonstrate the practical application of the framework in the role of patient monitoring where anomalous excursions in data, which exist for pre-defined intervals of time, can be detected and clinical alerts generated.

The process of Temporal Abstraction is an important phase within Intelligent Data Analysis (IDA) [8] where domain knowledge is applied to data in order to extract context-based information. The clinical domain is one where TA has played an especially relevant role due to the inadequacy of statistical or purely numerical methods in the analysis of data for purposes of diagnosis, therapy and prognosis. This is primarily because the underlying physiological mechanisms which give rise to the data are poorly understood making the construction of quantitative models for analysis purposes unreasonable [98]. Furthermore, the context of analysis changes with time as disease states progress and medication changes, meaning what were *normal* data values at one time may be very *abnormal* at another.

My goal was the development of a framework for MOTA which addressed areas within the field of TA which were found to be lacking and in my opinion, impeding the further evolution of TA-based systems.

7.1 Thesis Summary

Initially, in section 1.2, I defined the term Multi-dimensional Online Temporal Abstraction (MOTA) as being a dual-layered concept of dimensionality within the data and abstraction environments. *Level One* implies the temporal abstraction of data emanating from a single patient. In the case-study area, this involves multiple physiological data streams such as heart rate, SaO2, CO2 and blood pressure which constitute a single data stream-set. There may be a single TA on the data stream-set or multiple TAs. *Level Two* means the abstraction of data from multiple patients, again there could be one or multiple TAs on the data stream-sets. This particular area of TA was found to be poorly developed and constitutes the primary focus of the research provided by this dissertation.

In Chapter 2, I presented a comprehensive literature survey which reviews the major contributions made in the development of TA-based frameworks. The review was based on factors which I believe to be important to the continued growth and progress of such systems and also those where there is a deficit of current research and interest. Four criteria were used as the basis of the review, these were:

- *Data*: The clinical area which gives rise to data is of importance to TA mechanisms which will be applied. For instance, the characteristics of the Diabetes Mellitus domain is very different, in terms of the data produced and the desired abstractions to be constructed, to the area of mechanical ventilation monitoring. With higher data sample rates and abstractions frequencies, constraints are imposed upon TA processes which may not exist in lower frequency domains.

- *TA Complexity*: I was interested in the complexity available in TAs. Simple TAs, such as shifts over or under a particular threshold value for a pre-defined period of time, are commonly used in making diagnoses from patient data. Extending TA to enable the detection of more complex abstractions, such as multiple parameters each exhibiting a particular abstraction at the same time

or multiple parameters having a sequential dependency between their individual TAs, is an area worthy of further exploration and development as some conditions, such as those mentioned in section 3.2.2.3, including Sepsis [6, 83] and Periventricular Leukomalacia (PVL) [7], have been shown to exhibit advance indicators such as these. Early detection of such conditions can allow intervention at an earlier stage of the disease and potentially improve health outcomes.

- *TA and Data Dimensionality*: This particular area is one of importance in terms of the lack of research directed toward the provision of systems which can apply TA within a multi-dimensional environment as defined in section 1.2. Level One dimensionality has been achieved to varying degrees although, to my knowledge, no previous work has concentrated on specific methods for achieving robust performance in regard to the more challenging aspects of this level of dimensionality, such as applying multiple abstractions to the same data stream parameter. Additionally, there are no known developments allowing the application of TA within a Level Two environment where multiple TAs are obtained from multiple patient data stream-sets and the results combined to allow automatic alerting across a ward of patients.

- *TA Knowledge and Reasoning*: Temporal abstraction is a knowledge-driven process and as such, the knowledge level and reasoning which underpins the creation of TAs is of importance. As revealed by Foster et. al. [90], there exists a gap between clinical research and clinical management which corresponds to the gap between data mining and data abstraction. Providing a link between these presently disparate concepts constitutes a worthy direction for IDA systems of the future as it allows knowledge obtained from machine learning processes to be applied to the abstraction of data which can carries the potential to improve clinical management.

The outcome of the literature review provided in Chapter 2 was the identification

of current open research areas and those requiring further research in order to ensure the continued evolution of TA-based frameworks keeps up with the increasingly data-driven world we now exist in. In the conclusion of Chapter 2 (section 2.6), I discussed the development of the five thesis hypotheses in regard to the results of the literature review; I will address these again shortly.

The case-study environment of the Neonatal Intensive Care Unit (NICU) was outlined in Chapter 3. I began by introducing statistics which highlight the importance of the NICU in providing care for seriously ill and often very premature babies. The nature and frailty of the premature infant is another factor deserving a high level of importance. I discuss typical care practices for the first hours of life and characteristic routine measurements such as the APGAR score which provide an indication of how well the baby handled the birthing process and is adapting to its new environment. The area of NICU monitoring is particularly relevant to this thesis as the proposed work is to be applied within the monitoring context. Again, typical procedures are listed such as NICU flow charts for data recording and the threshold alarming characteristics of current monitoring equipment. The pitfalls of the NICU flowchart include the lack of a temporal view of data to detect subtle shifts and trends and the possibility for variations in the baby's vital signs to go undetected when they occur between the thirty to sixty minute recording times. The threshold alarming model has been widely shown [28, 29, 167] to be both unreliable and to produce a large number of false alerts (up to 86%).

The data and monitoring environment is analysed and here I demonstrated how the multi-dimensionality defined in section 1.2 exists in the real life environment of neonatal intensive care. The high frequency sample rates of physiological data measured from sensitive transducers, in combination with the multi-dimensional environment, serves to highlight the computationally challenging facets of the thesis proposals. At the conclusion of Chapter 3, I exposed the limiting aspects of current NICU monitoring equipment which are confronted by the proposed MOTA framework.

The issue of temporal reasoning is dealt with in Chapter 4 where prior to explaining the temporal model that drives the temporal abstraction mechanisms, I provide a brief overview of related work in temporal reasoning, especially that which has been applied to clinical environments. To perform online and incremental TA of time-stamped data, I demonstrate how the Event Calculus can be adapted from a formalism that has been primarily used to reason backward over stored data, to one which offers a reliable and efficient way to reason in real-time. Interestingly, axioms originally developed for the representation of dynamic change, such as *Trajectory* and *Releases*, were adaptable to the domain of MOTA and allowed the tracking of time itself; necessary for online TA and also the process of temporal interpolation.

The architecture of the MOTA framework is presented in Chapter 5. Here, the software components of the framework are introduced and explained with regard to their interconnection, communication and individual responsibilities. The high level presentation and an example TA process "walkthrough" provides the reader with a holistic view of the system from TA rule specification to rule execution, alerting and querying. The methods that the MOTA framework uses to solve the dimensionality problem become apparent through the schema for amalgamation of results from multiple PMMs, where each PMM addresses *Level One* dimensionality and their *combination* and conjunction with the ADR, caters for *Level Two* dimensionality.

Low level implementation details such as programming languages and environments, class diagrams and the PMM threaded execution model are included in Chapter 6. The programming methods used by PMMs in executing TA rules on patient data is covered along with information regarding the rule engine and the technique used for representing Event Calculus fluents, which are essentially temporal abstractions.

The second half of Chapter 6 offered an evaluation of the MOTA framework divided into three categories: i) testing of the PMM to establish whether MOTA can be performed within a 1 second reasoning cycle, ii) testing of the methods for temporal interpolation and iii) testing for the ability to perform TA across multiple

data stream-sets or *Level Two* dimensionality.

This thesis has contributed to numerous areas within both Health Informatics and Computer Science. Health Informatics contributions include the potential of improved patient monitoring through the use of a time-oriented method for analysing patient data on a continuous basis which provides an improved alarming model and allows clinicians to analyse data emanating from multiple patients. As previously mentioned, recent medical research indicates that certain conditions may exhibit early indicators in physiological data which, if detected, would allow early intervention and lead to improved health outcomes. Current monitoring equipment falls short of the task in three areas: i) where data is to be analysed with respect to time, ii) where multiple data stream parameters are correlated based on their temporal relationships and iii) where numerous events or intervals are to be accrued over a given time frame. The first two of these points are provided by the temporal model proposed in Chapter 4 and slight extensions to the model and its implementation can enable the accumulation of pre-defined intervals.

Contributions to Computer Science include the Event Calculus-based (EC) temporal model which offers a formal method for performing temporal abstraction in an online fashion. I have applied axioms which were designed for the representation of continuous change which permits a counter to be used for keeping track of time. Temporal interpolation, which requires measurement of the gaps between contextually similar intervals, can now be incorporated into the model and valid abstraction interval lengths can be pre-defined and then measured as they eventuate. This extends the application context of the EC to a real-time temporal reasoning model which can be implemented through a forward chaining rule engine.

As revealed by the literature survey, TA has been primarily developed and used to abstract data from stored singular data-sets and research has not investigated the possibility of applying TA to a multi-dimensional environment, as defined in section 1.2. To provide a level of organisation and control in such a complex setting, mechanisms for semantic representation and co-ordination of TA processes and parameters are

necessary. I have endowed the MOTA framework with such means in the form of the ADR and ECS components. Expanding TA into the realm of monitoring multiple data-sets represents a significant step in the evolution of TA-based systems, especially when abstraction techniques are able to operate in a real-time sense.

7.2 Conclusions

Crucial components in the Constructive Research method are to develop a new *construction* and to demonstrate the solution works. During this research, a prototype was constructed in order to ascertain the practicality and also performance of the proposed methods. The evaluation of the prototype was provided in the preceding chapter and the results it provided were encouraging. More specifically, the evaluation procedure has corroborated thesis hypotheses one to four stated in section 1.3.3 as follows:

Hypothesis 1: "A framework can be defined to enable temporal abstraction within a multi-dimensional data environment in real time. This framework is named the Multi-dimensional Online Temporal Abstraction (MOTA) Framework."

In section 1.2, I defined the two primary levels of dimensionality which are addressed by the MOTA framework, *Level One* and *Level Two*. Evaluation procedures were designed to evaluate these separately, namely tests 1 and 3. Test 1, which consisted of parts 1a and 1b, was aimed at assessing Level one dimensionality or the temporal abstraction of data emerging from a single patient consisting of numerous data streams. Test 1a involved determination of the number of relevant abstractions contained within the 17 different data stream-sets that were collected from the NICU at Nepean Hospital. The gold standard was a visual analysis of the data using a proprietary graphical data analysis package. This first test correctly identified all valid TAs within the data stream which validated the ability of a PMM to perform multi-dimensional online TA although only 6 rules were used making it relatively simple in a computational

sense. In order to provide a higher load on the PMM under evaluation, test 1b utilised an extended ruleset of 30 rules which was executed on each of the 17 data stream-sets one stream-set at a time. The concern was for some data points to be missed due to the increased usage of resources possibly causing the 1 second processing window to be breached. This did not occur and all identified intervals were detected by the PMM under test at a sample rate of 1Hz.

Test 3.1 gauged the ability of the MOTA framework to perform multi-dimensional TA at Level two dimensionality; or the abstraction of data emanating from numerous patients. The components involved are the ADR, MAM, Rule and Query Builder, *ECSWriter* and Script Compiler. Test 3.1 involved inserting all 30 test rules into the ADR and sending simulated *SendTaResult* messages from the ESP Server indicating that either a *true* or *false* result had been determined for a particular rule. The Rule and Query Builder was also assessed in this test as it was used for the entry of rules followed by the *immmediate* querying of ADR contents to verify correct relationships between domain entities (patients and rules). Visual checking of the ECS and executable rule scripts that resulted from the 30 rules entered (test 3.2) was carried out to validate correct functionality of the *ECSWriter* and Script Compiler.

A 100% success rate was achieved in terms of alert generation based on the rules that had returned a *true* finding. No false alerts were generated where the rule had an unchanged status field of *false*. The results of *immediate* queries verified correct operation of the Rule and Query builder which is responsible for instantiating the correct relationships between domain entities. The ECS and executable rule scripts were deemed correct after visual analysis and execution of the rule scripts on selected data sets further substantiated the fact that both of these items were free from error. Moreover, the rule scripts are directly generated from the ECS; any error within the ECS will create a similar anomaly in the script concerned.

Tests 3.1 and 3.2 verified that the MOTA framework components involved in the process of multi-dimensional TA at *Level two* dimensionality operated successfully.

Hypothesis 2: "A method to semantically *represent* the complex multi-dimensional relationships between data streams, parameters and TAs can be defined to support TA within such complex environments."

This method is catered for by both the Event Correlation Specification (ECS) and the Active Domain Registry (ADR). The ECS is a permanent record of the objects which have been instantiated during the execution of a TA rule. It shows the relationships between patients, rules and data parameters and also includes information regarding TAs themselves, such as interval lengths, temporal gaps and so on. The ECS was tested in section 6.4.2.1, Test 3.2 using a rule-set which included 30 separate TA rules, each assigned to a different patient ID. The resulting 30 patient specific ECSs provided a semantic representation of associations between data streams, parameters and TAs which were then used by the Script Compiler to generate 30 executable TA rule scripts for use by PMMs. For a holistic representation of the same, the global ECS includes all domain entities within the one file. The global and patient specific ECSs were visually checked for correctness and successful execution of the resulting TA rule scripts provided further evidence of their accuracy whilst also verifying reliable operation of the Script Compiler.

The ADR provides a runtime instantiation of the contents of the ECS and is therefore a runtime model of the relationships between domain entities. The ECS defines the way in which multi-dimensional TA is performed within the MOTA framework whilst the ADR is an *implementation* of this definition.

Hypothesis 3: "The Event Calculus can be extended and applied to cater for temporal interpolation and abstraction of real time data."

The theoretical underpinnings of the temporal model used by the PMMs for

abstracting data was detailed in Chapter 4. Chapter 5 revealed the practical implementation details of that model, and Test 1a, section 6.4.1.1 validates that the EC can indeed be extended to enable online temporal abstraction. I used a one second window of time as the measure by which reasoning and TA tasks must be completed and no tests indicated a breaching of that time indicating the threaded design of the PMM in conjunction with the EC-based temporal model was a satisfactory combination for abstracting time-stamped data which is sampled at a rate of 1Hz.

Hypothesis 4: "The framework can support the definition of TA rules by allowing an interface to an analytical processor which mines physiological data to reveal new associations between data parameters thus providing a bridge between data *mining* and data *abstraction* processes."

In parallel to the work presented by this thesis, research is being carried out by other researchers in our team to define data mining processes which carry the potential to drive TA mechanisms within the MOTA framework. As described in section 5.1.4, the interface to the Analytical processor is an XML-based representation of the rule which follows the format of the ECS. An XML parser can then read in the rule parameters and instantiate relevant ADR objects which will initialise the processes involved in rule execution and automatic alerting. Prior to execution, clinical validation of rules thus discovered would be proceed utilising an interface which is part of the Analytical processor.

Hypothesis 5: "Proposals one to four can be applied within a Health and Medicine context to enrich Clinical Management."

The example rule-sets used in this evaluation were provided by clinicians at the Nepean Hospital NICU as being indicative of impending serious degeneration in the health status of premature babies. Two of the rules included two separate parts describing the time-based trajectory of a particular physiological parameter. The rules were modelled as temporal abstractions and relevant domain

objects instantiated by the system. At present there are no existing tools for clinicians to use where they can define such rules, run them on patient data in real time and receive almost instantaneous feedback in the form of alerts.

Previous and existing monitoring equipment has and does typically operate using the threshold alarming model which, as discussed in sections 1.3.1.1 and 3.4, has many associated disadvantages and problems. The temporal model developed for this research includes the temporal dimension hence alleviating many of the problems of the threshold model whilst offering a simple mapping from the clinically defined rule to the executable one.

7.2.1 Research Limitations

This research includes a number of limitations which I have categorised as being either due to the limitations of the prototype or limits imposed on actual MOTA framework design.

The current implementation is a prototype which was built as a vehicle for testing the theories I have proposed, those being multi-dimensional TA and an EC-based model for performing online TA of time-stamped data. Section 6.5 discussed a number of limitations which emerged mainly through the process of the evaluation procedure.

Limitations of the MOTA framework itself primarily hinge around the provision of better interfacing to users, in terms of both input and output. The command line interface to the Rule and Query Builder which is used in the prototype is not designed for ease of use or intuitive operation of the functions within. The underlying functionality has been provided and is currently exposed in the form of Java interfaces, leaving graphical user interface development open for future research.

7.2.1.1 Prototype Limitations

Through the evaluation provided in chapter 6, with respect to "Level One Dimensionality" and the PMM, a number of limitations have come to light as follows:

1. TAs are limited to level shifts; positive and negative. As mentioned in section 4.6.2, extending the framework to enable further abstractions, such as trend and Area Under the Curve (AUC), would greatly improve the support that the MOTA framework can offer to the case study area of neonatal intensive care. However, the design offered by this research makes it a relatively simple matter to connect a separate module which calculates trend or AUC as long as it can supply a series of time-stamped values to the PMM concerned. Temporal abstractions based on the stream of such values can then proceed in the same way as using raw time-stamped data originating from bedside monitors.

2. *Complex* TAs are limited to multiple instances of *simple* concurrent abstractions. As discussed in section 4.7, modifications to the temporal model to enable creation of TAs where there exists a *sequence* of events would allow clinicians to define rules where one event precedes another. For example: "If heart rate falls below 150bpm for 20 seconds and then blood pressure falls below 24mm/Hg for a further 60 seconds, then alarm". Section 4.6.2 describes how this can be incorporated into the temporal model.

3. To allow cumulation and counting of the time spent where data points are inside a given abstraction range. This would be necessary where AUC calculations were required and is a desirable value in clinical terms where it is hypothesised that numerous excursions 'out of range' may have a cumulative effect similar to a continuous period of time spent outside the given threshold. The necessary modifications to the design of the PMM to allow this to happen are minimal. Essentially, the PMM must not terminate when an abstraction is found in the data, but rather continue monitoring and counting the time within the each successive encountered abstraction interval. The very same process is already part of the PMM logic and is used to count 'out of range' fluents.

4. Currently, PMMs assume that data is free from noise and other artifact such as disconnections from the transducer that sources the data stream. This gives

rise to two problems. i) During valid intervals, if disconnection or noise occurs, then the abstraction is indeterminable and ii) completely invalid intervals due to disconnection or patient movement will cause false alerts to be triggered.

5. At present, time granularity is fixed at 1Hz meaning that if data arrives at a higher frequency, such as 10Hz, then 9 out of 10 data values will not be read and assessed by PMM TA processes. The current implementation reads a single datum at a rate of one per second which is set by the 'sleep' time of PMM threads. For example, decreasing the value of sleep time to 0.5 seconds would theoretically increase the processing rate to 2 Hz, and as long as data arrives at a rate of 2 values per second, then the speed of the system increases. This has not been evaluated but providing an option to the user to pre-select the sample rate prior to executing their TA rules would be a valid extension to the work.

   Some of the fast waveform data, such as ECG and respiratory data, are sampled at rates of 512Hz and 128Hz respectively, and further research is already proceeding toward the efficient abstraction of such signals.

Although there are limitations in terms of the complexity of abstractions and noise and artifact filtering, what I have provided is a temporal abstraction system where incremental real-time TA is performed using a formal temporal model which offers possibilities for extension, both in terms of the temporal model itself and where different types of abstractions may be required, such as AUC. The PMM receives a stream of time-stamped data values and processes them into interval-based abstractions that are relevant to the domain area. The design allows multiple data streams to be processed concurrently adding a degree of multi-dimensionality to TA mechanisms.

7.2.1.2 MOTA Framework Limitations

With respect to the MOTA framework as defined by this work, the following limitations, which relate to "Level Two" Dimensionality, have become apparent:

1. Currently there is no Graphical User Interface (GUI). The prototype Rule and Query Builder includes a command line, text-based, interface for interacting with the underlying functionality which includes querying and rule operations such as insertions and deletions. Recent work on intelligent visualisation of temporal abstractions [86], and evaluation thereof [249], has shown significant benefits associated with the provision of graphical interfaces. Such an interface would be built on top of the existing functionality by simply utilising the Rule and Query Builder Java interface and calling the appropriate methods detailed in section 5.1.3.4.

2. Automatic alerting provided by the MAM is currently routed to the Rule and Query Builder command line interface and alerts displayed as simple text messages. As discussed in section 5.1.3.1, a third party toolkit, an SMS provider and an internet connection, can be utilised to channel alerts via an SMS system. Alerts can also be directed to a visual interface if one is provided.

3. Historic querying involving the TA Rule Base was not tested nor implemented although details of how this can be achieved, along with a worked example in SQL, was provided in section 5.2.2.

Whilst I have listed extensions to the MOTA framework, such as the capability for visualisation and SMS alerting as limitations, the framework has been designed in such a way as to facilitate such expansion through interfaces to the primary software modules.

The physical organisation of MOTA framework components is flexible in regard to the positioning of PMMs. They can be installed at the bedside or exist as internal software components within the SMS, which resides in a NICU server room. I have suggested the positioning of PMMs at the bedside in order to provide instant feedback to clinical staff on the floor and, in this scenario, a graphical interface would be required on each notebook machine that runs the PMM process. Presently, a simple command line interface provides visual feedback for alerts via text messages. This

functionality could be extended and connected in to a graphical display of anomalous intervals which give rise to alert states.

7.2.2 Future Work

Considering the above limitations, numerous extensions and further work is both possible and desirable. The MOTA framework, in its current state, is a prototype which has validated the thesis hypotheses and offered encouraging results to support my original motivation for the project. Communication and overall architectural requirements are offered by the present prototype design. This creates a platform on which to build further elements which can enhance existing methods and offer a more complete patient monitoring tool.

I believe extending the current formal model to include more complex abstractions would be a most desirable addition. As previously mentioned, the modification required to allow AUC abstractions is minimal and the provision of TA mechanisms where there exists a sequential, rather than concurrent, relationship between parameters would also be a worthy addition. Also in the area of the PMM, filtering techniques to detect disconnect events, patient movement and other artifacts is not only desirable, but essential in the environment of intensive care medicine where transducers are subject to the temperature changes and physical movements of the patient they are connected to.

The EC has been shown to be a useful method for performing temporal projection or prediction [189,196], where given an initial domain state, and a sequence of domain events and an event narrative, we seek a resulting domain state. This notion finds application in patient monitoring where, for example, we can provide suitable warning if a parameter, at its current rate of increase, will attain a pre-defined dangerous level in the future.

Graphical interfaces for temporal abstraction frameworks and exploration of time-oriented data are interesting areas of research which is providing promising results [116,249]. In conjunction with further development of historical querying mechanisms,

research such as this could be applied to the MOTA framework for the development of interfaces to the ADR and PMM allowing visual exploration of abstracted intervals by clinical staff to support clinical research and management.

Temporal Abstraction is not a concept solely limited to the clinical domain and previous work has demonstrated its applicability to other areas such as traffic control [96] and the monitoring of gas turbine data [42]. The MOTA framework could easily be implemented in any domain requiring knowledge-based analysis of time-stamped data. Industrial process control provides a similar environment to that of the intensive care unit regarding the streaming data environment where multiple sensors feed data into a central *control unit* for analysis and automated alerting to potentially dangerous patterns or shifts in the data. Business event monitoring would benefit from the notion of an 'extended event', as discussed in section 4.3, as most such systems have little notion of the history of events leading up to the current time. Interval-based event detection proposes a solution to this problem through the definition of complex events being based upon an interval rather than an instantaneous time-point. This effectively brings the field of event detection closer to that of temporal reasoning and abstraction, which is a knowledge-based discipline. To further the alignment of these two previously disparate fields of research, the design and research of TA-based frameworks for event monitoring environments represents a viable avenue for further exploration.

7.2.3 Final Conclusion

Most clinical temporal abstraction frameworks have operated on a single patient's data and multiple concurrent high frequency abstractions have proven problematic. It is evident from the literature survey provided in this dissertation, that for IDA systems to successfully move forward into the future where clinical environments are becoming increasingly data-intensive, the ability for managing multi-dimensional aspects of data at high observation and sample frequencies must be provided. I have

addressed the issues of dimensionality and frequency of TA mechanisms in two distinct areas of the proposed MOTA framework. Firstly, I approached the abstraction of data streams emanating from a *single* patient and offered temporal abstraction through the union of a temporal model capable of real-time reasoning and an implementation based on a forward chaining rule engine. This was implemented through the Patient Monitor Module. Secondly, I addressed the concept of monitoring and abstracting data from *multiple* patients and proposed the central Active Domain Registry where accumulation and correlation of results returned from numerous Patient Monitor Modules can facilitate automated clinical alerting.

The potential for the use of research of this type to reduce mortality and morbidity rates is significant. Researchers within our Health Informatics Research team, some of whom are now located in Canada, are progressing research with collaborating Neonatologists in the utilisation of NICU patient physiological data to progress the development of condition-onset prediction models for use by monitoring technologies such as the MOTA framework detailed in this dissertation.

Appendix A

List of Common Abbreviations

Abbreviation	Meaning
ADR	Active Domain Registry
BP	Blood pressure
BPM	Beats per minute
DSS	Decision support system
EC	Event Calculus
ECS	Event Correlation Specification
ESP	Event Stream Processor
ICU	Intensive Care Unit
IDA	Intelligent Data Analysis
MAM	Medical Alert Monitor
MDTA	Multi-dimensional Temporal Abstraction
mm/Hg	Millimeters of Mercury (a measure of pressure)
MOTA	Multi-dimensional Online Temporal Abstraction
MVI	Maximum Validity Interval
NICU	Neonatal Intensive Care Unit
PMM	Patient monitor module
SaO2	Blood oxygen saturation
SMS	Solution Manager Service
TA	Temporal Abstraction

Bibliography

[1] H. Wu, B. Salzberg, and D. Zhang, "Online event-driven subsequence matching over financial data streams," in *ACM SIGMOD International Conference on Management of Data*, (Paris, France), pp. 23–34, ACM Press, 2004.

[2] H. Balakrishnan, M. Balazinska, D. Carney, U. Çetintemel, M. Cherniack, C. Convey, E. Galvez, J. Salz, M. Stonebraker, N. Tatbul, R. Tibbetts, and S. Zdonik, "Retrospective on aurora," *The VLDB Journal The International Journal on Very Large Data Bases*, vol. 13, no. 4, pp. 370–383, 2004.

[3] Y. Tan, Y. Wang, and H. Li, "A management strategy of monitor data in ICU based on data stream technology," in *Medical Information Systems: The Digital Hospital, 2004. IDEAS '04-DH. Proceedings. IDEAS Workshop on*, pp. 163–169, 2004.

[4] C. McGregor, "e-baby web services to support local and remote neonatal intensive care," in *Health Informatics Conference*, (Melbourne, Australia), CD-ROM, HISA Ltd, 2005.

[5] G. A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for processing information," *The Psychological Review*, vol. 63, pp. 81–97, 1956.

[6] P. Griffin and R. Moorman, "Toward the early diagnosis of neonatal sepsis and sepsis-like illness using novel heart rate analysis," *Pediatrics*, vol. 107, no. 1, pp. 97–104, 2001.

[7] S. Shankaran, J. C. Langer, N. Kazzi, A. R. Laptook, and M. Walsh, "Cumulative index of exposure to hypocarbia and hyperoxia as risk factors for periventricular leukomalacia in low birth weight infants," *Pediatrics*, vol. 118, no. 4, pp. 1654–1659, 2006.

[8] N. Lavrac, E. Keravnou, and B. Zupan, "Intelligent data analysis in medicine," in *Encyclopaedia of Computer Science and Technology* (A. Kent, ed.), vol. 42, pp. 113–157, New York, USA: Marcel Dekker, Basel, 2000.

[9] A. L. Harris, C. C. Chen, and W. J. Perkins, "Medical knowledge discovery systems: data abstraction and performance measurement," *Knowledge Management Research & Practice*, vol. 2, no. 2, pp. 95–102, 2004.

[10] J. T. Sandefur, *Discrete Dynamical Systems, Theory and Applications*. Oxford: Clarendon Press, 1990.

[11] M. Deistler, "Linear system identification - a survey," in *From Data to Model* (J. Williams, ed.), pp. 1–25, Berlin, Heidelberg, New York: Springer, 1989.

[12] M. Imhoff, M. Bauer, U. Gather, and D. Lohlein, "Time series analysis in intensive care medicine," *Applied Cardiopulmonary Pathophysiology*, vol. 6, pp. 203–281, 1997.

[13] M. Imhoff, M. Bauer, U. Gather, and D. Lhlein, "Statistical pattern detection in univariate time series of intensive care on-line monitoring data," *Intensive Care Medicine*, vol. 24, no. 12, pp. 1305–1314, 1998.

[14] R. Fried, U. Gather, and M. Imhoff, "Pattern recognition in intensive care online monitoring," in *American Medical Informatics Association Symposium*, (Washington DC), pp. 184–188, 2001.

[15] M. Dojat, N. Ramaux, and D. Fontaine, "Scenario recognition for temporal reasoning in medical domains," *Artificial Intelligence in Medicine*, vol. 14, no. 1-2, pp. 139–155, 1998.

[16] S. Miksch, A. Seyfang, and C. Popow, "Abstraction and representation of repeated patterns in high-frequency data," in *The Fifth Workshop on Intelligent Data Analysis in Medicine and Pharmacology (IDAMAP-2000), Workshop Notes of the 14th European Conference on Artificial Intelligence (ECAI-2000)* (N. Lavrac and S. Miksch, eds.), (Berlin, Germany), pp. 32–39, 2000.

[17] J. C. Augusto, "Temporal reasoning for decision support in medicine," *Artificial Intelligence in Medicine*, vol. 33, no. 1, pp. 1–24, 2005.

[18] A. Seyfang and S. Miksch, "Advanced temporal data abstraction for guideline execution," in *Computer-based Support for Clinical Guidelines and Protocols. Symposium on Computerized Guidelines and Protocols (CGP 2004)* (K. Kaiser, S. Miksch, and S. Tu, eds.), vol. 101, pp. 88–103, IOS Press, 2004.

[19] I. J. Haimowitz, P. P. Le, and I. S. Kohane, "Clinical monitoring using regression-based trend templates," *Artificial Intelligence in Medicine*, vol. 7, no. 6, pp. 473–496, 1995.

[20] R. Bellazzi, C. Larizza, P. Magni, S. Montani, and M. Stefanelli, "Intelligent analysis of clinical time series: an application in the diabetes mellitus domain," *Artificial Intelligence in Medicine*, vol. 20, no. 1, pp. 37–57, 2000.

[21] Y. Shahar and M. A. Musen, "Knowledge-based temporal abstraction in clinical domains," *Artificial Intelligence in Medicine*, vol. 8, no. 3, pp. 267–298, 1996.

[22] M. Stacey, C. McGregor, and M. Tracy, "An architecture for multi-dimensional temporal abstraction and its application to support neonatal intensive care," in *Engineering in Medicine and Biology Society, (EMBS 2007)*, (Lyon, France), pp. 3752–3756, IEEE, 2007.

[23] M. Stacey and C. McGregor, "Temporal abstraction in intelligent clinical data analysis: A survey," *Artificial Intelligence in Medicine*, vol. 39, no. 1, pp. 1–24, 2007.

[24] P. Laws, S. Abeywardana, J. Walker, and E. A. Sullivan, "Australia's mothers and babies 2005," vol. 20 of *Perinatal Statistics*, Sydney: AIHW National Perinatal Statistics Unit, 2007.

[25] "Nsw mothers and babies," *NSW Public Health Bulletin*, vol. 18, no. S-1, 2007.

[26] C. McGregor, G. Bryan, J. Curry, and M. Tracy, "The e-baby data warehouse: a case study," in *35th Annual Hawaii International Conference on System Sciences*, pp. 3018–3024, 2002.

[27] R. Lister, G. Bryan, and M. Tracy, "The e-babies project: Integrated data monitoring and decision making in neo-natal intensive care," in *8th European Conference on Information Systems*, (Vienna, Austria), pp. 72–76, 2000.

[28] C. L. Tsien and J. C. Fackler, "Poor prognosis for existing monitors in the intensive care unit," *Critical Care Medicine*, vol. 25, no. 4, pp. 614–619, 1997.

[29] M.-C. Chambrin, P. Ravaux, D. Calvelo-Aros, A. Jaborska, C. Chopin, and B. Boniface, "Multicentric study of monitoring alarms in the adult intensive care unit (icu): a descriptive analysis," *Intensive Care Medicine*, vol. 25, no. 12, pp. 1360–1366, 1999.

[30] I. J. Rampil, "Intelligent detection of artifact," in *The Automated Anesthesia Record and Alarm System* (J. S. Gravenstein, R. S. Newbower, and A. K. Ream, eds.), pp. 175–190, Boston, USA: Butterworths, 1987.

[31] D. Garfinkel, P. V. Matsiras, and J. H. Lecky, "Poni: An intelligent alarm system for respiratory and circulation management in the operating rooms," in *Symposium on Computer Applications in Medical Care. American Medical Informatics Association*, pp. 13–17, 1988.

[32] M.-C. Chambrin, "Alarms in the intensive care unit: how can the number of false alarms be reduced?," *Critical Care Medicine*, vol. 5, no. 4, pp. 184–188, 2001.

[33] N. McIntosh, J.-C. Becher, S. Cunningham, B. Stenson, I. A. Laing, A. J. Lyon, and P. Badger, "Clinical diagnosis of pnneumothorax is late: Use of trend data and decision support might allow preclinical detection," *Pediatric Research*, vol. 48, no. 3, pp. 408–415, 2000.

[34] D. Foster, "Agent-based intelligent decision support for neonatal analysis and trend detection," tech. rep., Health Informatics Research, School of Computing and Mathematics, University of Western Sydney, 2007.

[35] R. Bellazzi, C. Larizza, and A. Riva, "Interpreting longitudinal data through temporal abstractions: An application to diabetic patients monitoring," in *Second International Symposium on Advances in Intelligent Data Analysis, Reasoning about Data* (X. Liu, P. R. Cohen, and M. R. Berthold, eds.), (London, UK), pp. 287–298, Springer-Verlag, 1997.

[36] C. Larizza, R. Bellazzi, and G. Lanzola, "An http based server for temporal abstractions," in *Intelligent Data Analysis in Medicine and Pharmacology (IDAMAP 99) - a workshop at the AMIA 1999 Symposium*, (Washington, DC), 1999.

[37] M. J. O'Connor, W. E. Grosso, S. W. Tu, and M. A. Musen, "Rasta: A distributed temporal abstraction system to facilitate knowledge-driven monitoring of clinical databases," in *10th World Congress on Medical Informatics (MedInfo2001)*, vol. 84, pp. 508–512, IOS Press, 2001.

[38] G. Carrault, M. O. Cordier, R. Quiniou, and F. Wang, "Temporal abstraction and inductive logic programming for arrhythmia recognition from electrocardiograms," *Artificial Intelligence in Medicine*, vol. 28, no. 3, pp. 231–263, 2003.

[39] J. Hunter and N. McIntosh, "Knowledge-based event detection in complex time series data," in *Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making*, vol. 1620 of *Lecture Notes in Computer Science*, (Aalborg, Denmark), pp. 271–280, Springer-Verlag, 1999.

[40] R. Bellazzi, C. Larizza, P. Magni, S. Montani, and G. D. Nicolao, "Intelligent analysis of clinical time series by combining structural filtering and temporal abstractions," *Lecture Notes in Computer Science*, vol. 1620, p. 261, 1999.

[41] J. Yu, J. Hunter, E. Reiter, and S. Sripada, "An approach to generating summaries of time series data in the gas turbine domain," in *International Conference on Info-tech & Info-net (ICII2001)*, (Beijing), pp. 44–51, 2001.

[42] J. Yu, E. Reiter, J. Hunter, and S. Sripada, "Sumtime-turbine: A knowledge-based system to communicate time series data in the gas turbine domain," in *IEA/AIE-2003* (P. Chung, ed.), pp. 379–384, Springer, 2003.

[43] J. Yu, E. Reiter, J. Hunter, and S. Sripada, "A new architecture for summarising time series data," in *Third International Conference on Natural Language Generation (INLG-2004), Poster Session*, 2004.

[44] J. Hunter, G. Ewing, Y. Freer, R. Logie, P. McCue, and N. McIntosh, "Neonate: Decision support in the neonatal intensive care unit; a preliminary report," in *9th Conference on Artificial Intelligence in Medicine in Europe, (AIME 2003)* (P. B. Michel Dojat, E.T.Keravnou, ed.), pp. 41–45, Springer, 2003.

[45] A. Seyfang and S. Miksch, "Asgaard's contribution to the guideline representation comparison," tech. rep., Vienna University, Institute for Software Technology, 2001.

[46] A. Seyfang, S. Miksch, and M. Marcos, "Combining diagnosis and treatment using asbru," *International Journal of Medical Informatics*, vol. 68, no. 1-3, pp. 49–57, 2002.

[47] S. Miksch, W. Horn, C. Popow, and F. Paky, "Utilizing temporal data abstraction for data validation and therapy planning for artificially ventilated newborn infants," *Artificial Intelligence in Medicine*, vol. 8, no. 6, pp. 543–576, 1996.

[48] S. Y. Belal, A. F. G. Taktak, A. Nevill, and A. Spencer, "An intelligent ventilation and oxygenation management system in neonatal intensive care using fuzzy trend template fitting," *Physiol. Meas.*, vol. 26, pp. 555–570, 2005.

[49] M. Dojat, F. Pachet, Z. Guessoum, D. Touchard, A. Harf, and L. Brochard, "Neoganesh: a working system for the automated control of assisted ventilation in icus," *Artificial Intelligence in Medicine*, vol. 11, no. 2, pp. 97–117, 1997.

[50] A.-S. Silvent, M. Dojat, and C. Garbay, "Multi-level temporal abstraction for medical scenario construction," *International Journal of Adaptive Control and Signal Processing*, vol. 19, no. 4, pp. 377–394, 2005.

[51] L. Chittaro and M. Dojat, "Using a general theory of time and change in patient monitoring: Experiment and evaluation," *Computers in Biology and Medicine, Time-oriented Systems in Medicine*, vol. 27, no. 5, pp. 435–452, 1997.

[52] S. Miksch and A. Seyfang, "Finding intuitive abstractions of high-frequency data," in *Workshop: Knowledge-Based Information Management, in conjunction with the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making (AIMDM99)*, (Aalborg, Denmark), pp. 13–26, 1999.

[53] S. G. Sripada, E. Reiter, J. Hunter, and J. Yu, "Summarizing neonatal time series data," in *research note sessions of the EACL03*, (Budapest, Hungary), pp. 167–170, 2003.

[54] S. G. Sripada, E. Reiter, J. Hunter, J. Yu, and I. Davy, "Modelling the task of summarising time series data using ka techniques," in *ES2001*, pp. 183–196, 2001.

[55] S. Miksch, A. Seyfang, W. Horn, and C. Popow, "Abstracting steady qualitative descriptions over time from noisy, high-frequency data," in *Artificial Intelligence in Medicine and Medical Decision Making (AIMDM'99)* (W. Horn, Y. Shahar, G. Lindberg, S. Andreassen, and J. Wyatt, eds.), pp. 281–290, 1999.

[56] A. Semrl, "Real-time monitoring of dense continuous data," in *Workshop: 'Computers in Anaesthesia and Intensive Care: Knowledge-Based Information Management' during the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making (AIMDM'99)*, (Aalborg, Denmark), 1999. http://www.ifs.tuwien.ac.at/ silvia/caic/#papers.

[57] H. Chaudet, "Extending the event calculus for tracking epidemic spread," *Artificial Intelligence in Medicine*, vol. 38, no. 2, pp. 137–156, 2006.

[58] C. Combi and L. Chittaro, "Abstraction on clinical data sequences: an object-oriented data model and a query language based on the event calculus," *Artificial Intelligence in Medicine*, vol. 17, no. 3, pp. 271–301, 1999.

[59] Y. Shahar, "Knowledge-based temporal interpolation," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 11, no. 1, pp. 123–144, 1999.

[60] W. Horn, "Ai in medicine on its way from knowledge-intensive to data-intensive systems," *Artificial Intelligence in Medicine*, vol. 23, no. 1, pp. 5–12, 2001.

[61] S. Jonsson and K. Lukka, "There and back again: Doing interventionist research in management accounting," in *Handbook of Management Accounting Research* (C. Chapman, A. Hopwood, and M. Shields, eds.), pp. 373–396, Berlin, Heidelberg, New York: Elsevier, 2007.

[62] E. Kasanen and K. Lukka, "The constructive approach in management accounting," *Journal of Management Accounting Research*, vol. 5, pp. 243–264, 1993.

[63] K. Lukka, "The key issues of applying the constructive approach to field research," in *Management expertise for the new millennium. In Commemoration of the 50th anniversary of the Turku School of Economics and Business Administration.* (T. Reponen, ed.), pp. 113–128, Turku School of Economics and Business Administration, 2000.

[64] K. Lukka, "The constructive research approach," in *Case study research in logistics* (L. Ojala and O. Hilmola, eds.), pp. 83–101, Turku School of Economics and Business Administration, 2003.

[65] Y. Shahar and M. Molina, "Knowledge-based spatiotemporal linear abstraction," *Pattern Analysis and Applications*, vol. 1, no. 2, pp. 91–104, 1997.

[66] C. McGregor, M. Purdy, and B. Kneale, "Compression of xml physiological data streams to support neonatal intensive care unit web services," in *IEEE International Conference on e-Technology, e-Commerce and e-Service*, (Hong Kong), pp. 486–489, IEEE, 2005.

[67] C. McGregor, J. Heath, and M. Wei, "A web service based framework for the transmission of physiological data for local and remote neonatal intensive care," in *IEEE International Conference on e-Technology, e-Commerce and e-Service*, (Hong Kong), pp. 496–501, IEEE, 2005.

[68] C. McGregor and J. Schiefer, "A web-service based framework for analyzing and measuring business performance," *Information Systems and E-Business Management*, vol. 2, no. 1, pp. 89–100, 2004.

[69] C. McGregor, J. Schiefer, and M. z. Muehlen, "A shareable web service-based intelligent decision support system for on-demand business process management," *International Journal Business Process Integration and Management*, vol. 1, no. 3, pp. 156–174, 2006.

[70] C. McGregor, "Mobility in healthcare for remote intensive care unit clinical management," in *Handbook of Research in Mobile Business: Technical, Methodological and Social Perspective* (B. Unhelkar, ed.), pp. 83–95, Hershey, PA, UWA: IDEA Group Publishing, 2005.

[71] C. McGregor, B. Kneale, and M. Tracy, "Bush babies broadband: On-demand virtual neonatal intensive care unit support for regional australia," in *3rd International Conference on Information Technology and Applications (ICITA)*, (Sydney, Australia), pp. 113–117, IEEE, 2005.

[72] W. J. Clancey, "Heuristic classification," *Artificial Intelligence*, vol. 27, no. 3, pp. 289–350, 1985.

[73] C. Combi and Y. Shahar, "Temporal reasoning and temporal data maintenance in medicine: Issues and challenges," *Computers in Biology and Medicine*, vol. 27, no. 5, pp. 353–368, 1997.

[74] M. Kahn, "In pursuit of time's arrow: temporal reasoning in medical decision support," in *4th Conference on Artificial Intelligence in Medicine*, (Munich), pp. 3–6, IOS Press, 1993.

[75] E. Coiera, "Intelligent monitoring and control of dynamic physiological systems," *Artificial Intelligence in Medicine*, vol. 5, no. 1, pp. 1–8, 1993.

[76] Y. Shahar, *A knowledge-based method for temporal abstraction of clinical data*. PhD thesis, Stanford University, 1994.

[77] S. Chakravarty and Y. Shahar, "A constraint-based specification of periodic patterns in time-oriented data," in *Sixth International Workshop on Temporal Representation and Reasoning (TIME-99)*, (Orlando, FL, USA), pp. 29–40, 1999.

[78] S. Chakravarty and Y. Shahar, "Specification and detection of periodic patterns in clinical data," in *Fourth Workshop on Intelligent Data Analysis in Medicine and Pharmacology IDAMAP-99)*, (Washington, DC, USA), pp. 20–31, 1999.

[79] S. Chakravarty and Y. Shahar, "Acquisition and analysis of repeating patterns in time-oriented clinical data," *Methods of Information in Medicine*, vol. 40, no. 1, pp. 410–420, 2001.

[80] C. Fuchsberger and S. Miksch, "Asbru's execution engine: Utilizing guidelines for artificial ventilation of newborn infants," in *'Intelligent Data Analysis in Medicine and Pharmacology' workshop, (IDAMAP 2003) as part of Artificial Intelligence in Medicine (AIME-03)*, (Protaras, Cyprus), pp. 119–125, 2003.

[81] M. M. Gaber, S. Krishnaswamy, and A. Zaslavsky, "Cost-efficient mining techniques for data streams," in *Second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation*, vol. 32, (Dunedin, New Zealand), pp. 109–114, 2004.

[82] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and issues in data stream systems," in *21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, (Madison, Wisconsin), pp. 1–16, ACM Press, 2002.

[83] M. Griffin, T. O'Shea, E. Bissonette, F. J. Harrell, D. Lake, and J. Moorman, "Abnormal heart rate characteristics are associated with neonatal mortality," *Pediatric Research*, vol. 55, no. 5, pp. 782–788, 2004.

[84] D. Boaz and Y. Shahar, "A framework for distributed mediation of temporal-abstraction queries to clinical databases," *Artificial Intelligence in Medicine*, vol. 34, no. 1, pp. 3–24, 2005.

[85] M. Ramati and Y. Shahar, "Probabilistic abstraction of multiple longitudinal electronic medical records," in *10th Conference on Artificial Intelligence in Medicine, AIME 2005* (S. Miksch, J. Hunter, and E. Keravnou, eds.), vol. 3581, (Aberdeen, UK), pp. 43–47, Springer Berlin / Heidelberg, 2005.

[86] Y. Shahar, D. Goren-Bar, D. Boaz, and G. Tahan, "Distributed, intelligent, interactive visualization and exploration of time-oriented clinical data and their abstractions," *Artificial Intelligence in Medicine*, vol. 38, no. 2, pp. 115–135, 2006.

[87] D. Cukierman and J. Delgrande, "Towards a formal characterization of temporal repetition with closed time," in *International Workshop on Temporal Reasoning and Representation (TIME)-98*, pp. 140–147, IEEE Press, 1998.

[88] P. Terenziani, "Qualitative and quantitative temporal constraints about numerically quantified periodic events," in *Temporal Representation and Reasoning, 1997. (TIME '97), Proceedings., Fourth International Workshop on*, pp. 94–101, 1997.

[89] Y. Shahar, S. Miksch, and P. Johnson, "The asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines," *Artificial Intelligence in Medicine*, vol. 14, no. 1-2, pp. 29–51, 1998.

[90] D. Foster, C. McGregor, and S. El-Masri, "A survey of agent-based intelligent decision support systems to support clinical management and research," in *Workshop on Multi-Agent Systems for Medicine, Computational Biology and Bioinformatics (MAS*BIOMED)*, (Utrecht, Netherlands), pp. 16–34, Autonomous Agents and Multi-Agent Systems (AAMAS 05), 2005.

[91] L. Chittaro, C. Combi, E. Cervesato, R. Cervesato, F. Antonini-Canterin, G. Nicolosi, and D. Zanuttini, "Specifying and representing temporal abstractions on clinical data by a query language based on the event calculus," in *Computers in Cardiology*, pp. 633–636, 1997.

[92] W. Horn, S. Miksch, G. Egghart, C. Popow, and F. Paky, "Effective data validation of high-frequency data: Time-point-, time-interval-, and trend-based methods," *Computers in Biology and Medicine*, vol. 27, no. 5, pp. 389–409, 1997.

[93] C. Larizza, A. Moglia, and M. Stefanelli, "M-htp: A system for monitoring heart transplant patients," *Artificial Intelligence in Medicine*, vol. 4, no. 2, pp. 111–126, 1992.

[94] S. Miksch, W. Horn, C. Popow, and F. Paky, "Context-sensitive and expectation guided temporal abstraction of high frequency data," in *The Tenth International Workshop for Qualitative Reasoning (QR-96)* (Y. Iwasaki and A. Farquhar, eds.), (Fallen Leaf Lake, California, USA), pp. 154–163, AAAI Press, Menlo Park, 1996.

[95] T. A. Russ, "Use of data abstraction methods to simplify monitoring," *Artificial Intelligence in Medicine*, vol. 7, pp. 497–514, 1995.

[96] Y. Shahar, "A framework for knowledge-based temporal abstraction," *Artificial Intelligence*, vol. 90, no. 1-2, pp. 79–133, 1997.

[97] M. Stacey, "Knowledge based temporal abstraction within the neonatal intensive care domain," in *CSTE Innovation Conference*, (Penrith, Australia), University of Western Sydney, 2005.

[98] S. Miksch, W. Horn, C. Popow, and F. Paky, "Time-orientated analysis of data in ICU monitoring," in *Intelligent Data Analysis in Medicine and Pharmacology: An overview* (N. Lavrac, E. Keravnou, and B. Zupan, eds.), pp. 17–34, Boston: Kluwer Academic Publishers, 1997.

[99] I. J. Haimowitz and I. S. Kohane, "Managing temporal worlds for medical trend diagnosis," *Artificial Intelligence in Medicine*, vol. 8, no. 3, pp. 299–321, 1996.

[100] L. Golab and M. T. Ozsu, "Data stream management issues - a survey," Tech. Rep. CS-2003-08, University of Waterloo, 2003.

[101] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, (San Diego, California), pp. 2–11, 2003.

[102] A. Arasu, S. Babu, and J. Widom, "The cql continuous query language: Semantic foundations and query execution," tech. rep., (Technical Report) Stanford University, 2003.

[103] M. Datar, A. Gionis, P. Indyk, and R. Motwani, "Maintaining stream statistics over sliding windows," in *ACM-SIAM Symposium on Discrete Algorithms(SODA)*, 2002.

[104] S. Gollapudi and D. Sivakumar, "Framework and algorithms for trend analysis in massive temporal data sets," in *Thirteenth ACM conference on Information and knowledge management*, (Washington, D.C., USA), pp. 168–177, 2004.

[105] Y. Zhu and D. Shasha, "Statstream: Statistical monitoring of thousands of data streams in real time," Tech. Rep. TR2002-827, New York University, 2002.

[106] N. Alon, Y. Matias, and M. Szegedy, "The space complexity of approximating the frequency moments," in *Annual ACM Symposium on the Theory of Computing*, (Pennsylvania, USA), pp. 20–29, 1996.

[107] J. Gehrke, F. Korn, and D. Srivastava, "On computing correlated aggregates over continual data streams," in *ACM SIGMOD International Conference on Management of Data*, (Santa Barbara, California, United States), pp. 13–24, 2001.

[108] A. Arasu, B. Babcock, S. Babu, J. Cieslewicz, M. Datar, K. Ito, R. Motwani, U. Srivastava, and J. Widom, "Stream: The stanford data stream management system," Tech. Rep. 2004-20, Stanford InfoLab, 2004.

[109] L. Chittaro and A. Montanari, "Efficient temporal reasoning in the cached event calculus," *Computational Intelligence*, vol. 12, no. 3, pp. 359–382, 1996.

[110] S. Chakravarty and Y. Shahar, "Capsul: A constraint-based specification of repeating patterns in time-oriented data," *Annals of Mathematics and Artificial Intelligence*, vol. 30, no. 1 - 4, pp. 3–22, 2000.

[111] M. K. Goldstein, B. B. Hoffman, R. W. Coleman, M. A. Musen, S. W. Tu, A. Advani, R. Shankar, and M. J. O'Connor, "Implementing clinical practice guidelines while taking account of changing evidence: Athena dss, an easily modifiable decision support system for managing hypertension in primary care," in *AMIA Annual Symposium*, pp. 300–304, 2000.

[112] M. Balaban, D. Boaz, and Y. Shahar, "Applying temporal abstraction in medical information systems," *Annals of Mathematics, Computing and Teleinformatics*, vol. 1, no. 1, pp. 56–64, 2003.

[113] M. A. Musen, S. W. Tu, A. K. Das, and Y. Shahar, "Eon: a component-based approach to automation of protocol-directed therapy," *Journal of the American Medical Informatics Association*, vol. 3, no. 6, pp. 367–388, 1996.

[114] J. H. Nguyen, Y. Shahar, S. W. Tu, A. K. Das, and M. A. Musen, "Integration of temporal reasoning and temporal-data maintenance into a reusable database mediator to answer abstract, time-oriented queries: The tzolkin system," *Journal of Intelligent Information Systems*, vol. 13, pp. 121–145, 1999.

[115] Y. Shahar and C. Cheng, "Model-based visualization of temporal abstractions," in *Fifth International Workshop on Temporal Representation and Reasoning (TIME '98)*, (Sanibel Island, Florida), pp. 11–20, 1997.

[116] Y. Shahar, D. Goren-Bar, M. Galperin, D. Boaz, and G. Tahan:, "Knave-11: A distributed architecture for interactive visualization and intelligent exploration of time- oriented clinical data," in *Intelligent Data Analysis in Medicine and Pharamcology (IDAMAP2003)*, (Protaras, Cyprus), pp. 103–110, 2003.

[117] A. Spokoiny and Y. Shahar, "An active database architecture for knowledge-based incremental abstraction of complex concepts from continuously arriving time-oriented raw data," *Journal of Intelligent Information Systems*, vol. 28, no. 3, pp. 199–231, 2007.

[118] A. Spokoiny and Y. Shahar, "A knowledge-based time-oriented active database approach for intelligent abstraction, querying and continuous monitoring of clinical data," in *11th World Congress on Medical INformatics (MEDINFO 2004)* (M. Fieschi, E. Coiera, and Y.-C. Li, eds.), (San Francisco, USA), pp. 84–88, 2004.

[119] J. Widom and S. Ceri, *Active Database Systems: Triggers and Rules For Advanced Database Processing*. Morgan Kaufmann, 1996.

[120] C. A. Mareco and L. E. Bertossi, "Specification and implementation of temporal databases in a bitemporal event calculus," in *Advances in Conceptual Modeling: ER '99* (P. P. Chen, D. W. Embley, J. Kouloumdjian, S. W. Liddle, and J. F. Roddick, eds.), vol. 1727 of *Lecture Notes in Computer Science*, (Paris, France), pp. 74–85, Springer, 1999.

[121] S. M. Sripada, "Efficient implementation of the event calculus for temporal databases," in *12th International Conference on Logic Programming*, pp. 99–113, MIT Press, 1995.

[122] J. F. Allen, "Towards a general theory of action and time," *Artificial Intelligence*, vol. 23, no. 2, pp. 123–154, 1984.

[123] E. Keravnou, "Temporal abstraction of medical data: Deriving periodicity," in *Intelligent Data Analysis in Medicine and Pharmacology (IDAMAP1996), 12th European Conference on Artificial Intelligence ECAI-96* (N. Lavrac, E. T. Keravnou, and B. Zupan, eds.), (Budapest, Hungary), pp. 61–79, Kluwer Academic Publishers, 1996.

[124] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting time series: A survey and novel approach," in *Data Mining in Time Series Databases* (M. Last, A. Kandel, and H. Bunke, eds.), pp. 1–22, World Scientific Publishing Company, 1993.

[125] D. Cukierman and J. Delgrande, "Characterizing temporal repetition," in *International Workshop on Temporal Reasoning and Representation (TIME) - 96*, pp. 80–87, IEEE Press, 1996.

[126] M. Barquin, L. Lievre, C. Dousson, A. Aghasaryan, T. V. Dong, E. Fabre, E. Benveniste, C. Jard, M. Cordier, and F. Levy, "Magda requirements," Tech. Rep. MAGDA/PF/LIV/001, The MAGDA project, 2000.

[127] M. Ghallab, "On chronicles: Representation, on-line recognition and learning," in *Principles of Knowledge Representation and Reasoning* (Aiello, Doyle, and Shapiro, eds.), pp. 597–606, Morgan-Kauffman, 1996.

[128] F. Wang, G. Carrault, R. Quiniou, M.-O. Cordier, and L. Luo, "Temporal reasoning based automatic arrhythmias recognition," in *Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003 IEEE International Conference on*, vol. 2, pp. 1114–1119 vol.2, 2003.

[129] A. Cohn, S. Rosenbaum, M. Factor, and P. Miller, "Dynascene: An approach to computer-based intelligent cardiovascular monitoring using sequential clinical scenes," *Methods of Information in Medicine*, vol. 29, pp. 122–131, 1990.

[130] C. Dousson, P. Gaborit, and M. Ghallab, "Situation recognition: representation and algorithms," in *International Joint Conference on Artificial Intelligence (IJCAI-93)*, (Chambery, France), pp. 166–172, 1993.

[131] S. Sharshar, L. Allart, and M. C. Chambrin, "A new approach to the abstraction of monitoring data in intensive care," in *Artificial Intelligence in Medicine (AIME 2005)* (S. M. e. al., ed.), vol. 3581, (Aberdeen, Scotland), pp. 13–22, Springer-Verlag, 2005.

[132] D. Calvelo, M. C. Chambrin, D. Pomorski, and P. Ravaux, "Towards symbolization using data-driven extraction of local trends for icu monitoring," *Artificial Intelligence in Medicine*, vol. 19, no. 3, pp. 203–223, 2000.

[133] M. Kahn, L. M. Fagan, and L. B. Sheiner, "Combining physiologic models and symbolic methods to interpret time varying patient data," *Methods of Information in Medicine*, vol. 30, pp. 167–178, 1991.

[134] M. Kahn, J. Ferguson, E. Shortcliffe, and L. Fagan, "Representation and use of temporal information in oncocin," in *Ninth Annual Symposium on Computer Applications in Medical Care* (M. J. Ackerman, ed.), (Los Alamitos, California, USA), pp. 172–176, IEEE Comput. Soc. Press, 1985.

[135] A. Seyfang, S. Miksch, W. Horn, M. Urschitz, C. Popow, and C. Poets, "Using time-oriented data abstraction methods to optimize oxygen supply for neonates," in *Artificial Intelligence in Medicine (AIME01)*, (Cascais, Portugal), pp. 217–226, 2001.

[136] M. Factor, "The process trellis architecture for real-time monitors," in *2nd ACM SIGPLAN Symposium on Principles & Practice of Parallel Programming*, (Seattle, Washington, United States), pp. 147–155, ACM Press, 1990.

[137] E. Coiera, "Intermediate depth representations," *Artificial Intelligence in Medicine*, vol. 4, no. 6, pp. 431–445, 1992.

[138] M. Dojat and C. Sayettat, "A realistic model for temporal reasoning in real-time patient monitoring," *Applied Artificial Intelligence*, vol. 10, no. 2, pp. 121–143, 1996.

[139] J. Fox, N. Johns, and A. Rahmansadeh, "Disseminating medical knowledge: the proforma approach," *Artificial Intelligence in Medicine*, vol. 14, pp. 157–182, 1998.

[140] E. Feigenbaum, "The art of artificial intelligence: Themes and case studies of knowledge engineering," Tech. Rep. STAN-SC-77-621, Department of Computer Science, Stanford University, 1977.

[141] E. Feigenbaum and P. McCurduck, *The Fifth Generation*. London: Pan Books, 1984.

[142] S. L. Achour, M. Dojat, C. Rieux, P. Bierling, and E. Lepage, "A UMLS-based knowledge acquisition tool for rule-based clinical decision support system

development," *Journal of the American Medical Informatics Association*, vol. 8, no. 4, pp. 351–360, 2001.

[143] R. Kosara and S. Miksch, "Metaphors of movement," *Artificial Intelligence in Medicine - special issue: Information Visualisation in Medicine*, vol. 22, no. 2, pp. 111–131, 2001.

[144] J. Augusto, "The logical approach to temporal reasoning," *Artificial Intelligence Review*, vol. 16, no. 4, pp. 301–333, 2001.

[145] L. Chittaro and A. Montanari, "Temporal representation and reasoning in artificial intelligence: Issues and approaches," *Annals of Mathematics and Artificial Intelligence*, vol. 28, no. 1 - 4, pp. 47–106, 2000.

[146] E. T. Keravnou, "A multidimensional and multigranular model of time for medical knowledge-based systems," *Journal of Intelligent Information Systems*, vol. 13, no. 1 - 2, pp. 73–120, 1999.

[147] I. Kohane, "Temporal reasoning in medical expert systems," Tech. Rep. Technical Report 389, Laboratory of Computer Science, Massachusetts Institute of technology, 1987.

[148] R. A. Kowalski and M. J. Sergot, "A logic-based calculus of events," *New Generation Computing*, vol. 4, pp. 67–95, 1986.

[149] R. Kowalski and F. Sadri, "Reconciling the event calculus with the situational calculus," *Journal of Logic Programming*, vol. 31, pp. 39–58, 1997.

[150] Y. Shoham, "Temporal logics in AI: Semantical and ontological considerations," *Artificial Intelligence*, vol. 33, no. 1, pp. 89–104, 1987.

[151] D. McDermott, "A temporal logic for reasoning about process and plans," *Cognitive Science*, vol. 6, pp. 101–155, 1982.

[152] T. Dean and D. McDermott, "Temporal database management," *Artificial Intelligence*, vol. 32, pp. 1–55, 1987.

[153] M. Vilain and H. Kautz, "Constraint propagation algorithms for temporal reasoning," in *Sixth National Conference on Artificial Intelligence*, (Los Angeles, USA), pp. 377–382, Morgan Kaufmann, 1986.

[154] M. Vilain, H. Kautz, and P. V. Beek, "Constraint propagation algorithms for temporal reasoning," in *Readings in qualitative reasoning about physical systems* (D. Weld and J. D. Kleer, eds.), pp. 373–381, San Francisco, USA: Morgan Kaufmann Publishers Inc., 1989.

[155] L. Chittaro, A. Montanari, and A. Provetti, "Sceptical and credulous event calculi for supporting modal queries," in *11th European Conference on Artificial Intelligence (ECAI'94)*, pp. 361–365, Wiley, Chichester, 1994.

[156] R. Kowalski, "Database updates in the event calculus," *The Journal of Logic Programming*, vol. 12, no. 1-2, pp. 121–146, 1992.

[157] I. Cervesato, L. Chittaro, and A. Montanari, "What the event calculus does and how to do it efficiently," in *Joint Conference on Declarative Programming - GULP-PRODE'94* (M. Apuente and I. Ramos, R. Barbuti, eds.), (Peniscola, Spain), pp. 336–350, 1994.

[158] I. Cervesato, M. Franceschet, and A. Montanari, "A guided tour through some extensions of the event calculus," *Computational Intelligence*, vol. 16, no. 2, pp. 307–347, 2000.

[159] C. Evans, "The macro-event calculus: Representing temporal granularity," in *Pacific Rim International Conference on Artificial Intelligence - PRICAI'90*, (Nagoya, Japan), IOS Press, 1990.

[160] M. P. Shanahan, "Representation of continuous change in the event calculus," in *Ninth Conference on Artificial Intelligence - ECAI'90*, (Stockholm, Sweden), pp. 598–603, 1990.

[161] A. Salatian and J. Hunter, "Deriving trends in historical and real-time continuously sampled medical data," *Journal of Intelligent Information Systems*, vol. 13, no. 1-2, pp. 47–71, 1999.

[162] R. A. Haddad and T. W. Parsons, *Digital Signal Processing, Applications, and Hardware*. Computer Science Press, 1991.

[163] M. H. Bhlen, R. T. Snodgrass, and M. D. Soo, "Coalescing in temporal databases," in *22nd International Conference on Very Large Data Bases (VLDB'96)* (T. M. Vijayaraman, A. P. Buchmann, C. Mohan, and N. L. Sarda, eds.), pp. 180–191, 1996.

[164] E. Coiera, *Reasoning with Qualitative Disease Histories for Diagnostic Patient Monitoring*. PhD thesis, University of New South Wales, 1989.

[165] C. L. Tsien, "Event discovery in medical time series data," in *American Medical Informatics Association (AMIA) Symposium*, pp. 858–862, Hanley & Belfus, 2000.

[166] C. L. Tsien, I. S. Kohane, and N. McIntosh, "Multiple signal integration by decision tree induction to detect artifacts in the neonatal intensive care unit," *Artificial Intelligence in Medicine*, vol. 19, no. 3, pp. 189–202, 2000.

[167] T. Lawless, "Crying wolf: false alarms in a pediatric intensive care unit," *Critical Care Medicine*, vol. 22, pp. 81–85, 1994.

[168] E. Koski, A. Makivirta, and T. Sukuvaara, "Frequency and reliability of alarms in the monitoring of cardiac post-operative patients," *Int J Clin Monit Comput*, vol. 7, pp. 129–133, 1990.

[169] P. V. Le, "A clinical trial of trendx: An automated trend-detection program," Master's thesis, Masters of Engineering, Massachusetts Institute of Technology, Massachusetts, USA, April 1996.

[170] M. Desouza, "Automated medical trend detection," Master's thesis, Masters of Engineering, Massachusetts Institute of Technology, Massachusetts, USA, 2000.

[171] J. Li and T.-Y. Leong, "Using linear regression functions to abstract high frequency data in medicine," in *AMIA Annual Symposium*, (Los Angeles, USA), pp. 492–496, 2000.

[172] L. Fagan, *VM: Representing Time Dependent Relations in a Medical Setting*. PhD thesis, Stanford University, 1980.

[173] R. Davis, B. G. Buchanan, and E. H. Shortcliffe, "Production rules as a representation for knowledge-based consultation program," in *Readings in Medical Artificial Intelligence: The First Decade* (W. J. Clancy and E. H. Shortcliffe, eds.), pp. 98–130, Reading, MA, USA: Addison-Wesley, 1984.

[174] J. Wallis and E. Shortliffe, "Explanatory power of medical expert systems: studies in the representation of causal relationships for clinical consultations," *Meth. Inform. Med.*, vol. 21, pp. 127–136, 1982.

[175] R. Davis, "Knowledge acquisition in rule-based systems: knowledge representation as a basis for system construction and maintenance," *ACM Sigart Bulletin*, vol. 63, p. 23, 1977.

[176] association for women's obstetric health neonatal nurses (AWOHNN) and national nurses association for neonatal nurses (NANN), *Core Curriculum for Neonatal Intensive Care Nursing*. St Louis, Missouri USA: Elsevier Saunders, 3rd ed., 2004.

[177] G. B. Merenstein and S. L. Gardner, *Handbook of Neonatal Intensive Care*. The C.V. Mosby Company, 2nd ed., 1989.

[178] M. Tracy, L. Downe, and J. Holberton, "How safe is intermittent positive pressure ventilation in preterm babies ventilated from delivery to newborn intensive care unit?," no. 89, pp. 84–87, 2004.

[179] J. J. Volpe, "Brain injury in the premature infant: Overview of clinical aspects, neuropathology, and pathogenesis," *Seminars in Pediatric Neurology*, vol. 5, no. 3, pp. 135–151, 1998.

[180] J. J. Volpe, "Brain injury in the premature infant - from pathogenesis to prevention," *Brain and Development*, vol. 19, no. 8, pp. 519–534, 1997.

[181] P. B. Pandit, K. O'Brien, E. Asztalos, E. Colucci, and M. S. Dunn, "Outcome following pulmonary haemorrhage in very low birthweight neonates treated with surfactant," *Arch. Dis. Child. Fetal Neonatal Ed.*, vol. 81, no. 1, pp. F40–44, 1999.

[182] M. I. Levene, C. L. Fawer, and R. F. Lamont, "Risk factors in the development of intraventricular haemorrhage in the preterm neonate," *Arch Dis Child*, vol. 57, no. 6, pp. 410–417, 1982.

[183] R. M. Ward and J. C. Beachy, "Neonatal complications following preterm birth," *BJOG: an International Journal of Obstetrics and Gynaecology*, vol. 110, no. 20, pp. 8–16, 2003.

[184] E. S. Ogata, G. A. Gregory, J. A. Kitterman, R. H. Phibbs, and W. H. Tooley, "Pneumothorax in respiratory distress syndrome: incidence and effect on vital signs, blood gases and ph," *Pediatrics*, vol. 58, pp. 177–183, 1976.

[185] V. Y. H. Yu, P. Y. Wong, B. Bajuk, and W. Szymonowitz, "Pulmonary air leak in extremely low birthweight infants," *Arch Dis Child*, vol. 61, pp. 239–241, 1986.

[186] A. Hill, J. M. Perlman, and J. Volpe, "Relationship of pneumothorax to occurrence of intraventricular hemorrhage in premature newborns," *Pediatrics*, vol. 69, pp. 144–149, 1982.

[187] M. Tracy, "Clinically relevant fall in mean blood pressure and oxygen saturation for premature infants." in personal communications, July 2007.

[188] Preemie.info, "Photographs of neonatal intensive care: humidicrib and sensor attachments." http://www.preemie.info. Access date: June 2005.

[189] E. T. Mueller, *Commonsense Reasoning*. San Francisco, USA: Morgan Kaufmann, 2006.

[190] J. Allen, A. M. Frisch, and D. J. Litman, "The Rochester dialogue system," in *National Conference on Artificial Intelligence*, (Pittsburgh, USA), pp. 66–70, 1982.

[191] A. Nakhimovsky, "Temporal reasoning in natural language understanding: the temporal structure of the narrative," in *Third conference on European chapter of the association for computational linguistics*, (Copenhagen, Denmark), pp. 262–269, Association for Computational Linguistics, 1987.

[192] R. Snodgrass, "Temporal databases," in *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, vol. 639, pp. 22–64, Berlin / Heidelberg: Springer, 1992.

[193] J. Clifford, S. Gadia, S. Jajodia, A. Degev, R. Snodgrass, and A. Tansel, eds., *Temporal databases: theory, design, and implementation*. Benjamin-Cummings Publishing Co., Inc., 1993.

[194] M. Shanahan, "Reinventing shakey," in *Logic-based artificial intelligence*, pp. 233–253, Norwell, MA, USA: Kluwer Academic Publishers, 2000.

[195] J. Knottenbelt and K. Clark, "Contract-related agents," in *Computational Logic in Multi-Agent Systems*, vol. 3900, pp. 226–242, Berlin / Heidelberg: Springer, 2006.

[196] A. Farrell, M. Sergot, M. Salle, and C. Bartolini, "Using the event calculus for tracking the state of normative contracts," *International Journal of Cooperative Information Systems (IJCIS)*, vol. 14, no. 2/3, pp. 99–129, 2005.

[197] C. Efstratiou, A. Friday, N. Davies, and K. Cheverst, "Utilising the event calculus for policy driven adaptation on mobile systems," in *Third International Workshop on Policies for Distributed Systems and Networks*, (Monterey, CA, USA), pp. 13–24, 2002.

[198] A. K. Bandara, E. C. Lupu, and A. Russo, "Using event calculus to formalise policy specification and analysis," in *4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'03)*, (Lake Como, Italy), p. 26, IEEE Computer Society, 2003.

[199] P. Yolum and M. P. Singh, "Reasoning about commitments in the event calculus: An approach for specifying and executing protocols," *Annals of Mathematics and Artificial Intelligence*, vol. 42, no. 1-3, pp. 227–253, 2004.

[200] E. Keravnou, "Temporal vagueness in medical reasoning," *International Journal of Systems Research and Information Science*, vol. 7, pp. 3–28, 1995.

[201] E. T. Keravnou, "Temporal diagnostic reasoning based on time-objects," *Artificial Intelligence in Medicine*, vol. 8, no. 3, pp. 235–265, 1996.

[202] E. Keravnou, Y. Shahar, D. G. M. Fisher, and L. Vila, "Temporal reasoning in medicine," in *Foundations of Artificial Intelligence*, vol. Volume 1, pp. 587–653, Elsevier, 2005.

[203] A. Galton, "Time and change for AI," in *Handbook of Logic in Artificial Intelligence and Logic Programming* (D. Gabbay, C. Hogger, and J. Robinson, eds.), vol. 4, New York, USA: Oxford University Press, 1995.

[204] J. McCarthy and P. J. Hayes, "Some philosophical problems from the standpoint of artificial intelligence," in *Machine Intelligence 4* (B. Meltzer and D. Michie, eds.), pp. 463–502, Edinburgh University Press, 1969.

[205] R. Miller and M. Shanahan, "Some alternative formulations of the event calculus," in *Computational Logic: Logic Programming and Beyond*, vol. 2408, pp. 95–111, Heidelberg: Springer Berlin, 2002.

[206] M. Shanahan, "The event calculus explained," in *Artificial Intelligence Today, Volume 1600 Lecture Notes in Computer Science* (M. J. Woolridge and M. Veloso, eds.), pp. 409–430, Berlin: Springer, 1999.

[207] A. R. Post and J. Harrison, James H., "Protempa: A method for specifying and identifying temporal sequences in retrospective data for patient selection," *J Am Med Inform Assoc*, vol. 14, no. 5, pp. 674–683, 2007.

[208] C. Larizza, G. Bernuzzi, and M. Stefanelli, "A general framework for building patient monitoring systems," in *Artificial Intelligence in Medicine (AIME'95)*, (Pavia, Italy), pp. 91–102, 1995.

[209] R. Dechter, I. Meiri, and J. Pearl, "Temporal constraint networks," *Artificial Intelligence*, vol. 49, no. 1-3, pp. 61–95, 1991.

[210] A. K. Mackworth, "Consistency in networks of relations," *Artificial Intelligence*, vol. 8, no. 1, pp. 99–118, 1977.

[211] E. C. Freuder, "A sufficient condition for backtrack-free search," *Journal of the ACM*, vol. 29, no. 1, pp. 24–32, 1982.

[212] U. Montanari, "Networks of constraints: Fundamental properties and applications to picture processing," *Information Sciences*, vol. 7, pp. 95–132, 1974.

[213] E. Schwalb and L. Vila, "Temporal constraints: A survey," *Constraints*, vol. 3, no. 2-3, pp. 124–149, 1998.

[214] Y. Shahar, "Dynamic temporal interpretation contexts for temporal abstraction," *Annals of Mathematics and Artificial Intelligence*, vol. 22, no. 1 - 2, pp. 159–192, 1998.

[215] J. Schiefer and C. McGregor, "Correlating events for monitoring business processes," in *6th International Conference on Enterprise Information Systems*, (Portucalense, Porto - Portugal), pp. 320–327, 2004.

[216] S. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie, "High speed and robust event correlation," *Communications Magazine, IEEE*, vol. 34, no. 5, pp. 82–90, 1996.

[217] G. Vigna, F. Valeur, and R. A. Kemmerer, "Designing and implementing a family of intrusion detection systems," in *9th European software engineering conference held jointly with 11th ACM SIGSOFT international symposium on Foundations of software engineering*, (Helsinki, Finland), pp. 88–97, ACM New York, 2003.

[218] P. Konana, A. Mok, L. Chan-Gun, W. Honguk, and L. Guangtian, "Implementation and performance evaluation of a real-time e-brokerage system," in *21st IEEE Real-Time Systems Symposium*, (Orlando, FL, USA), pp. 109–118, IEEE, 2000.

[219] F. Yamanaka and T. Nishiya, "Application of the intelligent alarm system for the plant operation," *Computers & Chemical Engineering*, vol. 21, pp. 625–630, 1997.

[220] N. H. Gehani, H. V. Jagadish, and O. Shmueli, "Composite event specification in active databases: Model & implementation," in *18th International Conference on Very Large Databases*, (Vancouver, Canada), pp. 327–338, 1992.

[221] P. Pietzuch, B. Shand, and J. Bacon, "Composite event detection as a generic middleware extension," *Network, IEEE*, vol. 18, no. 1, pp. 44–55, 2004.

[222] J. Carlson and B. Lisper, "An interval-based algebra for restricted event detection," in *Formal Modeling and Analysis of Timed Systems*, vol. 2791, pp. 121–133, Springer Berlin / Heidelberg: Springer, 2004.

[223] D. Zimmer and R. Unland, "On the semantics of complex events in active database management systems," in *15th International Conference on Data Engineering (ICDE'99)*, (Sydney, Australia), p. 392, IEEE, 1999.

[224] D. C. Luckham, "Rapide: A language and toolset for simulation of distributed systems by partial orderings of events," in *DIMACS Partial Order Methods in Verification - Workshop IV*, (Princeton University, New Jersey USA), pp. 329–357, 1996.

[225] N. W. Paton and O. Diaz, "Active database systems," *ACM Computing Surveys*, vol. 31, no. 1, pp. 63–103, 1999.

[226] S. Gatziu and K. Dittrich, "Detecting composite events in active database systems using petrinets," in *Fourth International Workshop on Research Issues in Data Engineering. Active Database Systems (RIDE-AIDS)*, (Houston, TX, USA), pp. 2–9, IEEE, 1994.

[227] S. Gatziu and K. Dittrich, "Events in an active object-oriented database system," in *1st International Workshop on Rules in Database Systems (RIDS)* (N. Paton and H. Williams, eds.), (Edinburgh, UK), pp. 23–29, Springer-Verlag, 1993.

[228] S. Chakravarthy, V. Krishnaprasad, E. Anwar, and S.-K. Kim, "Composite events for active databases: Semantics, contexts and detection," in *20th International Conference on Very Large Data Bases*, (Santiago, Chile), pp. 606–617, Morgan Kaufmann Publishers Inc., 1994.

[229] S. Chakravarthy and D. Mishra, "Snoop: an expressive event specification language for active databases," *Data & Knowledge Engineering*, vol. 14, no. 1, pp. 1–26, 1994.

[230] N. H. Gehani, H. V. Jagadish, and O. Shmueli, "Event specifcation in an active object-oriented database," in *ACM SIGMOD International Conference on Management of Data* (R. T. Snodgrass and M. Winslett, eds.), (San Diego, California, USA), pp. 81–90, 1992.

[231] N. Gehani, H. V. Jagadish, and O. Shmueli, "Compose: A system for composite specification and detection," in *Advanced Database Systems*, vol. 759 of *Lecture Notes in Computer Science*, pp. 3–15, Berlin / Heidelberg: Springer, 1993.

[232] G. Liu and A. K. Mok, "An event service framework for distributed real-time systems," in *IEEE Workshop on Middleware for Distributed Real-Time Systems and Services*, (San Francisco, California, USA), 1997.

[233] P. R. Pietzuch, B. Shand, and J. Bacon, "A framework for event composition in distributed systems," in *4th International Middleware Conference*, vol. 2672 of *Lecture Notes in Computer Science*, (Rio de Janeiro, Brazil), pp. 62–82, 2003.

[234] A. Galton and J. C. Augusto, "Two approaches to event definition," in *13th International Conference on Database and Expert Systems Applications*, vol. 2453 of *Lecture Notes in Computer Science*, (Aix-en-Provence, France), pp. 547–556, Springer-Verlag, 2002.

[235] R. Adaikkalavan and S. Chakravarthy, "Formalization and detection of events using interval-based semantics," in *Eleventh International Conference on Management of Data* (J. R. Haritsa and T. M. Vijayaraman, eds.), Advances in Data Management 2005, (Goa, India), pp. 58–69, Computer Society of India, 2005.

[236] C. Liebig, C. Liebig, M. Cilia, and A. Buchmann, "Event composition in time-dependent distributed systems," in *Cooperative Information Systems, 1999. CoopIS '99. Proceedings. 1999 IFCIS International Conference on* (M. Cilia, ed.), pp. 70–78, 1999.

[237] A. Mok, C.-G. Lee, H. Woo, and P. Konana, "The monitoring of timing constraints on time intervals," in *Real-Time Systems Symposium, 2002. RTSS 2002. 23rd IEEE*, pp. 191–200, 2002.

[238] B. Morin and H. Debar, *Correlation of Intrusion Symptoms: An Application of Chronicles*. Lecture Notes in Computer Science, 2820 ed., 2003.

[239] F. Heintz, "Chronicle recognition in the WITAS UAV project a preliminary report," in *Swedish AI Society Workshop (SAIS2001)*, (Skovde, Sweden), 2001.

[240] N. Ramaux, D. Fontaine, and M. Dojat, "Temporal scenario recognition for intelligent patient monitoring," in *6th Conference on Artificial Intelligence in Medicine, (AIME97)*, vol. 1211 of *Lecture Notes in Computer Science*, pp. 331–342, 1997.

[241] R. Milne, C. Nicol, M. Ghallab, L. Trave-Massuyes, K. Bousson, C. Dousson, J. Quevedo, J. Aguilar, and A. Guasch, "Tiger: real-time situation assessment of dynamic systems," *Intelligent Systems Engineering*, vol. 3, no. 3, pp. 103–124, 1994.

[242] C. Dousson, "Alarm driven supervision for telecommunication networks: Online chronicle recognition," *Annals of Telecommunications*, vol. 51, no. 9-10, pp. 501–508, 1996.

[243] P. Laborie and J.-P. Krivine, "Automatic generation of chronicles and its application to alarm processing in power distribution systems," in *International Workshop on Principles of Diagnosis (DX 97)*, (Mont St Michel, France), pp. 61–68, 1997.

[244] C. McGregor and M. Stacey, "High frequency distributed data stream event correlation to improve neonatal clinical management," in *2007 inaugural international conference on Distributed Event-based Systems (DEBS07)*, (Toronto, Ontario, Canada), pp. 146–151, ACM Press, 2007.

[245] E. Keravnou, "Engineering time in medical knowledge-based systems through time-axes and time-objects," in *Temporal Representation and Reasoning, 1996. (TIME '96), Proceedings., Third International Workshop on*, pp. 160–167, 1996.

[246] G. Wuu and U. Dayal, "A uniform model from temporal and versioned object-oriented databases," in *Eighth International Conference on Data Engineering*, (Tempe, AZ, USA), pp. 584–593, IEEE Computer Society, 1992.

[247] E. Friedman-Hill, *Jess in Action*. Greenwich, CT, USA: Manning Publications, 2003.

[248] C. S. Forgy, "Rete: A fast algorithm for the many pattern/many object pattern match problem," *Artificial Intelligence*, vol. 19, no. 1, pp. 17–37, 1982.

[249] B. S. Martins, S. Yuval, G.-B. Dina, G. Maya, K. Herbert, V. B. Lawrence, M. Deborah, and K. G. Mary, "Evaluation of an architecture for intelligent query and exploration of time-oriented clinical data," *Artificial Intelligence in Medicine*, vol. 43, no. 1, pp. 17–34, 2008.