An experimental study and evaluation of a new architecture for clinical decision support - integrating the openEHR specifications for the Electronic Health Record with Bayesian Networks

Sevket Seref Arikan

Thesis submitted in accordance with the requirements of the University of London for the degree of Doctor of Philosophy

University College London

March 2016

I, Sevket Seref Arikan, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Dedication

This Thesis is dedicated to my wife Zeynep.
It was your love and support that made this work possible. I could not possibly thank you enough. You truly are my soulmate.
I love you.

Abstract

Healthcare informatics still lacks wide-scale adoption of intelligent decision support methods, despite continuous increases in computing power and methodological advances in scalable computation and machine learning, over recent decades. The potential has long been recognised, as evidenced in the literature of the domain, which is extensively reviewed.

The thesis identifies and explores key barriers to adoption of clinical decision support, through computational experiments encompassing a number of technical platforms. Building on previous research, it implements and tests a novel platform architecture capable of processing and reasoning with clinical data. The key components of this platform are the now widely implemented openEHR electronic health record specifications and Bayesian Belief Networks.

Substantial software implementations are used to explore the integration of these components, guided and supplemented by input from clinician experts and using clinical data models derived in hospital settings at Moorfields Eye Hospital. Data quality and quantity issues are highlighted. Insights thus gained are used to design and build a novel graph-based representation and processing model for the clinical data, based on the openEHR specifications. The approach can be implemented using diverse modern database and platform technologies.

Computational experiments with the platform, using data from two clinical domains – a preliminary study with published thyroid metabolism data and a substantial study of cataract surgery – explore fundamental barriers that must be overcome in intelligent healthcare systems developments for clinical settings. These have often been neglected, or misunderstood as implementation procedures of secondary importance. The results confirm that the methods developed have the potential to overcome a number of these barriers.

The findings lead to proposals for improvements to the openEHR specifications, in the context of machine learning applications, and in particular for integrating them with Bayesian Networks. The thesis concludes with a roadmap for future research, building on progress and findings to date.

Acknowledgements

First and foremost, I'd like to express my immense gratitude to my advisor, Professor David Ingram. He has been an amazing mentor, a wise teacher and a great friend during the writing of this thesis, and guided me with infinite patience so that I can understand what I'm really trying to do. I sincerely hope that I will have the privilege of having his company in the future.

Thomas Beale and Dr Sam Heard have both been as kind and generous as a friend can be, never leaving a question unanswered, always offering help, and sharing all of their experience and wisdom to help me better understand openEHR. Their friendship has been as much appreciated as their knowledge, if not more.

The staff and students at CHIME could not have been more helpful. I would like to thank my second supervisor, Professor Dipak Kalra for his insightful comments and questions, which helped me set the boundaries of my study. I am also thankful for Dr Matthew Darlison's continuous support, and his ability to point at the solutions which I could not see.

I would like to thank both Hugh and Heather Leslie, Heath Frankel and the rest of Ocean Informatics for being the great colleagues and friends they are and always supporting my efforts for the PhD, even when they had to carry huge loads on their shoulders.

Dr. Ian McNicoll of freshEHR and Mr. Bill Aylward of Moorfields Eye Hospital have both helped me a lot. Dr McNicoll's clinical modelling help and Mr. Aylward's help with the concepts of ophthalmology made the experimental approach of this thesis possible. Dr Tony Shannon's feedback and input regarding openEHR models have been very useful for getting Opereffa implementation started.

I've had the great fortune of having a family that never stopped supporting me, from the moment they heard I'd be moving to the UK, until this very moment. My parents, Neziha and Yilmaz Arikan, along with my aunts Guler and Sukran Arikan went above and beyond to help me get a good education. I hope this thesis makes them proud. I am sorry that Birtanem could not see this thesis finished, but I'd like to think she'd be proud too.

I am grateful to my brother Onur Arikan for making sure that our parents were never left alone. Ergin Yilmaz helped me so much to make sure that I can chase my dream. Both Necati and Gulsum Kurt saw me as their own son and treated as such, for which I'd like to extend my most sincere thanks, love and respect to them.

Contents

Tables

Figures

Equations

List of abbreviations

Abbreviation	Meaning	Page
ADL	Archetype Definition Language	52
AI	Artificial Intelligence	20
AM	Archetype Model	58
AMD	Age Related Macular Degeneration	208
AQL	Archetype Query Language	21
BMI	Body Mass Index	143
BN	Bayesian Network	23
CCR	Continuity of Care Record	61
CDA	Clinical Document Architecture	61
CDS	Clinical Decision Support	20
CDSS	Clinical Decision Support System	44
CIMI	Clinical Information Modelling Initiative	50
CKM	Clinical Knowledge Manager	204
CLIPS	C language integrated Production System	51
CSER	Clinical Sequencing Explanatory Research	50
CTE	Common Table Expression	173
CTS	Clinical Trial Simulation	255
DAG	Directed Acyclic Graph	75
DOM	Document Object Model	126

DSTF	Draft Standard for Trial Use	62
EAV	Entity Attribute Value	158
EAV/CR	Entity Attribute Value/Classes and Relationships	159
EHR	Electronic Health Record	21
EM	Expectation Maximisation	102
EMF	Eclipse Modelling Framework	167
FHIR	Fast Health Interoperability Resources	62
FTI	Free T4 Index	98
GEHR	Good European Health Record	52
GEM	Guidelines Element Model	46
GENIE	Graphical Network Interface	79
GDL	Guideline Definition Language	50
GLIF	Guideline Interchange Format	92
GLIF3	Guideline Interchange Format 3	46
GTPQ	Generalized Tree Pattern Query	153
HIS	Hospital Information System	44
HL7	Health Level Seven	49
HL7 RIM	Health Level Seven Reference Information Model	66
HTML	HyperText Markup Language	128
ICD	International Classification of Diseases	42
ICU	Intensive Care Unit	62

IHTSDO	International Health Terminology Standards Development Organisation	42
Infoset	XML Information Set	86
ITS	Implementation Technology Specification	56
JAGS	Just Another Gibbs Sampler	71
JSON	Javascript Object Notation	187
MCMC	Markov Chain Monte Carlo	71
MLM	Medical Logic Module	46
OWL	Web Ontology Language	87
PC	stands for Peter Spirtes and Clark Glymour, authors of the algorithm	98
PCR	Posterior Capsular Rupture	196
PGM	Probabilistic Graphical Model	70
POC	Proof Of Concept	167
PTPQ	Partial Tree Pattern Query	153
RDBMS	Relational Database Management System	152
RIM	Reference Information Model	61
RM	(openEHR) Reference Information Model	52
ROC	Receiver Operator Characteristics	29
SM	Service Model	58
SMILE	Structural Modelling, Inference and Learning Engine	87
SNOMED CT	Systematised nomenclature of medicine clinical terms	42
SQL	Structural Query Language	34

T3	Triiodothyronine	98
T4	Thyroxine	98
T4U	Thyroxine resin uptake	98
TDD	Template Data Document	214
TDS	Template Data Schema	215
TPQ	Tree Pattern Query	221
TRIM	Templated RIM	66
URI	Uniform Resource Identifier	28
VL	Vitreous Loss	196
WTP	Web Tools Platform	215
XDM	XPath 2.0 Data Model	126
XML	Extensible Markup Language	22
XPath	XML Path Language	152
XQuery	XML Query	126
XSD	XML Schema Document	126
XSLT	XSL Transformations	129

Chapter 1: Introduction

This introductory chapter presents the problem that the thesis attempts to solve, the objectives set for providing solutions, the methods employed and the research contributions made, followed by a description of the structure of the thesis.

1.1 Research Context and Motivation

The use of computers to help clinicians in their decision-making, referred to as Clinical Decision Support (CDS) in this thesis, is a long-standing and active field of research. Integrating the decision-making capabilities of computers with the practice of medicine presents numerous challenges (Clancey and Shortliffe 1984), (Robert A. Greenes 2014) and these challenges have been a significant area of focus for artificial intelligence (AI) research, long before the use of computers became prevalent in other fields of daily life (Ledley and Lusted 1959b), (De Dombal et al. 1972), (Leaper et al. 1972) (Edward H. Shortliffe et al. 1975).

The complex and multi-dimensional nature of clinical decision-making requires a multi-disciplinary view of the processes involved, in order to improve the outcomes. The existence of a large body of research on understanding how clinicians' reasoning works (Ledley and Lusted 1959a), how expert knowledge and clinical data can be transformed into a computable form (Markwell, Sato, and Cheetham 2008), (Aikins 1980), (David Ingram 2002) , how they can be shared (M. A Musen 1992), (Beeler 1998), mathematically processed (Spiegelhalter and Knill-Jones 1984) and represented (Luciani and Stefanini 2012), shows that improvements in CDS depend on a combination of contributions from many different fields of research. Knowledge engineering, statistical modelling, artificial intelligence, information systems design and implementation and large scale data processing are all relevant in the development of better CDS, encompassing a vast intersection of domains of scientific research.

Even where successful outcomes have been achieved in the integration between components of this multi-disciplinary field of research, and these have been adopted within experimental innovations in the practice of medicine (Robert A. Greenes 2014), the widescale use of CDS is still elusive, despite the increase in processing power and the emergence of large scale data processing architectures and frameworks. Individual implementations can benefit from developments in science and engineering but the CDS demonstrated thereby is, typically, still

localised, case-specific and isolated, in general. The difficulty of integrating clinical data that originates from multiple information systems contributes significantly to this situation.

The reasons for this less than expected level of adoption of CDS are not purely based on problems with technology. A significant part of the problem lies in the difficulty of making the increases in computing capability available to clinicians in ways that enable them to integrate that capability with care processes. The difficulty of expressing clinical knowledge in the form of mathematical concepts such as probability, makes it hard for clinicians to use CDS approaches that require communication based on this language (Leaper et al. 1972) (R. A Greenes 2007). Such difficulties have led to an increasing divide between what is computationally possible, such as the use of graphical probabilistic models, and what is actually usable in the CDS implementations, since statistical methods for CDS depend on or benefit from availability of more data and greater computing power.

The emergence of new electronic healthcare record standards that are based on information models, primarily in response to requirements for data integration between different clinical information systems, presents an opportunity to overcome some of the most significant problems of CDS adoption. The openEHR electronic health record (EHR) specifications (Beale et al. 2006) provide a capable, flexible and mature representative of these standardisation efforts. openEHR's scope goes beyond the integration of health data across systems. It provides a comprehensive domain model and domain specific languages and tools that allow clinicians to express clinical concepts using this model. This strongly clinician-driven approach to defining clinical data allows complete information system implementations based on the domain model, in a technology agnostic way. openEHR also provides a query language called the Archetype Query Language (AQL) (Ma, Frankel, and Beale 2014) that allows querying of clinical data based on the domain model of openEHR.

The combination of openEHR's clinician-driven approach to defining clinical data, its support for a high level domain specific query language and its technology agnostic nature, makes possible a health computing platform that can support both clinical information systems development and CDS functionality. The use of such a platform provides an inherent solution to data integration issues, but its real, currently underutilised potential lies in the use of clinician input to build CDS systems, allowing clinicians to take control of functionality that is normally isolated from them by non-clinical, hard to grasp concepts, such as probability calculus.

A particular CDS approach based on the use of Bayesian Networks (BNs) (Koller and Friedman 2009) offers advantages similar to those offered by openEHR, in terms of letting domain experts define domain concepts without having to tackle complex implementation details. Bayesian Networks provide clinicians with a user interface that lets them quickly see an overall picture of clinical variables relevant to a patient's condition. This user interface hides the complexity of processing of probability concepts, while still providing the advantages of probabilistic reasoning in decision-making, thus offering a solution to the problem of integrating powerful statistical methods with clinical care for better CDS.

Despite its advantages and the increasing worldwide adoption of the openEHR methodology for EHR implementation, there is currently no systematic integration of openEHR methodology with probabilistic inference methods that are used highly effectively in other domains. This presents an opportunity to build a comprehensive health computing platform that includes decision support as a first class functionality. Therefore, based on the conceptual similarity between openEHR and BNs in their support for efficient representation of domain concepts, the fundamental research question this thesis seeks to answer is:

Can openEHR support clinical decision methods based on Bayesian Networks, by providing a model driven health computing platform that supports clinical data interoperability, clinical information systems development and machine learning functionality, and what, if any changes are required to the openEHR specifications to achieve this?

In its attempt to answer this question, the thesis uses the openEHR specifications as a basis for implementing several large-scale and innovative software frameworks, to make possible a hands-on and experimental approach to the testing of the openEHR specifications and methodology, in the context of CDS. Previously released open source libraries for openEHR, freely available clinical modelling tools and clinical models, and a number of open source libraries for software development and machine learning are used throughout the development of these experimental frameworks. The thesis uses research on XML data representation and XML persistence in relational databases, along with research on representation of clinical data using both relational and non-relational persistence systems, to deliver its research outcomes.

A logical breakdown of the fundamental research question provided above leads to the following specific research motivations:

- *Evaluation of the clinical modelling capabilities provided by openEHR in a CDS setting in which clinical models are used to define CDS related concepts.*

  It is frequently conjectured that EHR driven approaches can deliver better health IT and can be used to implement CDS. A fundamental assumption that must be proven true, for this conjecture to be true in the context of openEHR based CDS, is that the openEHR methodology and its implementations can support a Bayesian Network (BN) based decision-making mechanism. This assumption implies that openEHR's clinical modelling capabilities, the scope of concepts covered by these capabilities and the expressiveness of openEHR-based clinical models, can support definition of CDS concepts. Since CDS concepts are related to but not necessarily the same as the clinical concepts, the extent to which openEHR methodology can support representation and computation of both EHR and CDS concepts must be explored.

- *Analysis of the feasibility and characteristics of a software architecture for CDS based on openEHR and BNs.*

  openEHR's technology agnostic nature means that it has no dependence on any particular programming language or platform. Even though this independence is an advantage that allows openEHR to be implemented with any platform of choice, it usually establishes the implementation as a technology specific task with very little if any focus on robust, reusable generic software architectures for support of the implementation process, per se. The introduction of CDS functionality increases the complexity of an openEHR implementation even further, since establishing the links between clinical data and CDS mechanisms is also a platform specific requirement and task.

  The openEHR specifications cannot themselves include suggestions for software architecture or CDS implementation, due to the vast range of available options, but this does not mean that a high level, yet highly adoptable, generic architectural approach to implementation cannot be identified. To our UCL team's knowledge, there has currently been no research effort in this direction.

- *Analysis of the suitability of BNs as a decision-making engine for openEHR based CDS.*
  BNs are already used in many clinical decision making scenarios, but their use in a context in which data is defined, persisted and queried using openEHR constitutes a highly specific setting. Various factors come into play in this setting, such as: the use of openEHR data types instead of arbitrary methods for representing data; the use of AQL as the means of data access, which may or may not introduce issues of expressiveness based on AQL's features and the structure of the openEHR clinical models that are used in AQL queries.
  Existing uses of BNs in non-openEHR based clinical settings support their usefulness for CDS, but offer no helpful information for the specific setting this thesis sets out to evaluate. Therefore, the efficacy of BNs in an openEHR-based approach to CDS is an open research question.

- *Identification of any revealed shortcomings of openEHR in delivery of a generic CDS platform, along with potential approaches to overcoming these shortcomings.*
  The motivations of this thesis are not limited solely to analysis of key aspects of openEHR and BN integration, or to the identification of shortcomings revealed in such an integration. The work aims also to lead to proposed changes in the specifications, based on research outcomes achieved in the thesis, to contribute to improvement of the openEHR methodology in the context of CDS implementation. This objective has hitherto received rather little attention, compared with that devoted to clinical information systems implementation.

1.2 Research Scope and Objectives

The scope and research motivations of the thesis embody the conjunction of a number of individually vast research topics. This places feasibility limits on an implementation experiment driven approach. Therefore, the thesis scope and experiments undertaken are defined and constrained by the most fundamental and relevant elements of openEHR, BNs and software architecture, as follows:

- The openEHR clinical models and the use of openEHR methodology in general, throughout the thesis, focus on an information model approach to clinical data representation. openEHR supports the use of clinical

terminologies and they are included in the discussion at various points, where relevant. But these terminologies were not used in building the underlying clinical models for the experiments described, primarily due to time constraints. They offer significant benefits and would certainly be included in future work.

- The use of BNs is limited to discrete networks - that is, networks with discrete random variables only. BNs belong to the family of graphical models, as explained in Chapter 4, and other types of graphical models are excluded from the experiments. The number of available tools and frameworks that support discrete networks has been a key factor in this decision: significant implementation effort would be required to make use of other types of BNs, for which there is far less tooling available.
- The persistence abstraction for openEHR developed in the thesis is applicable to a number of persistence systems. The thesis provides an implementation on top of a relational database and other suggested implementations are left for future research.
- Access to clinical data to be used in the experiments is a major limiting factor in the scope of the thesis. The thesis uses both real patient data and synthetic data. Anonymised real patient data, publicly available from the UCI machine learning repository (Bache and Lichman 2013), is used for the experiment on detection of thyroid diseases in Chapter 5, as an example of BN based CDS. Attempts to use real patient data for the more substantial experiment described in Chapter 9 has led to multiple problems which reveal a major barrier to be overcome in future research studies similar to the experiments presented in this thesis. Access to existing, high-quality research data is subject to rigorous ethical approval and related rules and regulations. However, these approval processes require information that cannot be provided in advance for some machine learning approaches, such as the list of variables from the data set that will be used in the research. Even though existing regulations and associated processes are in place for good reasons and have been introduced with great care, machine learning use cases, such as identifying BN variables automatically, do not fit well within these existing processes. Moreover, extensive effort to produce a data set from an existing operational clinical legacy system, which had been in routine use at a world leading research centre, has failed due to data quality issues, as discussed in Chapter 9 in Section 9.2. Therefore, synthetic data is

created and used for the experiment discussed in Chapter 9. Availability of large scale clinical data for research could have allowed this thesis to explore a wider set of CDS scenarios. The extensive effort required to build a data set suitable for this final experiment have taken considerable time, which could have been used to expand the thesis scope instead.

The objectives of the thesis are determined with respect to their foundational nature for development of a robust health-computing platform that can support CDS research in as many future directions as possible, within the scope defined above. These objectives are set out as follows, along with the expected contributions that will follow from their achievement:

1) *To test the suitability of both openEHR and BNs for expressing clinical concepts, and computations on these concepts, in a CDS setting.*
   Such an analysis will help in identifying the overlapping and disjoint concepts used in an openEHR based CDS setting. Identification and classification of these concepts will then enable testing of the adequacy of the current scope and expressiveness of openEHR for CDS implementation. **The results of this test will contribute to establishing a baseline for openEHR's support for CDS modelling, thereby providing evidence for openEHR to use in systematically improving its capabilities.**

2) *To define a novel architecture for openEHR implementation which can support both clinical application development and CDS implementation scenarios, across a number of implementation technology options.*
   Clinical modelling itself provides no information about the software implementation required to process the models. An implementation architecture that can be used with multiple software platforms would enable the advantages of different platforms to be exploited in providing information system and CDS functionality. Such an architecture is the key to establishing a robust, experimental platform for future openEHR CDS research. **The definition of this novel architecture will help openEHR overcome the challenge of staying technology agnostic while encouraging and enabling systematic implementation utilising different implementation technologies.**

3) *To inspect the relationship between clinical information system and CDS system implementations based on openEHR*

The most common approach to software architecture when designing CDS systems is to assume that CDS will be developed as a standalone system. Defining a unified architecture for both clinical information system and CDS implementation eliminates many integration problems, but different use cases for clinical data access lead to orthogonal requirements at the software design and implementation level. **Understanding how requirements imposed by clinical information systems functionality and CDS functionality interact with one another in an openEHR implementation context, will offer the opportunity to develop a generic implementation architecture that can be customised and optimised for specific scenarios without losing its unified platform advantages.**

4) *To test the query capabilities and decision making performance of the openEHR and BN components of the new architecture with high volume clinical data.*

The query capabilities and performance of openEHR AQL has to date only been evaluated in the context of clinical information systems implementation. Consequently, the features provided by AQL have hitherto only aimed to address this functionality. Using AQL in a CDS context, in which a large amount of data for many patients must be accessed, quite probably alongside non-clinical care data and controlled by an algorithmic inference mechanism, presents a very different use case than that seen in the clinical information system context. In the clinical information system context, data access is focussed on a single patient, and is therefore rather small in terms of data volume and the reasoning process always has a human actor, namely the clinician. The viability of openEHR as a generic health-computing platform, in the face of requirements to process hugely increasing data volumes, will depend on its capability to enable performant AQL queries at a large scale, with support also for operations on data that are specific to machine learning methods and scenarios. **The achievement of this objective will provide observations and feedback from an actual CDS implementation, which is rarely provided in detail by openEHR implementers. This type of feedback will present an opportunity to develop, for example, a version of AQL supporting probabilistic search**

criteria, in order to support clinical data processing for machine learning implementations.

1.3 Research Methodology

Extensive software implementation, based on real life requirements gathering and literature review, is used as the primary research methodology to achieve the research objectives of the thesis, based on the completion of the following tasks:

1) An extensive preliminary literature review of CDS history and approaches is performed. This literature review shows how CDS approaches have evolved in the last five decades and what the current problems are. The openEHR standard and methodology is classified as an extension and evolution of the data bank approach to CDS identified in the literature review. A discussion of the openEHR specifications and methodology is provided in Chapter 3, which shows how openEHR can connect information models to CDS.
2) A software development framework which uses openEHR models to support automatic, web based user interface generation along with automatic persistence and retrieval of clinical data is developed to serve as a test bed for the clinical information system development aspects of openEHR implementation. The requirements for this framework are gathered via collaboration with a clinician.
3) A literature review is performed for the use of BNs in medicine, in order to determine their fitness for the purposes of providing a generic reasoning mechanism for CDS. This literature review is supplemented by an implementation experiment that uses a BN to diagnose thyroid diseases in a non-openEHR setting, using published, anonymised real patient data.
4) A tree based representation of openEHR data along with a Tree Pattern Query (TPQ) representation of AQL, is developed for persistence abstraction. This builds on a literature review of XML data representation and persistence methods, common approaches to handling clinical data in relational databases, and the findings of the preceding openEHR-based clinical information system implementation experiment.
5) The tree based persistence and query abstraction is implemented on top of a relational database server. This implementation is then populated with

simulated data for half a million patients and integrated with a BN implementation.

6) A real life CDS scenario from the domain of ophthalmology is identified and openEHR clinical models representing the relevant clinical concepts are developed, mainly by clinicians and an expert clinical modeller. These models are extended to address the CDS scenario specific data requirements. An existing implementation of the identified CDS scenario, based on an alternative statistical approach, is used to compare and contrast with the combined openEHR and BN based approach developed in the thesis.
7) Finally, the clinical models, tree based persistence implementation and synthetically generated data are used to estimate the risk for a clinical operation using a BN. Following an extensive real patient data analysis on a legacy system in Moorfields Eye Hospital, synthetic data generation is adopted, due to data quality issues. Risk estimation performance of the BN is measured via use of the Receiver Operator Characteristics (ROC) curve (Bradley 1997).

1.4 Contribution

The experimental results and contributions made to the field are as follows:

1) A detailed analysis of openEHR in a CDS implementation scenario is provided. The analysis builds on openEHR models developed for clinical care records, which are later extended to support CDS implementation. This pushes openEHR methodology beyond its role in supporting interoperability and clinical information system implementations. By targeting such a large scope, the thesis reveals issues which might have been neglected or missed in studies that focus on limited scenarios, in an isolated manner. To our knowledge, this thesis is the first study that has targeted such a comprehensive analysis of openEHR implementation challenges.
2) The orthogonality of the implementation challenges posed by two openEHR-based scenarios - clinical information systems implementation and CDS implementation - is demonstrated in the implementation of the Opereffa framework. This experiment clearly shows that design choices related to persistence implementation can significantly limit the use of openEHR for machine learning and population scale data analysis. This is a key finding for implementers and could not have been straightforwardly deduced from the

openEHR specifications alone. This finding also shows that openEHR adoption is potentially vulnerable to implementation challenges that are not currently addressed by the openEHR specifications.

3) The development of the new persistence abstraction for openEHR, presented in the thesis, based on tree data structure and TPQs , provides a solution to the challenge of keeping openEHR platform technology independent, without introducing a steep learning and implementation curve for the implementation of fundamental required functionality, such as data persistence, for each and every implementation platform. The implementation of the persistence abstraction, on top of a relational database, shows that this original approach, as described in Chapter 8, can be used with a mainstream persistence option. The use of published research on XML data representation and persistence, along with published research on managing clinical data, establishes a new link between openEHR implementation and research outcomes from both computer science and information retrieval. This is an original contribution to a key challenge influencing the wider adoption of openEHR adoption, which has hitherto been treated as case-specific software development activity. It paves the way for future research on EHR persistence, in collaboration with rich and alive research activity on graph processing, especially at large-scale.
4) The use of openEHR clinical models for CDS identifies shortcomings of current openEHR methodology, as revealed in the case of a specific machine learning scenario. The clear focus of openEHR models on clinical data leaves administrative and demographic data out of the models, which are focused solely on particular clinical concepts. Even though this approach is soundly justified in the openEHR specifications, it nonetheless leaves significant CDS variables, such as the professional experience of the clinician performing an operation or the age of the patient, outside the scope of the clinical models, although these variables are required for a CDS implementation. The thesis suggests extensions to the openEHR specifications to allow inclusion of relevant but non-clinical data in openEHR models as metadata. Requirements to manage this metadata are also recognized, in scenarios such data exchange between systems or updates to existing clinical data. The use of openEHR models in the experimental CDS of Chapter 9 also shows that the use of standardised clinical terminologies within the information models is critical for ensuring reliable

automated processing of the described clinical data. The examples reported show that, even though a human end-user of an information system can understand and process textual information, failure to use standardised terminologies to represent the associated clinical semantics can introduce potential ambiguity in the implementation of machine learning use-cases, and that the openEHR clinical models are not automatically immune to this kind of problem.

5) The use of AQL for large-scale queries shows that AQL can successfully express a subset of clinical data that spans multiple clinical models, to provide data input to a BN implementation. This finding strengthens the suggestion that AQL can be used for querying requirements beyond those of patient-centred clinical information systems. The thesis proposes extensions to the AQL specification, based on the data processing requirements exemplified by the experimental BN integration with openEHR. These extensions can be implemented in the form of function call support within AQL queries, allowing AQL support to be integrated with BNs and other machine learning methods for CDS, with markedly less effort than would otherwise be required.
6) The thesis implements BN based decision making for two different settings. By providing both non-openEHR and fully openEHR based scenarios, the thesis identifies issues with BNs that are independent of the use of openEHR, such as problems associated with lack of observations for combinations of values of clinical variables. The openEHR based BN implementation also shows that integrating machine learning frameworks with openEHR implementation requires consideration of scalability, and consequently parallelisation, of associated data processing. This finding provides a strong incentive to focus in the future on large scale, parallel processing frameworks for openEHR implementation.
7) Overall, the combination of the contributions described delivers the definition and experimental validation of a flexible software architecture that can potentially support orthogonal software architecture requirements for clinical data processing in CIS and CDS systems, across multiple levels of data scale, which are currently served via separate, dedicated solutions.

1.5 Thesis structure

The thesis consists of a further 9 chapters, followed by an appendix. The contents of these are structured as follows:

Chapter 2 *Clinical Decision Support and Clinical Information Systems*.

Begins with the history of CDS and its key concepts, followed by a definition of a key problem in CDS, identified by the thesis and named "The Detachment Problem". This view of barriers to wide-scale adoption of CDS makes EHR concepts a promising candidate basis for a computable health platform. The recent literature reviewed in this chapter shows that, despite advances in computing power and CDS capabilities, the fundamental problems identified almost five decades ago still exist. The review also shows that there is a convergence towards the use of information models for CDS implementations in the health informatics domain.

Chapter 3 *The openEHR Specifications and Their Relationship to Clinical Decision Support.*

Provides background relevant to this thesis about the openEHR standard and the methodology implied by its specifications and tools. A discussion of openEHR's features is provided, which shows how openEHR can be used for an information model based approach to CDS. These arguments underlie the suggested use of openEHR methodology as a common platform for interoperability, clinical information system and CDS development. However, this platform is only half of the solution for a generic CDS solution that can be reused across different clinical domains. The second half: BNs, is introduced in the following chapter.

Chapter 4 *Bayesian Networks for Clinical Decision Support and Their Integration with openEHR*

BNs are introduced as a generic decision making mechanism which is a member of a family of probabilistic reasoning methods called graphical models. The use of BNs in medicine is analysed via literature review, performed to answer specific questions regarding the feasibility of using BNs as a generic, robust CDS mechanism. The review shows that there is sufficient evidence to support the suggested use of BNs for the purposes of this thesis. Therefore, an integration architecture combining openEHR methodology and BN concepts is developed. This integration architecture completes the high level description of the openEHR based CDS platform idea introduced in the summary section of Chapter 3. Experiments

based on software implementation of both components of this unified architecture are described in chapters 5 and 6.

Chapter 5 A Pilot Bayesian Network Implementation Experiment Using Thyroid Disease Data

An existing BN framework is used to diagnose thyroid diseases, based on publicly available real patient data. This experiment aims to explore the use of BNs as a CDS mechanism and is intentionally performed in a non-openEHR setting. This approach makes it possible to observe if and how the use of BNs in an openEHR context is different than the well-established approach of exporting clinical data to a simple format, such as comma separated values, for non-clinical uses. The experimental results show that access to larger volume of data improves the outcomes for key steps in BN development, such as defining the network structure. However, in integrating data from different sources, the existence of outliers in the data and missing observations for combinations of clinical variables, are challenges that must be dealt with. The experiment also shows that using existing tools and frameworks for implementing data processing and BN functionality is not an option, but a necessity, due to the infeasibility, due to time and resource constraints, of developing this functionality from scratch. Therefore, the capability of a health computing platform to be accessible to existing tools is found to be a critical factor for its success.

Chapter 6 *A Pilot openEHR Based Clinical Information System Implementation Experiment – The Opereffa Open Source Framework*

Discusses the development of a clinical information system implementation framework based on openEHR. This substantial implementation, which is available as open source software, shows that openEHR is capable of supporting a generic platform approach for clinical application development, but that this capability does not fluently extend to supporting CDS functionality, especially if AQL is not considered as a first class design requirement. The findings of this chapter, along with the non-openEHR BN experiment in Chapter 5 underlie the importance attached to the development of the persistence abstraction in Chapter 7.

Chapter 7 *Persistence Abstraction for openEHR*

An abstract representation of openEHR data and query semantics of AQL is developed and implemented. This representation, named XINO, solves the problem

of having to develop data representation and query processing mechanisms for each implementation platform. The representations for data and query semantics is built on the large amount of research on XML processing and persistence, with a focus on the requirements of openEHR data and AQL. An extension of the Tree Pattern Query (Lakshmanan, Wang, and Zhao 2006) representation of AQL is developed. An essential goal of this approach is to allow openEHR to be efficiently implemented on top of different persistence systems and thereby to use their specific features and advantages. In order to prove the achievability of this goal with the abstract representation at hand, a proof of concept implementation is developed on top of the open source relational database Postgresql (Momjian 2001). This implementation, details of which are discussed in Chapter 8, is then used as the data source for a comprehensive CDS implementation in the domain of ophthalmology in Chapter 9.

Chapter 8 *XINO Architecture for Persistence*

The persistence abstraction developed in Chapter 7 is implemented using the open source relational database Postgresql (Momjian 2001). This implementation, named XINO-P, proves that the platform agnostic XINO architecture can be mapped to widely used relational databases, via generation of SQL based on the TPQs. Recognising the well-known challenges of representing hierarchical structures in relational form (Celko 2012), the mappings developed in this chapter and their use in Chapter 9 to serve clinical data to a BN implementation, provide crucial proof of the feasibility of XINO. Moreover, this chapter shows how XINO's tree based approach can be mapped to the native capabilities of a persistence system, which opens the door to many persistence implementations of openEHR on different persistence systems. Chapter 7 and 8 therefore present the definition and implementation of a generic persistence framework for openEHR, filling a critical current gap in feasible technical pathways to wider openEHR adoption.

Chapter 9 *An Experimental openEHR Based Clinical Decision Support Implementation for Ophthalmology: Risk Estimation for Cataract Operations*

An end to end implementation of a CDS based on XINO and BNs is developed, for estimating the risk in cataract operations. The CDS scenario is based on an existing published clinical research study, in order to benefit from its published design as well as provide a rationale for comparison of an openEHR/BN

CDS approach with an established alternative statistical method that relies on data extraction and logistic regression analysis(Narendran et al. 2008) .

The XINO implementation is driven by openEHR models, which were initially developed in the Moorfields Eye Hospital and later extended for the purposes of the CDS implementation described here. The results of the analysis of clinical ophthalmology data kept by Moorfields Eye Hospital in its existing clinical system, show that the quality and amount of existing data is insufficient to implement the full scope of the experiment. Synthetic data generation is therefore used to simulate realistic datasets for half a million patient operations, which are then persisted to the XINO framework. This data is then consumed by an existing open source BN implementation and the decision making performance of this combined architecture is evaluated through multiple, computationally intensive iterations to produce ROC curves (Bradley 1997). The elements used in the BN consist of data items from the openEHR models with data access defined by AQL queries.

This extensive implementation uses openEHR models to drive every aspect of a CDS scenario and provides valuable insight into the issues and opportunities arising therefrom, such as extending AQL to establish a generic probabilistic query capability, named in the thesis as Probabilistic AQL. Other findings include, but are not limited to, the confirmation of issues relating to missing observations, as identified in Chapter 5, the effects of modelling clinical data with openEHR data types, and the limitations of BN inference achievable, exacerbated by the lack of parallelisation of computations.

Chapter 10 *Conclusions and Future Research*

Findings from Chapters 7, 8, and 9 are discussed and extended to an overview of future research potential. The chapter concludes that currently emerging big-data ecosystems provide an exciting opportunity for openEHR methodology for CDS implementation. The most significant advantage of these eco-systems is their ability to integrate machine learning and data persistence in a scalable manner. This unified approach to processing large scale data allows parallelization of data processing, which has been found to be a bottleneck in the XINO and BN integration scenario discussed in Chapter 9. The thesis concludes with the view that, despite shortcomings identified, that must be addressed mostly through extensions to the current specifications, the openEHR standard provides a capable platform for health computing and CDS implementation, with the possibility of exciting future research.

Appendix I *Synthetic Data Generation*

The method used to generate synthetic patient data for cataract operations is explained.

1.6 Summary

The thesis provides an analysis of the published openEHR standard for the EHR, as the basis of a health computing platform that support data interoperability, clinical information systems development and CDS implementation. The motivation for the research described stems from the current lack of published research on the use of openEHR as the basis for a generic clinical data platform, despite its mature and flexible design and widespread adoption, internationally.

The research objectives are achieved by means of experiments based on large scale software implementations and by literature review. The results provide insights about the openEHR standard and its key use cases, which will be of wide interest to implementers and researchers. The results also pave the way for a number of new research directions for openEHR, most notably for extending AQL and using big-data frameworks.

Chapter 2: Clinical Decision Support and Clinical Information Systems

This chapter provides an overview and discussion of relevant research, covering a time span from the origins of the field in the 1960s to the present day. At the outset of the PhD project in 2008, a literature review was conducted, which clarified both motivation and rationale for the research to be conducted. The scope of this review is the intersection of artificial intelligence (AI), information systems and CDS. CDS is here positioned as a bridge between the two other domains. This historical review was subsequently supplemented, and described in a separate section devoted to literature published during the period of the PhD work, from 2008-2015. This shows that, despite new approaches, based in the main on the use of information models to tackle recognized historic challenges of CDS, some long-identified fundamental problems remain unchanged. The chapter makes extensive reference to other related research domains, and their evolution to the present day.

2.1: History and Key Concepts

The use of computers to help clinicians in decision-making is a long-standing and active field of research. (Clancey and Shortliffe 1984) provides some of the key aspects of using artificial intelligence for decision-making in medicine (p. 1-17).

This work is significant, since it was produced at a time where a paradigm shift in AI was happening. It reports at a convenient point in time where some of the key paradigms have matured enough to be thoroughly evaluated, and successor paradigms are emerging to provide potential solutions to problem issues introduced by their predecessors. Therefore, (Clancey and Shortliffe 1984) is a convenient anchor for observing the evolution of the field, and it will be referred to often in this chapter.

An important term from (Clancey and Shortliffe 1984) is *knowledge base*, which identifies a key component of software-based approaches to clinical problem solving in medicine. Currently this term does not have a strict meaning. It may apply to an electronic repository of patient information taken as a stored form of knowledge or to a set of rules for determining the appropriate action in an information system. It may even apply to a set of mathematical definitions.

This thesis considers knowledge base as any coherent form of computable knowledge representation that encapsulates statements about the domain concepts. The existence of a knowledge base in an information system implies that

domain concepts or rules have been identified and separated from the rest of the software. This component usually has a key role in the way a system behaves in response to some input. Because of its theoretical and engineering advantages, a knowledge base has been adopted as a central component in many systems, across the clinical domains that are within the scope of this thesis. The idea of defining knowledge in a formal, processable and flexible form has strong similarities to some of the current mainstream software development methodologies such as model driven architectures. The knowledge base can also act as a unifying concept across research domains within the scope of this thesis. Therefore, the following discussion adopts a knowledge base centric view of the CDS domain that will be extended to other relevant domains as required.

(Clancey and Shortliffe 1984) provides a set of different dimensions on which to compare knowledge based systems (p 11). Upon classifying CDS implementations as knowledge based systems, a subset of these dimensions provides convenient criteria for discussion of the different paradigms that have hitherto been adopted in decision support systems. This subset consists of: (1)content, (2)hypothesis formation and evaluation, (3)management of uncertainty, (4)data collection and explanation and (5)knowledge acquisition. These high-level concepts are still relevant today in the context of CDS design and implementation.

Similarly, the methods for CDS identified in (E. H Shortliffe, Buchanan, and Feigenbaum 1979) are mostly still relevant, although they have evolved. Especially methods based on accumulating data in a computer processable form has become comprehensive enough to encompass others by providing a platform, on top of which other functionality can be built. The five paradigms given below are mainly based on the classification of (E. H Shortliffe, Buchanan, and Feigenbaum 1979).

Symbolic reasoning

Symbolic reasoning has been a widely studied field of artificial intelligence since at least the 1970s. This paradigm attempted to mimic the reasoning process of human beings, by employing methods such as rules representing expert knowledge. The knowledge base (Clancey and Shortliffe 1984) corresponds to a combination of these rules and related domain concepts within this symbolic reasoning paradigm.

The advantage of this approach is that the content of the knowledge base is expressed in a form that is meaningful to human beings. (E. H Shortliffe, Buchanan,

and Feigenbaum 1979) focuses on MYCIN (Shortliffe, 1976), which is a representative of the symbolic reasoning approach.

MYCIN chose to use formal rules to represent and evaluate expert knowledge. Since rules are easy for experts to understand and communicate, this approach contributes to achieving better intelligibility. By using such rules for its knowledge base, MYCIN is able to provide data collection and explanation of decisions in a form that is relatively easily understood by clinicians.

A decision support system is capable of performing reasoning on its knowledge base, but the reasoning capability on its own does not guarantee successful outcomes. The rules are used to process the content stored in the knowledge base, so the quality of that content is critical. More efficient communication with physicians during construction of the rules within the knowledge base improves the overall success of the system. Reasoning is performed by way of execution of rules, and when these rules are in a format that is understandable by humans, following the process is also easier. This helps the clinicians make better use of the outputs from the system, since they can see the reasoning behind the decisions made. (Edward H. Shortliffe et al. 1975) explains how this explanation mechanism works in MYCIN. Even though rules based representation provides advantages in CDS, rules are not sufficient as the sole basis for symbolic reasoning. What is inherent in the clinical decision-making process is the uncertainty. The clinician deals with uncertainty using evidence and his observations, and his actions follow accordingly. Rules may encode these actions, but the decision mechanics for activation of rules has to handle uncertainty. MYCIN employs *certainty factors* for this purpose. (E. H Shortliffe and Buchanan 1975) says that the adoption of the certainty factors in MYCIN was introduced as an approximation to conditional probabilities, referring to issues associated with the use of statistical methods such as Bayesian methods.

This approach is not free of problems. (Duda and Shortliffe 1983) recognizes the problem with the semantics of the values of this approximation in the clinical context; the certainty factors are not probabilities, hence their meaning is not as clear as the rules which they guide. The relationship between probabilities and certainty factors has been explored by (Barclay Adams 1976), which points to potential issues that may arise with wider use of the certainty factors approach, even though the approach performs well in MYCIN. In summary, symbolic reasoning provides convenient methods for knowledge engineering and explanation of decisions based on rules, but handling uncertainty is not easy compared to statistical methods such as the Bayesian approach.

Bayesian Methods

(E. H Shortliffe, Buchanan, and Feigenbaum 1979) considers Bayesian statistical approaches as one of the major paradigms of Clinical Decision Support. Impressive diagnostic accuracy is not uncommon in systems where Bayesian methods of inference are used, and this is a point in favour of the approach (De Dombal et al. 1972).

However, diagnostic accuracy is not the only criterion for successful CDS. The Bayesian method requires inputs in the form of probabilities. This requirement introduces significant challenges for knowledge engineering as providing domain knowledge in the form of probabilities has not proved a familiar and convenient method for clinicians. Compared to expressing knowledge in the form of rules or using explanations based on their execution, conditional probabilities are significantly less convenient as a domain language for clinical decision support. With a statistical approach, the quantitative expression of probabilities turns into a language embracing data collection, knowledge acquisition, and even explanation of the reasoning, as well.

According to (Leaper et al. 1972) which discusses the problems associated with the use of probabilities for knowledge engineering, two applications of the Bayesian approach on the same set of data, using different sources for probabilities, result in significantly different levels of performance. When the conditional probabilities are obtained with the help of software, from patient data, the performance of the system can end up being significantly better compared to the case where the conditional probabilities are obtained from direct input of experts.

Potential problems with the Bayesian approach extend deep into probability theory and assumptions made about conditional probabilities. Due to the computational challenge of handling dependencies between input variables and outcomes of interest, most of the uses of the Bayesian approach end up with a form known as *naive Bayes*. In a simple clinical decision-making setting, this form uses random variables to represent diagnoses and symptoms. In this mathematical form, the candidate diagnoses are presumed to be mutually exclusive and their set is exhaustive. The symptoms are assumed to arise independently of one another. This set of assumptions does not realistically or adequately reflect the expert opinion in many cases.

Given that the naïve Bayesian paradigm can perform quite well even with these constraints, it is still an alternative to the symbolic reasoning paradigm.

(Spiegelhalter and Knill-Jones 1984) provides a useful discussion of the statistical methods in CDS. According to this study, the principal barrier to wider use of statistical methods (including Bayesian methods) is related to difficulties in building the knowledge bases (probabilities). Even though recent advances in Bayesian modelling and computation may help deal with incorrect or inadequate assumptions about probabilities, the problems associated with using probabilities as a descriptive language for a knowledge base still remain.

Efficient representation of domain knowledge is a significant factor for successful decision support, as demonstrated in both the symbolic and probabilistic paradigms. The modern extensions of the next paradigm focus heavily on this goal.

Data bank analysis

A term coined in (E. H Shortliffe, Buchanan, and Feigenbaum 1979) is “Data Bank Analysis for Prognosis and Therapy Selection”, and this term identifies a further CDS paradigm which has seen exponential growth in popularity compared to the others. What was described as a “Data Bank” in the 1970s has evolved into today’s electronic healthcare records, if one refers to the key features and goals of such systems, as described in (E. H Shortliffe, Buchanan, and Feigenbaum 1979), which positions the medical record repository as a tool to provide access to large amounts of data for better care management. The principal suggested use is clinician access to a library of past cases that are relevant to a current patient’s case, and consuming that information to make decisions, an approach that is widely adopted by modern clinical information systems.

Since the umbrella term used for this approach in modern systems is EHR, this thesis considers the EHR as the modern representation of data bank paradigm. (Kalra and Ingram 2006) provides an in depth, up to date exploration of the concept, showing how EHR became a unifying platform for many purposes. The diversity of the studies referenced by (Kalra and Ingram 2006) is proof for this unifying, platform centric view of the EHR. In the following excerpt from this work, the authors refer to CDS functionality that can be provided through the use of the EHR:

“The widescale use of decision support and alerting systems that interact with patient records is considered an essential informatics solution to the prevention of errors”

The growth in adoption of the EHR alone cannot deliver the potential benefits for decision support. The focus on EHR in the medical informatics domain is obvious, supported by the discussion in (Kalra and Ingram 2006), but the extent to which EHR implementations have managed to support better care via decision support is another matter. (Linder et al. 2007) finds no significant improvement in 14 of the 17 indicators they examine in the context of EHR use.

This thesis acknowledges the potential benefits of decision support in EHR systems, but it also observes that these benefits are not necessarily reflected into clinical information systems due to a number of reasons. Also, the focus of this thesis on EHR does not imply that a crucial method for representing and processing clinical data, that is, use of coding systems in medicine is ignored in the context of CDS. The practice of coding health data significantly predates the emergence of computers, given that the International Classification of Diseases (ICD) was first published in 1893 (WHO 2015), (James J. Cimino 1996). The use of codes for representing clinical concepts is a cross cutting component of both symbolic reasoning (J. J. Cimino 2011) and EHR based approaches to clinical decision-making (James J. Cimino 1996) and is used at the national level (de Lusignan et al. 2001).

The use of healthcare terminologies as components of healthcare information models, which are the building blocks of modern EHR implementations, is a well established approach (R. A Greenes 2007), (Al Rector et al. 2006),(Mori 1995), which has led to recognition of the use of these terminologies by information model standards development groups (Zanstra et al. 1998). Therefore, this thesis assumes that the EHR is an encapsulating concept which makes use of terminologies such as ICD-10 (World Health Organization 1992) or SNOMED-CT (IHTSDO 2015) to fulfil requirements that depend on existence of semantic identifiers for information model elements.

2.2: The Detachment Problem in Clinical Decision Support

The performance of decision support systems is dependent on a number of variables, such as the amount of clinical data available to the decision-making mechanism, the ease of building and maintaining the knowledge base that allows processing of the data and the execution methods and performance of the software that implements the decision support mechanism. The components of decision support systems, which these variables are associated with, are usually research topics in their own right, connecting the CDS domain to other research domains

from other disciplines, and sometimes they have further dependencies on other variables, such as the choice for the underlying computing platform.

The efficiency of the collective functionality of the components of CDS determines the performance of the implementation, and failure to establish a sufficient level of integration between these components leads to a detachment problem.

The detachment problem describes the state in which CDS cannot deliver the desired and expected outcomes due to integration inefficiencies, the scope of which includes both CDS components and data. The nature of the problem can be broadly described in terms of:

- Conceptual detachment
  CDS implementations cannot fully benefit from a promising method such as probabilistic reasoning, because it cannot incorporate and utilise the relevant concepts in an efficient, acceptable way.
- Data detachment
  A CDS implementation cannot access clinical data to function as expected, or it can only access the data in limited and specific ways, which limit the benefits achievable, even if the CDS method is broadly applicable. In the first case, the CDS is detached from data, and in the second case, the CDS is detached from other settings and systems where it could have been useful.

An example of conceptual detachment would be handling uncertainty in a rule based system, exemplified by MYCIN's certainty factors which represent probabilities. The conceptual detachment that necessitates the use of certainty factors is the inconvenience faced by the domain experts when they need to provide probabilities to express uncertainty.

Data detachment can be exemplified in most clinical systems integration scenarios for a CDS implementation. In probabilistic models, access to a higher number of instances of a set of domain variables allows more precise discovery of the nature of relations among these variables. Accessing to more domain variables that may have significance in the model is also of key importance. In cases where data is divided among different systems, such as patient demographics data and laboratory data residing in different systems, or past clinical data scattered among various institutions, only a subset of these detached data subsets is available to probabilistic model implementation, diminishing its performance.

These problems have been recognized by the research community. Various studies, implicitly and explicitly discuss either the connection between information

systems and decision support, or the lack of it. (E. H Shortliffe 1987) provides an early discussion of the issues, referring to them as “logistical issues”. He discusses the requirement to provide the same data separately to different systems, where there is no connection between systems.

The following quotation from (Shiffman 1994) expresses the importance of access to data. Referring to (Shortliffe EH, Perreault LE, Wiederhold G, and Fagan LM. eds. 1990) he says:

“Successful use of decision support tools is dependent on their integration into routine data management tasks”

(Sim et al. 2001) recognizes the difficulty of building the links between routine data and decision support systems as follows:

“Significant financial and organizational resources are often needed to implement CDSSs, especially if the CDSS requires integration with the electronic medical record or other practice systems”

(E. H Shortliffe 1993) also underlies the problem where the CDS cannot become available to users:

“I believe that the greatest barrier to routine use of decision support by clinicians has simply been inertia; systems have been designed for single problems that arise infrequently and have generally not been integrated into the routine data-management environment of the user”

(Müller et al. 2001) provides a case study in which an existing, standalone CDS is integrated with a Hospital Information System (HIS). The study outlines the requirements necessary to unify decision support functionality with software based medical information management to integrate an abdominal pain scoring system and a hospital information system. (M. A Musen, Shahar, and Shortliffe 2006) expresses the importance of this kind of integration in the following statement:

“We need more innovative research on how best to tie knowledge-based computer tools to programs designed to store, manipulate, and retrieve patient-specific information”

This quotation implicitly assumes that software that is designed to store and process patient specific information is different from knowledge based software, i.e. the decision support software in our context.

This differentiation between information storing, and information processing software is worth noting. The difference has its roots in the history of these two types of systems. As discussed before, the initial expectation from EHR systems has been to store as much information as possible and to retrieve information when the clinician needs it. The decision support systems must process the data, and consequently, the related domain knowledge must be represented in a consistently and efficiently computable form.

The problem of detachment of key components of CDS such as clinical data, decision-making mechanism and knowledge representation is actually an undesirable side effect of the evolution and specialization of these components. This specialization has been recognized by early works such as (E. H Shortliffe 1987). Understanding the current state of these components and the relations between them is crucial to improving their connectivity.

Despite the increase in use of computers in healthcare, potential improvements to clinical practice that can be provided by clinical decision support systems, and quite clear recognition of the problems with CDS approaches, the implementations of CDS systems have not become widely available in the health informatics domain. This rather small amount of adoption is worth recognizing as the first point regarding the current state of such systems, even though the capability of existing clinical decision support systems may far exceed the early systems of the past. This point is helpful as a motivation for questioning the successful and less than satisfactory aspects of current clinical decision support systems.

(R. A Greenes 2007) is a recent study that provides a detailed treatment of the CDS domain, which identifies key problems and issues in CDS and suggests possible solutions. The following list of observations and references to relevant works are taken from this work, with additional comments where necessary.

- CDS systems are hard to develop due to the difficulties encountered in constructing the knowledge base component required to drive them. The construction of a knowledge base is followed by maintenance and improvements, effectively extending the knowledge representation task into a knowledge management lifecycle.
- In order to make use of wider clinical input in the construction of CDS systems, knowledge representation must employ standard methods that allow sharing of results across systems and between clinicians.
- Even though CDS based diagnosis has been a strong focus of interest, it has been rarely implemented and used in actual everyday clinical practice, beyond

its place of origin. Prognosis related CDS implementations seem to be more widely used(R. A Greenes 2007).

- The relationship between the Clinical Information System and the CDS system is critical in terms of the capabilities of the CDS. The following quotation from (R. A Greenes 2007) refers to this relationship in the context of interactions required for the CDS system to perform successfully:

"For these kind of interactions to work, the specification of the data elements needed by CDS must be compatible with those in the IT system, and the actions that CDS determines should be performed must be capable of being carried out by the IT system."

- Regardless of the capabilities of the CDS system, the implementation should consider the workflow of clinicians during the clinical processes. (R. A Greenes 2007) gives an example which, for lack of this consideration, led to complaints from clinicians about the performance of a computerized physician order entry system, as described in (Shabot 2004)
- There has been a continuous effort to formalize and improve various aspects of CDS systems. This formalization is required to enable easier creation and management of knowledge bases and facilitate integration with Clinical Information Systems. Various specifications have been created as a result of this requirement, and they have been in continuous evolution to respond to increasing complexity of information systems.

  For example the Medical Logic Module (MLM) devised by (George Hripcsak 1994) and implemented in Arden Syntax (George Hripcsak 1994) is a good example of formalization of multiple CDS components within a single specification. Arden Syntax, as explained in (G. Hripcsak et al. 1994) and (George Hripcsak 1994), includes components to define the trigger event for invoking decision support, the logic that will be executed as a result, the action that will be performed in response to execution of that logic, and finally a data mapping from the underlying clinical data source to the MLM components. (R. A Greenes 2007) provides several independent contributions in the field which have formalized clinical actions as computable guidelines: Guidelines Element Model (GEM) (Shiffman et al. 2000), Guideline Interchange Format 3 (GLIF3) (Boxwala et al. 2004) and GELLO (HL7 2005). These formalisms focus on defining the logic and actions in a computable way. (R. A Greenes 2007) also cites (Peleg et al. 2003) for a comparison of computer interpretable guideline methods.

Other published components of Arden Syntax, such as its data mappings, have also been influential in the development of modern CDS approaches. EHR standards now provide a much richer, and formal method for representing and mapping clinical data between systems. Thus, modern decision support implementations can now make use of the clinical information models developed using these standards, rather than creating CDS system specific methods for data representation and mapping. In fact, modern EHR specifications have scope that goes beyond data representation: they can now be used to model actions that need to be carried out during the care process, such as prescription and administration of medications.

Despite continuing efforts to formalize the representation of key components of CDS systems, some key mechanisms that underlie their decision-making capabilities, and problems associated with them, have not greatly changed. Probabilistic approaches, such as BNs , for assessing and processing clinical data are still not widely used, beyond the research exemplars and according to (R. A Greenes 2007), simpler mathematical approaches have been dominant, as expressed in the following quote:

"Just as in the foregoing discussion relating rule-based systems and more sophisticated knowledge representation paradigms, simple understandable models (e.g., linear and logistic regression, score systems) have far outweighed in number and utilization the more sophisticated machine learning models (e.g., support vector machines, neural networks, and recursive partitioning algorithms), many of which remain limited to research applications."

However, even the most common and well understood statistical methods may require extra steps during statistical model building, such as performing a transformation on some of the covariates in a logistic regression. These types of tasks, as outlined before, tend to block the use of efficient probabilistic methods, due to clinicians having difficulty in handling probabilistic concepts and operations.

These observations lead to two main conclusions in the context of this thesis: First, there has been great effort devoted to developing formal methods for providing CDS. Second, powerful and promising methods for handling intrinsic uncertainty are still not widely available in CDS system implementations. Therefore, potential improvements to CDS need to explore the formal methods of defining and processing clinical data as the underlying approach.

In the larger context of medical informatics, another field of research characterized by a recent strong focus on the formalization of clinical data is electronic health records. This parallel attempt to improve the state of the art via

formal representation and information processing methods in both EHR and CDS research is a significant unifying characteristic for both domains.

2.3: EHR, Computable Health and Clinical Decision Support

The EHR paradigm provides all the functionality that the data bank approach to Clinical Information Systems aims to provide. However, during its evolution the EHR became more than a data storage formalism. Many factors have contributed to the evolution of the EHR concept, such as increased capacity and lower cost of computer hardware and software, evolving ethico-legal requirements, greater prominence of requirements for shared care and cost effectiveness considerations. The following factors have been highly influential in changing the concept of an EHR from a data store to an infrastructure for computation:

- The requirement for data sharing across various clinical information systems and, as a consequence, the requirement for EHRs to be accessible using the different technologies that are the basis of those information system implementations.
- The requirement for the EHR to provide functionality to support as many scenarios from different clinical domains as possible, leading to the requirements for conceptual coherence, data integrity and interoperability.

Standardization efforts for the EHR can be considered as the most successful method for handling these requirements. Modern EHR standards have usually avoided focusing on selected clinical domains or technologies. They introduce methods that allow definition of clinical data in a consistent way regardless of the clinical domain that the data comes from. EHR standards also address the issue of being available for implementation in multiple technologies, through the publication of a range of implementation technology specifications.

Through this approach, modern EHR standards have defined computable health information platforms, which can both exchange data and allow development of information systems using standards based data representation (Wollersheim, Sari, and Rahayu 2009), (Lopez and Blobel 2009), (P. H. Cheng et al. 2004, 7), (Kuhn 2007).Due to these infrastructural considerations, additional key components of health informatics software, such as demographics can now be positioned on top of the EHR, using it as a platform. This trend can be clearly seen in studies such as (Kalra and Ingram 2006) where many requirements of medical care are discussed in the context of EHRs, with references to other works that also support this view.

In the context of CDS, this approach makes the EHR is an obvious candidate for the underlying source of clinical data. The data mapping component of Arden Syntax (George Hripcsak 1994) is built on this approach, with the aim of connecting the large variety of clinical information systems that contain relevant data to a CDS formalism. The CDS implementation can delegate the responsibility for accessing clinical data to the EHR implementation, by using a formally defined information model as a source of data.

Recent developments in CDS related formalization attempts show that this division of responsibility for clinical data access is becoming a common approach. GELLO's (HL7 2005) close relationship to object models and Health Level 7 (HL7) is one such example.

2.4: Current State of EHR and CDS integration

A review of most recent studies on CDS reveals two key findings: the most fundamental problems related to CDS adoption still remain, but recent research is offering solutions to these problems, with notable emphasis on the conceptual integration between EHR and CDS, which is the focus of this thesis.

As may be expected, the problems reported by these studies have different architectural and technological contexts from their predecessors reported two decades ago, but their nature stays remarkably similar, albeit that a more standards focussed approach is recognisable.

From an architectural point of view, the standalone CDS implementations that require duplicate data entry are rarely adopted (Khalifa 2014). The lack of integration of clinical systems is a potential disruption to clinical workflow when accessing data that is required for CDS (Berner 2009) and much of health data is still not in machine-understandable form (Mark A. Musen, Middleton, and Greenes 2014). The maintenance of formally expressed knowledge for CDS is a challenge, both in central, service oriented and embedded architectures, in which, the CDS functionality is directly included in a clinical software with local specialisations (Berner 2009).

Even though the availability of specialised devices for clinical tasks helps clinicians, the data produced by these devices can only be used for CDS if they are part of an integrated architecture (Mark A. Musen, Middleton, and Greenes 2014). The use of terminologies such as SNOMED-CT (IHTSDO 2015) is an improvement, but data semantics problems still exist (Wright et al. 2015) and improvements in

standards that target CDS are defined as a priority (Wright et al. 2015; Kawamoto 2010)

These solutions focus on standards built on information models, in line with the approach developed in this thesis, and the collaborative use of multiple e-health standards for CDS, such as using openEHR's clinical data modelling approach and data types with the Virtual Medical Record (vMR) defined by HL7 (González-Ferrer et al. 2013). The standards harmonization work of the Clinical Information Modelling Initiative (CIMI 2015), which uses concepts parallel to those of openEHR's to represent CIMI models, attempts to provide well defined methods for this type of collaborative use of different standards. (Tao et al. 2013).

Currently, both openEHR and HL7 methodologies are used for the integration of CDS and EHR concepts and the use of EHRs is seen as an improvement for tasks that require large scale data, such as analysis of multimorbidity (Fraccaro et al. 2015). This line of thought is extended by the prediction of the future of the EHR as a vehicle for delivery of CDS (Mark A. Musen, Middleton, and Greenes 2014). These emerging solutions are also benefitting from the increasing availability of distributed computing frameworks and cloud architectures, such as the use of Hadoop in a cloud setting for building different types of applications based on EHR standards. Examples of these new kinds of applications are web and mobile applications (Bahga and Madisetti 2013; Bahga and Madisetti 2015) and data mining (Batra et al. 2014; Robert A. Greenes 2014). The availability of open source and model driven EHR platforms such as openMRS (Mamlin et al. 2006) enable easier development of these types of applications (MacLeod et al. 2012; Fraser et al. 2012) even in low-resource settings (Mohammed-Rajput et al. 2010).

The adoption of EHR as a platform is found also in new initiatives that aim to extend the practice of clinical care with new types of data, such as the CSER Electronic Medical Record Working Group, which was created to explore informatics issues related to integration of genomics data with EHRs and CDS (Tarczy-Hornoch et al. 2013). This integration, which, as of 2013, is implemented solely by embedding PDF files into patient EHRs, marks the addition of a new type of data to the EHR scope. Consequently, standards for processing genomic data in the context of EHRs are a requirement for enabling their use in CDS implementations (Overby et al. 2013; Tarczy-Hornoch et al. 2013).The HL7 Clinical Genomics Work Group is actively working on the development of standards for communication of genomic data (HL7 2015a).

From an openEHR perspective, the most significant recent development is the integration of CDS and EHR concepts in the openEHR Guideline Definition Language (R. Chen and Corbal 2015), which is currently in the process of being incorporated in the openEHR specifications. GDL enables the expression of rules that process clinical data based on openEHR data types and is used for real life CDS implementations (Chen 2012). Prior studies on openEHR based CDS show that use of external rule languages such as CLIPS (Riley 2015) with openEHR concepts for CDS, is also possible (Chen 2009).

The recent literature shows that a platform based approach to EHR and CDS integration, based on different architectures and technologies, is becoming the predominant approach, in both research and implementation efforts, and that openEHR is widely and actively in use. The research motivations of this thesis are thus confirmed as relevant and of interest in CDS implementations and in other studies.

2.5: Summary

Despite the improvements in the implementation of the various computational methods for providing clinical decision making capability, almost five decades of multidisciplinary effort is still unable to deliver widely usable CDS. A strong focus on the use of information models, and standards based on these models, is the current dominant approach for tackling these well-recognized limitations of CDS, and this approach implies the emergence of health computing platforms based on standards, in line with the architecture that this thesis work set out to define, using openEHR.

Chapter 3: The openEHR Specifications and Their Relationship to Clinical Decision Support

This chapter provides an overview of the openEHR specifications (David Ingram 2002), (Beale et al. 2006). The focus is on the concept of computable healthcare and how it helps in sharing information and behaviour with an aim to build the links between capabilities of openEHR, CDS and AI research.

The openEHR specifications, which are freely accessible in the form of multiple documents from the web site of openEHR foundation (http://www.openehr.org), provides a modern design for an EHR solution, which can potentially fulfil the requirements described in (Kalra and Ingram 2006). A more detailed exploration of the requirements of a modern EHR design can be found in the Good European Health Record GEHR project (D. Ingram 1995), (Lloyd et al.) deliverables. The openEHR specifications have their roots in the GEHR project.

Both GEHR and openEHR specifications tackle a key issue in healthcare IT: providing a standard method for computing healthcare related information, based on unified support for information models and terminologies. These specifications represent the evolution of software engineering and information systems design in clinical informatics. There are other initiatives such as Health Level Seven (HL7) and (ISO/EN 13606 2012), which are related to openEHR in terms of their goals and content (Schloeffel et al. 2006). The ISO/EN 13606 standard is based on a subset of the openEHR specifications.

The scope of these specifications covers both clinical and technical domains. An in-depth comparison of these specifications is out of the scope of this thesis. openEHR is the specification and the standard this thesis will build on. The following section provides an overview of the aspects of openEHR that enable computable health, followed by a discussion of the relationship between these aspects and probabilistic methods.

3.1: The openEHR Standard and Methodology

The fundamental characteristic of openEHR is its use of archetypes (Beale and Heard 2007a), (Beale and Heard 2008a) expressed via Archetype Definition Language (ADL) (Beale and Heard 2008b). The archetypes define clinical models via specifying constraints on structure and values of a reference model (RM). This approach is defined as two-level modelling (Beale and Heard 2008a).

The two-level modelling approach of openEHR is built on the accumulated results of a series of large scale research projects that took place over more than twenty years (David Ingram 2002), refining the results of research projects such as Synapses, which is based on the idea of an object model supported by a data dictionary (Grimson et al. 1997), (Grimson et al. 1998), (Bisbal, Stephens, and Grimson), and GEHR (Lloyd et al.).

The fundamental components of openEHR are brought together in a process that produces outputs which can be used to implement clinical information systems that support a wide variety of functionality. Even though this process it not explicitly named in the openEHR specifications, the design of the fundamental components and the ways they are meant to be used, which is clearly explained in the specifications, implicitly describes a methodology.

This thesis refers to this methodology as the openEHR methodology, referring to a clinical model driven software implementation lifecycle with an iterative nature. Therefore, the term openEHR methodology refers to a superset of openEHR specifications, extending them with the processes that make use of them. A high level representation of both the primary components of openEHR standard and the openEHR methodology that encapsulates them is provided in Figure 1.

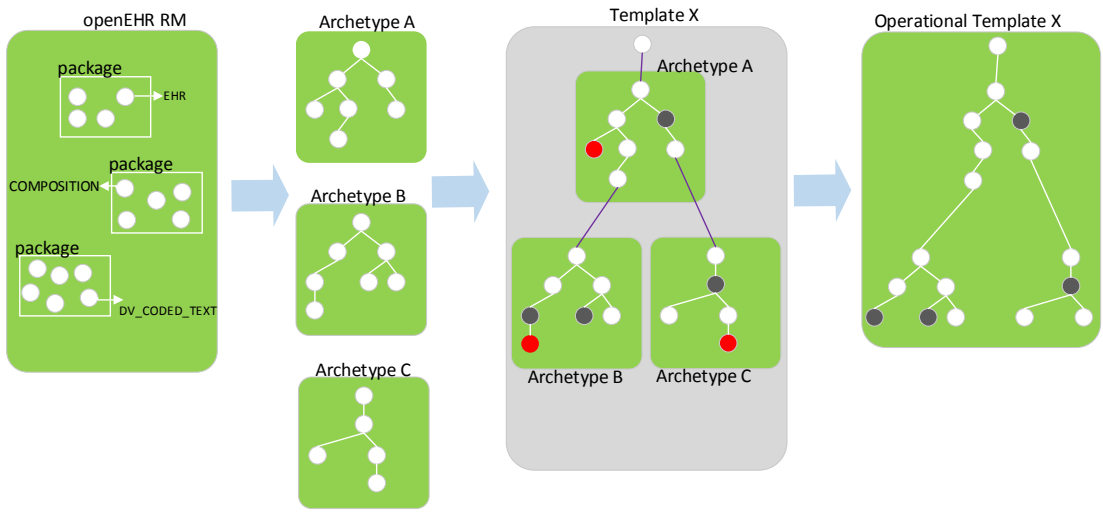


Figure 1: openEHR RM, Archetypes and Templates

The first level of the two-level modelling approach of the openEHR methodology is the RM, which consists of a limited number of types defined in detail. The term “type”, when used in the context of openEHR RM in this thesis, refers to the widely adopted mechanism of data abstraction as implemented by most object oriented languages without implying an approach taken by a particular

programming language. The openEHR specifications use the term “class” as well, without explicitly defining the relationship between the terms type and class. This thesis assumes that these terms are used interchangeably, and adopts the same approach. RM types will be written in uppercase, such as COMPOSITION, ELEMENT, or LIST_ITEM to distinguish them from programming language types or data item names.

These types, which collectively define the contents of the RM, address the requirements of representing values and structures with a focus on clinical concepts, including, but not limited to EHR (Beale et al. 2008e), Demographic (Beale et al. 2008a), Data Structures (Beale et al. 2008d), and Data Types (Beale et al. 2008b).

The RM is the basis of clinical models, called openEHR archetypes (Beale and Heard 2007a). The types in RM, organised under packages as shown in Figure 1, are brought together to define archetypes, using the ADL (Beale and Heard 2008b). The archetypes use RM types to compose clinical models to represent concepts such as blood pressure measurement or a list of allergies.

An archetype can use RM types such as COMPOSITION and their fields to define the structure of a clinical concept as well as allowed values of data such as a limited number of codes for a field that has the type DV_CODED_TEXT. This practice of composing clinical models represented by archetypes based on RM types is the second level of two-level modelling approach. The development of archetypes is most frequently described as clinical modelling. The practice of creating downstream artefacts of archetypes in Figure 1 also falls under this description. Figure 1 shows how a number of archetypes can be modelled using RM types. Archetypes also support the use of terminologies via the use of both capabilities of RM types and term and terminology binding capabilities of archetypes. Term binding allows an archetype-local identifier of a data item to be associated with a term from a specific terminology. Terminology binding allows a set of terms from a terminology to be defined as the valid values of a data item.

A key trait of archetypes is that they are meant to represent maximal data sets. That is, all data items that could be considered under a clinical concept should be included in the archetype for that concept. The term “data item” as used in this thesis refers to a clinical concept included in an openEHR archetype or another modelling artefact derived from an archetype. All the nodes in the diagram in Figure 1 that are in the containers that lie to the right of the openEHR RM represent data items, regardless of what their RM type or the complexity of that RM type is.

The archetypes are meant to be the basis of interoperability in the openEHR methodology and the maximal dataset approach ensures that if a particular archetype is used by different systems, there is ideally no need to add extra data items, leading to a modification to the archetype, which consequently may break compatibility with other systems. However, following this approach to avoid modifications to archetypes for adding content has the downside of archetypes representing many data items, not all of which may be required in every scenario.

There is also the possibility of the definition of a clinical concept optionally containing other clinical concepts with sufficient complexity that requires their own archetypes, in which case archetypes need to be included in other archetypes in various combinations, based on the clinical modelling requirements at hand.

These requirements are fulfilled via openEHR templates (Beale and Heard 2007b). As shown in Figure 1, an openEHR template can be used to bring together a number of openEHR archetypes. Archetypes can be included in other archetypes using their slot mechanism, their fields can be further specialised, such as limiting the set of codes allowed for a DV_CODED_TEXT field to an even smaller subset or some fields which are not needed can be removed. The modifications in a template can never conflict with the definitions of archetypes, they can only introduce further constraints on data items. If a data item is defined as mandatory in an archetype, it cannot be removed in a template, but the set of values defined as valid for that data item can be limited to a smaller subset.

The benefit of templates is that they allow specialisation of archetypes for a specific scenario without the need to introduce new archetypes. This approach keeps the number of shared archetypes to a minimum and encourages their re-use.

The use of templates for further customisation of archetypes introduces another modelling artefact to the openEHR methodology. Even though templates are implementation specific, their role and capabilities overlap with the second level of two-level modelling. At the time of the writing of this thesis, the openEHR specifications are being updated to remove the difference between templates and archetypes to eliminate the need for another downstream modelling artefact. This thesis focuses on the currently established method of openEHR implementation based on templates.

The clinical model defined by a template goes through a final transformation as shown in Figure 1 to create an operational template. An operational template is still a clinical model, which is based on RM types, but it is meant for deployment to a software implementation based on openEHR, and it cannot be modified further as a modelling artefact. The software implementation of openEHR is responsible for the

representation and management of actual clinical data generated during clinical care. The data represented by the openEHR implementations is referred to as a "data instance" or "RM based data" in this thesis. The term instance refers to the distinction between the clinical models which are blueprints of values that are generated during clinical care and the actual values stored and processed by openEHR implementations.

The process of going from a set of RM types to artefacts that are used by software implementations depicted in Figure 1 shows how openEHR methodology supports building clinical information systems based on domain models. The iterative nature of the methodology, not depicted in Figure 1, comes from the versioning support of archetypes defined in openEHR specifications. If a new use case for a system identifies a data item that should belong to an archetype, a new version of the archetype is develop based on the previous version, and then made available so that the process in Figure 1 can be repeated with the new version of archetype. The term "use case" refers to a scenario, in which a party, which may be an end user, or an information system, makes use of a particular functionality provided by an information system, which may be referred to as "system" for brevity.

The division of responsibilities between modelling artefacts depicted in Figure 1 imply that templates and their downstream artefacts are more implementation oriented than archetypes. The clinical models used in this thesis for experiments based on implementation are therefore discussed at the template level, but they are always produced following the approach in Figure 1.

Even though templates are more related to clinical information systems implementation than archetypes, they are still independent of any particular programming language or framework. The architecture of openEHR explained in Beale et al. (2007) does not assume or demand a specific technology for its implementation. Instead, the Implementation Technology Specification (ITS) approach of openEHR (Beale and Heard 2008a) provides mappings from openEHR concepts to technology stacks such as XML (Bray et al. 1997) or Java (Gosling et al. 2005). Actual implementations of these mappings can be used as the core of many different information systems in various medical domains.

Archetype based clinical models can represent concepts from many different clinical domains by bringing together a small set of data types in different combinations. The openEHR methodology provides a comprehensive solution to communication problems of medical informatics which are discussed in (M. A Musen 1992) in depth via use of these clinical models as a domain specific

language that acts as a means of liaison between clinicians and software developers, and is used globally (openEHR Foundation 2015).

Through shared models, different systems can share both data and behaviour. When a particular software implementation is built on the data types and structures provided by openEHR, it becomes capable of functioning in all systems that process data using openEHR. Therefore, the interoperability of health information between multiple systems delivers a key benefit by design: the decoupling of implementation technologies of clinical information systems from the clinical information processed by those systems.

In the context of decision support, this decoupling can improve the outcomes of two fundamental scenarios:

- When the required clinical information resides in multiple systems
- When a particular capability is required in multiple systems.

These scenarios provide a generalization of the interaction of multiple clinical information systems, which can be improved by openEHR in the following ways:

- The openEHR methodology provides a robust way to represent clinical data via two-level modelling, based on its ability to express many clinical concepts in addition to its technology agnostic specifications.
  The effective outcome of this design is the ability to exchange clinical data between clinical information systems in many clinical domains, independent of software platforms used for clinical information systems implementation.
- Information models are not the only formal way to represent and process knowledge and data. Use of standardized terminologies for the same purpose is a common method in many information systems. Terminologies such as SNOMED-CT (IHTSDO 2015) can also address key knowledge management requirements of healthcare-informatics, and their relationship to EHR specifications is an active topic of research and discussion. (Markwell, Sato, and Cheetham 2008) discusses the integration of SNOMED-CT to both HL7 and openEHR. (Al Rector et al. 2006) evaluates the use of ontologies to perform terminology to EHR bindings. (A. L. Rector 2001) defines a framework for allocating information in a setup where terminologies and information models are used together.
  openEHR's support for binding information models to terminologies extends its capability for clinical data exchange beyond information models, which addresses a larger set of use cases.

- openEHR specifications contain a query language for data access (in draft form at the time of writing of this thesis), specified in the same technology independent way as with the other parts of the specifications. This query language named Archetype Query Language (AQL) allows access to clinical data based on openEHR RM types, taking the platform independent clinical data representation concept even further by defining how this data representation should be queried. When a particular behaviour that relies on RM based data is implemented in a clinical information system, its portability to other systems can be improved if its data access mechanism is based on AQL.

As a health computing platform openEHR aims to support a substantial set of functionality building on the core capabilities described above. The diagram in Figure 2, taken from (Beale et al. 2006) shows how key concepts are distributed to layers which build on each other. This diagram shows the multi-layer vision of the health computing platform along with the relationship between abstract specifications and how abstract specifications are related to layers of the platform. The abstract specifications for RM, archetype model (AM), and service model (SM) allow definition of artefacts and functionality for the layers of the health computing platform.

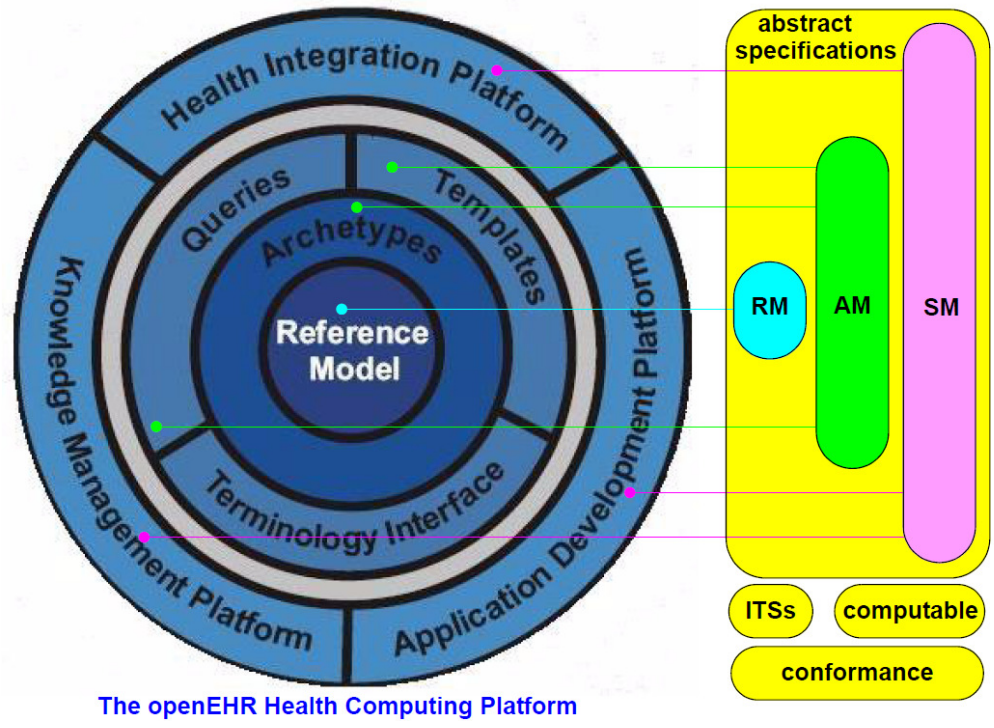


Figure 2: The openEHR Health Computing Platform

The key research question for this thesis is, whether the health computing platform depicted in Figure 2 can successfully support probabilistic methods for CDS to improve their availability to clinical information systems.

3.2: Information Models and Clinical Decision Support

As discussed in Chapter 2, historically there has been a significant overlap between AI and CDS. Most of the methods that process clinical data to arrive at conclusions have their origins in AI research, which in turn builds on the results of other fields of research with varying levels of abstractness. Therefore, various branches of mathematics, set theory, statistics, information theory, computer science are connected to clinical practice through CDS to an extent that depends on the nature of the AI approach used.

The adoption of results of research from the large domain of AI for CDS is a complex procedure due to the vast scope of medicine and consequently the variety of data generated during medical care. The MLM concept used in Arden Syntax (George Hripcsak 1994), which is discussed in Chapter 2 shows that making use of even a rather simple decision-making mechanism, such as the rule based approach, for CDS can be a challenge due to complexity of underlying data. The rule based functionality of the CDS implementation based on Arden syntax can be disrupted due to changes in the format of the clinical data (Jenders et al. 1995). Integration of the decision-making logic to actual clinical systems may suffer from various problems such as the lack of support from Arden syntax for complex data types required to represent clinical data (Peleg et al. 2001) or efforts required to implement integration for each clinical information system (Samwald et al. 2012).

The problems encountered during integration of Arden syntax implementation to clinical information systems are independent of the capabilities of the decision-making mechanism, and underlying AI research. Therefore, integration of CDS functionality to EHR requires an in depth analysis to discover if successful adoption of the reasoning method for CDS is feasible in the context of the integration of interest.

When a particular reasoning approach is employed in the EHR standards based CDS context, the extent to which it can be supported is dependent on both the design of the underlying standard and the nature of the CDS approach. The successful integration of a reasoning approach is dependent on aspects of these two components. This chapter concludes with a brief discussion of some of the key traits of openEHR, which represents the first component of EHR and probabilistic AI

methods integration for CDS. The other component, BNs will be discussed in the next chapter

Some of the key computational features of openEHR that are immediately relevant in a CDS implementation are as follows:

- Domain specific data types

  The openEHR data types model a wide range of concepts from the clinical domain, and they also allow the use of standardised clinical terminologies to encode values. This allows the representation of both numeric data and nominal variables in a computable way. openEHR data types encapsulate all the clinical data that will be created and used within an openEHR-based system. Representation and access to clinical data are provided by a small number of types that are basic to computation about a large variety of clinical values.

- Constraint based model definition

  The constraints defined by the openEHR ADL on RM types are not only structural constraints. They can cover any attribute of the RM types, including data types and their values. Having the capability to define valid data through constrained attributes allows openEHR-based systems to reject clinical data immediately at its creation if it does not comply with the constraints of the model. This means that outliers, missing values or inconsistent values in clinical data will either not exist, or they'll be at a minimum.

- Coherent and consistent abstraction

  openEHR's features allow using same formal definitions of clinical data for all operations related to data processing. Clinical models, clinical data that complies with those models, and finally access to clinical data through a custom query language all use same components of the specification. This allows all implementations of openEHR to compute solely on the specification, without falling back to an implementation specific aspect. An important point worth noting is that the robustness of the openEHR type system and modelling methodology does not mean full coverage for all computations that may be required. There is inevitably a limit to robustness, and exploring that limit in the context of probabilistic CDS is one of the primary goals of this work.

3.3: Relevant standards

The focus of this thesis is on the integration of two high-level concepts, EHR and CDS, explored through experimental implementations of two representative methodologies – openEHR and Bayesian Networks. This choice reflects the

significant overlap between the features of openEHR and Bayesian Networks and the integration requirements identified by CDS research. Other prominent electronic health standards also have features that overlap with these requirements but they were considered less suitable for the research goals of this thesis, as confirmed in an evaluation performed at the outset of the research. Nonetheless, these standards, namely HL7, ISO/EN 13606 and SNOMED-CT, are still actively developed and used. Therefore, their relevance and important traits are discussed in the following sections.

3.3.1 HL7

The HL7 standard has a strong focus on the concept of messaging between healthcare systems. Even though the standard itself has gone through major changes between its second version, that was released in 1998, and its third version released in 2005, the emphasis on messaging has not changed. This emphasis is significant in the context of this thesis, since formalising messages that are exchanged between systems does not necessarily imply or necessitate implementation of these systems on the basis of the same clinical data models that are used for messaging. The focus on messaging does not imply that a clinical information system cannot be completely based on HL7 standards. At least one software framework, Tolven (Tolven Institute 2015b) has shown that this is possible. However, the information model used in HL7 V3 does not have fundamental EHR concepts at its core, in the way that openEHR does. This does not mean that HL7's modelling capabilities are strictly limited to messages. A subset of the standard that consists of a Clinical Document Architecture (CDA) (Benson 2012) for exchanging clinical documents, and a Continuity of Care Record (CCR) (Benson 2012) for expressing the critical care history of patients, extends HL7's scope to the exchange of clinical documents and transfer of patients' existing records to new systems. In particular, the CCR overlaps with EHR concepts, due to its potentially longitudinal record nature.

The Reference Information Model (RIM) introduced by HL7 V3 supports an approach similar to that provided by openEHR, reusing a small number of data types to represent a large number of clinical concepts. However, HL7's approach to developing domain models does not align with the object oriented approach to domain modelling as much as that of openEHR. For example, the specialisations of

a core set of high level classes are expressed via codes in HL7, whereas openEHR uses only strict inheritance rules to express type information. HL7 introduces methods such as omission of member attributes of classes along with cloning of classes, which do not easily map to object oriented modelling concepts. openEHR's approach, on the other hand, introduces the Archetype mechanism to provide a single method for reusing its core reference information model components, and this mechanism is fully specified and implemented as reusable open source software libraries using object oriented languages. HL7's support for the use of terminologies is extensive, allowing similar capabilities to openEHR.

The complexity encountered by implementers in making use of the HL7 information model introduced in V3, is currently being addressed by a new addition to the HL7 standard, named Fast Health Interoperability Resources (FHIR) (HL7 2015d) (Bender and Sartipi 2013). At the time of the writing of this thesis, FHIR is in Draft Standard for Trial Use (DSFT) state, but it is likely to replace the complicated modelling practices of HL7 V3 with a simpler framework based on the concepts of resources and, in addition, a much stronger focus on implementation (HL7 2015d).

FHIR represents a step change in the way HL7 information models are created and extended, but it is not the only recent development of this kind. Another relevant standards initiative, established in 2011 in its early form, is the Clinical Information Modelling Initiative (CIMI) (CIMI 2015), which aims to deliver logical models which can be used to produce multiple downstream physical data representations. The importance of CIMI, especially in the context of HL7, lies in its approach based on a reference model and archetypes, strongly influenced by early contributions from the openEHR community At the time of the completion of the writing of this thesis, CIMI is actively engaged on establishing CIMI models as the basis of FHIR profiles (HL7 2015b), based on a harmonisation of models from different standards and terminologies, using the logical models to be developed by CIMI. Therefore, this approach implies introduction of two level modelling in the HL7 domain, via logical model harmonisation provided by CIMI.

These developments lead to the following observation: HL7 is an electronic health standard with a large community and its adoption is definitely capable of providing support for machine processable health data. However, this support is not focused on the concept of the EHR and clinical system implementation. Even though recent initiatives, as described here, are paving the way to easier implementation of these EHR concepts within an HL7 based system architecture, these same initiatives are also continuously changing the information modelling methods of HL7. In contrast, the information modelling paradigm of openEHR has

not changed or needed to change since its inception, making it a more straightforward and less volatile option on which to base an experimental approach to EHR and CDS integration.

3.3.2 ISO/EN 13606

The ISO/EN 13606 standard aims to achieve semantic interoperability in electronic health record communication (CEN/ISO 13606 Association 2015). It aims to enable communication of all, or a part of, an EHR between EHR systems. ISO/EN 13606 has the EHR concept at its core, and is also built on the dual modelling approach of openEHR, based on a reference model and constraints defined by archetypes. This conceptual similarity between ISO/EN 13606 and openEHR is not a coincidence: ISO/EN 13606 is a subset of the openEHR specification and was developed under the leadership of founding members of the openEHR Foundation, based at UCL.

Even though its scope emphasises exchange of information rather than implementation of EHR systems, ISO/EN 13606's reference model and use of archetypes allows it to be used for implementation of clinical information systems, sometimes making use of existing openEHR archetypes, enabled by the very close relationship between the two standards (Cornet 2015). There is at least one operational implementation of the standard as an EHR system (Austin et al. 2011). The standard has been used for automated generation of user interfaces (Kohler et al. 2011) and web applications (Menárguez-Tortosa, Martínez-Costa, and Fernández-Breis 2011). These use-cases have focused on application development. Thus, ISO/EN 13606 is used both for data exchange (Nogueira Reis et al. 2015), (C. Rinner, Wrba, and Duftschmid 2007) and application development.

The close relationship with openEHR and the existence of research and implementations that address data exchange and persistence, are positive aspects of ISO/EN 13606 in the context of the research goals of this thesis. However, ISO/EN 13606 presents a number of problems in the same context, that make it a less than ideal option for implementation work. Firstly, as with HL7 at the outset of this thesis project, the use of the ISO standards is governed by rules that are less liberal than those for openEHR adopters: (Austin et al. 2013) points out that programmers are forbidden to add details of the standard to an implementation artefact, due to IP restrictions.

The second and more limiting problem is that ISO/EN 13606 lacks a publicly accessible and usable, shared implementation technology specification. Such a specification, for example, is frequently provided in the form of an XML schema, which enables exchange of well-formed documents between implementations, providing a lowest common denominator for implementation. Not only does ISO/EN 13606 not provide an XML schema, but, according to (Austin et al. 2013), it cannot be directly represented in this form, forcing implementers to find workarounds for developing a suitable XML schema for data exchange.

As a result, implementers that make use of ISO/EN 13606 either develop their own XSDs or they try to re-use the ones provided by other research groups or implementers. Even though the problems introduced by differences between such schemas can be eliminated technically during the integration of systems, this practice is no better than the inevitable manual process that was required in most applications based on the still very widely used HL7 V2 messaging standard, which suffered from a similar lack of rigour and coherence, though for different reasons.

The lack of easily accessible clinical models or a modelling community for ISO/EN 13606, is also a disadvantage as a candidate for use in this thesis, although the use of openEHR models, made possible by the significant overlap of the methodologies, can to some extent alleviate this problem.

Therefore, even though, in principle, it offers many of the advantages of openEHR , in all aspects the use of ISO/EN 13606 would be less efficient and straightforward than the use of openEHR, and the end result would be a platform that could not offer a standard method of integration with other systems, based on XML.

3.3.3 SNOMED CT

The systematised nomenclature of medicine clinical terms (SNOMED CT) is a clinical terminology that is maintained by The International Health Terminology Standards Development Organisation (IHTSDO 2015). The use of a clinical terminology, in conjunction with an EHR standard, is necessary to express, independently of human language, the semantics used to record clinical information and the structure of that information within an EHR implementation.

SNOMED CT provides a framework for expressing concepts and relationships to define semantics and is used by HL7, ISO/EN 13606 and openEHR to clarify semantics and improve semantic interoperability. The evolving capability of

SNOMED CT to express complex meanings (Benson 2012) has led to an increasing overlap between SNOMED CT and EHR standards that make use of it (Martínez-Costa et al. 2015) (Markwell, Sato, and Cheetham 2008). This thesis does not address this overlap nor approaches to manage it. SNOMED-CT provides capabilities to express complex concepts, but these require careful and consistent use, as do the information modelling capabilities provided by EHR standards, in order correctly to express meaning (Alan Rector and Iannone 2012).

SNOMED -CT is increasingly in use, internationally (Lee et al. 2014), and can be used for reasoning about records. Its size and complexity (with more than 300,000 concepts and 1.4 million relationships (Benson 2012)) posed a significant implementation challenge for its use in this thesis, although an open source terminology server was used in one of the early experiments. Its importance is acknowledged and problems that could be alleviated with the use of terminology support in the experiments reported, are recognized and discussed in the body of the thesis. A complete integration of SNOMED CT with the models created and experiments conducted, had to be left out of scope, due to time and resource constraints.

3.4: Relevant frameworks

Given the breadth of the EHR and CDS integration that this thesis tackles, an evaluation of all relevant health IT frameworks and standards based on the implementation driven approach of the thesis, is not possible. However, two potentially relevant software frameworks were briefly evaluated at the beginning of the project in late 2008, for an assessment of advantages they might offer in collaboration with openEHR, or as an alternative to it, especially for ease of software implementation. Even though these frameworks were not used in the thesis, their on-going progress has been continuously monitored during the progress of the project and the writing of the thesis, since they are both based on information models and helped in identifying trends in EHR implementation.

The two frameworks that were considered are Tolven and openMRS. The key characteristics of these frameworks in the context of the research goals of this thesis are summarised below.

- **Tolven**

  The Tolven platform (Tolven Institute 2015b) is built on the HL7 V3 (Beeler 1998, 3) Reference Information Model (RIM) (HL7 2015c) . However, Tolven extends the standard HL7 RIM with the aim of providing an application framework. These extensions, the most significant one being Templated RIM (TRIM) (Tolven Institute 2015a), provide capabilities similar to that of openEHR's two level modelling approach. Tolven documentation also emphasizes its attempt to break out of the HL7 message focused approach.

  Tolven provides a generic mechanism that processes clinical data in the form of documents, where documents can contain standards based content as well as non-standard data. Rules governing these documents are then used to process content, which is normalised according to the HL7 RIM data types. This document processing mechanism can be extended via the use of software plugins, which is the mechanism offered by Tolven for implementing new functionality. Tolven offers a pre-defined set of clinical content models along with a web based user interface and other functionality that provides a web based application for use by health care providers and patients.

  Even though the scope and functionality of Tolven enables its use as a platform for clinical information systems and CDS (Aziz, Rodriguez, and Chatwin 2014), (Kondylakis et al. 2012), (Welch et al. 2014), a number of issues arise in the context of the research goals of this thesis.
  First of all, Tolven's functionality is an extension of HL7, which, at the time of starting this PhD project (2008) did not have an intellectual property policy as liberal as that of openEHR. The HL7 standard only became freely usable in 2012. In comparison, the openEHR Foundation has been offering excellently documented standards, completely free of charge, since its establishment in 2003.

  The other issue associated with Tolven, as far as this thesis is concerned, is the extension of the HL7 RIM standard with Tolven specific modelling mechanisms, such as the TRIM. This Tolven specific modelling approach does not align with the goal of using a platform that is completely based on a global EHR standard , that was adopted for this project and thesis. Compared with the very rigorously defined and continuously, internationally, reviewed clinical models of openEHR, along with freely its available software tooling, the TRIM is a niche approach with much less widespread adoption and support.

  Finally, Tolven's software architecture, based on open source and flexible components, is designed to be extended via its plugin mechanism and includes

functionality such as user authentication, which is not included in the scope of this thesis. Its architecture and existing implementation allows fast development of web based clinical applications, but only a subset of its components and functionality are relevant for the work of this thesis. Isolating that subset, without having to deal with ripple effects in terms of code refactoring elsewhere, would be potentially a very large task, with no guarantee of being able to use Tolven's existing extension mechanism under these circumstances.

These findings, in addition to there being significantly less published documentation in comparison with openEHR, led to the early elimination of Tolven as an experimental implementation platform. However, these finding are specific to the aims of this thesis and do not imply inferiority of Tolven itself, which is successfully used for both application development and research.

- **OpenMRS**

  OpenMRS (OpenMRS Inc 2015) is an EHR implementation that is used extensively in low-resource settings in developing countries (Mohammed-Rajput et al. 2011). Its goal of enabling EHR functionality in highly demanding environments where both basic infrastructure and human resource are scarce, requires that its functionality can be reused and extended with a minimum amount of effort (Allen et al. 2007).

  Therefore, OpenMRS provides a data model and functionality that is comprehensive and extensible. The data model supports both clinical and demographic concepts. OpenMRS also offers capability to define and create user interfaces. Therefore, it can be defined as a self-contained, extensible EHR implementation and a platform.

  The single, most significant disadvantage of OpenMRS as a candidate platform for the experiments required in this thesis is that its information model is not directly built on a particular EHR standard. Instead, a flexible information model is used with consideration of standards such as ICD10 (World Health Organization 1992) for terminology and HL7 (Fraser et al. 2013) for messaging.

  Despite the attention given to use of these standards, some studies have found the integration of openMRS's data model with capable ontologies such as SNOMED-CT (IHTSDO 2015) to be problematic (Halland, Britz, and Gerber 2011). Recent research is focusing on adopting standards based APIs for connecting OpenMRS to other applications, using the relatively new Fast Healthcare Interoperability Resources (FHIR) (Bender and Sartipi 2013)

standard from HL7 (Kasthurirathne et al. 2015). This design is presented as a better alternative than OpenMRS's current approach to interoperability with other systems, which has been described as neither sustainable nor generalizable (Kasthurirathne et al. 2015) and non-trivial (Waters et al. 2010). These findings position OpenMRS as a platform that offers benefits for implementations where there is a need to develop new functionality as quickly as possible, using a non-standards based internal data model. Even though these benefits have allowed OpenMRS to be used successfully in clinical care and research projects such as workflow integration (Yu and Wijesekera 2013), the lack of a standards based information model limits the usability of the potential research outcomes as compared to an openEHR based approach.

Despite its shortcomings in the context of the research aims set out in this thesis, OpenMRS provides significantly better documentation than does Tolven, and its data model and extension mechanisms allow efficient development of clinical applications, as proven by its many deployments around the world (Mohammed-Rajput et al. 2011). Therefore, had it been available in suitably complete form at the time, and were it to have adopted a less application centric design and focused on fully standards based information models for EHRs, OpenMRS would have been a preferable framework, compared with Tolven, for adoption in the project described in this thesis.

3.5: Summary

The openEHR specifications provide a method for expressing domain information in a computable way. They go beyond the data bank definition of early systems, by providing a computable health platform. They also support other relevant formalisms, such as clinical terminologies.

Artificial Intelligence research in clinical decision support developed formal representations of clinical domain knowledge earlier than researchers in the EHR domain. More recently, principally due to greater recognition of the requirements for semantic interoperability, EHR research has begun to focus on the formal representation of domain knowledge. This common convergence towards formal methods of information representation, is the unifying characteristic of both CDS and EHR, on which new approaches towards better CDS may be formulated.

The features of openEHR make it a modern example of an EHR formalism, and even though some of the advanced functionality defined in the openEHR specifications is still not universally implemented, this thesis classifies openEHR as a mature EHR specification and a good candidate for hosting complex CDS approaches, due to its strong support for formally defining data. This naturally leads next to a discussion of the CDS approach adopted, namely Bayesian Networks, in an openEHR context. The next chapter provides a discussion of the integration of BNs with openEHR, and the potential benefits of an openEHR based CDS implementation.

Chapter 4: Bayesian Networks for Clinical Decision Support and Their Integration with openEHR

A BN is a probabilistic model which belongs to a larger family of models called Probabilistic Graphical Models (PGM). PGM research in AI has been gaining traction in the last two decades and outcomes of this research is used for tasks such as clustering, reasoning, classification and decision-making (Larrañaga and Moral 2011), (Koller and Friedman 2009). Increasing interest in PGMs in general and BNs in particular is mainly a result of the availability of significantly more computing power, removing the necessity for restrictive assumptions imposed by the more simplistic Bayesian methods deployed in the 60s and 70s.

BNs have some traits that make them convenient and capable decision-making tools for CDS. Some of these traits bear similarities to model driven approach of openEHR at a high level, which makes BNs a good candidate for a CDS mechanisms based on an openEHR implementation. This chapter explains what BNs are, their promise and their potential relationship to the openEHR specifications, in the context of computable health and decision support based on it. The discussion begins with fundamentals of Bayesian methods of handling uncertainty and extends to more complex settings.

4.1: Bayesian Approach to Uncertainty

Bayes' theorem, which is at the root of Bayesian statistics, was published after Thomas Bayes's death. Richard Price, a friend of Thomas Bayes found an essay after Bayes had passed away, and he sent it to the Philosophical Transactions of the Royal Society of London. (Bayes and Richard, 1763)
The mathematical form of Bayes' theorem is a very simple formula, as follows:

$$P(X|Y) = \frac{P(Y|X).P(X)}{P(Y)}$$

Equation 1: Bayes' theorem

Despite this simple form, the effect of the Bayesian approach to probability has been profound. Bayes' theorem is actually a restatement of conditional probability. Equation 1 describes the relationship between two random variables. It shows that the probability of variable X taking a particular value, given that the value of random variable Y is known, is proportional to likelihood of Y taking its known

value times prior probability of X's particular value. Both X and Y can be vectors of random variables, which lets this form to represent the relationship between multiple random variables. The interpretation of this simple function has been a major topic of discussion between statisticians who follow different interpretations of probability. The dominant school of statistics uses the concept of frequency of a particular event taking place (the frequentist approach), and Bayesian approach uses one's belief, or judgement about the value of a random variable to interpret Bayes' theorem. Despite the fundamental difference in interpretation of probability, the literature on Bayesian approach to statistics usually provides clear explanations of how Bayesian concepts are related to their frequentist counterparts (Bolstad 2004).

A key advantage of the Bayesian approach to statistics is the ability to map the inference process to three key components: prior probability, evidence (or observation), and posterior probability. The posterior probability is a modification of the prior probability based on the observation. The power of the Bayesian approach lies in the applicability of this basic idea to a large range of statistical inference tasks, employing various probability distributions (Gelman et al. 2004). The applicability of updating the prior probability to posterior through observation can be extended to more complex settings, without abandoning the fundamental principles. The complexity of settings in this context refers to both the complexity of domain concepts and their relationships, and the mathematical methods required to represent and perform inference on those domain concepts.

Both in the single random variable and vector of random variables (joint distribution) cases, Bayes' theorem requires either nested summations (in case of discrete distributions) or multiple integrals (in case of continuous distributions) for both normalization constant and posterior distribution. As the number of variables included in the model increases, the complexity of summations and integrations lead to analytically intractable calculations.

Numerical approximation methods can be used for handling these calculations, but their use in high dimensional integrals can be problematic when the inference task at hand introduces hundreds of variables (Sloan 2000). Sampling techniques such as Markov Chain Monte Carlo (MCMC) are used as a means of approximation for these high dimensional integrals (Kloek and Van Dijk 1978). These techniques make increasing computing power more accessible to Bayesian methods through various software implementations, such as WinBugs (Lunn et al. 2000) and JAGS (Plummer 2003). The availability of these tools makes sampling based inference on Bayesian models a common practice today.

As discussed in Chapter 2, Bayesian approach to decision-making has been a widely used approach in clinical decision support even before the availability of sampling based inference, despite the lack of capability to handle complex relations among variables. Therefore, with sampling methods enabling inference on more complex models, one of the major barriers to adoption of Bayesian inference in CDS has become less of a challenge.

The following set of examples introduces some simplified clinical decision-making contexts, demonstrating the use of Bayesian approach before going into details and discussing improvements achieved with the increasing availability of computing power. The examples aim to demonstrate the wide applicability of the Bayesian approach by mapping the fundamental components of Bayesian thinking to clinical decision-making . In keeping with this goal, the examples are intentionally kept simple in terms of the underlying probabilistic concepts.

4.2: Bayesian Reasoning in the Clinical Domain

The clinical scenario that is going to be modelled with the Bayesian approach is a very simple one. In this scenario, there exists a particular disease, with a known prevalence, and there is a test for the disease. The disease either exists or not, and the test produces either a positive outcome (meaning that the disease exists) or a negative one. The test is not a perfect one; it has a certain success rate, so it will generate either false positives or false negatives sometimes. A rewrite of Equation 1 produces the following equation:

$$p(D_{d=true}|T_{t=true}) = \frac{p(T_{t=true}|D_{d=true}).p(D_{d=true})}{p(T_{t=true})}$$

Equation 2: Conditional probability of a disease

In this equation, D represents the disease which may exist, in which case D has the value true. T is the test, which may produce a positive outcome (for the existence of the disease), which is expressed with the value true. A representation of the causal relationship between these clinical concepts is provided in Figure 3.

What Equation 2 does is to express a clinical scenario using Bayesian concepts of probability. The prevalence of the disease is the prior probability of the disease. The test is an observation related to the disease. The probability of the disease given the test outcome is the posterior probability.

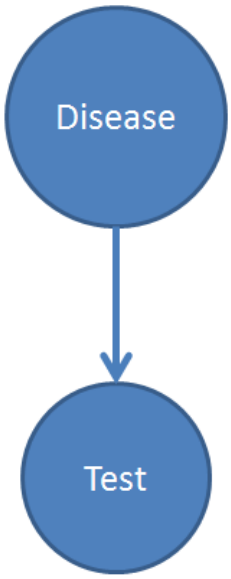


Figure 3: Causal relationship: Disease and Test

As simple as it may be, this example demonstrates some fundamental aspects of Bayesian approach to clinical decision support:

- It specialises a purely mathematical definition and places it into a clinical context. The context free variables in Bayes' theorem in Equation 1 become variables from the clinical domain.
- It expresses the relation between the disease and the test mathematically through the assumption of probabilistic dependence between the test and disease probabilities.
- The fundamental concepts of Bayesian model, prior probability, observation and posterior probability successfully expresses the clinical scenario

Another example of modelling a clinical scenario with a Bayesian approach would be linking a set of diseases to possible symptoms, following the naïve Bayes approach mentioned in Chapter 2, as depicted in Figure 4.

In Figure 4, the disease variable can take a value from a set of limited amount of values, such as tuberculosis, cancer, asthma or pneumonia. Each symptom such as cough, fever, weight loss is given its own random variable in the model.

This probabilistic model demonstrates the limitations of the naïve Bayesian approach. The disease variable assigns probabilities to various diseases, so the diseases are mutually exclusive. Therefore the model cannot support queries such as "what is the probability of a patient having both asthma and pneumonia?" The symptoms are also independent of each other, so this is a rather simple view of the

clinical scenario; a clinical condition such as diabetes could cause problems which themselves could be subject to diagnosis, in which case the existence of these conditions is not independent of each other.

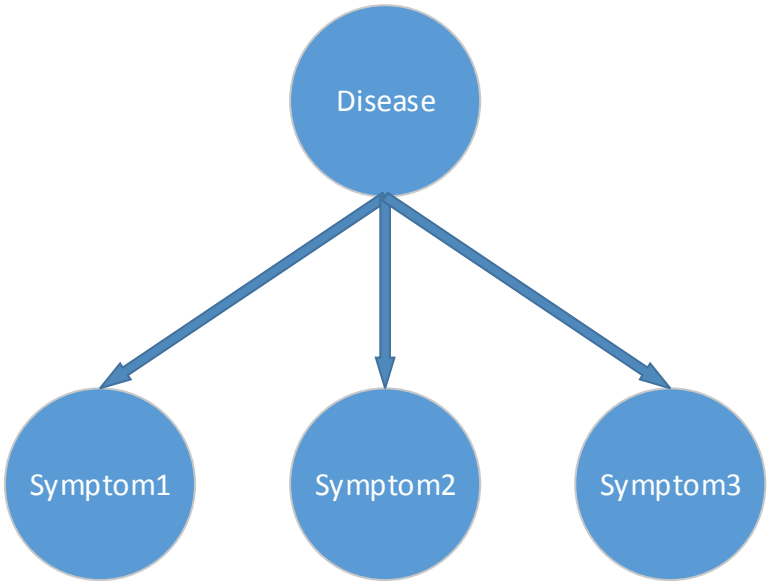


Figure 4: Causal relationship: multiple variables

The reasons for the use of this oversimplified approach to modelling relationships between variables are the computational advantages of this approach and its classification performance. The good classification performance which has led to widespread use of this approach is related to the nature of classification tasks, that is, a correct ranking of the probabilities of outcomes, rather than the precise probabilities is what matters (Hand and Yu 2001)

Even though it performs well (Hand and Yu 2001), one cannot guarantee that the assumption of independence can deliver the best performance in all decision-making tasks. In addition to this, given the large amount of decision-making settings, both in a diagnosis and prognosis context, classification is not the only function that a CDS system must support. Therefore, there has been continuous interest in the research domain for delivering alternatives and extensions to this simple probabilistic method.

One such example, a well-established probabilistic method in decision-making , used for both classification and for estimation of actual values of interest, is regression. Use of various regression methods is a common approach to decision-making in medicine (R. A Greenes 2007). Despite the common frequentist approach, Bayesian methods for regression exist and they allow the use of Bayesian approach beyond the naïve setting (Gelman et al. 2004).

Bayesian regression methods can help deal with the issues introduced by the oversimplification of the naïve approach, and they can also extend the capabilities for decision-making beyond classification. However, their use, especially the building of the probabilistic model from domain concepts is not as straightforward as the naïve model. Therefore, they provide a solution to the representational inaccuracy of the naïve model at the price of less efficient communication with domain experts.

BNs provide another alternative to the simple model in Figure 4 that can deal with the issues introduced by independence assumptions, while keeping the advantage of easily expressing domain concepts and relationships between them. Therefore, they present a powerful alternative to well established probabilistic methods for clinical decision-making .

4.3: Bayesian Networks

The building blocks of BN concept has been introduced by the seminal work of Judea Pearl titled *Probabilistic Reasoning in Intelligent Systems* (Pearl 1988).

A BN encodes a joint probability distribution using a directed acyclic graph (DAG). It consists of nodes, representing random variables, and arcs representing dependency relationships between variables. The expression of dependency relations via graph representation allows factoring of the joint probability distribution, which in turn allows efficient inference methods.

This factoring is built on a specific interpretation of the directed arcs that connect nodes. The directed arcs in a BN represent the dependency relationships of variables they connect and they are used to derive a key property that follows dependency relationships: conditional independence (Koller and Friedman 2009). Conditional independence is a relationship between random variables in which a random variable is independent of another variable, given that the value of a third variable is known. It can be generalized to vectors of variables; hence, it is applicable to the whole of a BN. The precise definition of conditional independence is as follows:

Given three random variables X, Y and Z, X and Y are independent, if value of Z is known, that is:

$$p(X|Z,Y) = p(X|Z)$$

When the value of Z is not known, X and Y are not independent variables.

A BN allows identification of conditional independence relations among variables by using methods that exploit information from its graph based representation. A fundamental concept that helps identify conditional independence relationships of nodes in a BN is D-Separation (Geiger, Verma, and Pearl 1990). D-Separation defines a set of rules that allow identification of the independence relationship of any two nodes of a BN. Algorithms such as Bayes-Ball (Shachter 1998) allow checking the nature of the relationship between nodes using the definition of D-Separation.

The advantage of a BN is that this key mathematical property, which allows efficient probabilistic inference by avoiding unnecessary calculations, is encoded in the graph structure by the domain expert in the form of arcs that connect variables. Domain information is represented in the form of relations among variables by the domain expert, but the resulting graph structure encodes key probabilistic properties without any specific effort for doing so, consequently extending the expressiveness of the naïve Bayesian approach without giving up on the benefits of easily expressing domain concepts.

Despite their capability to overcome some of the issues introduced by the naïve models, BNs have their own limitations. For example, the definition of a BN is built on a DAG, meaning that the interactions among random variables (nodes) in the model cannot create directed loops. This topological constraint of BNs introduces limitations in terms of modelling of clinical concepts such as the inability to model feedback loops among variables since interactions between variables may lead to infinite loops of probability updates in response to observation of values of variables in the model. No calculus has been developed to deal with these loops (F. V. Jensen 2002) .

Some limitations of BNs can be overcome by relaxing topological constraints such as the directed, non-cyclic nature of arcs in a DAG or by using continuous probability distributions as nodes of the BN. These changes introduce new graph topologies and node types, which are studied in depth in the larger context of PGMs (Koller and Friedman 2009). These extended representations can reason on more complex relationships than the ones expressed by DAGs of BNs.

Due to significant size and scope of the research on PGMs, this thesis limits its focus to BNs as the inference mechanism for CDS. Other members of the family of PGMs, are not considered within the scope of this work, but they will be discussed briefly when the context requires to do so, to draw the boundaries of the capabilities of BNs and to identify potential future extensions to CDS mechanism developed in this thesis.

The existence of extensions to BNs should not be interpreted as a sign of their failure to handle decision-making in non-trivial scenarios. The underlying joint probability distribution nature of BNs links them to various significant research domains such as AI, statistics, computer science and machine learning (Korb and Nicholson 2003), (Russell and Norvig 2002), (Bishop 2007). Outcomes of research from these domains have enabled successful use of BNs in many scenarios, CDS being one of them (Pourret, Naïm, and Marcot 2008).

Therefore, BNs have been chosen as the preferred method of probabilistic CDS from the family of PGMs for this thesis, based on the fine balance they offer between inference capability and modelling simplicity.

The key concepts of BNs that provide this fine balance are discussed next, along with relevant extensions.

4.4: Key Concepts of Bayesian Networks

The components of a BN can be classified into two groups: qualitative and quantitative components. The qualitative components are nodes and arcs, describing the structure of the network. The quantitative components are the parameters of the probability distributions, which are represented by the nodes. The qualitative components are frequently referred to as the structure of the network, while the quantitative components are called as the parameters of the network.

Figure 5 provides an example network with a sample probability distribution for random variable RV1. The probability distribution that RV1 belongs to defines two outcomes: True of False, represented by T and F in the table.

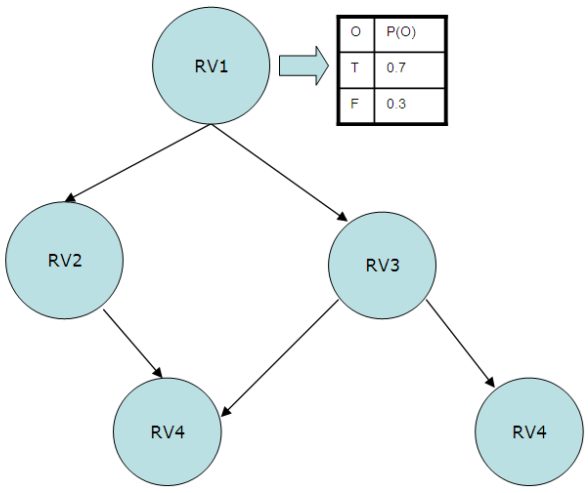


Figure 5: A simple Bayesian Network

In this network, domain concepts are transferred into random variables, and arcs that connect the variables encode dependency relationships.

The use of nodes and arcs make it easy for a domain expert to build probabilistic models in a particular domain without substantial knowledge of probabilistic concepts. However, development of a BN is not a trivial operation that consists of merely representing expert knowledge as a DAG. It is a complex knowledge engineering process that requires various tasks to be performed (Pradhan et al. 1994), (Julia Flores et al. 2011). These tasks can be classified into two broad groups:

- Tasks related to structure of the network, such as learning the structure from collected data, which is referred to as structure learning (Buntine 1996), eliciting structure from domain experts and testing claimed relations among variables using collected data.
- Tasks related to parameters of the network, such as learning parameters of probability distributions from data (Neapolitan 2004), which is referred to as parameter learning, eliciting priors from data or experts or both (Julia Flores et al. 2011), performing inference based on observation and performing simulations.

The knowledge engineering process based on these tasks can be a combination of both human input and algorithmic discovery of network components. A human expert such as a clinician could easily define the structure of a BN based on concepts from the medical domain without requiring any clinical data but defining the parameters of the BN requires assigning values to conditional probabilities: a task computers perform significantly better than humans via discovering the parameters from data (Leaper et al. 1972). In case of BNs with a large number of variables, automatic learning of structure of BNs (Neapolitan 2004), (Koller and Friedman 2009) can help human experts by providing an initial BN for further improvement.

Defining both the structure and parameters of a BN produces a probabilistic model, which supports inference based on observations of the values of the probabilistic variables represented by the nodes of the BN. The actual use of BNs for decision-making is based on this operation.

Inference on a BN is the calculation of updated probabilities of the random variables of a joint probability distribution in response to an observation. The observation, also called the evidence, is the observed value of one or more nodes

of the BN and it is used to calculate updated probabilities of the remaining nodes in the network.

Figure 6 contains an example network, which can be used for clinical decision-making via performing inference. This network is provided as an example with the software GENIE (Druzdzel 1999), a freely available tool for developing and performing inference on BNs. The BN is originally from (Lauritzen and Spiegelhalter 1988).

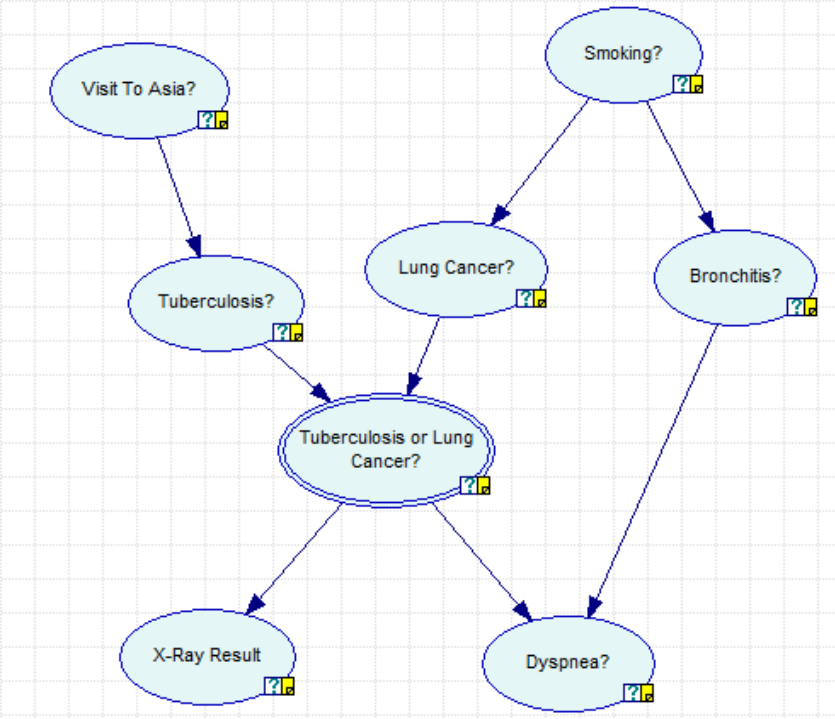


Figure 6: BN for clinical diagnosis

The arcs in the BN in Figure 6 represent the interactions between variables. A visit to Asia has an effect on the probability of someone having Tuberculosis, smoking has an effect on both Lung Cancer and Bronchitis, and so on. Figure 7 shows both the structure and the parameters of the network using GENIE's support for displaying BNs in different formats.

Figure 7 shows that the BN is describing a joint probability distribution where no observation has been performed. In this state, the nodes represent the prior probabilities of outcomes. When an observation is performed, that is, the value of one of the variables is observed and therefore known for certain, the other variables are assigned updated probabilities.

Figure 8 shows new values of random variables updated in response to a change in the “Smoking” variable. Inference based on this observation updates probabilities of some variables. This new piece of information, that a person is a smoker increases the probability of “Lung Cancer”.

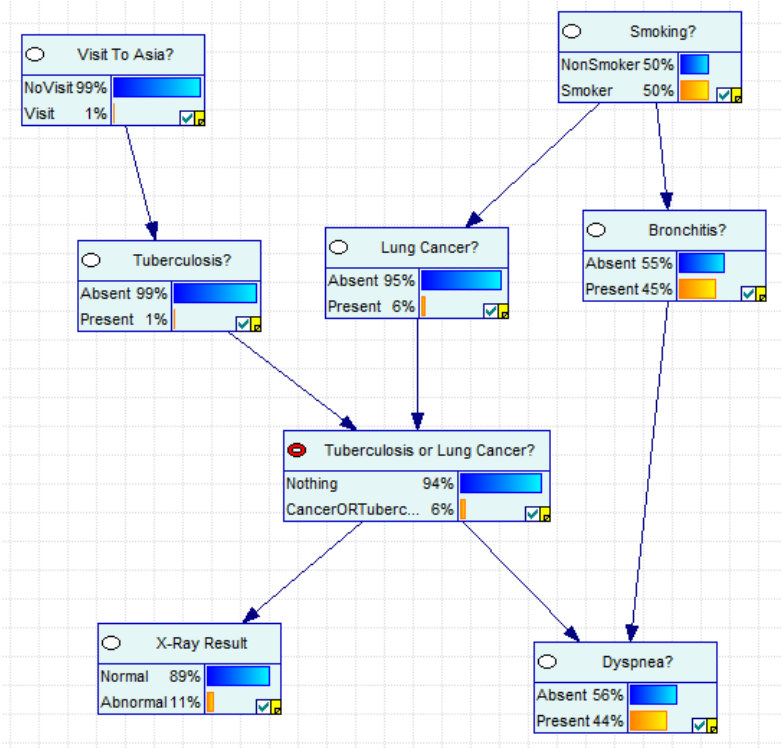


Figure 7: BN with node probabilities

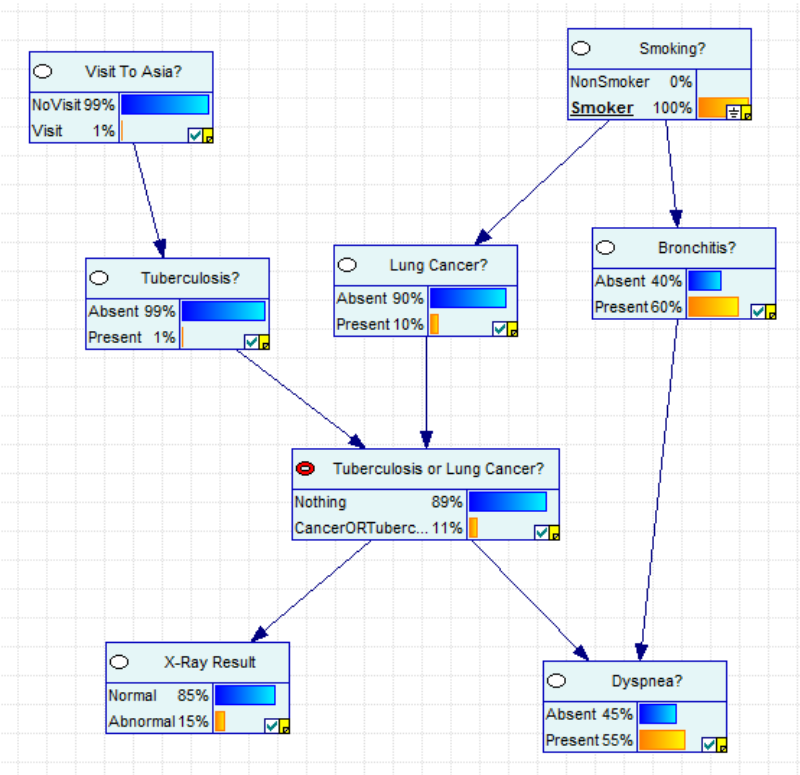


Figure 8: BN with an observation

An important point to note in Figure 8 is that the probability of Tuberculosis has not changed, and neither has Visit To Asia's. Intuitively, this makes sense. Finding out that a person smokes should not have any relation to that person's recent travel history. The underlying inference method that performs this update is based on conditional independence. The conditional independence relations between variables stop the effect of observation of "Smoking" variable's value from propagating to Tuberculosis and Visit To Asia.

The Tuberculosis or Lung Cancer is a deterministic node. This node allows definition of two states, "Nothing" and "CancerOrTuberculosis". The reason this node is called deterministic is that its outputs are defined based on rules, which map values of its parent nodes to its outputs. This node can be thought of a transformation node on the network, which will generate the value "Nothing" when both Tuberculosis and Lung Cancer nodes have the value Absent. This rule based generation of values based on outputs of other nodes allows the BN to express new semantics which may not have been considered and recorded as a clinical variable during clinical care. This deterministic node is an example of extending the BN formalism, which helps the network represent a larger set of domain concepts.

The updating of probabilities (inference) is an important topic in PGM research in general and BN research in particular. Inference methods for updating the probabilities of a BN can be classified into two categories as exact inference and approximate inference. Choosing an inference method for a BN is a case specific task that requires an awareness of the relative benefits and complications of the method chosen.

Exact inference in Bayesian networks calculates probabilities without any loss of precision. Some of the most common exact inference algorithms have their roots in the method introduced by (Pearl 1986). Pearl's approach establishes a method to propagate information within the graph structure of BN, where information represents the observation of the value of a random variable. The propagation of this information corresponds to updating other random variables in the network based on the modelled dependence relationships among the variables. Various approaches have been developed on this idea of "message propagation" which updates local probabilities of nodes based on observations (Lepar 1998).

Exact inference methods have constraints related to topology of the graph and probability distributions of nodes. The network must be a DAG. Some exact inference algorithms can be used with BNs but they can not be used for some extensions of BNs that can include continuous probability distributions in their

nodes. This problem can be dealt with approximating continuous distributions via discrete ones, but in this case the number of intervals chosen for discretization has an effect on the inference performance of the BN, and the inference is no longer exact due to discrete approximation.

In exact inference on discrete variables, each variable has a conditional probability table, including all possible combinations of all parent variables. The BN in Figure 8 provides an example of this setting. The number of entries in conditional probability tables is determined by the number of parents of a node as well as the number of intervals of discrete distributions represented by these nodes. When a large number of parents and a large number of intervals for discrete distributions exists simultaneously, the size of the conditional probability tables can grow large. Inference can become intractable in this setting.

Using an extension of the BN such as conditional Gaussian BNs (Shenoy 2006) may solve this problem. This extension to BNs uses Gaussian distribution to represent continuous variables and allows discrete and continuous variables to co-exist subject to some structural limitations such as continuous nodes not being allowed to have discrete children. This extension allows representing domain concepts that include continuous variables (such as age, temperature, weight etc.) without a discretisation based approximation and without losing the capability for exact inference (Lauritzen and Jensen 2001).

Even though some extensions of BNs allow use of continuous variables and support exact inference, these extensions still have their limitations such as using a Gaussian distribution to represent a continuous variable which may not be realistic representation for all the variables. Further extensions to BNs may include other distributions (Moral, Rumí, and Salmerón 2001), (Krauthausen and Hanebeck 2010) fewer topological constraints (Koller, Lerner, and Angelov 1999), (Schrempf and Hanebeck 2004) and more compact representations such as decision trees (Su and Zhang 2005). These extensions increase the expressive power of BNs, but the resulting joint probability distributions may not allow exact inference anymore. In this case, approximate inference methods may be used.

Approximate inference methods allow keeping the benefits of more expressive extensions to BNs without completely losing the capability to perform inference. Most widely used approximate inference methods in BNs is based on sampling algorithms such as Gibbs sampling (Pearl 1988), (Neal 1993). Less frequently used approximation methods also exist, such as variational approximation (Murphy 1999; Jaakkola and Jordan 1997), but they are not

discussed in depth in this thesis, mostly due to significantly larger amount of literature and tooling available to sampling methods.

Sampling methods are built on the idea of drawing samples from the posterior distribution of the joint probability distribution encoded by the BN and analysing the characteristics of the posterior distribution based on these samples. This approach has wide applicability to a large set of calculations (Gilks, Richardson, and Spiegelhalter 1996), inference in a BN is only one of them.

Approximate inference via sampling can be used for both BNs and their extensions (Langseth et al. 2009), (Koller and Friedman 2009), (Brewer, Aitken, and Talbot 1996). Generic sampling tools such as WinBUGS (Lunn et al. 2000) and JAGS (Plummer 2003) provide the capability to define graphical models using a number of probability distributions for the nodes. These tools support a domain specific language to build a probabilistic model along with features for analysis of the sampled data and visualisation of graphical models.

The generic approach of these sampling frameworks for graphical models bears a resemblance to openEHR's approach for building clinical models. A number of probability distributions can be brought together in a graphical model in an infinite number of combinations to model domain concepts and relationships between them.

It should be noted that despite its flexibility, these sampling frameworks still have their limitations. They allow the use of a pre-defined set of distributions, which can be used to build graphical models, including BNs, within the capabilities of the domain specific languages they support. These constraints attempt to guarantee that the sampling operation can be performed, though sampling methods may not always converge to stable results. The limitations of these sampling frameworks is not mathematical; they can be extended with the outcomes of research (Wabersich and Vandekerckhove 2013). The current approach of these tools is to set a good balance between rather stable results from sampling based on a set of probability distributions and the ways they can be used together in a graphical model to express a decision-making context, i.e. their expressiveness.

Therefore, even though the currently available, well known tools do not support all extensions of BNs, this limitation can be overcome via custom implementations of more advanced approaches or extensions to existing tools for both modelling and inference. These improvements constitute an important line of future research for better CDS beyond the current scope of this thesis. Such future research can be built on outcomes of numerous studies that improve on the existing methods adopted by the current tools.

Studies that focus on a large variety of topics such as improving the performance of sampling when unlikely evidence is encountered (J. Cheng and Druzdzel 2000), (Yuan and Druzdzel 2003), improving performance of discretisation (Kozlov and Koller 1997), (Di Tomaso and Baldwin 2008), improving sampling performance for very large BNs (C. S. Jensen and Kong) and parallel inference (Vasanth Krishna Namasivayam, Pathak, and Prasanna) are all examples of research that can be used to improve a BN based CDS approach.

The family of PGMs and computation tools stemming from BNs and leading to these potential future extensions spans a vast domain for research. This thesis identifies the basic, yet powerful definition of BNs as introduced by (Pearl 1988) as the central point of this domain as well as a knowledge engineering methodology (E. H Shortliffe, Buchanan, and Feigenbaum 1979) that can be extended to more complex and capable representations and inference methods depending on the decision-making or prediction tasks at hand. BNs allow the same principles of knowledge engineering to be used in many scenarios with multiple options for adjusting the balance between the expressiveness of the domain model and the performance and accuracy of inference. However, BNs have not been identified as the CDS mechanism for this thesis based on these advantages alone. The findings related to overall advantages of BNs is complemented by a rather specific appraisal based on a review of the uses of BNs in medicine, as provided in the next section

4.5: Bayesian Networks in Medicine

The integration of BNs into the domain of medicine through medical informatics creates a context in which many variables from different disciplines interact. Different types of clinical data and processes from medicine, as well as many components of information technology and concepts of BNs and their extensions are all connected in such a context.

This large and complex set of relations makes it impossible to suggest that BNs can improve outcomes in every single CDS scenario over alternative methods. However, a review of existing studies that explore the use of BNs in various settings in medicine would help evaluate their performance and potential as a generic, widely applicable knowledge engineering and inference framework for CDS. To this end, a literature review was performed using the facilities provided by www.sciencedirect.com. The search facility of this research repository returned 6423 articles in response to the search phrase “Bayesian networks clinical” (in “all fields” field in the search form).

A set of studies relevant to decision-making based on BNs in medicine has been identified through a detailed evaluation of the first 250 members of this result set, along with 308 results returned from the same search performed for the publication “Artificial Intelligence In Medicine”. These studies cover BNs as well as their various extensions from the family of PGMs. The review has been performed with the goal of answering the following questions:

- Do BNs provide a clinical modelling formalism that allows clinicians to define domain concepts in a large number of clinical domains?
- Do BNs help domain experts define clinical scenarios without having to deal with the underlying mathematical models? How expressive BNs are for describing the various components of clinical decision-making ?
- Can BNs provide feedback about the reasoning process so that clinicians can interpret the outcomes?
- Can BNs perform inference at least as well as the more established methods of probabilistic modelling in CDS domain?

These questions address the critical requirements for using probabilistic methods in CDS, and positive answers to them, provided by the findings of existing studies, provide evidence for the feasibility of using BNs for CDS. The following sections provide the findings of the review, performed mainly with a focus on answering the questions above, treating mathematical and computational aspects of the studies to be of secondary concerns.

4.5.1: Bayesian Networks as CDS Models

openEHR’s capability to model concepts from a multitude of clinical domains requires that, for a CDS framework to be consistently integrated to openEHR, it must also be able to represent these concepts. Therefore, a BN based CDS approach must be applicable across different clinical domains to support openEHR integration.

An evaluation of the boundaries of the expressiveness of BNs for all CDS scenarios would not be feasible for this thesis. However, successful use of BNs in a variety of clinical decision-making scenarios from different clinical domains indicate sufficient expressiveness. The review shows that BNs are used in a variety of clinical domains, in scenarios such as choosing antibiotics for treating severe infections (Andreassen et al. 1999), modelling clinical performance of pancreatic cancer patients (Hayward et al. 2010), diabetes monitoring (Riva and Bellazzi

1996), insulin therapy management (Andreassen 1992), diagnosing the stage of oesophageal cancer (van der Gaag et al. 2002) and diagnosis of pneumonia cases in ICU (Lucas et al. 2000).

Further uses of BNs include diagnosing heart problems (Long, Fraser, and Naimi 1997) and nasopharyngeal cancer (Galán et al. 2002), analysis of adverse drug reactions (Cowell et al. 1991), estimating survival in malignant skin melanoma (Sierra and Larrañaga 1998), neuromuscular diagnosis (Xiang et al. 1993), ovarian tumour classification (Antal et al. 2003), (Antal et al. 2004), analysis of tuberculosis epidemiology (Getoor et al. 2004), diagnosing pyloric stenosis (Alvarez, Poelstra, and Burd 2006), classifying SPECT images (Sacha, Goodenday, and Cios 2002), predicting blood glucose concentration (Ramoni et al. 1995), diagnosis of breast cancer (Kahn Jr et al. 1997), analysis of dynamics of organ failure in intensive care unit (Peelen et al. 2010) and monitoring laboratory errors (Doctor and Strylewicz 2010).

Even this small scale literature review shows that BN approach to CDS can address clinical scenarios with considerable variety of concepts from domains such as cardiology (Díez et al. 1997), (Verduijn et al. 2007), psychiatry (Chevrolat et al. 1998), neurology (R. Chen et al. 2012), ophthalmology (Tucker et al. 2005), urology (Montironi et al. 1996), (Montironi et al. 2002) and oncology (X.-H. Wang et al. 1999), (Smith et al. 2009).

Based on the substantial variety of both the clinical cases and the clinical domains these cases belong to, the expressiveness of BNs as a CDS modelling formalism, using their extensions when necessary, is deemed sufficient for expressing CDS concepts for the diverse set of clinical applications that can be developed based on openEHR.

However, this sufficiency does not necessarily imply that BNs provide an easy to use knowledge elicitation tool. The degree of convenience with which this expressiveness can be put to use for building CDS models must be assessed.

4.5.2: Communication with Domain Experts

A significant difficulty of the probabilistic approach to CDS is that the underlying mechanism for inference has a highly abstract nature. The use of statistical terms for expressing relationships between clinical concepts is not a convenient language for extracting knowledge from domain experts. The graph based representation of BNs can improve the efficiency of this process.

The studies that discuss the use of input from domain experts for defining the structure of BNs partially support this claim while pointing at potential challenges of the process.

Clinicians can use the graph based representation of BNs to encode domain concepts and their relationships with the help of a knowledge engineer, who explains the options available to them in terms of relationships between BN nodes and defining clinical concepts (Onisko 2003)

The advantages of this approach for building an expert system is described as follows by (Gappa, Puppe, and Schewe 1993):

"The most important precondition for knowledge acquisition systems by experts is that the underlying model is sufficiently tailored to the domain and/or its problem solving strategy, so that the expert can easily get acquainted to it. For this, well-chosen graphical knowledge representations can greatly support the understandability of the knowledge model and thus the model building of the expert."

The emphasis of (Gappa, Puppe, and Schewe 1993) on the importance of predefined concepts for a knowledge model bears resemblance to the approach adopted by openEHR:

"The usefulness and efficiency of a knowledge acquisition tool crucially depends on the adequacy of the predefined concepts of its underlying knowledge model and therefore it is important to ask which of the concepts may not be that easy to understand and thus are hardly or not at all instantiated."

Attempts to improve the process for defining the structure of graphical models include automated interviews (Luciani and Stefanini 2012) and joining human input with automated learning of structure from data in a knowledge engineering workflow (Julia Flores et al. 2011).

The availability of software tools such as HUGIN (Andersen et al. 1989), WinBUGS (Lunn et al. 2000) and SMILE (Druzdzel 1999),which help follow these knowledge engineering practices, support the claim that BNs present a valid knowledge representation option for automated reasoning (Long 2001).

The similarities and relationships between BNs along with their extensions and some well-established knowledge engineering methods such as OWL, Web Ontology Language (McGuinness and Van Harmelen 2004), is both an opportunity to improve current BN implementations and a probable topic for future research. For example, using existing ontologies for network construction (Fenz 2012) allows previously encoded knowledge to be used, which is a method suggested more than 20 years ago (Gappa, Puppe, and Schewe 1993).

Expert input from clinicians or the use of domain ontologies can lead to requirements that cannot be expressed with a BN that is based on the fundamental definition of (Pearl 1988) which has limitations in terms of both network topology and probability distributions that can be expressed as network nodes.

Therefore, some concepts such as temporal aspects of a clinical case or making decisions based on the results of inference require capabilities beyond the fundamental definition of BNs used by this thesis.

Temporal aspects of a clinical case are implicitly included in prognostic decision-making scenarios. These aspects may be evident, such as in the case of estimating the value of a clinical variable given a fixed length of time, for example in estimating reoccurrence of cancer in a five year period (Gevaert et al. 2006). These inference tasks can be performed without any explicit representation of temporal aspects. Inference requirements with more complex temporal aspects are represented by Dynamic Bayesian Networks (Murphy 2002).

The review has identified uses of BNs that address temporal aspects of diagnosis or prognosis for tasks such as blood glucose time series analysis (Riva and Bellazzi 1996), pneumonia treatment at the intensive care unit (Lucas et al. 2000), reasoning about cardiovascular disorders with temporal relations (Long, Fraser, and Naimi 1997), modelling the spread of cancer (Galán et al. 2002), and analysis of organ failures (Peelen et al. 2010) and ventilator-associated pneumonia (Charitos et al. 2009) in the intensive care unit.

The availability of studies that use BNs for knowledge engineering, complemented by the possibility of using existing knowledge encoded in other forms makes BNs a viable option for user friendly development of CDS models. The studies show that BNs can model many key components of the decision-making context using the same consistent representation.

4.5.3: Explaining the Reasoning Process

Understanding the reasoning used by the CDS mechanism is a crucial advantage for a clinician. No matter how successful a particular CDS approach is, a black box implementation makes it impossible for a clinician to follow the reasoning process. Including feedback from such a system in the care process becomes particularly problematic if the feedback conflicts with the clinician's opinion.

Rule based approaches to CDS make it easy to deal with this potential problem by providing access to rules that were used in inference. For probabilistic approaches, the mechanics of providing this functionality is more complicated due

to underlying probabilistic reasoning mechanism. Despite this relative difficulty, explanation of reasoning in BNs has been an active field of research.

Capabilities such as explaining the reasoning of a BN (Haddawy, Jacobson, and Kahn Jr. 1997),(Elvira 2002),(C. Lacave and Díez 2002),(C. Lacave, Luque, and Díez 2007),(Carmen Lacave, Oniśko, and Díez 2006) , generating verbal explanations from BNs (Druzdzel 1996), and graphical explanation of reasoning (Madigan, Mosurski, and Almond 1997) allows clinicians to follow reasoning process.

Despite the availability of explanation methods for BNs, bridging the gap between the mathematical reasoning and clinical explanations is not as easy as rule based systems, especially when approximate inference methods are used.

4.5.4: Inference Performance of Bayesian Networks

Even though this thesis places a strong emphasis on high-level, graphical representation capabilities of BNs and the advantages they provide over other probabilistic methods, these traits of BNs alone are not sufficient to suggest that they perform better than the alternatives for classification or prediction tasks. Therefore, BNs should provide performance at least on par with well established probabilistic methods for these tasks. Evidence of such performance complements other advantages of BNs, making them a good overall option for CDS.

Studies that compare well established probabilistic methods such as logistic regression with BNs usually report similar performance for classification and prediction tasks. One such example is the prediction of clinical performance of pancreatic cancer patients (Hayward et al. 2010) in which BNs perform better in various predictive tasks compared to logistic and linear regression, with the exception of predicting tumour size. BNs are considered as a successful replacement for certainty factors (Heckerman and Shortliffe 1992) as well as an improvement over naïve Bayesian model (Sakellaropoulos and Nikiforidis 2000) based on more precise definition of dependencies between variables.

4.5.5: Summary of Findings

The findings of the literature review show a level of use of BNs that sufficiently support their capability to address various requirements of a widely applicable CDS framework. However, the integration between the underlying sources of clinical data and BNs is not the focus of these studies. Consequently, the formal representation of domain concepts in information systems is not included in

the scope of BN based CDS research, arguably with the exception of using OWL (McGuinness and Van Harmelen 2004), which provides a formal representation of domain concepts.

BN based CDS in an openEHR context introduces a new approach, which formalises the data access aspect of BN implementation that is of secondary importance to these studies. In this new approach, openEHR specifications provide a well defined set of capabilities and services for computable health across multiple systems by generalising underlying data sources to an openEHR representation as discussed in the following section.

4.6: Integrating openEHR Methodology with Bayesian Networks

The expected benefit of introducing openEHR as the underlying clinical data representation for BN modelling and implementation is a significant contribution to the solution of the isolation problem outlined in Section 2.2. The benefits of high level representation of domain concepts provided by both openEHR and BN are similar. Both approaches allow complex operations to be performed on domain concepts based on this representation. The particulars of this similarity are the basis of a logical architecture for their integration.

A major barrier to developing better clinical information systems is the incomplete or incorrect representation of requirements. Clinicians mostly provide software requirements in their own terms through analysts and developers. Requirements are transformed into software by developers, and individual developers may understand the same requirements in different ways. As a result, the quality of the representation of domain concepts in software is dependent on the level of understanding of the domain the developers possess. As the domain gets more complicated, precisely expressing domain concepts in software becomes harder, leading to less accurate representations. Clinicians can only indirectly influence the content and behaviour of information systems through developers, since only developers are capable of creating computable concepts.

openEHR allows domain experts to create computable models based on clinical concepts as explained in Section 3.1. Therefore, clinicians drive the clinical information system development process through domain models without going into details of the software development domain. This approach eliminates the inaccuracy that stems from the rather traditional software requirement analysis practices.

BNs provide a similar improvement for building probabilistic models, allowing domain experts to define the statistical representation of domain concepts without tackling complex probabilistic terms such as joint probability distributions or conditional independence. Furthermore, it is possible to use a combination of a small number of probability distributions to represent an arbitrary number of domain concepts, similar to openEHR's reference model. Therefore, both approaches allow clinicians to extend their control to non-clinical domains based on similar principles.

An important research question that originates from this observation is therefore to what extent these similarities could support an integration between openEHR specifications and BN based CDS. Two primary software implementations are required in order to answer this question via the experimental approach adopted by this thesis: an implementation of the openEHR specifications sufficient for the chosen scope, and an implementation of a BN based inference engine. The relationships between these two software components are first identified and discussed based on a logical architecture, partial implementation of which is used as the basis of hands on experiments.

4.7 Logical Architecture for openEHR and Bayesian Networks Integration

The purpose of the logical architecture for openEHR and BN integration is to identify the nature of the relationships between the components of the integration. These relationships define how key features of both openEHR methodology for clinical application development and BN based CDS can be connected. Consequently, they provide the architectural guidelines for actual software development to achieve this integration. However, these guidelines still require an appraisal for feasibility of implementation. A relationship defined in the logical architecture may suffer from performance problems or laborious efforts such as large scale data mapping tasks. Therefore, testing the assumptions of the logical architecture through software implementation is the method of appraisal adopted in this thesis as discussed in Chapters 5, 6 and 9.

Studies that discuss the use of CDS to improve clinical care frequently refer to EHR implementations as the platform that hosts CDS functionality. (Kuperman et al. 2007), (Bates and Gawande 2003), (Kalra and Ingram 2006). EHR specifications and information systems based on them provide an answer to a well-known problem in the decision support domain: access to data.

The integration of computer interpretable guidelines to HL7 Reference Information Model (Beeler 1998, 7) in (Peleg et al. 2001) provides a healthcare specific example of standards based data access based on GLIF (Boxwala et al. 2004), similar to use of Arden Syntax for development of a data and query model (Jenders, Corman, and Dasgupta 2003). These studies show that the integration between CDS and EHR concepts has problems such as incompatibilities between the HL7 RIM and Arden syntax (Peleg et al. 2001). Developing solutions for impedance mismatch of EHR systems and computerised guidelines have been suggested as a method of overcoming these integration problems (Schadow, Russler, and McDonald 2001). The problems identified by these HL7 focused studies show that computability of EHR and CDS concepts do not guarantee a problem free integration.

The logical architecture is the starting point of an analysis similar to these studies, with three goals:

- To introduce probabilistic AI based decision support into openEHR
- To identify problems in the process
- To develop and offer solutions to identified problems

Figure 9 provides the main components of the logical architecture.

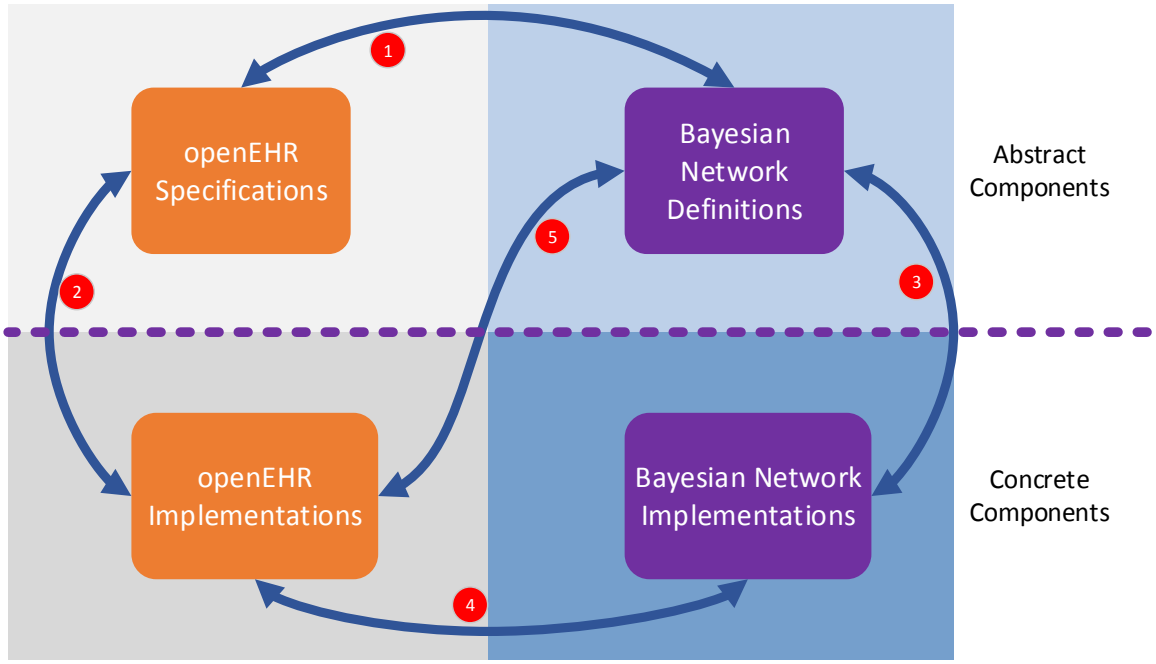


Figure 9: Logical architecture for openEHR – Bayesian Network integration

The elements of the diagram in Figure 9 provide components of openEHR and BN implementations. The components are classified into two groups as abstract and concrete.

openEHR specifications and BN definitions are the abstract components which are independent of implementation aspects such as programming languages or algorithms for exact or approximate inference. These components support a knowledge engineering approach by allowing domain experts define domain concepts.

Concrete components represent the various software implementations of both openEHR specifications and BN definitions. Numbered connections in the logical architecture in Figure 9 represent possible relationships in the context of integration. The following is a discussion of these relationships, describing the scope of research and implementation implied by them, referring to connection numbers in the logical architecture.

1) *Integrating openEHR's modelling approach with Bayesian Network Definitions*

   openEHR's domain models and their underlying formalism can support the implementation of multiple aspects of an information system. Various openEHR implementations already use the openEHR archetypes (Beale and Heard 2007a) for a number of implementation tasks such as data validation and persistence or user interface generation. Reusing openEHR archetypes as a repository of clinical domain concepts extends the use of openEHR models to CDS development and has the potential to improve the construction of BNs , allowing clinicians to identify domain concepts easily.

2) *Implementation of the openEHR specifications*

   An actual implementation of openEHR specifications is required to observe both the benefits of openEHR methodology and its problems in the context of clinical information systems implementation and CDS integration. Aspects of implementation such as data access performance and scalability, which are affected by the underlying architecture and technology platform, must be observed as well since they are at least partial determinants of performance in every operation on openEHR data. Therefore, a testbed is crucial for an in depth analysis of both the openEHR methodology and aspects of its implementation in a CDS context.

3) Bayesian Network implementation

The efficacy of the integration between openEHR specifications and BN based CDS cannot be observed without a BN implementation. Similar to openEHR methodology, a domain concept can be represented in multiple ways using BNs. Inference on a BN can be performed via different algorithms, as discussed in Section 4.3, and these algorithms can be implemented using a number of platforms. Even though the availability of various options for BN implementations is acknowledged in the logical architecture, time limit for this thesis allows only a subset of these options to be used through freely available and open source tools. However, both comparing the performance of different inference algorithms as well as exploring the scalability of these algorithms, especially through parallelisation (Vasanth Krishna Namasivayam, Pathak, and Prasanna), (X.-L. Wu et al. 2012), (Neiswanger, Wang, and Xing 2013) are key future research topics identified by the logical architecture.

4) Integrating Bayesian Network implementations with openEHR implementations

The integration described by the logical architecture defines openEHR implementations as the source of clinical data. Since BN implementations need to access clinical data for inference, methods of data access for this specific goal must be developed. Accessing clinical data for purposes such as learning the parameters of a network is a significantly different scenario from an openEHR implementation point of view than patient centric data access, which is the case for clinical information systems built on openEHR. The difference is due to increase in the volume of data that is used in the former case. Technology independent, widely applicable methods for large volume data access for openEHR implementations need to be developed considering the fact that both ends of the relationship in the logical architecture can be built on different technologies. The transformations from the object oriented (Meyer 1988) openEHR data types to rather primitive data types required by BN inference algorithms are also a key part of this relationship. The findings of research focusing on this relationship have the potential to introduce changes and additions to openEHR specifications for large scale clinical data processing.

5) *Integrating openEHR implementations with Bayesian network definitions*

openEHR implementations, which provide access to clinical data, complement openEHR clinical models for defining BNs by contributing to both structure and parameter learning.

Automated structure learning algorithms, which make use of clinical data that is provided by openEHR implementations, can create BNs with an initial set of variables and relationships, which can be improved upon by the domain experts. Especially when the number of clinical variables is large, this initial network can shorten structure definition process.

If the BN structure is specified with full or partial domain expert input, the clinical data can be used to validate the dependency relationships asserted by the structure, or to discover additional, unspecified ones.

The parameters of a BN can be learned from previously collected data, based on the BN structure. This approach enables domain experts to define outcomes for clinical variables of interest, without having to specify probabilities of these outcomes, i.e. the parameters of the network.

4.8: Summary

Bayesian approach to handling uncertainty has improved significantly from the days of naïve Bayesian classifiers. Due to increased processing power, more advanced methods, such as BNs have matured to the point of being a reliable option for many clinical decision-making tasks. The research about probabilistic graphical models, with BNs being the dominant type of model, has delivered satisfactory and promising outcomes in many clinical domains.

Based on the suggestion that openEHR is a mature, modern representative of the EHR concept, the integration of openEHR and BNs for CDS is chosen as the focus of this thesis, with the goal of exploring the sufficiency of EHR concepts in supporting better CDS via this integration.

The suggested method of research is a set of experiments performed on a testbed, which is an implementation of a logical architecture for integration of openEHR and BN concepts both at the specification and implementation levels. The results of the experiments form the basis of a software architecture for openEHR implementation as well as suggestions for changes and additions to openEHR specifications to support BN based CDS.

To identify how the use of the openEHR approach to modelling and processing clinical data affects BN based CDS implementation process, an isolated, rather basic BN has been developed and used for classification initially. The discussion of this scenario provided in the next chapter aims to identify characteristics of the BN approach without an underlying openEHR platform. How these characteristics change in the context of openEHR integration is analysed in depth in the chapters that follow after.

Chapter 5: A Pilot Bayesian Network Implementation Experiment Using Thyroid Disease Data

The BN approach to managing uncertainty provides multiple options for building domain models and inference on them. This chapter examines the process of developing and using a BN. There is no assumption of clinical information system integration and clinical data is provided in the form of a flat text file.

The BN is designed as a classifier which uses clinical data of patients with thyroid problems. Well known software tools for BNs are used, to achieve reliable results and to complete experiments in an acceptable time frame. The primary focus of the experiment is on issues such as data quality and problems and barriers related to it, what modelling and inference options are available given the domain concepts, and what are the advantages and disadvantages of these options. The actual classification performance of the BNs is of secondary importance.

5.1 The Setting of the Experiment

The BN approach is used for diagnosis of thyroid disorders. This clinical domain was selected based on the availability of clinical data from the UCI machine learning repository (Blake and Merz 1998) which provides access to various data sets. The experiment develops a BN with discrete conditional probability tables. Steps for building the network such as pre-processing of data, learning their structures and parameters were performed. The BN was then used for classification to diagnose thyroid disorders.

5.2 Processing the Raw Data

The thyroid data that was used in the experiment had data quality problems such as outliers and missing observations. These problems were identified via an analysis of data using the open source WEKA machine learning workbench (Hall et al. 2009). WEKA was used to remove the data instances with one or more missing values for clinical variables used in the BN, resulting in a data set of 5635 rows. After missing values were removed, the data set was processed with open source statistical language and framework: R (R Development Core Team 2008) to remove outliers. The final step in processing raw data was transforming the diagnosis outcomes to consistent values using features of freely available text editors.

The pre-processing phase of thyroid data consisted of typical examples of the trivial, yet time consuming tasks that are frequently performed when data exported from multiple sources needs to be analysed. Even though there were many freely available tools such as WEKA or R, the process was time consuming, and sometimes required dealing with the data interoperability problems introduced by the tools themselves, such as different interpretations of structure of data in comma separated files.

The efforts required to pre-process data even in this small scale experiment demonstrated the advantages of standards driven clinical information systems approaches, which can significantly decrease the pre-processing time required for data analysis by allowing automatic data integration from multiple sources in addition to performing data validation during data entry.

5.3 Learning the Network Structure

The relationships between thyroid disorder related concepts for the BN were first learned via automatic structure learning, followed by manual modifications. The structure of the BN was learned via GENIE (Druzdzel 1999), a freely available BN development and inference environment. GENIE's built in discretisation algorithm was used on continuous variables prior to learning the network structure. The following variables from the data set, identified via a limited literature review for thyroid disorders, were used as the nodes of the BN:

- Age: The age of the patient
- T4(thyroxine): The measurement of the major thyroid hormone secreted by the thyroid gland
- T3(triiodothyronine): The measurement of the T3 hormone, which is the result of the conversion of T4.
- FTI(free T4 index): The form of T4 in the blood, which can exert effects on target tissues.
- T4U: Thyroxine resine uptake test results
- Diagnosis: The diagnosis in the data set.

Learning the structure of the BN using the PC algorithm (Spirtes, Glymour, and Scheines 2000) implementation in GENIE with gradually increasing amounts of data shows how the performance of structure learning benefits from more data.

The automatically learned BN structure in Figure 10, based on a data set of 1000 elements, connects all predictor variables to the response variable

(DIAGNOSIS). The connections from predictor variables to response variable are directional, and the direction of the connections is correct. However, the structure in Figure 10 has bidirectional connections between predictor variables in addition to the non directional connection between T3 and FTI nodes.

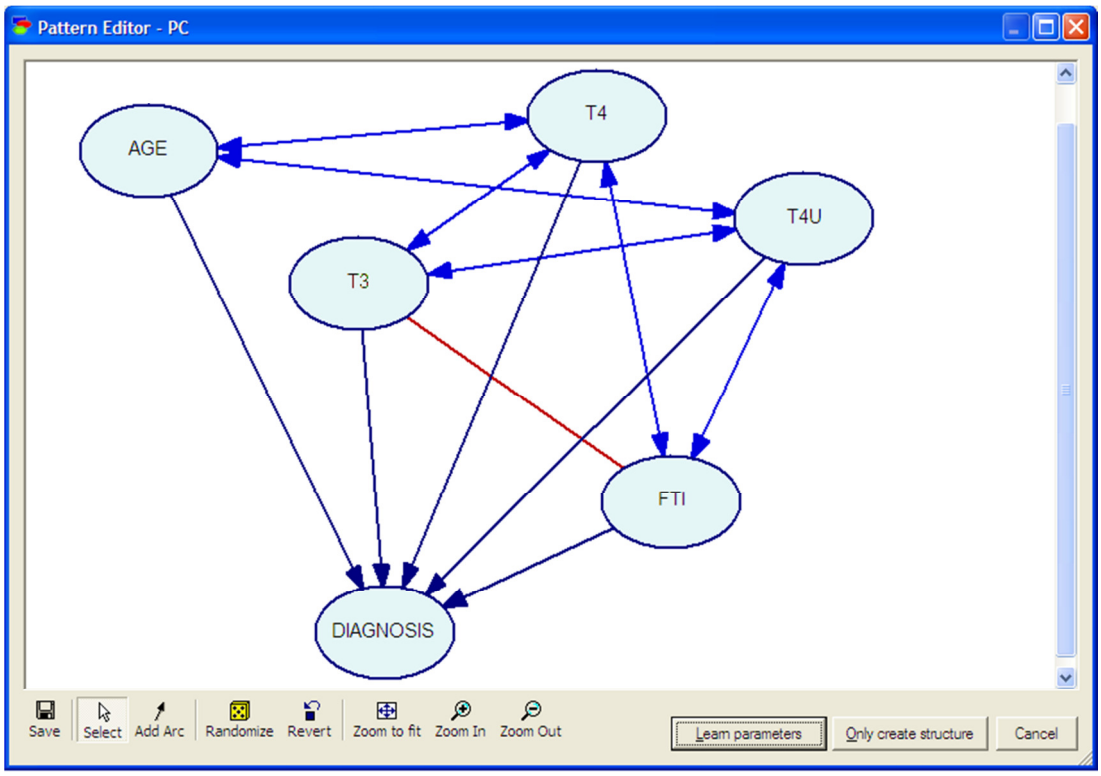


Figure 10: BN structure, learned from 1000 observations

The BN structure in Figure 11 was produced by increasing data set size to 5000 elements. This BN has no bidirectional arrows, showing that the nature of the relationships between domain concepts was learned more precisely. The undirected connections between predictor variables and response variable still exist, and the network is still not a directed acyclic graph.

GENIE's support for providing background information during structure learning allows expert input to be used alongside the relationships discovered from data. This feature was used to provide the background information in Figure 12 to the structure learning algorithm, along with the previously used data set with 5000 elements.

The background information in Figure 12 was used to express the relationship between the predictor variables and the outcome variable at a basic level, and led to the BN structure in Figure 13.

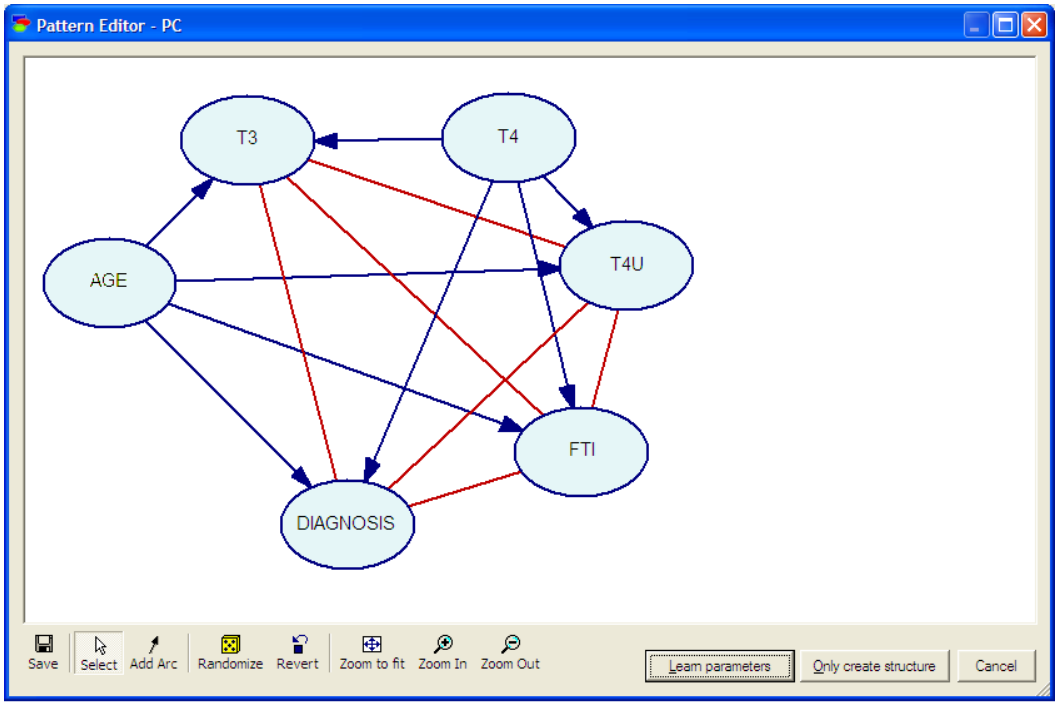


Figure 11: BN structure, learned from 5000 observations

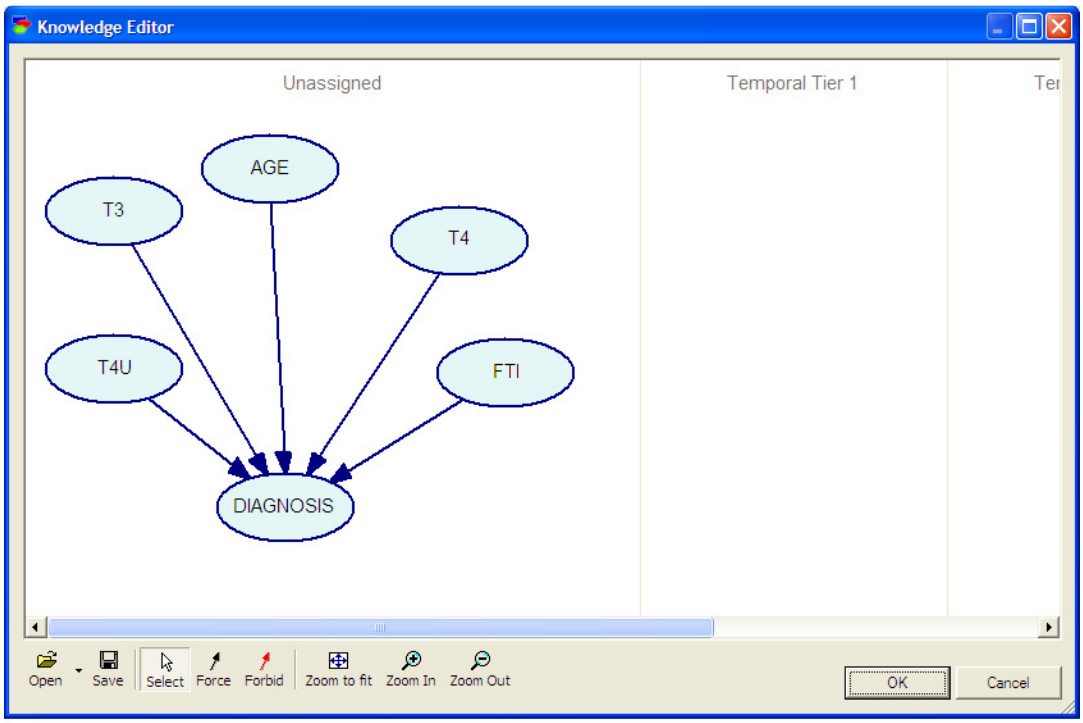


Figure 12: Background information for BN structure

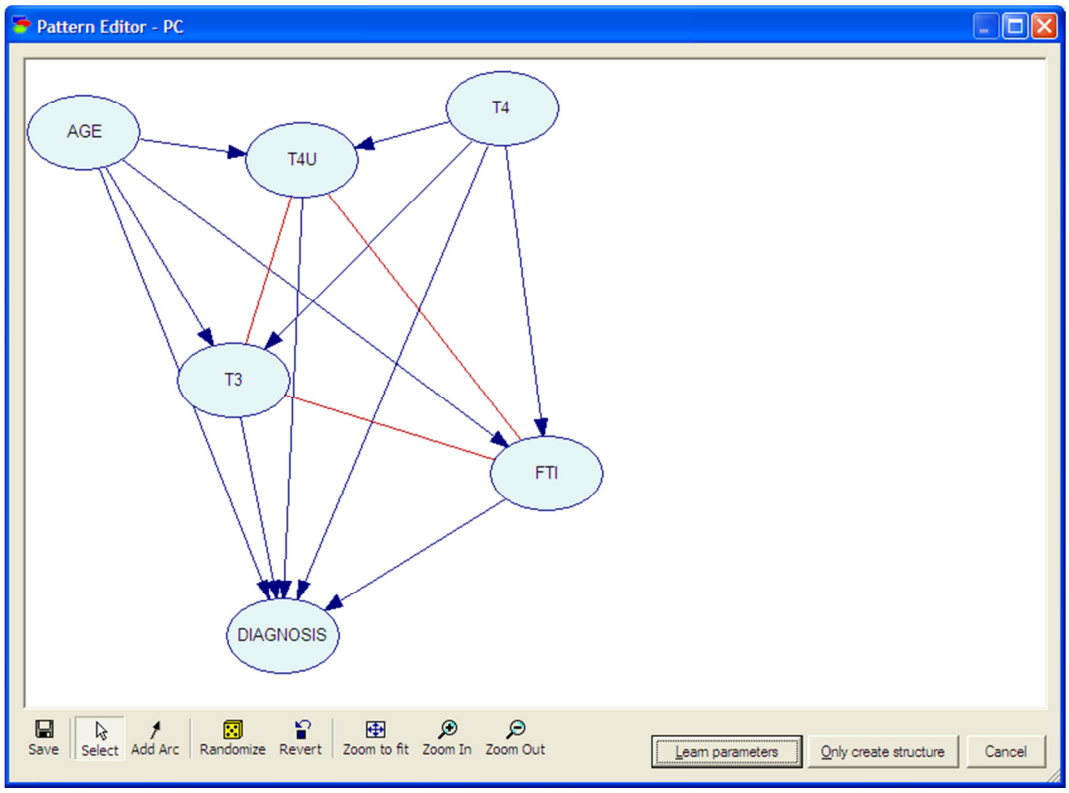


Figure 13: BN structure, learned with background information and 5000 observations

Other than the three undirected connections, the BN in Figure 13 has no problems in terms of required directed acyclic graph structure for inference. By deleting these undirected connections, the structure of the BN in Figure 14 was obtained.

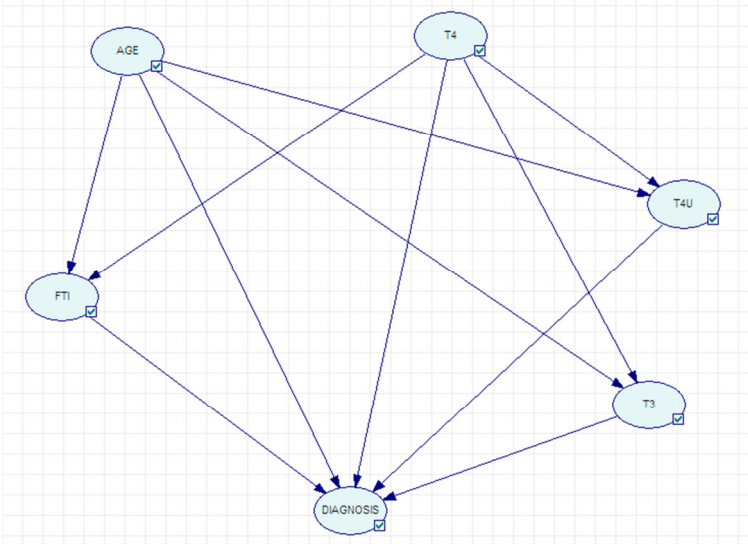


Figure 14: BN structure used in the experiment

The structure of the BN in Figure 14 does not necessarily reflect the precise relationships between domain concepts, especially for the connections between predictor variables. However, the process which results in this structure shows how both human input and clinical data can be used together to model a clinical decision-making context.

5.4 Learning the Network Parameters

The parameters of the BN were learned completely from data using the Expectation Maximization (EM) algorithm (Dempster, Laird, and Rubin 1977) implementation in GENIE. Only 1000 of the 5635 records in the dataset were used for parameter learning to avoid overfitting the data. The operation took less than 2 seconds on an Intel Core2 Duo based system (3.33 GHZ), running Windows XP resulting with the BN in Figure 15.

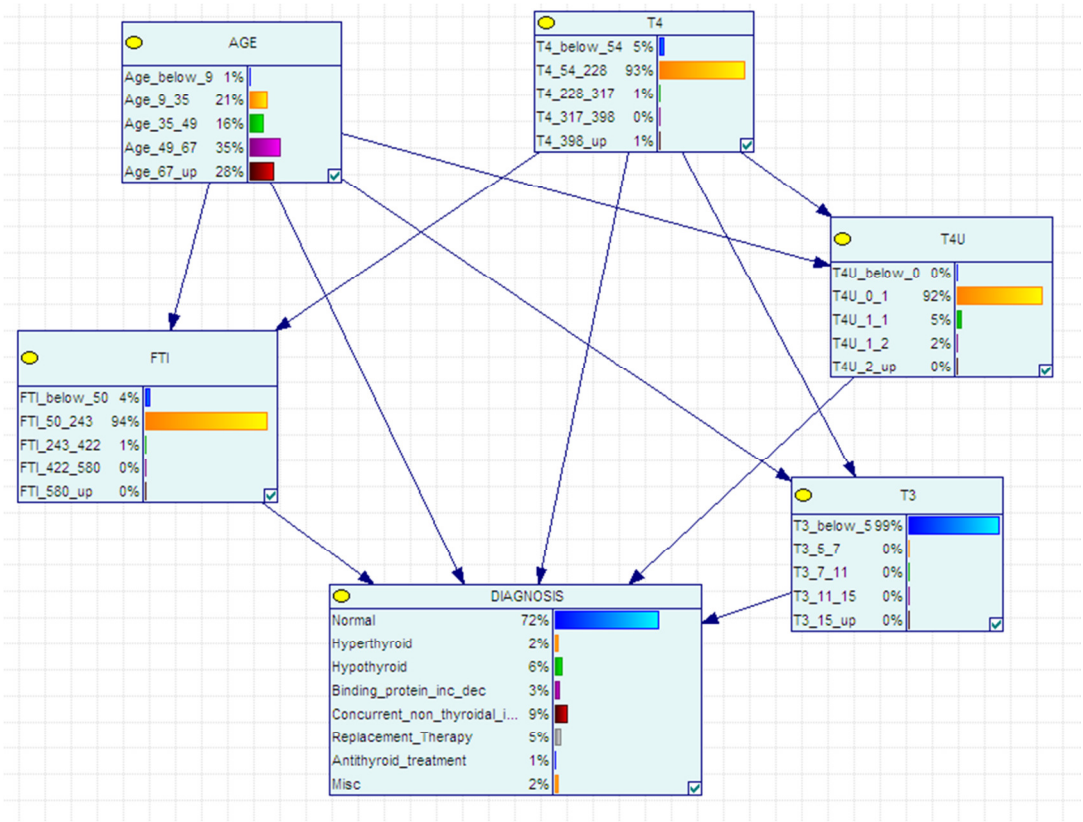


Figure 15: The distributions of nodes, learned via EM

The probability distributions defined by the nodes of the BN in Figure 15 (i.e. parameters of the network) assign zero to conditional probabilities of some combinations of outcomes of events that are represented by the nodes, which is a problem that stems from insufficient number of observations for these outcomes

(the 0% frequencies in Figure 15 do not represent these conditional probabilities, they are actually values close to 0, rounded down for compact display of nodes only). Even though the EM algorithm can deal with missing observations to an extent, some of the conditional probabilities end up with zero assigned to them, effectively describing these outcomes as impossible. This problem can be dealt with in various ways, such as changing the discretisation parameters so that extremely rare events do not end up as individual outcomes but instead they are categorised under outcomes with higher probabilities. However, this approach leads to less precise approximations to continuous distributions and in some cases in which the rare outcome must be specifically included in the model, it may not be an option at all.

Availability of more data can solve this problem by allowing the EM algorithm to assign non-zero probabilities to rare outcomes using more observations that include these outcomes.

Despite these problems, due to there being insufficient observations for some outcomes, the advantages of learning parameters from the data are evident. A large number of conditional probabilities would have to be specified by a domain expert even for the rather small BN in Figure 15. Not only would the expert have had to provide these conditional probabilities, but the resulting distributions would have had to comply with the basic rules of probability, such as requiring conditional probabilities of outcomes of events to sum to correct values.

5.5: Performing Inference on Bayesian Network

The following experiment used the clustering algorithm of (Lauritzen and Spiegelhalter 1988), which is an exact inference algorithm, to perform classification on the 4635 instances of the data set that were not used for parameter learning. Each instance was used to set the values of predictor variables of the BN, which were then used to predict the diagnosis node probabilities. These updated probabilities for the diagnosis node were used to select the classification outcome, which corresponds to the diagnosis outcome with the highest probability, after the update. This outcome was compared with the actual diagnosis as recorded in the data instance, to determine if the predicted diagnosis as correct. These operations were performed using a Java wrapper around the SMILE (Druzdzel 1999) library, which is the underlying library used by GENIE.

The inference process used with the test set failed for 41 of the 4636 data instances, due to the SMILE library attempting to process impossible observations.

These observations were deemed impossible due to their probabilities having the value 0 in the BN. These problematic data instances were subsequently excluded from the analysis of the inference results. However, they show that problematic parameters in a BN may lead to an inability to process real patient data. This behaviour may cause problems when patient data for patients with rarely encountered conditions is being processed.

The overall classification performance of the BN is provided in Table 1. The BN predicted the correct diagnosis in 77% of cases. Table 2 provides further details of the test results.

Correctly classified instances:	Incorrectly classified instances:	Total
3541	1054	4595

Table 1: Classifier performance

Number of healthy instances	**3424**
Number of unhealthy instances	**1171**
Correct prediction for unhealthy instances	**147**
Incorrect prediction for unhealthy instances	**1024**
Breakdown of incorrect prediction for unhealthy instances: found healthy when not	984
Breakdown for incorrect prediction for unhealthy instances: found the wrong disease	40
Incorrect prediction for healthy instances	**30**
Correct prediction for healthy instances	**3394**

Table 2: Detailed breakdown of classification results

A potential approach for improving the classifier performance, given that 3.9% (40/1024) of the classification errors is due to misclassification of a thyroid disorder, is to produce a binary output from the classifier as healthy or not healthy, leaving further evaluation of the findings to the clinician who uses the CDS system.

The classification performance of the BN can also be defined in terms of Sensitivity and Specificity (Loong 2003), which are used to assess accuracy of clinical tests (Parikh et al. 2008), (Lalkhen and McCluskey 2008).
The definitions of Sensitivity and Specificity in a clinical decision-making context, along with the terms these definitions are based on, are as follows:

- True positive: A test outcome that correctly diagnoses a condition
- False positive: A test outcome that incorrectly diagnoses a condition when the condition does not exist.
- True negative: A test outcome that correctly finds that a condition does not exist.
- False negative: A test outcome that incorrectly finds that a condition does not exist when the condition exists.
- Specificity: number of true negatives / (number of true negatives + number of false positives)
- Sensitivity: number of true positives / (number of true positives + number of false negatives)

	Healthy (real)	Unhealthy(real)
Healthy(predicted)	Tn: 3394	Fn: 984
Unhealthy(predicted)	Fp: 30	Tp:187
Total:	3424	1171

Table 3: Classifier performance

The sensitivity and specificity values for the BN classifier, based on the classification of test outcomes in Table 3 is as follows:

Specificity: 3394 / (3394 + 30) = 0.99
Sensitivity: 187 / (187 + 984) = 0.15

The high specificity of a test means that it performs well in identifying lack of a condition. Therefore, a positive outcome from such test would rule in a condition. The high sensitivity of a test means that it performs well in identifying the presence of a condition. Therefore, a negative outcome from such a test would rule out a condition. Based on these definitions, the outcome of the experiment is a BN that could be used by a clinician to rule in a condition.

Evaluating the performance of the BN based on sensitivity and specificity characteristics, is a more informative definition of its properties than simply providing the overall classification performance with the test data set.

5.6 Summary

Despite its limited scope and depth, the experimental development of a BN for thyroid disorder diagnosis has helped to identify key characteristics of the BN based approach to CDS.

The effort required to clean and transform data so that it can be used for structure and parameter learning was found to be significant. This data preparation phase required modifications to data, such as discretisation of continuous values, which can directly affect the performance of the resulting BN.

The size of the dataset available for structure and parameter learning was a key determinant of the quality of the results for both processes. Using human input as background information allowed better use of existing data for structure learning. Using existing data for parameter learning makes it possible for clinicians to avoid complex and error-prone aspects of defining BN probabilities.

However, the dataset size available for parameter learning is crucial for developing BNs that can accurately reflect the relationships between clinical concepts. Not having access to sufficiently descriptive data leads to a BN that may be unable to process some patient data, if the values of observations do not fall within the learned parameter boundaries. Defining the structure of a BN is less susceptible to lack of clinical data, especially if the number of domain concepts is small, and domain experts can easily identify them and define their relationships.

Detailed evaluation of the performance of a BN, with an emphasis on decision-making strategies specific to a particular clinical domain, can help clinicians make better use of its capabilities.

The experiment showed that, especially with the use of existing data, it is possible to develop and use a BN for CDS with minimum exposure to the underlying probabilistic concepts. The limited scope of the experiment leaves out many extensions and features of BNs that was discussed in Section 4.4. Yet, even such a basic implementation allows relevant clinical concepts to be used for purposes of probabilistic inference.

The inefficiencies and problems identified during different phases of the experiment overlap with well known issues encountered in clinical systems integration, for which solutions are provided by the openEHR specifications. The identification of key domain concepts as the components of the BN is another stage

of the experiment that can benefit from the openEHR methodology, as discussed in Section 4.7 in the discussion of the logical architecture for systems integration.

The pilot experiment identified aspects of a BN based approach to CDS that can be improved by exploiting the capabilities of openEHR. The experimental use of openEHR methodology to this end, is discussed in detail in the chapters that follow.

Chapter 6: A Pilot openEHR Based Clinical Information System Implementation Experiment – The Opereffa Open Source Framework

The openEHR specifications place significant emphasis on the term computable health, a concept that underlies all the functionality that openEHR is expected to provide, including better clinical decision support.

The scope, depth and quality of the openEHR specifications now provides a solid basis for computable health. However, the technology independent nature of the specification makes it hard to envision the extent to which the concept can be realised in practice, in the face of many implementation challenges. These challenges include limitations of the technologies used for the implementation and achieving the breadth and variety of functionality that a full openEHR implementation should support.

Therefore, the only reliable method to assess the sufficiency of the openEHR specifications for building clinical information systems and supporting CDS integration within these systems, is experiment and observation based on technical and clinical implementation. The Opereffa framework was undertaken as a proof of concept implementation of the openEHR specifications to serve this purpose.

6.1: Design and Implementation

Opereffa was developed in the Java programming language using open source technologies. It implements primary components of an openEHR based information system and provides EHR functionality to support clinical care. Its goal is to provide a workbench for experimenting with openEHR implementation and for observing the effects of design decisions made, on system characteristics such as performance and ease of integration with other software.

Opereffa's design positions it as a framework that can support core functionality for building an openEHR based system. Figure 16 provides a conceptual overview of the approach adopted.

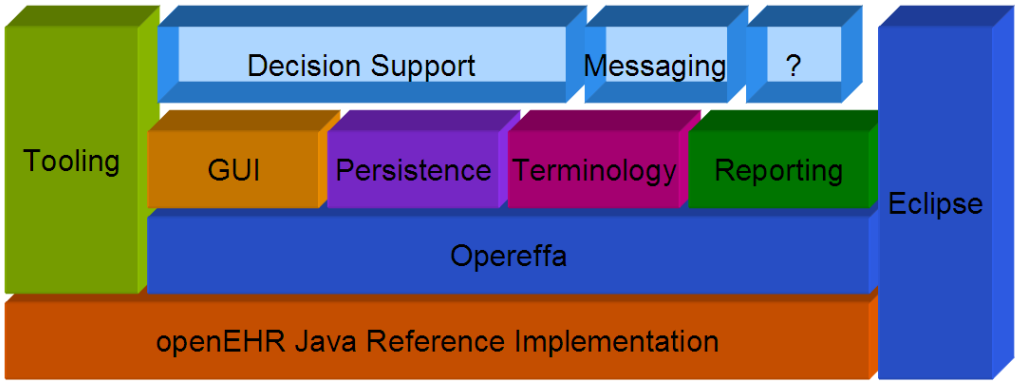


Figure 16: Opereffa framework and relevant concepts

Some of the components in Figure 16 represent pre-existing open source software projects, such as the Java based implementation of various openEHR tools and the Eclipse platform (desRivieres and Wiegand 2004) from the Eclipse Foundation. A clinical application development framework that supports the construction of a modern clinical information system based on openEHR was assembled using these components. Opereffa's implementation scope was limited to a necessary subset of these components, due to time limitations and relevance to the goals of this thesis.

Opereffa has been an open source effort from its outset, to enable wider feedback about the validity of its design and its approach to openEHR implementation. During its development, it was downloaded in over 70 countries and was used in a number of projects as well as academic studies. The software architecture of Opereffa is depicted in Figure 17.

The Opereffa software architecture comprises openEHR tooling and runtime components. The tooling, which is integrated into the Eclipse development environment, used pre-existing open source openEHR libraries to generate user interface code for Java Server Faces (Mann 2005), which is a Java based software framework for web applications development.

The automatically generated user interfaces contain data entry and display fields that correspond to data items defined in the openEHR clinical models. These user interfaces can be customised, in terms of visual styles, within the Eclipse development environment, which provides the user interface generation capability via a plugin developed for Opereffa. When the user interface code is deployed to the Java Server Faces environment, it automatically becomes available for data entry and display to users as an experimental clinical information system, which is accessible with a standard web browser. Figure 18 is a screenshot of an

automatically generated user interface, which includes data entry fields generated from an openEHR archetype, in addition to integration with an open source terminology server, LexBIG (Pathak et al. 2009), for performing searches and selection of applicable SNOMED-CT (IHTSDO 2015) terms.

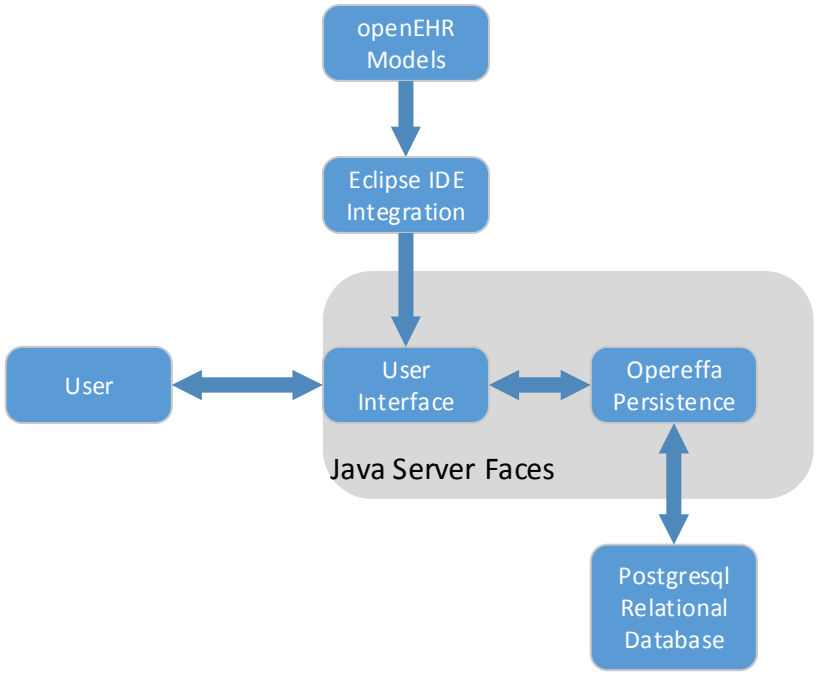


Figure 17: Software architecture of the Opereffa framework

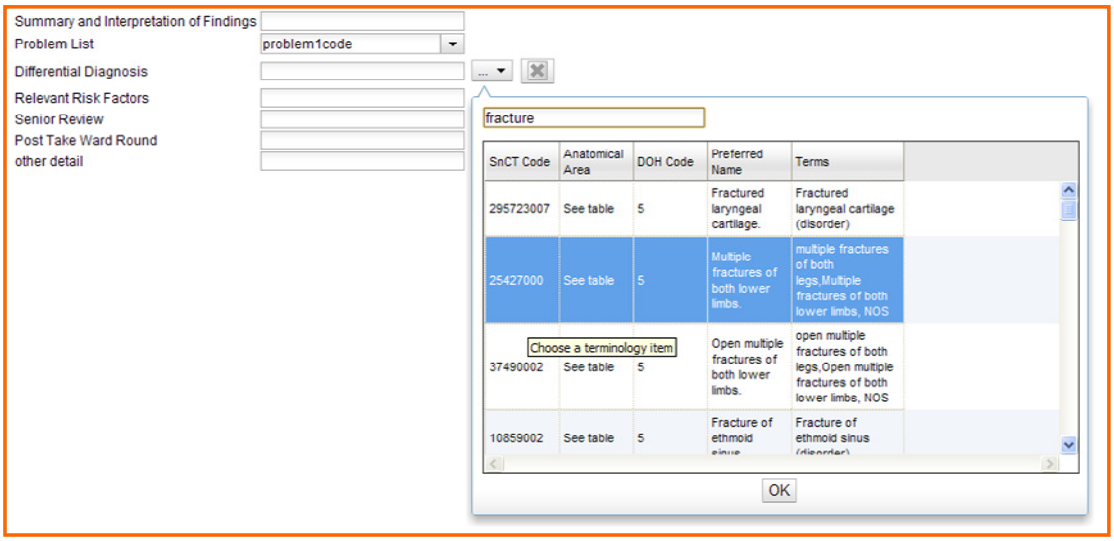


Figure 18: Screenshot from Opereffa User Interface

Filling in the fields of an automatically generated user interface, and then saving the contents, invokes the persistence implementation of Opereffa, which stores the data collected into the open source Postgresql (Momjian 2001) database.

Requests from the user interface to display previously saved documents also invoke this software component to fetch the data from the database and display it using the automatically generated user interface code, thereby also allowing users to update existing data.

The Opereffa design focuses on the concept of a clinical application driven directly by openEHR clinical models. A small number of such archetypes, developed with input from Dr. Tony Shannon, an active member of openEHR community, were used to assess the feasibility of the model driven clinical application development approach and to generate test data.

Opereffa demonstrated the capability to add new user interfaces based on openEHR clinical models, without any modification required to other parts of the system. This showed that a small number of components driven by openEHR models can support clinical records for a large variety of clinical domains - a key design goal of the openEHR specifications (Beale and Heard 2008a).

The Opereffa architecture represents clinical data by using information in openEHR archetypes to associate actual data values with their relevant openEHR RM types. Figure 19 shows how this association is implemented.

Opereffa maps these types from the openEHR specifications to corresponding Java classes. These Java classes, named as wrapper classes in general, associate the definitions of data items in openEHR archetypes with actual clinical data, entered by users through user interfaces, which are also generated from the same data items.

The elements on the right-hand side of Figure 19 represent implementations of these wrapper classes in the Java programming language. Their structure matches the structure of the openEHR model on the left of Figure 19, which is represented in a simplified form, for clarity.

The “Event” shown on the left-hand side of Figure 19 is a type defined by the openEHR specifications, and it has a time and a state, in addition to other properties, which allow instances of this type to model an actual clinical event. The instance of the Event type contains fields, such as “data”, which is specified as an instance of ITEM_LIST openEHR type. The “data” field, with type ITEM_LIST, contains items, which are of type ELEMENT. Instances of ELEMENT type have values, which may be either a quantity, a terminology rubric, or some plain text.

The Opereffa persistence model is based on the capability of wrapper types to save their contents to a database and later read it back, using the Hibernate object relational mapping library (Bauer and King 2005).

The association between the RM types and their corresponding Java wrapper types allows Opereffa to create data, persist it in a suitable database and read it back, effectively performing all operations on data using the clinical model on which the data is based. The open source openEHR libraries allow access to all aspects of these clinical models, such as the definition of valid values for a data item. This allows Opereffa to perform data validation at entry, a feature implemented only to a limited extent. The database representation for the contents of the wrapper types uses the paths of corresponding data items from the openEHR archetype to save their position in a tree comprising all wrappers that together represent the archetype.

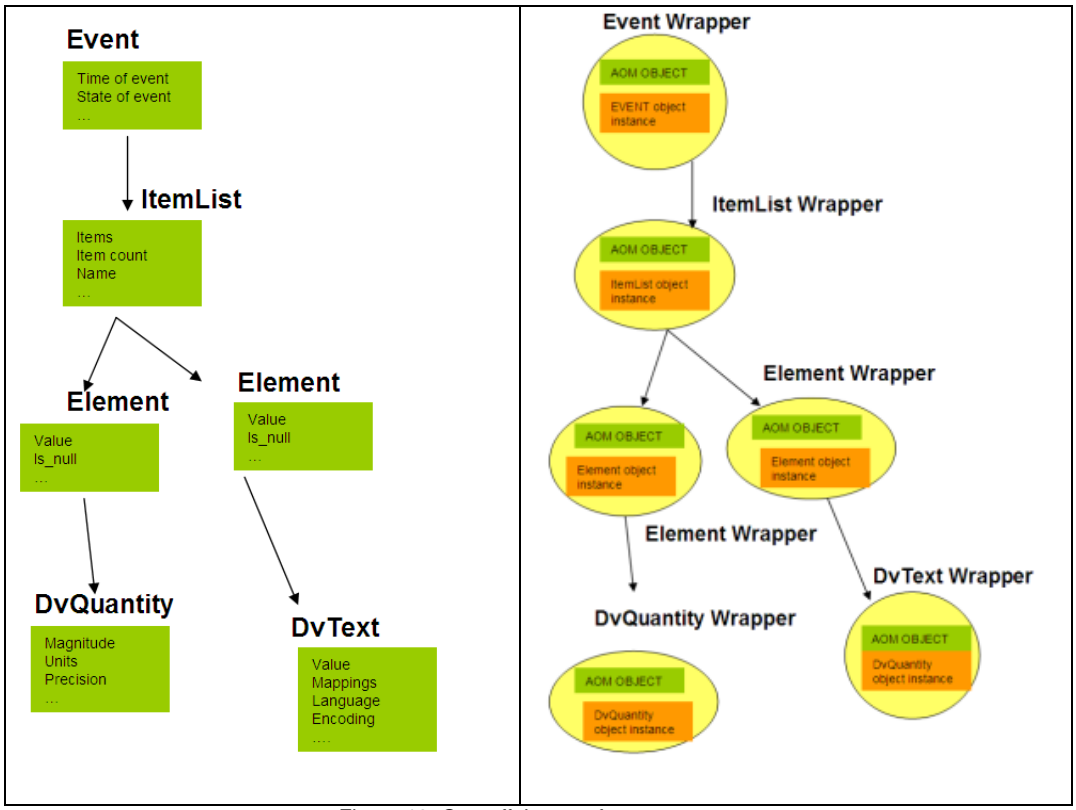


Figure 19: Opereffa's use of wrappers

None of the methods used in the Opereffa implementation is technology specific. The Java platform was chosen based on the availability of open source libraries for openEHR and other functionality that was required to implement the architecture in Figure 17, as quickly and reliably as possible.

6.2: Findings

The Opereffa experiment showed that a flexible, web based clinical information system can be developed with a model driven approach using open

source clinical modelling tools, along with open source libraries for processing openEHR models.

Automatic generation of user interfaces, complemented by a generic persistence layer that connects these user interfaces to a relational database, provides a flexible solution that can process clinical data from various domains. However, a purely web based application approach to building a clinical information system places limitations on the functionality achievable, and decreases its flexibility, especially in the persistence layer.

The initial and strong focus of the experiment was on providing access to patient data with a web based clinical information system. This does not require a generic openEHR data access method. There was no requirement in this experiment for sharing data with other systems. All the components of the Opereffa implementation, save for the pre-existing openEHR libraries, were developed from scratch, and customised to work with together to provide the envisioned functionality.

It has been observed that this approach leads to a strong specialisation of the openEHR implementation in all its layers, making it hard to expose data and functionality to other information systems. This problem reveals itself when new functionality that was not included in the initial requirements, becomes necessary. In the case of Opereffa, this new functionality corresponds to implementation of the openEHR AQL (Ma, Frankel, and Beale 2014), a domain specific query language for performing queries on openEHR data which was still in draft status, at the time of writing of this thesis.

The logical architecture for the openEHR and BN integration defined in Section 4.7 assumes multiple implementations for the openEHR specifications and learning BN network structures and parameters. The platform independent and openEHR specific nature of AQL makes it an appropriate method for connecting these implementations.

However, implementing AQL to integrate third party BN tools such as GENIE (Druzdzel 1999) and BNLearn (Marco Scutari 2009) on top of Opereffa requires the semantics of AQL queries to be supported by the persistence layer. It was observed that the Opereffa design, which assumes data access only from the web application, made it infeasible to support the requirements of AQL.

Two primary causes of this infeasibility are the data access assumptions of Opereffa and its focus on openEHR archetypes as a unit of user interface and persistence.

Opereffa's software architecture is designed to support flexibility for a clinical web portal, with a specific data access pattern. A list of previously committed documents is provided for a patient, and only one of them is displayed on the screen at a time. All of the data items that belong to a clinical document, based on an archetype as depicted in Figure 19, are fetched together from the relational database. Therefore, the unit of both writes and reads is single documents. AQL semantics allows for the description of parts of documents that satisfy specified conditions, and these parts can also be defined as query results. It defines hierarchical relationships between query components and it can set the scope of the search in an openEHR persistence implementation. Clinical data persisted by Opereffa does not include the relationships among data items that would be required to support these aspects of AQL queries. Therefore, the flexibility of the Opereffa persistence layer lies in its schemaless nature, but it does not extend to queries that make use of the structure of the data it contains.

The use of openEHR archetypes as the unit of persistence means that Opereffa replaces a significant component of the openEHR specifications in an ad-hoc way. The openEHR specifications define the openEHR RM (Beale et al. 2008e) as the means for representing actual clinical data, conforming to structural and value constraints defined by the openEHR archetypes. Opereffa uses the wrapper approach depicted in Figure 19 to represent actual data values, effectively replacing the RM representation with wrappers. The consequence of this approach to the persistence layer is that the data values are associated with custom paths based on wrapper types instead of archetype paths. AQL queries are based on archetype paths, assuming that the AQL implementation is capable of retrieving results using this information. This mismatch between the Opereffa persistence design and AQL's assumptions makes it unable to support AQL's way of describing data items.

The adoption of openEHR archetypes as the unit of the clinical model that Opereffa uses to drive user interface generation, in addition to persistence, means that all clinical domain input should be provided in the form of archetypes. This approach was not fully aligned with the openEHR methodology, which assumes that archetypes should be re-usable models with maximal data set properties rather than clinical system specific models. Therefore, Opereffa's interpretation of clinical models in the context of openEHR is not compatible with the widely adopted approach, which would diminish its capability to re-use globally available openEHR clinical models as well as its capability to share its models with other systems. This interpretation was sufficient to support the envisioned functionality for Opereffa, which was limited to a proof of concept EHR access portal that supports easy and

automatic generation of user interfaces in addition to data persistence support to serve these user interfaces. This definition of functionality did not include any clinical model or data sharing scenarios.

6.3: Summary

The primary finding from the Opereffa experiment described here is that the consideration of data access patterns, along with query result volumes, is a fundamental design requirement for openEHR implementations. Classifying some data access scenarios as of secondary importance for design leads to an architecture that makes their subsequent implementation infeasible, due to conflicts with the previous design choices.

The problems identified with AQL implementation do not mean, however, that Opereffa was a failed experiment. Despite its shortcomings in responding to the requirements of the logical architecture in Section 4.7, Opereffa has provided valuable insight into the capabilities of openEHR for model driven health data processing. In fact, the strong and worldwide interest in Opereffa, despite its extremely limited exposure, is evidenced by references made to it in many already published studies. These cover a range of topics, such as: generation of user interfaces, use of big data frameworks, mobile applications and clinical decision support (Kopanitsa et al. 2013), (H.J. Parashar et al. 2013), (Cd, S, and P 2009), (Velte et al. 2012), (Hem Jyotsana Parashar, Sachdeva, and Batra 2013), (Kohler et al. 2011), (Saxena, Sachdeva, and Batra 2015), (Kashfi and Jairo Jr 2011), (Batra et al. 2014), (Christoph Rinner et al. 2011), (Madaan et al. 2013), (Menárguez-Tortosa, Martínez-Costa, and Fernández-Breis 2011), (Sachdeva et al. 2011), (Madaan and Bhalla 2014), (Duftschmid, Chaloupka, and Rinner 2013), (Kohl 2012), (Sundvall et al. 2013), (Menarguez 2013).

The most significant conclusion from the Opereffa experiment was the crucial responsibility of the persistence layer to support fundamentally different kinds of data access patterns, within an openEHR driven approach. A more flexible persistence design is thus required to fulfil the requirements of the logical architecture for CDS integration, to ensure a unified software framework that is more robust in the face of highly variable patterns and volume of data access. A novel design exhibiting these properties is the focus of the next chapter, leading, in the following chapter to its implementation and evaluation in a comprehensive CDS setting.

Chapter 7: Persistence Abstraction for openEHR

Both clinical models based on openEHR and CDS built on BNs have the capability to support a wide range of functionality in their respective domains, and their integration leads to interactions between EHR and BN concepts, most of which require a BN based CDS implementation accessing clinical data via an openEHR implementation. Significant challenges exist in fulfilling this requirement.

The data access characteristics of particular interactions between openEHR and BN implementations can vary significantly. For example, survival prediction based on a BN and the value of a prognostic variable requires access to clinical data for a single patient, but learning the structure and parameters of the BN would benefit from access to clinical data of a large population of patients. Therefore, the capabilities of openEHR persistence implementation are crucial in robust CDS integration.

The openEHR specifications do not include the implementation of persistence of openEHR data or access to it via AQL in its scope. This keeps the openEHR specifications adequately concise, which lets implementers use a technology that is appropriate for their use cases without the risk of losing the benefits of openEHR compliance.

However, the high number of options for implementing openEHR persistence introduces the inevitable cost of developing the openEHR data representation and AQL support for each implementation technology. Given that implementation can follow a different design approach for each technology, a significant amount of repeated effort is likely to be required. This repeated effort is a limiting factor for implementing openEHR across a number of persistence technologies, especially to better support CDS integration. Each of these persistence technologies potentially offers a unique advantage such as large scale in memory data processing (Stonebraker and Weisberg 2013), batch data processing (Borthakur 2007), streaming data processing (Ranjan 2014) and more. Given that these advantages can help improve performance of different CDS scenarios, eliminating this limiting factor could potentially improve openEHR based CDS by making use of the results of ongoing research.

Another aspect of integration that needs to be considered is the effect of previous design decisions for persistence that are implemented prior to CDS integration. As discussed in Section 6.3, an openEHR implementation can fulfil functional requirements for a clinical information system and still fail to support data access for CDS integration.

Overcoming these challenges is necessary to benefit from the technology independent nature of both openEHR methodology and BNs in the context of their integration. Otherwise, integration of openEHR methodology and BNs cannot go beyond a series of case specific systems integration tasks, falling short of the unified architecture this thesis aims to develop.

This thesis develops abstract, robust and consistent representations of both openEHR data and the Archetype Query Language to overcome these challenges. These representations allow openEHR data persistence and AQL to be implemented on a number of persistence systems. Persistence system is used as an umbrella term in this thesis that refers to software that provides the capability to save data to a durable medium and read it back, such as relational databases, big data frameworks, graph and document databases.

An abstract definition of openEHR data and AQL processing with a focus on implementation across different persistence systems complements openEHR methodology without compromising its technology independent nature. This persistence abstraction establishes a balance between technology independence and implementations fully specialised to particular technologies. Consequently, it enables AQL based data access to a number of underlying persistence systems, providing a unified platform to support CDS based on BNs

An ideal abstraction should sufficiently support the following requirements to achieve this goal:

- Expressiveness
  The abstraction should be able to express openEHR RM based data and AQL semantics.
- Extensibility
  The abstraction should support extensibility to accommodate changes to the openEHR specifications and support extensions which may not be part of the specification, but deemed useful.
- Feasibility of implementation
  The abstraction should be implementable across a number of persistence systems.
- Consistent representation
  The abstraction should ideally have a consistent representation that can define data and operations on data across implementations on different platforms.
- Scientific relevance

The abstraction should have research associated with it that identifies its benefits and shortcomings in relation to its use in openEHR persistence, especially for supporting CDS.

The scope of the persistence abstraction consists of openEHR RM (Beale et al. 2008e) and AQL (Ma, Frankel, and Beale 2014).

7.1: openEHR Models and RM Data

Persistence abstraction for RM provides a representation of data that is based on RM types. The RM types, as discussed in Section 3.1, are defined independent of any particular technology. Therefore, openEHR data can be represented in many data formats, as long as the representation conforms to definitions of RM types. Textual formats such as XML, custom binary formats, or custom data structures based on built in type systems of programming languages can all represent RM data. The XML format is frequently used for openEHR data representation due to its strong adoption by many platforms as well as being human readable, and it allows openEHR data to be transformed to other formats when necessary.

The screenshot in Figure 20 shows the relationship between an openEHR model and RM data. The term openEHR model refers to an openEHR template. As discussed in Section 3.1, openEHR templates are modelling artefacts that are strongly associated with openEHR implementation and openEHR methodology encourages actual clinical data to be created based on them. Therefore, they will be referred to as openEHR models or clinical models in the context of persistence abstraction.

The screenshot from the freely available template designer tool in the left hand of the diagram in Figure 20 illustrates the clinical model, which is an openEHR template, and the visual representation of RM data to the right of the same diagram illustrates an RM instance that is valid according to this model.

The clinical data on the right is in XML format and a few data items such as the Systolic and Diastolic from the model are associated with actual data in the XML file.

The archetype path of the Systolic data item of the clinical model in Figure 20 provides a mechanism for referring to data items as defined in the openEHR specifications (Beale et al. 2008c). The archetype path consists of a root and a sequence of elements listed under the root in a parent/child format in which each

element except the root has a parent. Predicates that constraint archetype node identifiers to particular values can be placed on elements. The elements can be uniquely identified among their siblings using their archetype node identifier values in predicates, as exemplified by the [at0006] predicate on the events element of the path.

The archetype path of a data element is independent of the format that is used for representing actual clinical data, providing a semantically meaningful pointer to data without the need to know the underlying data format used by the openEHR implementation. It is the implementer's responsibility to provide access to actual clinical data pointed at by the archetype path.

The clinical model on the left in Figure 20 defines a set of data instances that fits the structure defined by the model along with the criteria for data values. The RM instance on the right is just one instantiation of data that is valid according to this model, based on the implementation of RM types using the XML type system, such as COMPOSITION and OBSERVATION types.

The XML data in Figure 20 is an example of RM instance data that could be created in many openEHR implementations. Regardless of its persisted form or the persistence system it is saved in, RM data such as this should be queryable in a platform independent way. This requirement is fulfilled by the Archetype Query Language, which is therefore within the scope of persistence abstraction along with RM data.

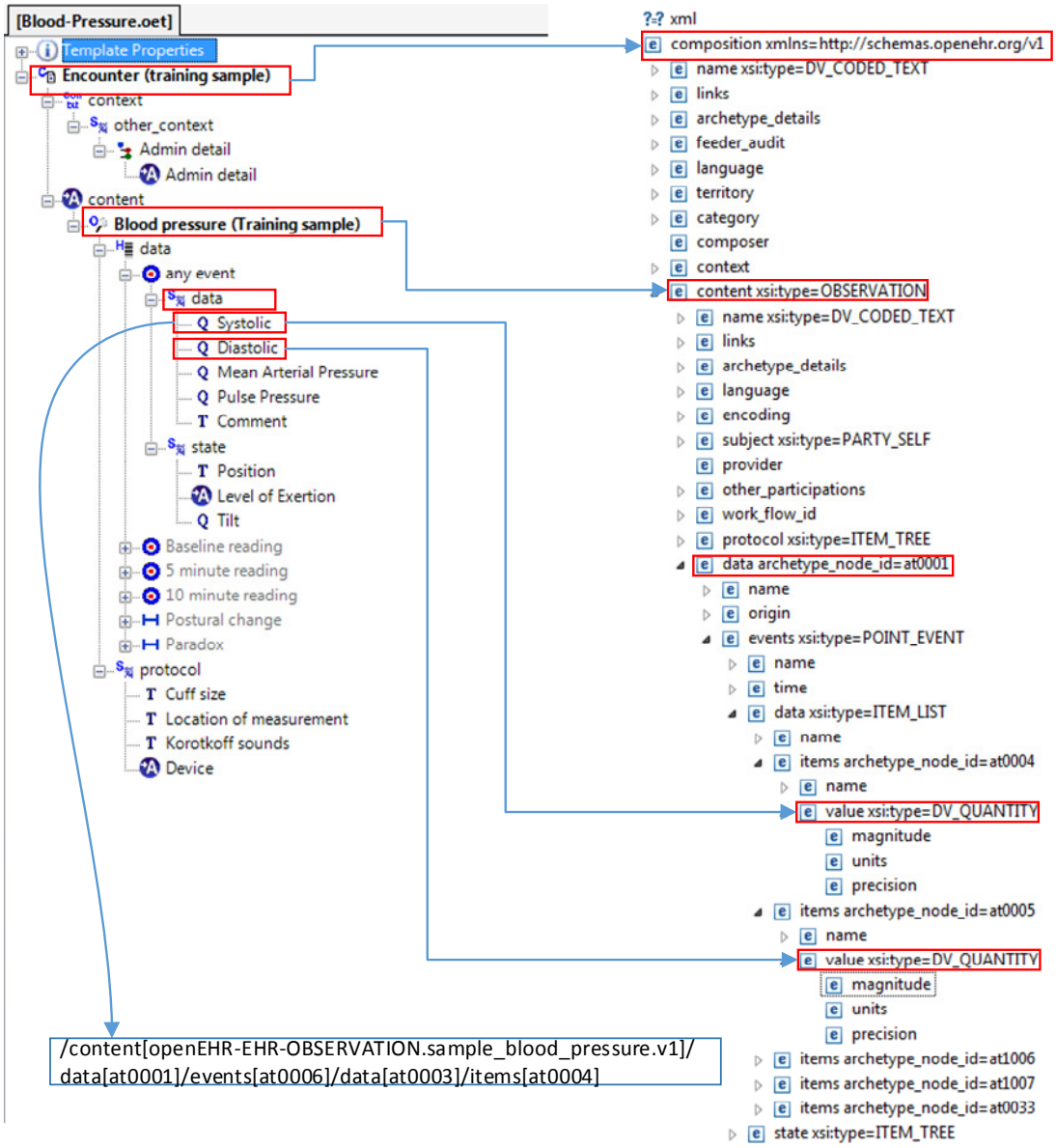


Figure 20: openEHR clinical model and RM based data instance

7.2: Archetype Query Language

Querying openEHR data using AQL is a platform independent method of data access, which is widely adopted by openEHR implementers, despite AQL not being part of the openEHR specifications at the time of the writing of this thesis. Similar to openEHR templates, it is likely to become part of openEHR specifications in a bottom up manner, following its adoption by implementers. Therefore, it is chosen as the means of data access in this thesis.

AQL queries consist of three major sections identified by three clauses in an AQL query: SELECT, FROM and WHERE. Their brief explanation is as follows:

- FROM

  The FROM clause defines the data points in an RM instance that will be used as reference points in other AQL clauses. They can be used directly or become reference points for accessing other data points through relative paths. This clause supports describing the hierarchical relationships of data points along with the use of predicates on their attributes such as their archetype node identifiers.

- SELECT

  The SELECT clause identifies the data points in an RM instance that the AQL query should return in its results. These points can either be the ones identified in the FROM clause or other data points that lie on a path relative to them.

- WHERE

  The WHERE clause allows expressing various constraints either directly on the data points identified by the FROM clause or on data points at a relative path to them, for the purposes of filtering results.

Figure 21 contains an example AQL query that should return the Systolic data point from the RM instances that match the criteria defined in the query. The right hand side of Figure 21 contains a template, which shows how AQL queries are defined by criteria based on clinical models.

This AQL query can be deconstructed as follows, based on the key clauses:

- FROM

  The FROM clause identifies three data points based on the RM types. The EHR is the highest level container in the openEHR specifications that contains all clinical data for a patient and it is not represented in the clinical model in Figure 21. Nonetheless, the AQL implementation is responsible for identifying the EHR instance that has the ehr_id value of ‘1234’. The EHR instance is given the alias ‘e’.

  FROM clause in this query uses the CONTAINS keyword to define a hierarchical containment constraint to identify the patient encounter element (of RM type COMPOSITION) that should reside within an EHR. The encounter element is given the alias ‘c’ and its archetype node id is constrained to ‘openEHR-EHR-COMPOSITION.encounter.v1’ via a predicate.

A second use of CONTAINS keyword introduces another containment constraint that requires a blood pressure observation (o) (of RM type OBSERVATION) to exist at some relative path to patient encounter (c). Connections in Figure 21 from c and o elements to clinical model on the right illustrate the hierarchical relationship the FROM clause is describing. Therefore, the FROM clause, identifies three data points with aliases e,c and o. These points are then used directly or indirectly in other clauses.

- SELECT

The SELECT clause uses only one data point identified by the FROM clause which has an RM type of OBSERVATION and alias ‘o’. SELECT clause uses this data point as the root of a path that identifies the actual data point of interest.

The data point defined as “Systolic” in the clinical model in Figure 21 is an RM object that represents a quantity. Its path relative to the root of the OBSERVATION typed RM object is “/data[at0001]/events[at0006]/data[at0003]/items[at0004]” based on the clinical model. By using the ‘o’ alias the SELECT clause identifies the Systolic data as the result to be returned from the query.

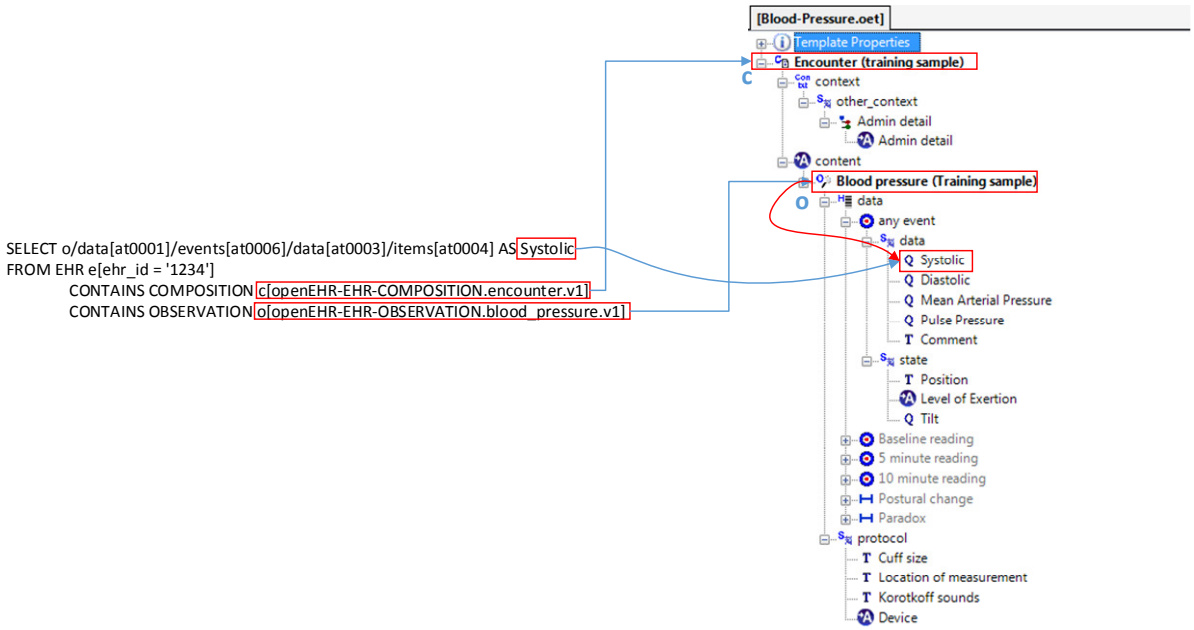


Figure 21: AQL query and openEHR clinical model

7.3: Structural Characteristics of openEHR RM

The structural characteristics of openEHR RM carry significance both in clinical modelling and implementation.

The RM, which is the starting point of clinical modelling in openEHR methodology, enforces a hierarchical structure for representation of clinical data. As depicted in the high level overview of openEHR RM type hierarchy diagram in Figure 22, which is taken from the openEHR EHR Information Model specification (Beale et al. 2008e), instances of EHR type (representing the concept of Electronic Health Record) create a single container for all clinical data that belongs to a particular EHR. Therefore, the EHR is the top-level concept.

The EHR instance is the container of actual clinical data that is represented by instances of the COMPOSITION type. Even though there are many other types in the RM, EHR and COMPOSITIONs under EHR instances are key determinants of structure of actual clinical data, as the diagram from openEHR EHR Information Model specification in Figure 23 shows.

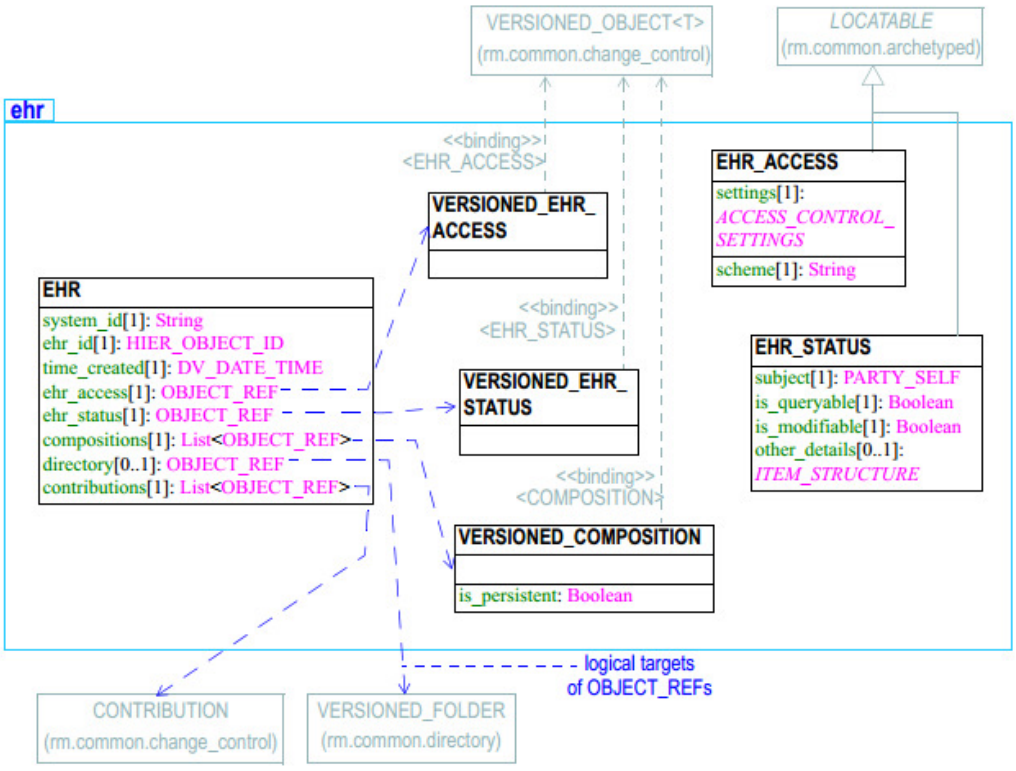


Figure 22: openEHR RM: EHR package

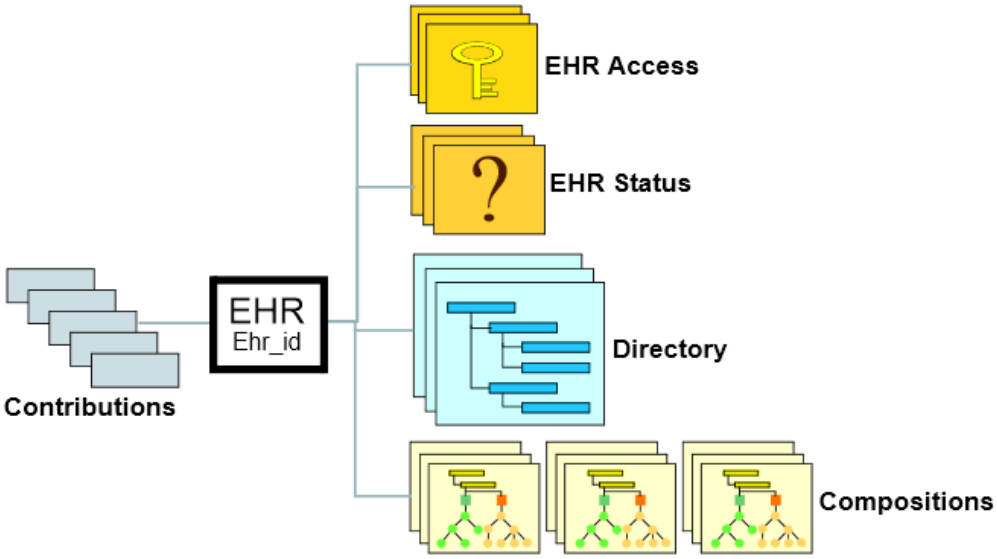


Figure 23: openEHR EHR: organisation of data

Regardless of the method and persistence system chosen for implementation, access to RM based data based on the structural characteristics of clinical models requires the implementation to make use of structural aspects of it. The functionality that is defined in the openEHR specifications, which uses archetype paths, as well as AQL query processing, which uses both relative paths and containment of data items, are examples of data access based on structural characteristics. Therefore, a persistence abstraction for openEHR and an implementation based on it must support the representation of these characteristics of RM based data.

Based on these requirements, persistence abstraction can be defined as platform independent representation and querying of structured clinical content where querying supports content model based access methods. Aside from its specific focus on clinical content, this definition bears noticeable similarity to capabilities of XML and query mechanisms it supports. This similarity is important since a large amount of research has been conducted on XML for content representation in addition to query processing, results and findings of which can contribute to the design of an openEHR specific persistence abstraction.

The inclusion of XML in openEHR specifications as an implementation technology specification, as discussed in Section 3.1, confirms XML's capability to represent openEHR data, strengthening the argument that its underlying content model may provide valuable insights for building a persistence abstraction for openEHR.

To this end, an appraisal of XML representation of openEHR along with XML's underlying data abstraction methods is performed. The findings of this appraisal are then used as the basis of the persistence abstraction for openEHR, without any dependencies on XML.

7.4: Appraisal of XML Representation of openEHR Data

7.4.1: Design and Goals of XML

The Introduction section of the XML specification (Bray et al. 1997) provides insight into XML's fundamental characteristics that has made it a ubiquitous data representation and thus exchange method:

"Extensible Markup Language, abbreviated XML, describes a class of data objects called XML documents and partially describes the behavior of computer programs which process them. XML is an application profile or restricted form of SGML, the Standard Generalized Markup Language [ISO 8879]. By construction, XML documents are conforming SGML documents.
XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form markup. Markup encodes a description of the document's storage layout and logical structure. XML provides a mechanism to impose constraints on the storage layout and logical structure. [Definition: A software module called an XML processor is used to read XML documents and provide access to their content and structure.] [Definition: It is assumed that an XML processor is doing its work on behalf of another module, called the application.] This specification describes the required behavior of an XML processor in terms of how it must read XML data and the information it must provide to the application."

XML documents have a well-defined structure and the specification clearly outlines the functionality that must be supported by software that will process XML data. Processing XML data has become a common capability for a large number of platforms through the implementation of XML processors in many different programming languages.

One of XML's goals is supporting a wide variety of applications. This goal is established by not only describing how XML documents are supposed to be processed but also by keeping XML independent of the concepts of a particular domain. XML refers to some fundamental concepts such as documents, elements, tags etc. but these concepts are domain neutral and the specification focuses on syntax and structure of these concepts when represented in textual form.

Through the use of these generic concepts, XML has been able to represent many types of data from various domains in the form of XML documents, including openEHR RM data. The openEHR foundation has published XML Schema Documents (XSD) that define the openEHR RM based on XML's type system and XML documents that comply with published XSDs represent RM based data.

The initial conclusion from this observation would be that XML is a convenient intermediate form for openEHR RM data. This intermediate form can be used to move openEHR data across systems by leveraging ubiquitous support for XML processing, delivering a successful solution to the requirement of representing clinical data consistently across different platforms. The data abstraction methods used by XML, which are discussed next, play a key role in its success since they enable consistent implementation of XML across platforms.

7.4.2: Data Abstraction Methods Used by XML

A set of XML related specifications define the data abstraction methods used by XML along with query languages that target XML data, such as XML Information Set Specification(Infoset) (Cowan and Tobin 2004), Document Object Model Specification (DOM) (Wood et al. 1998) and XQuery 1.0 (Boag et al. 2002) and XPath 2.0 Data Model (XDM) Specification (Fernández et al. 2002).

An in-depth discussion of these specifications is out of the scope of this thesis, but a high level overview of their relationship is provided in Figure 24. The diagram in Figure 24 outlines the relationship between the XML specification and three other specifications from World Wide Web Consortium (W3C) that provide abstractions of XML data at various levels.

The diagram also provides the goals of XML Infoset, DOM and XDM along with the increasing level of abstractions they build on each other to achieve these goals.

XML Infoset defines concepts such as Element Information Item or Attribute Information Item. These concepts are more abstract than the ones used in XML specification, for the purpose explained in the introduction of XML Infoset specification:

*"This specification defines an abstract data set called the **XML Information Set** (**Infoset**). Its purpose is to provide a consistent set of definitions for use in other specifications that need to refer to the information in a well-formed XML document"*

DOM and XDM in turn, are both built on the abstractions provided by Infoset. DOM's goal is to allow access to documents along with the capability to modify them. To achieve this goal, DOM provides a set of interface definitions, which define the functionality that must be implemented by software. The Node interface defined by DOM is a fundamental interface that allows access to documents. The concept of a node is a component of the larger concept of a tree that consists of nodes. Although the DOM specification does not contain a formal definition of the tree, "What is the Document Object Model" section of DOM Level 3 Core Specification (Nicol et al. 2001) establishes the the relationship between tree and node concepts as follows:

"In the DOM, documents have a logical structure which is very much like a tree; to be more precise, which is like a "forest" or "grove", which can contain more than one tree. Each document contains zero or one doctype nodes, one document element node, and zero or more comments or processing instructions; the document element serves as the root of the element tree for the document. However, the DOM does not specify that documents must be implemented as a tree or a grove, nor does it specify how the relationships among objects be implemented. The DOM is a logical model that may be implemented in any convenient manner. In this specification, we use the term structure model to describe the tree-like representation of a document. We also use the term "tree" when referring to the arrangement of those information items which can be reached by using "tree-walking" methods; (this does not include attributes)."

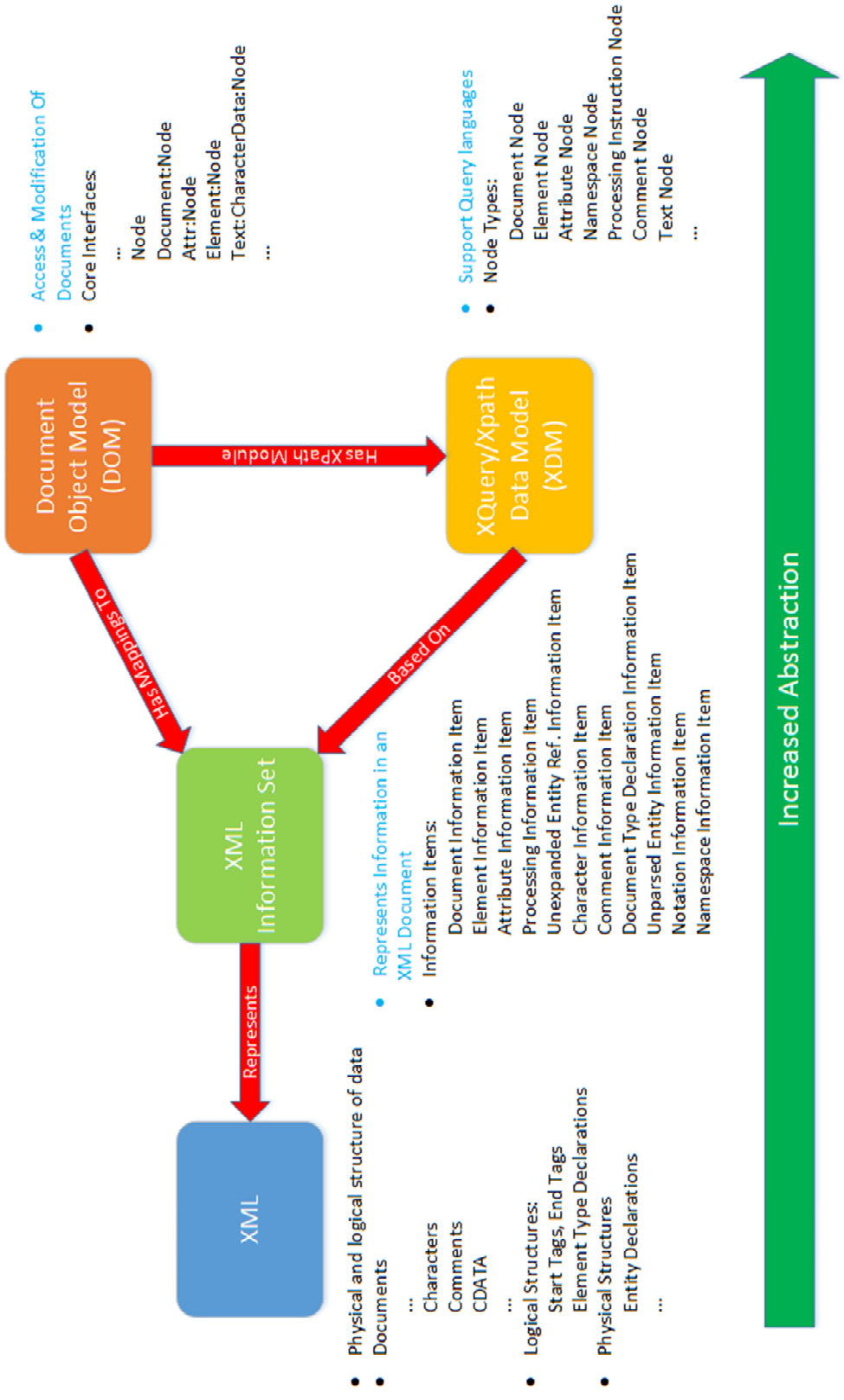


Figure 24: Abstractions of XML content

The Node interface and its related interfaces do not directly mention XML. In fact, DOM is capable of providing access to both XML and HTML (Berners-Lee and Connolly 1995). The concepts used by DOM to represent documents, such as

nodes and trees are more abstract then the underlying XML Infoset concepts, which are more specific to a document, such as Element Information Item. This increased abstraction is what allows DOM to provide a unified access model to different document types.

This approach is adopted by XDM as well, by introducing a data model based on XML Infoset, but with a focus on XML query languages, as described in "Introduction" section of XQuery 1.0 and XPath 2.0 Data Model specification:

"The XQuery 1.0 and XPath 2.0 Data Model (henceforth "data model") serves two purposes. First, it defines the information contained in the input to an XSLT or XQuery processor. Second, it defines all permissible values of expressions in the XSLT, XQuery, and XPath languages. A language is closed with respect to a data model if the value of every expression in the language is guaranteed to be in the data model. XSLT 2.0, XQuery 1.0, and XPath 2.0 are all closed with respect to the data model.
The data model is based on the [Infoset] (henceforth "Infoset"), but it requires the following new features to meet the [XPath 2.0 Requirements] and [XML Query Requirements]: …"

XDM focuses on read-only access for querying, but it uses abstractions similar to DOM, such as Document Node, Element Node and other Node types as shown in Figure 24. The "Terminology" section of XQuery 1.0 and XDM specification describes how these concepts are brought together:

"Nodes form a tree that consists of a root node plus all the nodes that are reachable directly or indirectly from the root node via the dm:children, dm:attributes, and dm:namespace-nodes accessors. Every node belongs to exactly one tree, and every tree has exactly one root node.
…
[Definition: A tree whose root node is a Document Node is referred to as a document.]
…"

Despite being more specific compared to DOM, XDM's definition of a tree structure that consists of nodes and it use for representing documents overlaps with DOM's approach based on the same concepts. The XPath module of DOM depicted in Figure 24 is proof for this overlap.

The key finding from the brief analysis of the relationships between the specifications included in Figure 24 is that these abstractions are the basis on top of which XML representation of openEHR is built, indicating the feasibility of representing openEHR data and queries that target this data with a small number of platform independent concepts.

7.4.3: Key Findings

The implementations of the interfaces defined by the specifications in Figure 24 enables XML form of RM based data to be processed in many platforms. Both the representation of content via DOM interfaces and the capability to query this content using specialised XML query languages such as XQuery (Boag et al. 2002) or XPath (Clark and DeRose 1999) make heavy use of the tree based abstraction. Therefore, the XML representation of openEHR indirectly shows the feasibility of using a tree abstraction for openEHR data

The diagram in Figure 25 shows how openEHR specifications and tree based abstractions provided by XML specifications are related to actual software implementations. The research that focuses on XML processing is also depicted in Figure 25.

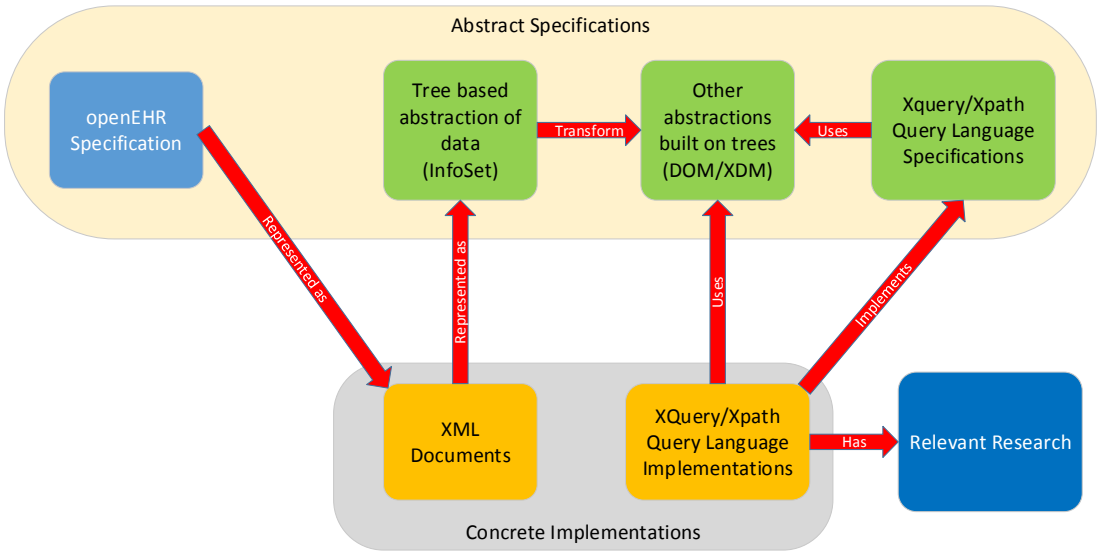


Figure 25: openEHR as XML: abstract and concrete components

The relationships presented in Figure 25 hints at the possibility of using XML representation of openEHR to accomplish the goals identified at the beginning of this chapter. This approach would build on the ubiquity of both XML documents and implementations of XML query languages using an XML document representation of openEHR RM data along with a mapping of openEHR AQL to XML specific query languages, as depicted in Figure 26.

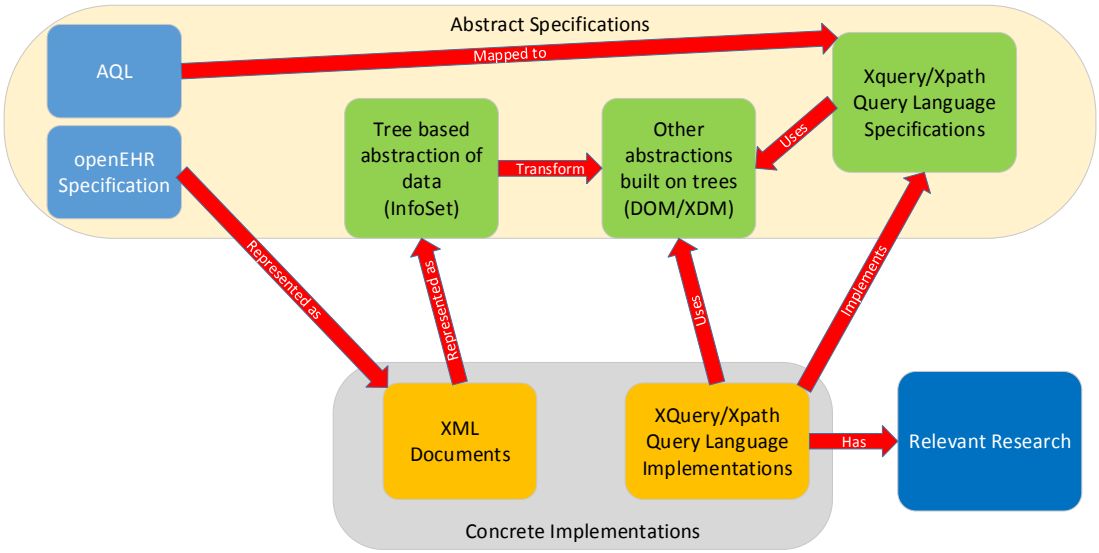


Figure 26: XML based openEHR persistence

The diagram in Figure 26 includes AQL alongside openEHR specifications to delegate both data representation and querying to XML and its query languages. Although this approach has the potential to fulfil the requirements of the persistence abstraction for openEHR, its complete reliance on XML is likely to introduce various problems. These problems stem from XML's fundamental traits that improve its versatility, which comes with the price of lower storage and computation efficiency compared to specialised data formats.

Human readability is one such trait that leads to XML documents using a textual representation and content layout that is not as space efficient as other alternatives. Implementations of powerful XML query languages, which are dependencies of the approach depicted in Figure 24, would need to be available in every context in which openEHR data is queried. Not all the features of these generic query languages are necessarily required to support the functionality of AQL, but there features are nonetheless implemented, potentially introducing computational overhead. This overhead may introduce performance issues in use cases where a high number of XML documents must be processed, such as epidemiological queries (Freire et al. 2012), even when optimised XML databases are used.

Another potential problem associated with delegating persistence abstraction to XML processing lies in the difference between availability of XML processors and feasibility of embedding them into other software. Given the large number of persistence systems that can be used for openEHR persistence, embedding an

XML processor to these systems may lead to a complex implementation step for openEHR persistence.

Due to these potential problems, despite the versatility and success of its underlying data models, direct use of XML does not sufficiently fulfil the requirements of persistence abstraction for openEHR as defined in this thesis. However, problems associated with XML's implementation do not necessarily rule out the use of its internal abstraction methods. The tree based abstraction of data is therefore used as the basis of a persistence framework for openEHR.

7.5: Tree Based Persistence Abstraction for openEHR

The experimental persistence abstraction for openEHR developed in this thesis uses the tree representation of RM data, based on the findings of appraisal of XML and related query language specifications. This approach has the benefit of excluding representation and processing requirements for data that are not relevant to RM or AQL query processing, achieving significant simplification for both persistence abstraction and its implementation. Figure 27 depicts the fundamental components of this architecture.

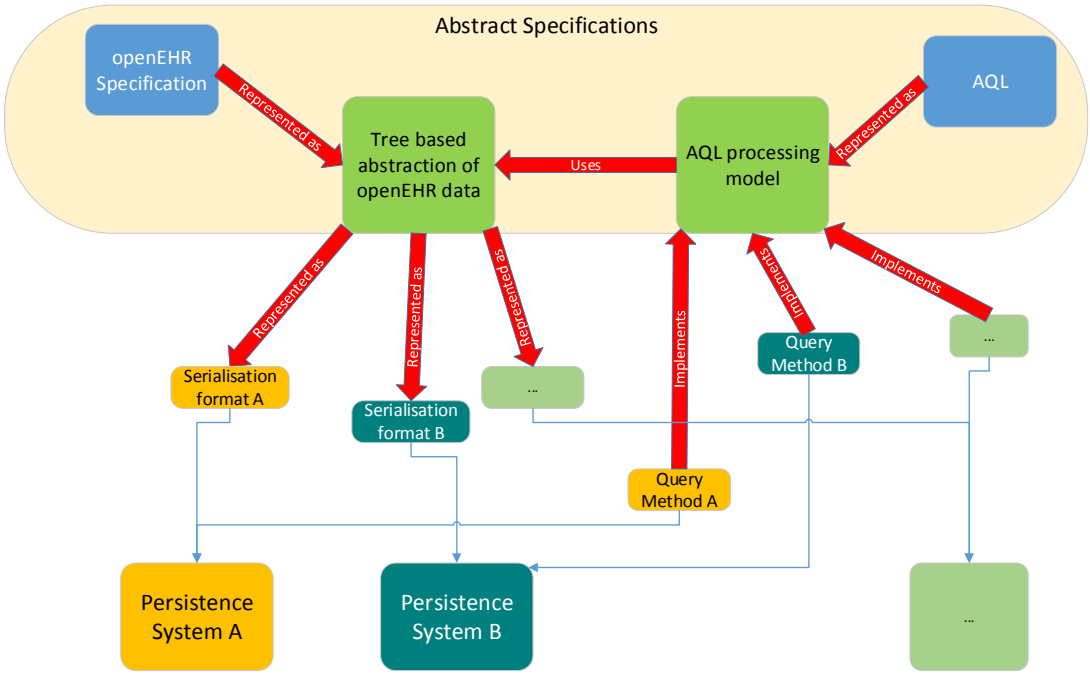


Figure 27: Tree based persistence of openEHR data

Both the RM based data representation and the AQL processing model use tree based abstractions in this architecture. A particular implementation of

persistence abstraction on a persistence system is based on two components: a serialisation format and a query method.

The serialisation format can be any data format supported by the persistence system to store data, and query method is any mechanism that allows access to data stored in the persistence system. Both the serialisation format and the query method are mapped to platform independent components of the persistence abstraction as depicted in Figure 27. These mappings enable different persistence systems such as relational databases, graph databases, or large scale data processing frameworks such as Hadoop (Borthakur 2007) to support AQL based on their native features, providing a unified data access method to RM based data. Therefore, the architecture in Figure 27 provides a generalisation of the openEHR persistence implementation based on the use of tree based persistence abstraction for openEHR.

This generalisation is built on the tree representation of openEHR data, with a strong specialisation on RM types instead of a generic content representation approach. The same specialisation is adopted for AQL processing as well. Tree representation and AQL processing based on this representation are defined in a technology independent way, similar to openEHR. Therefore, this generalisation is technology independent.

The specialisation in openEHR RM types and AQL processing reduces complexity in both representation and implementation by excluding all data representation and querying requirements that are not related to openEHR, in addition to eliminating the need for any intermediate representation and processing layers such as XML documents and processors. The difference between the two approaches is depicted in Figure 28.

When XML is used as the means of representing and querying openEHR data, the tree based representation and query mechanisms for openEHR are encapsulated within the relevant XML specifications and implementations. This approach corresponds to an implicit and limited abstraction of openEHR persistence via the use of XML.

Explicitly defined RM data and AQL abstractions remove the dependencies for XML storage and processing capability for a persistence system to be used for openEHR implementation. This approach also allows particular implementations based on persistence system specific serialisation formats and query methods to be optimised for openEHR persistence, which would not be possible in case of embedding an XML processor into persistence systems.

The first step in achieving these suggested benefits is the development of tree representation for RM data.

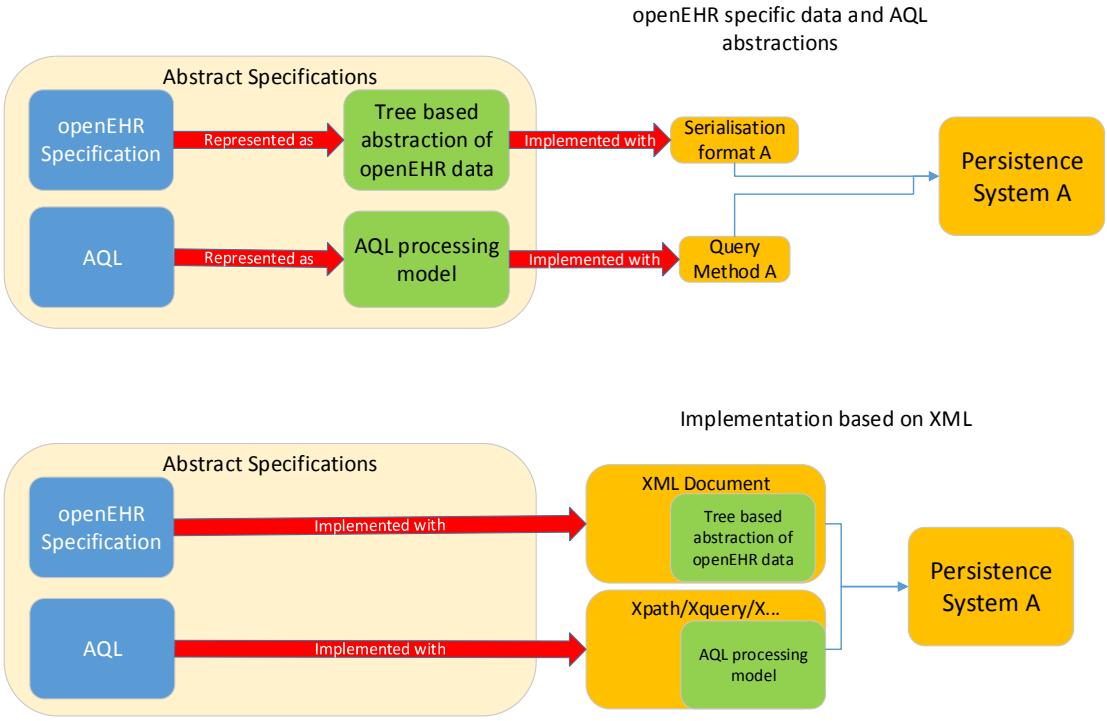


Figure 28: Implicit vs explicit tree based persistence

7.5.1: Tree-based Representation of RM data

In the context of this thesis, the term tree refers to tree data structure (Knuth 1968) and its computer science interpretation. The trees that represent RM data are singly rooted, and each node has at most one parent. All nodes of a tree are instances of the same data type. The data type used for nodes is a collection of key-value pairs, frequently implemented as a hash table (Cormen 2009). Using this data type enables nodes to represent named attributes with values. Therefore, when this thesis references an attribute of a node, the reference implies an entry in the collection of key-value pairs.

The connections between nodes represent the parent-child relationship in which a parent node may have zero or more child nodes. The connections (edges) between nodes are represented via consistently named attributes of nodes such as children or parent. The root node of a tree is the only node with a null value for the parent attribute. There are no constraints on the names of attributes that can be used. Therefore, any aspect of openEHR RM data can be represented with an attribute added to a node instance.

The mapping from openEHR RM data instance to a tree is therefore representing instances of the RM types as nodes of a tree.

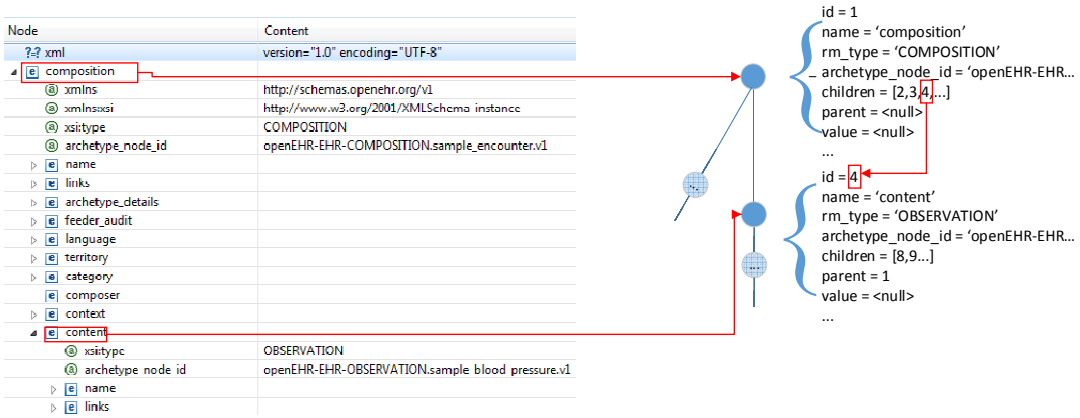


Figure 29: openEHR RM based data as tree

Figure 29 provides a high-level overview of this approach based on a visual representation of RM based data. The data in this figure is based on the blood pressure clinical model in Figure 20. The RM type instances are represented as nodes of the tree on the right. The tree nodes with ellipsis represent a group of nodes that are not included in full detail in the tree for the sake of clarity.

The root node of the tree represents the top-level object of the RM data with the RM type COMPOSITION. The root node of the tree represents the "composition" element of RM data. The attributes of the root node are partially included in Figure 29 to demonstrate how node attributes are used. The id attribute is the unique identifier of the node. The value of this attribute is used by parent and children attributes of nodes to express parent-child relationships. The children attribute of the root node is a collection of ids of its children, of which only the one with id 4 is individually presented in Figure 29. The rm_type attribute contains RM type of a data item, archetype_node_id contains the semantic identifier of a data item as defined by the openEHR specifications and name attribute contains the name of the field defined by the RM type. If an RM type has a field that contains an actual numeric or literal value, this value can be represented by a node attribute named "value". The value of "value" attribute is null for nodes which do not have a single, primitive value, as depicted in Figure 29.

The attributes of nodes of the tree in Figure 29 are only illustrative of the key-value nature of the nodes and not the precise list of attributes that every node has in the experimental implementation discussed in Chapter 9. These details are provided in Chapter 8. The flexible nature of key-value pairs based nodes allows both RM

data and other data that can be used by AQL processing implementations, such as parent-child relationships, to be expressed in a simple way. The creation and content of node level data is a key part of both this thesis and future research.

The tree based representation of RM based data provides the target for AQL processing, which is built on a small number of operations on trees.

7.5.2: Tree-based Abstraction of AQL Processing

The term AQL processing as used in this thesis refers to producing a result set of RM based data instances in response to applying the conditions defined in an AQL query on data. A tree based abstraction of AQL processing therefore implies expressing the semantics of these conditions based on constraints on nodes of tree based representation of RM. The conditions defined by an AQL query are distributed across the fundamental clauses of AQL, with potential dependencies on each other. The tree based abstraction is therefore developed based on the "FROM", "SELECT" and "WHERE" AQL clauses which are written in upper case in the rest of this chapter.

7.5.2.1: The 'FROM' AQL Clause as Source of Constraints on Trees

Identifying tree nodes defined by the conditions of the FROM clause requires two types of constraints to be applied: constraints on node attributes and constraints on node hierarchy. Figure 30 shows how FROM clause of an AQL query is associated with a tree that represents RM based data.

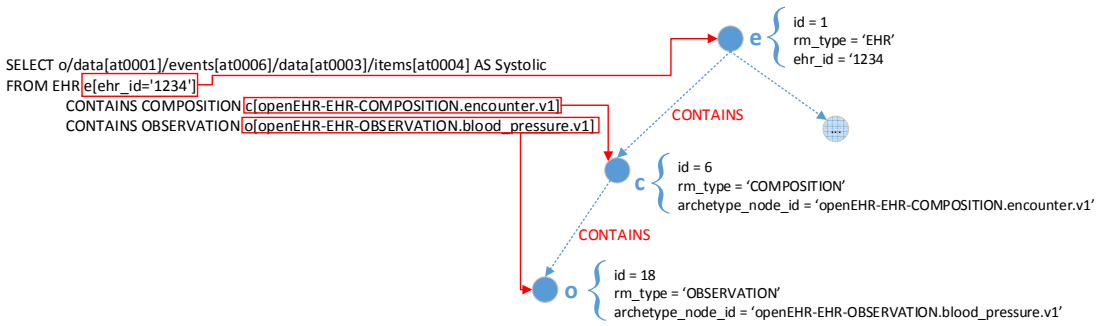


Figure 30: AQL FROM clause as constraints on a tree

A predicate such as [ehr_id='1234'] for a data item corresponds to a constraint on the tree node attribute ehr_id with value '1234'. The "EHR" type of the

same data item in the *FROM* clause is also expressed as a constraint on “rm_type” attribute.

The optional alias for the data item which is ‘e’ in this case is included in Figure 30 for convenience but it is not a constraint and is not part of the mapping from AQL to tree constraints. Since AQL assumes that an unquoted string in a predicate is a constraint on archetype node identifier, the data items with aliases ‘c’ and ‘o’ can be written as c[archetype_node_id=’ openEHR-EHR-COMPOSITION.encounter.v1’] which is consequently expressed as a constraint on a node attribute as depicted in Figure 30.

The constraints on node hierarchy are introduced by the *CONTAINS* keyword used in the *FROM* clause which expresses a “descendant of” relationship between data items. That is, given a containment constraint such as A CONTAINS B, there should exist a data item B that is accessible by recursively following child nodes where A is the root node. In other words, B should be a descendant of A. This definition includes direct parent-child relationships since they are also ascendant-descendant relationships. Figure 30 shows the descendant status of nodes via the use of dashed arrows which means there may be zero or more nodes between a parent and its descendant.

7.5.2.2: The ‘SELECT ‘ AQL Clause as Source of Constraints on Trees

The *SELECT* clause introduces constraints on both node hierarchy and node attributes as depicted Figure 31.

The *SELECT* clause uses the ‘o’ alias for the node identified in the *FROM* clause as the root of a path that ends with a data item of interest that should be returned as the result of the AQL query. In case of query in Figure 31, this item is given the alias ‘Systolic’.

The ‘Systolic’ data item defined by *SELECT* clause can be expressed as a series of hierarchical constraints on tree nodes similar to ones introduced by the *CONTAINS* keyword. However, these constraints are parent-child relationships between nodes as expressed by straight connectors in Figure 31. The nodes on the path from ‘o’ to ‘Systolic’, including ‘Systolic’ node itself are subject to node attribute constraints for the archetype_node_id attribute. ‘Systolic’ alias is included in the diagram in Figure 31 for convenience, similar to ‘o’ alias, but it is not related to any constraints.

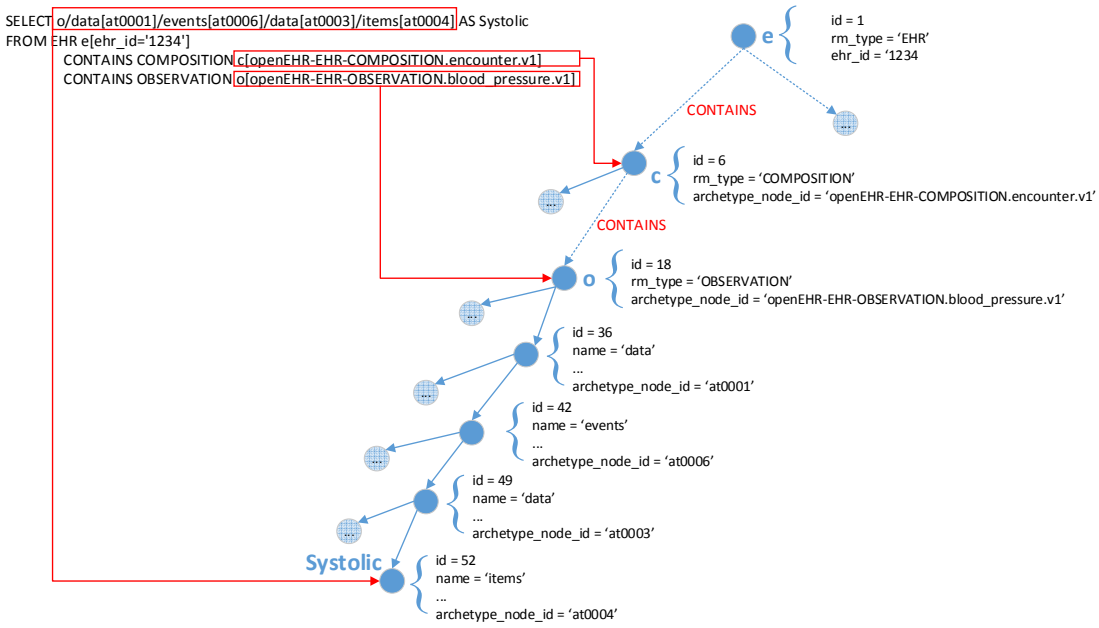


Figure 31: AQL SELECT clause as constraints on a tree

7.5.2.3: The 'WHERE' AQL Clause as Source of Constraints on Trees

The WHERE clause allows AQL queries to introduce further constraints either directly on data items defined in the FROM clause or data items accessible through relative paths. These constraints can be constraints on hierarchy or constraints on archetype attributes as shown in Figure 32.

Even though the sample query in Figure 32 points at a data element that is defined in the SELECT clause ('Systolic'), this is not necessarily the case all the time. The AQL syntax and semantics allow the WHERE clause to point at any node using a relative path based on the data items defined in the FROM clause. Therefore, the WHERE clause may introduce constraints on data items that are not included in the SELECT clause.

The example AQL query in Figure 32 actually uses this feature of the WHERE clause to refer to a node named 'value', which is a child node of the 'Systolic' node. The 'value' node has an attribute named 'value', and a constraint is placed on this attribute.

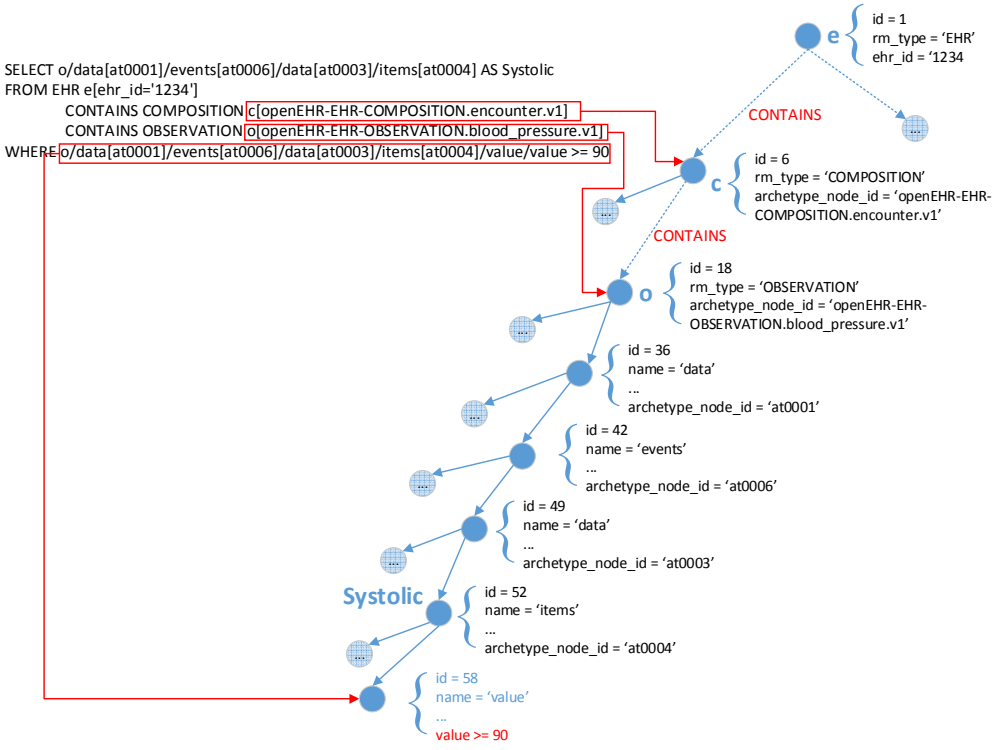


Figure 32: AQL WHERE clause as constraints on a tree

7.5.3: Mapping Tree-based AQL Processing to Tree Pattern Queries

The mapping of AQL query clauses to constraints on hierarchy and attributes of tree nodes establishes the AQL query semantics based on trees. However, the discussion in 7.5.2 is a textual definition of these mappings, even though it is supported by visual representation. This textual definition does not provide a means of expressing constraints on trees that can be processed by software implementation. Using the Tree Pattern Query (TPQ) representation, AQL queries can be expressed in a compact and platform independent way.

7.5.3.1: Tree Pattern Query Representation

A TPQ (Lakshmanan, Wang, and Zhao 2006) is a specialised representation that depicts the parent-child and ancestor-descendant relationships of nodes on a tree. TPQ processing, also called TPO matching, applies a TPQ on a tree and returns tree nodes that match the pattern defined by the TPQ.

The base TPQ representation for queries developed in this thesis expresses the ascendant-descendant relationship between nodes using double edges, and the

parent-child relationship with single edges following the definition from (Amer-Yahia et al. 2001). This base representation is extended with constraints on node attributes.

The use of TPQs to represent query semantics follows the same approach as in Section 7.5.2, based on AQL query clauses.

7.5.3.2: Mapping Tree Constraints of FROM AQL Clause to TPQ

FROM clause of AQL introduces only ascendant-descendant constraints on nodes in a tree, based on the CONTAINS keyword as depicted in Figure 30. Figure 33 provides an extension of this scenario: the TPQ representation of the FROM clause with constraints on attributes in addition to constraints on hierarchy.

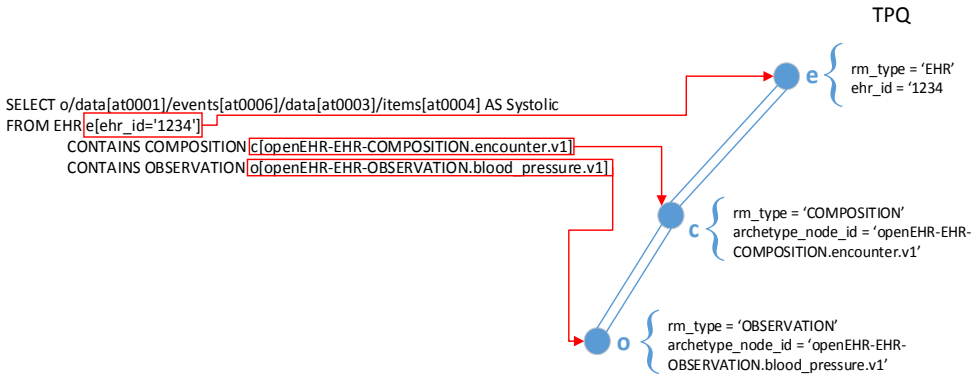


Figure 33: AQL FROM clause as a TPQ

Figure 33 shows that the constraints introduced by *FROM* clause can be expressed with the TPQ semantics. For the purposes of clarity, the diagrams that include TPQs do not show all the constraints on node attributes.

7.5.3.3: Mapping Tree Constraints of SELECT AQL Clause to TPQ

The *SELECT* clause may use relative paths to point at nodes based on the nodes defined by the FROM clause. These relative paths may include predicates which express constraints on node attributes.

In order to express relative paths with TPQ, the relative paths are transformed to an ascendant-descendant relationship. This transformation relies on the fact that every parent-child relationship introduced by the components of a path is also an ascendant-descendant relationship. This transformation is complemented by a constraint on an attribute, which uses the concept of a derived attribute. The

value of the derived attribute is based on the relationship between the root of the relative path and the last node on it: the absolute path of the last node on the relative path can be obtained by following the relative path on top of the absolute path of the root node. Figure 34 shows how relative path extension to TPQ is represented.

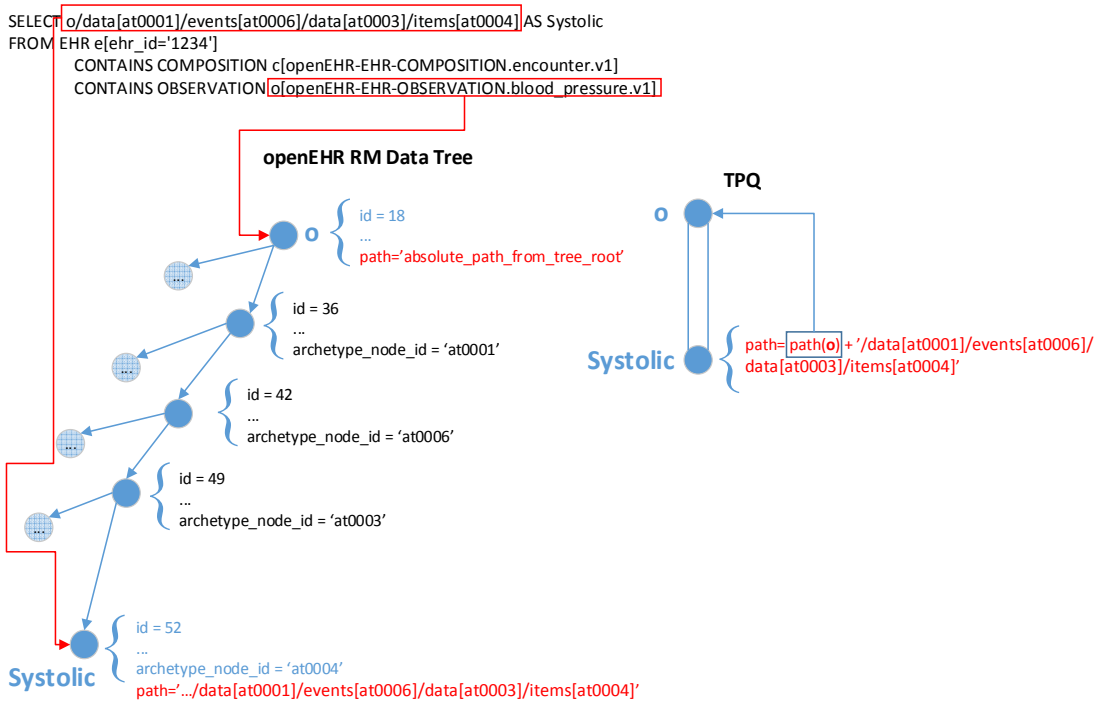


Figure 34: AQL SELECT clause as a TPQ

Figure 34 shows the RM based data tree and TPQ side by side. Each node on the tree has its absolute path from the root of the tree assigned to its path attribute. The value 'absolute_path_from_tree_root' of the path attribute of node **o** is a placeholder value used in the diagram for the purposes of clarity.

The node with the alias **Systolic** is reachable from **o** by following a series of nodes. Therefore, its absolute path from the tree root can be obtained by taking the absolute path of **o** and appending the relative path of each node on the path recursively. The reachability of **Systolic** node from **o** also implies that it is a descendant of **o**.

The TPQ in Figure 34 expresses this relationship through connecting **Systolic** to its ascendant **o** with a double edge and introducing a derived attribute constraint on a path by referring to the path value of **o**. Since TPQ represents a pattern and not any specific RM based data tree instance, the value of **o**'s path

attribute is referred to as **path(o)** which means that this value must be resolved by the actual implementation of TPQ.

The only actual value used in the derived attribute is the relative path of **Systolic,** which is available from the *SELECT* clause. This relative path is independent of the actual absolute path of **o** and **Systolic** so it can be used in the TPQ as it is. The underlying requirement for the derived attribute value approach is that the absolute path of every node from the root is assigned to its path attribute. The implementation details of this approach are discussed in Chapter 8.

7.5.3.4: Mapping Tree Constraints of WHERE AQL Clause to TPQ

The WHERE clause of AQL can represent complex conditions for filtering via the use of Boolean operators. Figure 35 depicts a rather simple example of the use of WHERE clause, in which filtering criteria for a numeric value is defined using a data item at a relative path to OBSERVATION **o**. In this simple case, the data item pointed at by the WHERE clause is represented as an anonymous node in the TPQ. The anonymous node uses the relative path representation approach in addition to another constraint on an attribute named value.

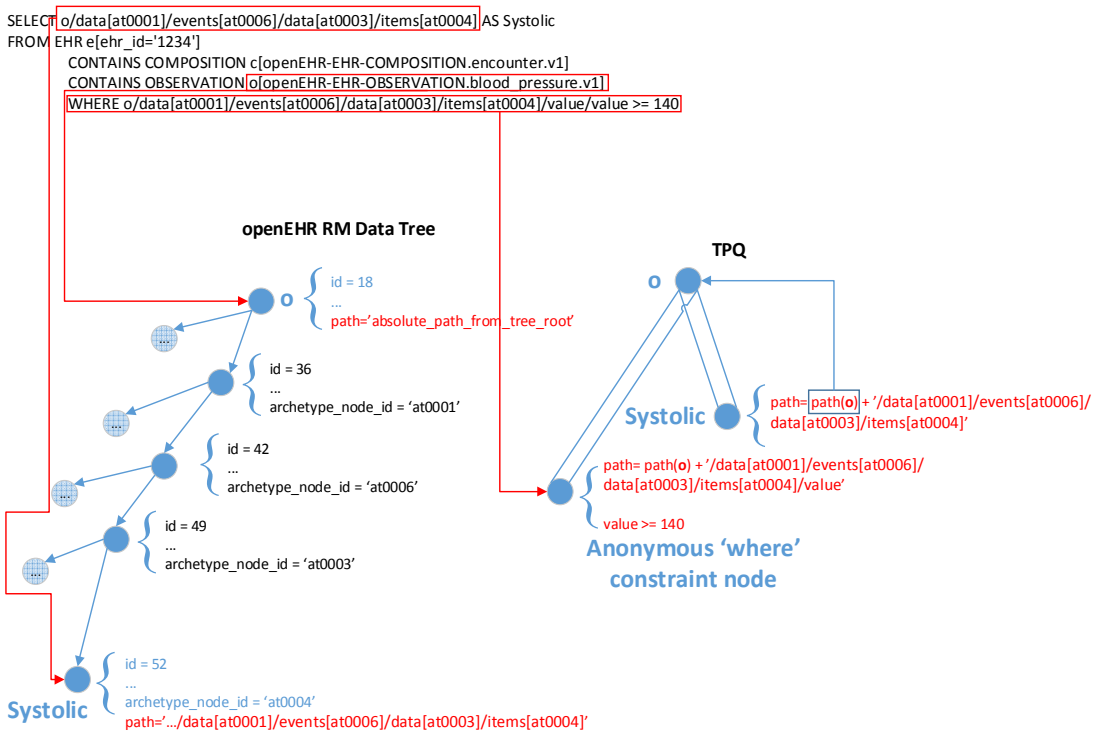


Figure 35: AQL WHERE clause as a TPQ

The TQP representation is sufficient to express the fundamental semantics of AQL clauses. However, the use logical operators for more complex queries as well as various unspecified aspects of AQL processing requires further extensions to basic TPQ representation.

7.5.4: Logical Operator Support in Tree Pattern Queries

The support for logical operators enables AQL queries to express complex logic for accessing RM based data. The Boolean logical operators are supported in the following ways in the AQL grammar:

- Combining CONTAINS expressions in the FROM clause

  AQL provides support for expressing ascendant-descendant relationships grouped together through AND, OR and NOT operators within the hierarchy defined by the FROM clause. The AQL specification (Ma, Frankel, and Beale 2014) describes support for these operators as:
  "*Boolean operators (AND, OR, NOT) and parentheses are used when multiple containment constraints are required.*"

- Combining WHERE clause conditions

  Multiple constraints can be introduced in the WHERE clause by connecting these constraints using Boolean operators (AND,OR,NOT). The Boolean operators can be used to connect path-constraint pairs, or they can be used to express multiple constraints in a predicate.

- The unspecified behaviour of SELECT clause

  AQL can define multiple data points as results. Even though the AQL syntax does not explicitly define any Boolean operator support in this context, AQL implementation needs to establish an implicit Boolean operator connecting multiple data items.

Previously developed mappings from AQL to TPQ representation cannot express queries which use these Boolean operators without extensions. The scope of extensions to TPQ representation to support Boolean operators is limited to AND and OR Boolean operators due to time constraints and rather frequent use of these operators. The draft AQL specifications include NOT and XOR operators as well.

The extensions to TPQ representation are discussed based on an extended version of the previously used AQL query, along with an openEHR template that is a modified version of the one depicted in Figure 20. The modified template defines a clinical encounter as before, but body mass index (BMI) concept has been added

alongside the blood pressure measurement. Figure 36 contains a screenshot of this clinical template, which is used as the target of the AQL query with Boolean operators.

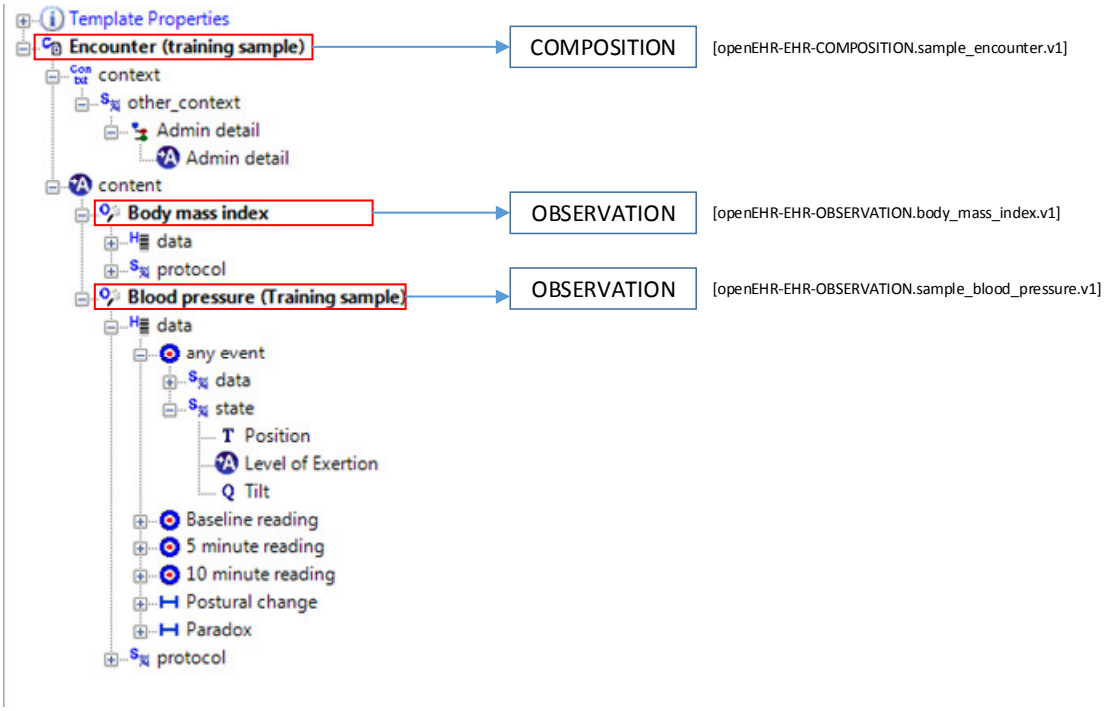


Figure 36: Extended openEHR template

7.5.4.1: Expressing Boolean Operators for FROM Clause in TPQs

The use of AND or OR Boolean operators within the FROM clause requires that the relationship between sibling data items are explicitly defined. The template in Figure 36 has a COMPOSITION with two OBSERVATIONS. An AQL query that selects both OBSERVATION data items is provided in Figure 37.

The query in Figure 37 selects two different data items, both having the same RM type: OBSERVATION. It is targeted at the clinical model represented by the template on the right and it needs to define the bmi and bpressure data items in the FROM clause so that they can be expressed as query results in the SELECT clause. The structure of the template makes body mass index and blood pressure data items siblings under the content field of the parent archetype (Encounter).

The query uses parenthesis and AND Boolean operator to explicitly describe the structure of the data the query is targeting. The Boolean operator is required to clarify the relationship between bmi and bpressure. Without this operator, AQL implementation could process data instances where there is only bmi data item (assuming OR) and another implementation could exclude the same data instances

(assuming AND). Therefore, when the TPQ contains multiple data items with a shared parent the interpretation for their existence must be explicitly expressed using Boolean operators.

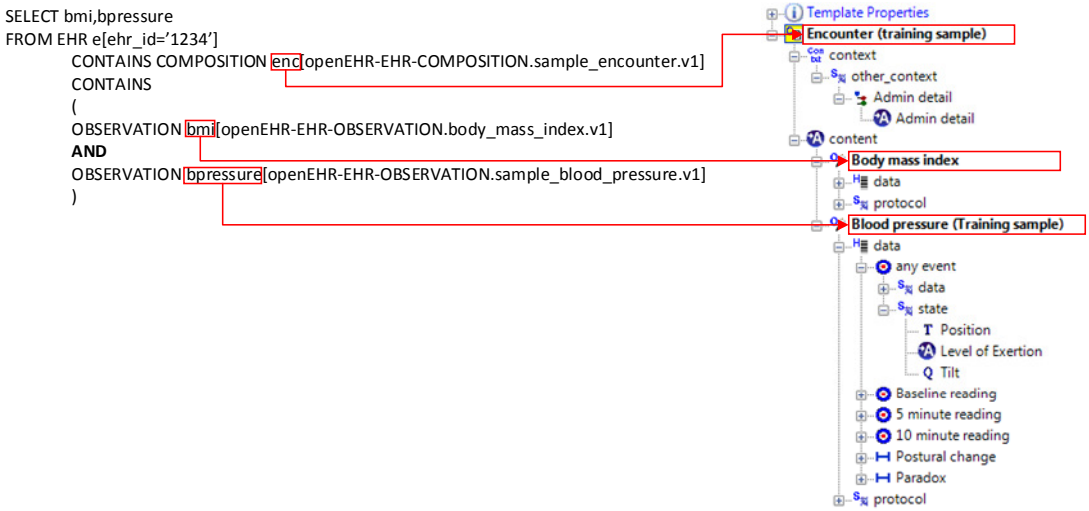


Figure 37: AQL with Boolean operators

This thesis uses a node representation of logical operators similar to (Izadi, Härder, and Haghjoo 2009) for containment constraints defined by the FROM clause. Figure 38 provides this representation based on the query from Figure 37.

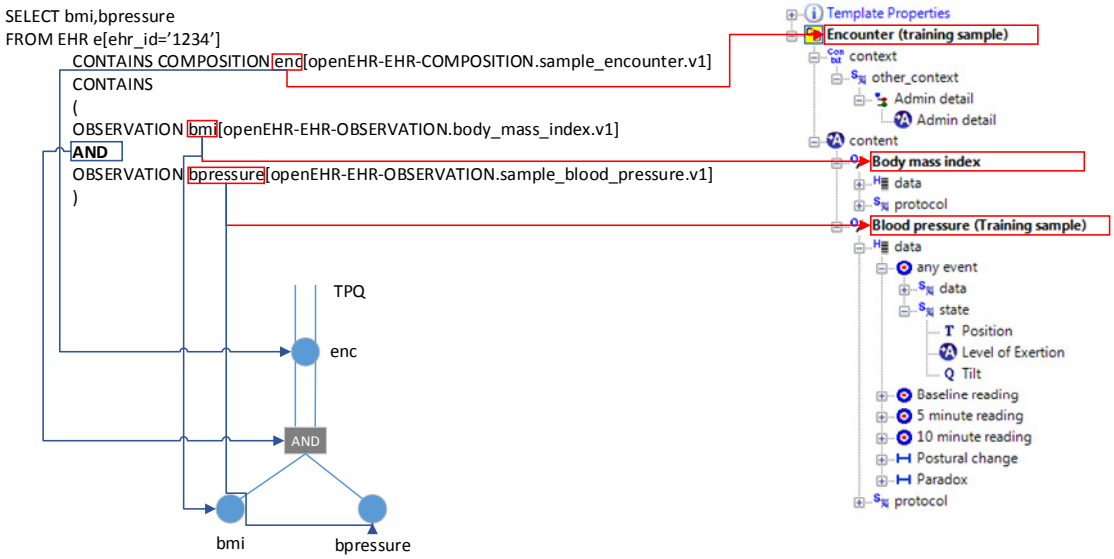


Figure 38: AQL with AND operator and its TPQ representation

Figure 38 depicts the AQL query, the template and the resulting TPQ for the FROM clause. The rectangular AND node is used to define the structural constraint

with a Boolean operator. The AND node is connected to ‘enc’ node with double edges to maintain the ascendant-descendant relationship, but it is connected to its operands with single edges to emphasize that they are operands of the AND node. The AQL query could have used OR operator in the CONTAINS statement which could then be expressed with the TPQ representation in Figure 39.

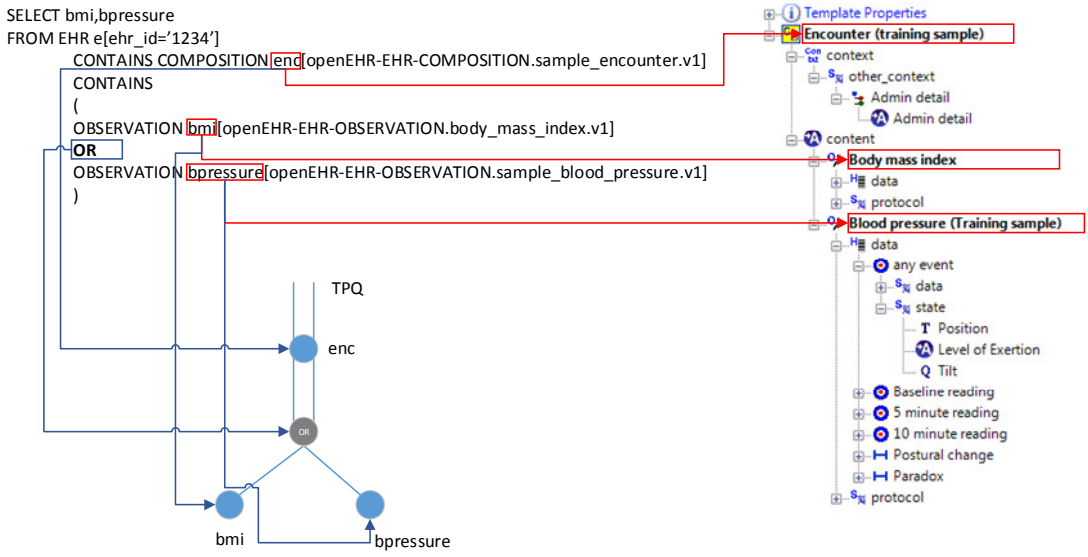


Figure 39: AQL with OR operator and its TPQ representation

The semantics of the connections of the OR node with other nodes in this diagram is the same as the AND node in Figure 38: the ascendant-descendant relationship is preserved by the OR node. Recursive uses of logical operators in the FROM clause can be expressed in the TPQ representation following the same pattern.

7.5.4.2: Expressing Boolean for SELECT Clause in TPQs

AQL specification does not include support for logical operators for the data items defined in the *SELECT* clause. However, consistent interpretation is required for these items in the context of TPQ representation. The data items defined by the SELECT clause are based on the ones defined by the FROM clause. They can be the same, or they can be descendants, which are accessible via relative paths. The TPQ representations of constraints on hierarchy and attributes that are introduced by SELECT clause have been discussed in Section 7.5.3.3. However, these constraints actually require special treatment because of the unspecified semantics of AQL query processing behaviour.

When there are multiple data items defined in the SELECT clause, AQL processing implementation may or may not allow returning empty values for the items that cannot be found in data. Figure 40 depicts various RM based data instances in tree form along with a TPQ that has data items introduced by the AQL SELECT clause.

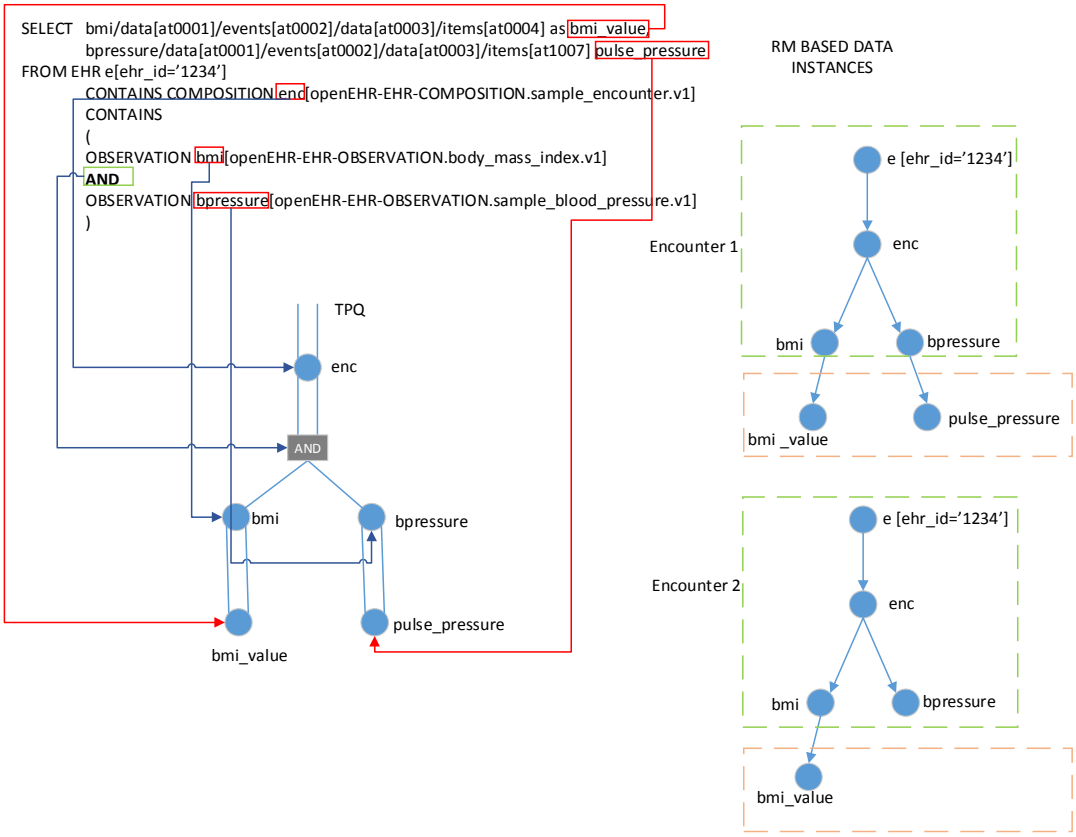


Figure 40: AQL SELECT clause with multiple data items

Figure 40 includes an AQL query that selects the bmi value and pulse pressure value based on the encounter template that has been used in Figure 36. The TPQ representation of the query includes 'bmi_value' and 'pulse_pressure' nodes in the TPQ using the descendant representation. The data instances on the right of the diagram in Figure 40 represent two different encounters during which RM based data instances have been created. However, the pulse pressure was not recorded in Encounter 2. Since the openEHR template defines pulse pressure as an optional value, these two data instances are both valid.

The TPQ here interprets the relationship between "bmi_value" and "pulse_pressure" nodes and their parents based on the same semantics expressed by the CONTAINS statement in the FROM clause. That is, the existence of a

“bmi_value” node somewhere below the “bmi” node is a condition that has to be satisfied by a data tree for that tree to provide a match for this TPQ. The same requirement exists for “pulse_pressure” and “bpressure” nodes. Therefore, Encounter 2 would not be considered a match for this TPQ.

This TPQ implies an AND operation on the nodes based on the AQL SELECT clause. The AND semantics is not explicitly defined by the AQL query but arises due to the way these nodes are included in the TPQ structure.

If the expected behaviour of the AQL processing implementation is to return pulse_pressure as an empty value for Encounter 2, the TPQ would be expressing a structural condition that would not correctly represent the expected implementation behaviour. This conflict reveals the requirement to distinguish TPQ nodes introduced by the SELECT clause from the ones introduced by the FROM clause.

Data trees that cannot satisfy the constraints on the hierarchy of the nodes introduced by the FROM clause should not be included in further processing. However, data trees that fully satisfy these constraints but only partially satisfy the constraints introduced by SELECT clause may be included in the results based on configuration of query processing or AQL implementation’s preference of one interpretation of SELECT clause over the other.

The interpretation that allows empty values to be returned for data items defined in the SELECT clause requires the TPQ to distinguish between nodes introduced by FROM and SELECT clauses. In this case, SELECT clause based nodes have an optional structural constraint, termed “optional containment” in this thesis. There is also a requirement to discard data instances in which none of the optionally contained TPQ nodes introduced by the SELECT clause exists. Therefore, this interpretation can be expressed by adding optional constraints on the hierarchy of nodes introduced by the SELECT clause nodes along with a Boolean operator that eliminates data instances that contain none of these nodes. Figure 41 depicts this approach.

Figure 41 depicts the optional constraint on hierarchy using dashed edges to ‘bmi_value’ and ‘pulse_pressure’ nodes from their respective parents. The OR Boolean operator ensures that data trees that contain none of the nodes from the SELECT clause are not returned since the existence of none of these nodes would result in a false Boolean value. This OR operator checks the existence of its operands and not their containment under their parents.

Introducing this OR operator node to the TPQ changes its structure from one in which each node in the TPQ has a single parent to one in which some nodes

having multiple parents, such as ‘bmi_value’ having ‘bmi’ and the OR operator node as parents.

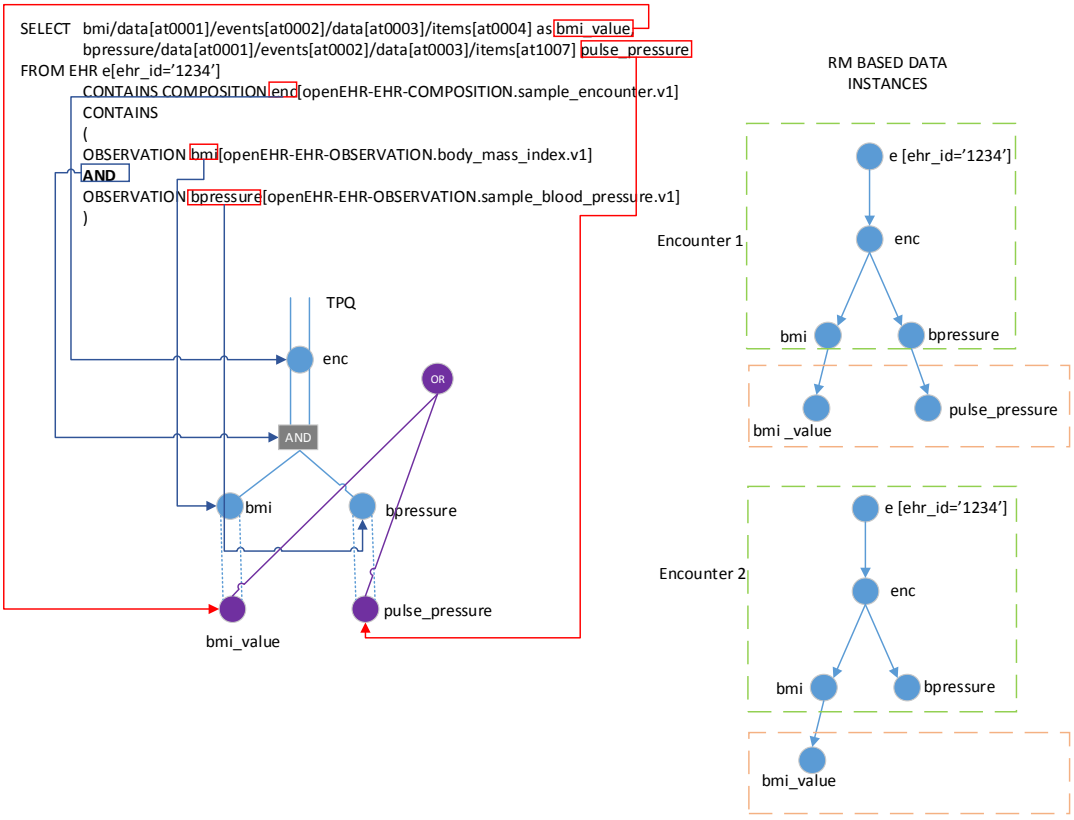


Figure 41: AQL SELECT clause: logical OR interpretation

7.5.4.3: Expressing Boolean Operators for WHERE Clause in TPQs

The WHERE clause supports the use of Boolean operators for connecting constraints on the data items it defines in addition to the capability to use nested Boolean operators. The optional containment representation is used for nodes introduced by the WHERE clause. The details of this requirement are discussed in depth in Chapter 8.

Figure 42 extends the query in Figure 42 with a WHERE clause that introduces multiple constraints. Figure 42 depicts an AQL query with three conditions in the WHERE clause which makes use of grouped Boolean operators. The diagram in this figure shows how Boolean operator nodes for WHERE clause constraints are represented in the TPQ, extending the previously introduced implicit OR operator based on the SELECT clause. For the purposes of clarity, attribute constraints are not explicitly depicted in the diagram in Figure 42.

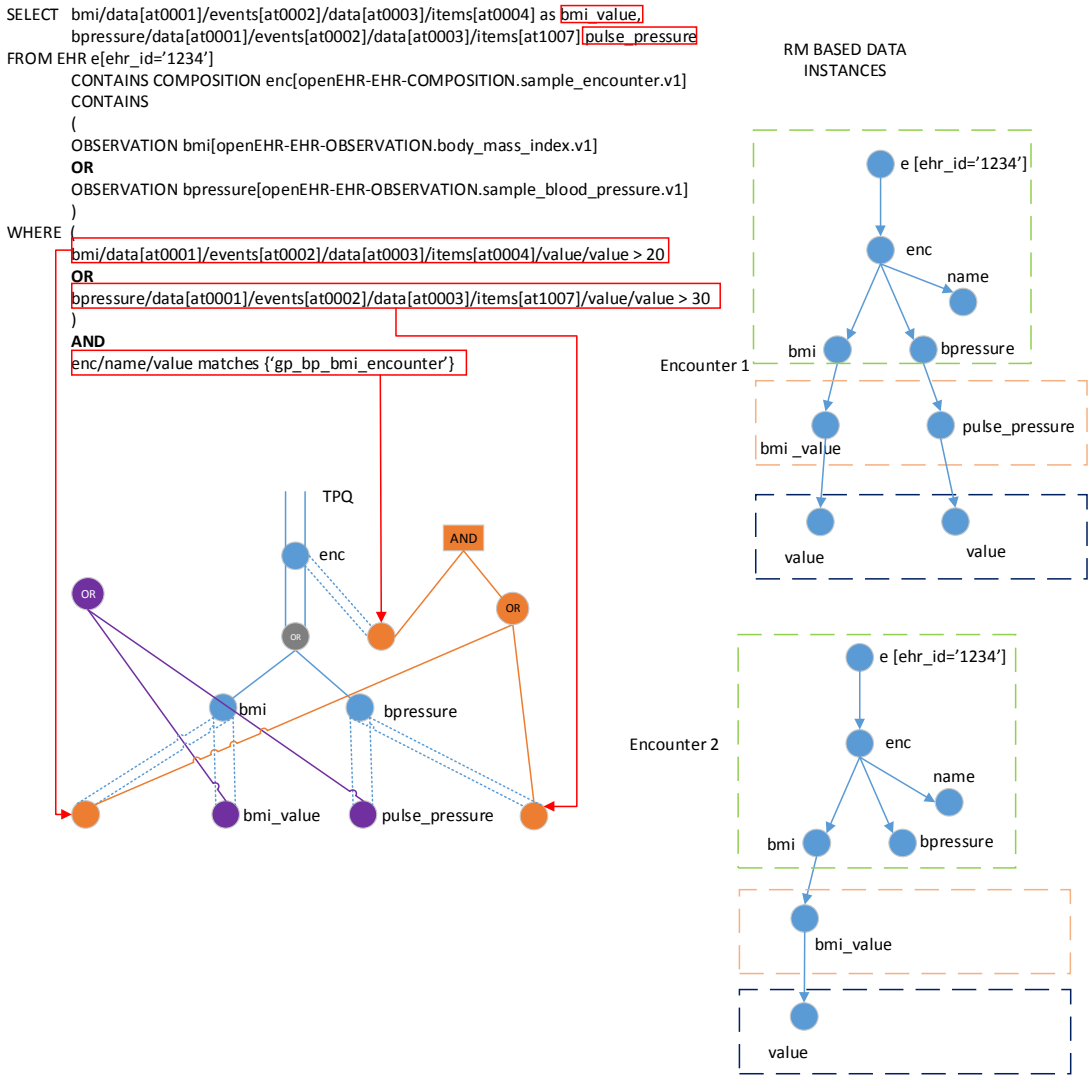


Figure 42: AQL WHERE clause with Boolean operators

The optional containment extension to TPQs is required for the TPQ to return values from the Encounter 2 data tree, as intended. Since the Encounter 2 data tree does not have the pulse pressure node, it also does not have the value node which would be a child of it. If the TPQ expresses the relevant constraint's node with a mandatory descendant connection to 'bpressure', the Encounter 2 data tree would be incorrectly excluded from query processing.

7.6: Relevant Research

The tree based abstraction developed in 7.5 aims to deliver the benefits of a flexible representation for RM based data. This approach leads to an openEHR specific model for persistence, but this specialisation does not necessarily mean

that outcomes of wider research on the underlying tree representation cannot be used.

Where available, findings from research on trees and operations on trees can offer the possibility of improving various aspects of the approach developed in 7.5 A review of tree processing methods has been performed to this end, with an initial focus on literature on XML processing. XML's successful use for representing openEHR data hints at the possibility of adopting research related to processing its underlying abstractions to improve the TPQ based AQL processing approach.

Improving the performance and capabilities of XML processing is an active topic of research. The use and processing of XML are relevant in a number of other fields of research, including but not limited to information retrieval, large-scale data processing and database systems, mostly due to XML's ubiquitous nature.

A relatively recent review of tree matching in the context of XML retrieval, (Tahraoui et al. 2013) provides a list of widely used methods for both exact and approximate matching. The exact matching methods covered in (Tahraoui et al. 2013), described as structural and holistic join methods, along with sequential matching methods, provide a number of options for implementing the TPQ matching developed in Section 7.5.

The structural join approach to twig pattern matching is defined as follows in (Tahraoui et al. 2013):

"… (1) decomposition, (2) matching and (3) merging. Firstly, a twig pattern is decomposed into a set of basic parent–child and ancestor–descendant relationships between pairs of nodes. In the second phase, each binary relationship is separately executed using structural join techniques and its intermediate results are stored for further processing. The final result is formed by merging these intermediate results. …"

The definition of twig patterns from the same study is:

"twig patterns, i.e., small trees"

The structural join approach is considered as an improvement over the traversal methods, which adopt the approach of walking the nodes for the target of the query one by one. As (Al-Khalifa et al. 2002) shows, as the size of the target for search operation grows the performance of traversal methods decreases.

The relational database implementation approach to structural joins (Al-Khalifa et al. 2002) is usually considered less efficient than specialized XML databases such as TIMBER (Jagadish et al. 2002). However, discussion of

'containment queries' (Zhang et al. 2001) using relational databases provide insight into implementation of applying constraints on hierarchies, which is relevant to TPQ based implementation of AQL. In depth analysis of the reasons behind performance problems with use of relational databases for queries on XML (Zhang et al. 2001) shows that various extensions to relational database features can improve performance:

"While it is premature to make concrete predictions, we are optimistic that by combining better join algorithms with better cache utilization, an RDBMS will be able to natively support containment queries efficiently"

The use of relational databases for processing XML, such as using mappings from XPATH to SQL, is not always considered inefficient (Tatarinov et al. 2002) and there are many studies that present methods for representing XML data in a relational setting (Harding, Li, and Moon 2003).

Despite their performance advantages over traversal methods, the structural join methods introduce the problem of generating a high number of intermediate nodes as a result of repeated join operations between the components of the query patterns. A family of tree matching methods classified as holistic twig matching deal with this problem using special data structures to decrease the number of intermediate results (Tahraoui et al. 2013).

A significant amount of research for XML processing focuses on introducing either variations of families of algorithms outlined in (Tahraoui et al. 2013) or developing various indexing or processing methods to improve the performance of existing approaches. Examples of such studies include using a look-ahead approach to improve holistic join performance (Lu, Chen, and Ling 2004), using structural indexes to decrease number of intermediate results (T. Chen, Lu, and Ling 2005), encoding tree structures in a relational database (Weigel, Schulz, and Meuss 2005), using indexing methods for high performance XML retrieval in relational databases (Weigel et al. 2003) as well as developing indexing methods and labelling schemes for tree pattern matching (H. Wang and Meng 2005), (Rao and Moon 2004), (H. Wang et al. 2003), (Lu et al. 2005), (Lu, Meng, and Ling 2011), (Barbay 2005), (Arion et al. 2007).

Research on efficient implementation of query languages for XML content provides methods that are applicable to tree structured data such as using indexes based on the trie data structure (Bodon and Rónyai 2003) for XPath query processing (Brenes et al. 2008) or extracting tree patterns from XQuery queries for faster query processing (Arion et al. 2006).

The query languages for XML content are implemented via different methods. The introduction of formal methods for implementing these languages involves dealing with language features that are significantly more complex than the features offered by AQL at the time of the writing of this thesis. Results of research on implementing these languages partially or fully, based on formal representation and methods, is potentially useful for implementing AQL processing based on the tree based approach. Partial Tree Pattern Queries (PTPQ)(X. Wu et al. 2011), (Theodoratos et al. 2006), Generalized Tree Pattern Queries (GTPQ) (Zeng, Jiang, and Zhuge 2011) and development of a special pattern matching language (Benzaken, Castagna, and Miachon 2005) are examples of research that focuses on features of these XML query languages.

The above studies show that the benefits realised through querying XML data with specialised languages have prompted a significant amount of research on methods for improving performance and integrating these languages into other well-established frameworks such as relational databases. Developing indexing methods and algorithms that improve performance of a particular aspect of a query is a common research topic.

Much existing research can be used to improve various aspects of the implementation of the openEHR persistence abstraction. In particular, methods for integrating XML content and query languages into relational databases can help in development of a relational database implementation that solves the design problems encountered with the pilot Opereffa framework, as discussed in Section 6.3. Wider research focused on specific aspects of queries on tree content, such as ascendant-descendant or parent-child relationships, will inform adoption of specialised algorithms for AQL processing, based on operations on trees.

In summary, tree based persistence abstraction is considered a valid approach to meeting the requirements identified in Section 7.1, while also bringing significant benefits from wider existing research findings.

7.7: Summary

The motivation for building a persistence abstraction for openEHR stems from the requirement to implement this key functionality for openEHR based on different options. Recent research has been delivering high-performance, specialised persistence systems for handling large data sets, based on distributed

computation to facilitate machine learning and data analysis, and complementing the capabilities of existing persistence systems such as relational databases.

Being able to implement openEHR persistence across a variety of persistence systems has been identified as a promising approach to support openEHR and CDS integration in a large number of settings, with the goal of making use of the comparative advantages of the underlying frameworks. This approach provides the bridge between electronic health records and research on scalable machine learning which in turn allows further research on the intersection of two major topics of research.

The feasibility of related implementations must be improved to enable the use of openEHR on multiple persistence systems. To this end, a new tree representation of RM based data was developed and TPQ representation chosen as the method for expressing AQL semantics using trees. This approach satisfies the requirements for expressiveness, extensibility, feasibility of implementation, consistent representation and scientific relevance identified in Chapter 7.
This chapter described a novel tree-based abstraction method, designed to support a multi-persistence system architecture for consistent openEHR and CDS integration. This is further explored experimentally in Chapter 9, with an implementation of this method for persistence of openEHR RM data, for a CDS based on a BN created for analysis of clinical data in the domain of ophthalmology.

Chapter 8: XINO Architecture for Persistence

This thesis explores the feasibility of an openEHR based CDS architecture via an experimental approach. The term “openEHR based architecture” means using all the support openEHR specifications provide for computable health for the components of the architecture whenever possible.

A unifying aspect of both clinical care and CDS use cases is data access. Even though openEHR methodology can be followed for design and implementation of both clinical information systems and CDS functionality, orthogonal data access patterns may require switching from an openEHR based approach to a more implementation and platform specific one based on the requirements.

Even though this specialisation may be required to benefit from the strengths of a particular technology such as document database or a distributed file system, it is a step back from the conceptual integrity of openEHR. An interesting research question is therefore, would it be possible to preserve the use of openEHR concepts for data access across different use cases and data volumes.

The tree-based persistence abstraction developed in Chapter 7 is the first component an openEHR persistence framework called XINO that has been developed to answer this question. The second component of XINO is the mappings from the tree structures and operations on them to functionalities of various persistence systems. The design goal behind XINO is to introduce a small number of operations that can be implemented across a variety of persistence systems, which leads to an openEHR persistence implementation.

Two key requirements must be fulfilled in order to achieve this design goal: RM based data needs to be persisted based on a representation that can be supported by different persistence systems and a number of previously defined operations on data must be implemented using the features of the target persistence system. The implementation used in this thesis is based on Postgresql relational database server (Momjian 2001).

The choice of a relational database as the target persistence system for implementation is intentional. Relational databases are used extensively in information systems implementation across a wide range of domains and due to their maturity, stability and emphasis on data consistency they are regularly used in healthcare information systems.

However, despite their capabilities relational databases present a challenging option for healthcare data modelling. The underlying relational model

(Codd 1970) can become too verbose and complicated when it comes to representing clinical data and performing operations on it, potentially introducing performance problems as well. It is worth mentioning that specialised software frameworks that can deal with characteristics of clinical data have been under development since as early as 1966. The development of MUMPS system (Bowie and Barnett 1976) at Massachusetts General Hospital, pre-dates Codd's hugely influential paper by 4 years and its derivatives are still used in successful, large-scale health information systems such as VistA (Brown et al. 2003).

Data creation and manipulation in relational database implementations are subject to more strict constraints in a relational database compared to persistence systems that handle large amounts of data such as Hadoop (Borthakur 2007). Relational databases provide strong support for data consistency, but this support leads to limits on performance as data size grows. Most of the recent large scale persistence systems are able to overcome performance and scalability limitations of relational databases by waiving guarantees provided by relational database implementations. Concepts such as eventual consistency (Vogels 2009) enable large scale distributed persistence systems, often characterised with the term 'NOSQL', to handle large volumes of data (Cattell 2011)

Therefore, due to both relational data modelling challenges and potential performance problems of relational databases encountered during processing of hierarchical data (Celko 2012), relational database implementation is probably the most exigent configuration for XINO.

The primary reason for choosing Postgresql for the particular XINO implementation used in this thesis, despite these challenges, is the size of the industry and research community that works on relational databases. A relational database implementation of XINO that can adequately support data access scenarios for both clinical care and machine learning provides a versatile openEHR based platform using a single persistence system. Therefore, this thesis has explored the feasibility of such a configuration by implementing XINO on a Postgresql database server.

8.1: Design Principles for Persisting openEHR Data in a Relational Database

The flexibility of relational algebra (Codd 1970) presents a number of options for persisting openEHR data in a relational database. Implementation specific extensions provided by different relational database servers increase the number of

these options if implementers decide to trade portability in exchange for benefits of specialisation.

It is not possible for this thesis to cover all options for persisting openEHR data in a relational database when an experimental approach based on implementation is adopted. Both the time frame and the skill set that would be required would be unattainable. However, a few guiding principles are used to arrive at a XINO based persistence design in a relational database. Different approaches to a relational implementation of XINO can be used, as long as they comply with these principles.

The first design principle, probably the one with the highest priority, is handling changes in the structure of data. openEHR is designed to represent a potentially infinite number of clinical concepts using a small number of data types, so not including this characteristic in persistence design is bound to produce an unmanageable implementation. Since data in a relational database must reside within tables defined by a schema, arrival of clinical data with continuously changing structure should not require changes to the database schema. Even though these changes could potentially be accommodated programmatically, i.e. new schemas and tables could be generated based on openEHR models that create the data, there is still the problem of not having an upper bound on the number of schemas that may be required. Therefore, a database schema that is resilient to changes in the structure of data is a crucial requirement from a design point of view.

The second design principle, which is introduced by this thesis' attempt to explore limits of openEHR in supporting both clinical care and CDS scenarios, is applicability to most, if not all persistence systems that can be considered as alternatives to relational databases, especially for machine learning tasks. This applicability is required to ensure that data volume does not introduce limits on functionality and use of a relational database is built on an approach that can be used with alternative persistence systems.

The third design principle, which could be considered implicit in any information system implementation, is performance. The persistence design should consider users' performance expectations from the openEHR implementation for both clinical care and CDS functionality. Precisely defining performance in settings as behaviourally complex and diverse as clinical information systems and CDS implementations, made even more complex by factors such as data volume, security, etc. is hard, if not impossible. However, this does not mean that performance can be dismissed as a design principle.

Conforming to these design principles for a large number of use cases defined by clinical information systems and CDS implementations is a challenge. Furthermore, these principles may conflict at times. Therefore, this is a multidimensional optimisation task; a particular implementation may choose to put more emphasis on a single principle.

8.2: Relevant Research

The Postgresql based implementation of XINO consists of mappings from the tree representation of data and TPQs (as described in Chapter 7) to relational data and SQL. This approach has been developed and improved through extensive experimental implementations, guided by the design principles set by this thesis, and relevant published research. Two pertinent lines of research provide valuable insight into how XINO's design goals can be accomplished: representing clinical data using the Entity-Attribute-Value (EAV) model and querying XML documents, especially in relational databases. As with the development of the abstract data representation for openEHR, the findings from these lines of research may not be directly applicable to a relational database based implementation, but they can be adopted and used for implementations based on other persistence systems.

8.2.1: EAV Approach to Relational Persistence

Persisting and processing data that has highly variable structure without having to make changes to the underlying relational database schema is a frequently arising requirement in software development. The diversity of both the type and structure of clinical data implies that clinical information systems design and implementation must often fulfil this requirement except in cases of systems targeting very limited clinical scope.

A particular relational data model that has been used extensively in clinical data representation, with the aim of addressing this requirement, is EAV. This model provides a high level of flexibility via representing Entities (such as a patient, an operation or any clinical concept), Attributes (such as age and gender of the patient) and Values (such as 33, the actual numeric value of a patient's age attribute) at the database level with three tables in its most common form. Derivatives of this most common form may use slightly different table structures.

This approach allows any concept to be defined at the database level by creating an entity, assigning attributes to it and creating data instances with actual data values that reference these entities and attributes. Change to domain

concepts, such as adding a new attribute, consists of inserting a new row to the attributes table. Following this step, new data instances with this attribute can be created by inserting a value for the attribute with a reference to its definition in the attributes table.

Discussions about this modelling approach regularly emphasise its shortcomings, especially in long-term management of data and performance problems associated with a large number of join operations, which are required to retrieve concepts that are represented as highly granular data items in the database. It is remarkable that despite heavy criticism and discouragement (Celko 2012) EAV modelling and its derivatives have found significant use in clinical data processing with relational databases.

The use of EAV and related approaches focusing on generic relational database schemas for clinical systems implementation has been evaluated in depth by various studies, taking into account the characteristics of clinical data, with a more detailed approach compared to the rather generic treatment of (Celko 2012).

These studies provide both positive and negative aspects of generic relational modelling for clinical data. (Helms and McCanless 1990) questions the suitability of relational databases for hierarchical clinical trial data, while (Johnson 1996) presents generic data modelling as a promising approach. As more implementations that use generic and EAV influenced designs emerge during the 90s, in parallel to the larger adoption of relational databases, methods for dealing with its shortcomings and problems are developed, for example for querying EAV data in biomedical databases (Nadkarni 1997).

Making use of automatically generated SQL is one such method, which can be implemented in a number of ways, such as developing a query kernel that generates SQL queries on an EAV database by making use of metadata (Nadkarni 1998). The use of metadata can be seen as a precursor of capabilities provided by AQL, in the sense that it allows query operations to be defined without the details of the actual relational design.

The extension of pure EAV to EAV/CR (Classes and Relationships) (Nadkarni et al. 1999) is another attempt to improve the EAV approach by making use of object-oriented data modelling. Despite having query performance disadvantages (R. S. Chen et al. 2000), this unified approach introduces a consistent domain modelling practice for EAV design and implementation.

However, the integration of object oriented concepts with an EAV design is not without challenges, as expressed in the following quote from (Dinu and Nadkarni 2007):

"Typically, not all classes in the data will meet the requirements for EAV modeling. Therefore, production schemas tend to be mixed, with a given class represented in either conventional, EAV or hybrid form as appropriate. The introduction of an EAV component to the schema, however, mandates the creation of a metadata component to capture the logical data model for the EAV data: in the absence of this, the EAV component is essentially unusable. The necessity (and difficulty) of creating a complex meta-schema, as well as a code framework that is driven by it, is one of the major factors that has inhibited the more widespread use of EAV data models: the availability of open-source schemas and frameworks may gradually change this."

The challenges of making use of well-defined domain concepts for EAV representation leads to hybrid approaches that use both EAV and rather traditional relational design (Dinu, Zhao, and Miller 2007). The requirement to establish a method for mapping the domain models to EAV representation of data is discussed in healthcare specific cases as well, such as extracting data from an EAV based EHR system to a format based on ISO/EN 13606 (ISO/EN 13606 2012) archetypes (Duftschmid, Wrba, and Rinner 2010). These studies show that using object oriented concepts to overcome data representation problems of the EAV model is a valid approach supported by research.

EAV's widely acknowledged performance issues have also been targeted by research. An adaptation based on the characteristics of data access can help in improving the performance of the EAV approach. Approaches such as extending the EAV model for read-only data warehouse implementations (Paul and Hoque 2011) assume particular access patterns, which enable design extensions to EAV that perform better.

Even though clinical information systems are normally used to perform patient-centric queries during clinical care, i.e. not fetching records of multiple patients, the nature of the clinical care process itself favours a similar "*read-optimized"* data access scenario. Patient care frequently includes access to a patient's medical history, which requires most, if not all the patient data to be accessed. As medical data accumulates in the patient's EHR, large volume reads are bound to happen, with relatively far fewer writes; each care episode may write some data, but it is likely to read all data that has been created before. This assumption makes read optimized EAV design a strong option for clinical information systems and CDS implementation.

Due to its flexibility, EAV model can also represent hierarchical characteristics of data by expressing parent-child relationships between data items by defining attributes such as 'parent' or 'children'. However, in a relational

database setting, this approach may lead to an arbitrary number of join operations, dependent on the complexity of the hierarchical relationships defined in the queries, and lead to significantly decreased performance as either the query complexity or data volume increases (Löper et al. 2012).

A high level decomposition of the topics covered in these studies revealed a number of key findings. First, despite being subject to criticism and suffering from well-known performance and data management issues for almost 25 years, the EAV design is still not obsolete. It is still considered as a design option for clinical data persistence with various extensions and modifications to help deal with the chronic problems it introduces. This is most probably due to the highly volatile structure of clinical data. Apparently, dealing with this volatility is so important that implementers are willing to forgo the performance gains that could be provided by less generic database schemas.

Second, building better defined representations of the clinical concepts is accepted as an improvement over the simplest EAV model. These improved representations provide mechanisms for data transformation, data extraction or improved query capabilities.

Finally, even though high level tools can isolate users from the complex SQL queries that would be required to access EAV data, the fundamental mechanism of table joins that must be used to build query results cannot be avoided. This introduces an inevitable performance problem, which would be further aggravated by the handling of hierarchical aspects of the data in an EAV model.

Ignoring for the moment the performance issues, re-evaluating these research findings in the light of the new assumption that all clinical data will be based on openEHR models, presents multiple new opportunities for improving an EAV based implementation.

The primary reason for opting for an EAV model for persistence - dealing with the structural volatility of data - is handled by openEHR by design. The openEHR RM guarantees that no clinical data instance will introduce a new entity type or attribute since all data is built of combinations of highly reusable types, brought together via archetypes. Therefore, the entity and attribute definitions are known in advance for every possible clinical data instance.

The type system introduced by the RM can be used to codify an EAV representation in advance, before any data is committed, and this encoding can be used to generate SQL queries automatically. The results from these queries will then populate instances of RM types without any need for semantic mapping, as would be required by some of the approaches mentioned previously. Therefore, the

openEHR RM and its type system can significantly eliminate one of the well-known problems of an EAV design. The openEHR RM can also improve the applicability of an EAV model to other persistence systems since the Entity and Attribute components of EAV can be encoded and kept out of the persistence layer, leaving only Value as a list made of most basic types representing actual data. This is a representation that can be supported by many persistence technologies including non-relational ones.

Not all aspects of an openEHR based approach to EAV are an improvement on the more traditional implementation. openEHR already offers a query language, AQL, which would be the natural choice to isolate users from writing SQL queries against the EAV. This is another improvement over various, case-specific approaches developed in other studies, but AQL queries have a strong focus on the hierarchy of data. Therefore, arbitrary join operations for enforcing the hierarchy constraints expressed in high level AQL would inevitably introduce performance issues.

Therefore, an openEHR based approach to an EAV design offers significant improvements, but in the context of relational databases some critical problems still remain, mostly around the difficulty of managing hierarchical aspects of structural data in a relational database. This difficulty is not specific to an EAV context - representing clinical data in a relational database is a frequently encountered requirement that can be interpreted as a particular instance of a more generic requirement, which is representing hierarchical data in a relational database.

This requirement has been studied in depth in its more general form due to the natural occurrence of hierarchical data in many domains. There are both relational data modelling approaches as discussed in depth in (Tropashko and Burleson 2007) and (Celko 2012) as well as custom extensions to the SQL language provided by relational database vendors. The methods discussed in both (Tropashko and Burleson 2007) and (Celko 2012) have wide applicability in a large number of scenarios. However, another field of research also provides a large number of results that are closely related to processing TPQs (which is how we represent AQL) as defined in Chapter 7: processing XML queries.

8.2.3: XML Query Processing

Processing of XML queries via query languages such as XPath (Clark and DeRose 1999) and XQuery (Boag et al. 2002) has extensive research associated

with it. Matching patterns in XML content has been studied in depth, due to XML's ubiquitous use in data exchange and storage. Research in this field focuses on processors for XML query languages, and storing XML in relational databases and native XML databases as well as in big data frameworks.

The results of these studies are relevant to the requirement of handling hierarchical aspects of AQL queries because the abstract design for AQL processing developed in Chapter 7 is similar to XML content processing. Moreover, these studies provide insight into both the specific scenario of storing XML content in a relational database and rather abstract algorithms that can be used in many implementation contexts beyond relational databases. Therefore, these findings can be used in multiple implementations of XINO, based on both relational databases and other persistence systems.

Current capabilities of AQL, especially in terms of expressing hierarchical relationships between query elements is functionally a subset of the capabilities supported by XPath and XQuery for the same purpose. Therefore, various methods and algorithms developed for XML processing, both in relational databases and other environments, may be considered insufficient to formalise or support complete scope of specialised XML processing languages but they may offer more utility in case of rather limited tree pattern matching cases for AQL processing.

Research publications on XML processing that are relevant to handling hierarchical aspects of AQL queries can be classified into two groups, in the context of a relational database based XINO implementation: relational and non-relational. The studies in the former group assume that XML is processed through use of a relational database, which implies use of SQL and widely supported database features such as indexes, while those in the latter group adopt a rather relaxed assumption regarding the means available for operations on XML content. This does not imply that studies in the second group are irrelevant though; modern relational databases support extension mechanisms to SQL that allow access to mainstream programming languages. Therefore findings from both these groups can be used in a relational database implementation.

(Zhang et al. 2001) discusses supporting "containment queries" in relational databases, defining the core concept of this study as follows:

"By "containment query" we mean queries that are based on the containment and proximity relationships among elements, attributes, and their contents."

This definition refers to components of XML content and (Zhang et al. 2001) discusses both performance issues and benefits of implementing queries that fall

within this definition. The approach to XML content representation adopted in this study of SQL based implementation of containment queries has some noticeable features in the context of XINO.

First, the representation uses an encoding of XML content that is aimed at efficiently performing containment queries. Use of an inverted index that encodes positions of elements in a document provides a more efficient means of answering hierarchical queries compared to simply expressing parent-child relationships in an EAV setting. Second, aside from this difference in handling hierarchy information, the representation of XML content in (Zhang et al. 2001) bears resemblance to an EAV approach, in the sense that the same relational schema can be used to persist any XML document without changes to it. Despite the performance problems it uncovers, this study concludes with an optimistic view of the use of relational databases for containment queries.

The data representation in (Zhang et al. 2001) is strongly influenced by the nature of the queries their study focuses on. Therefore, the representation of XML in this study is not a strong candidate for a generic representation method.

Representation of XML content in relational databases is a well explored research topic, which has produced many methods for this purpose. (Shanmugasundaram et al. 1999) discusses document specific representation of XML content in relational databases, based on the XML schema. Similarly, (Florescu and Kossmann 1999a) and (Florescu and Kossmann 1999b) discuss the relationship between various XML storage options in a relational database along with query performance using SQL. (Bohannon et al. 2002) establishes a cost-based optimisation method for finding the optimum relational representation of XML schemas where cost is defined by SQL query costs. (Du, Amer-Yahia, and Freire 2004) uses annotations of XML schemas to create the relational representation. Using XML schemas for constructing relational representation and querying is not always a straightforward method; it can lead to issues in query translation to XML form when XML schemas are recursive in nature (Fan et al. 2005) .

The different approaches to persisting XML employed by these studies show that whether to consider XML schema information or not for relational representation is very much a design choice. (Yoshikawa et al. 2001) defines these design options as “*Structure-mapping approach*” and “*Model-mapping approach*”; the former referring to XML schema driven relational schema construction and the latter referring to a fixed relational schema for all XML content. (Yoshikawa et al. 2001) uses the latter approach based on two key pieces of data: the full path of every XML node from the XML content root is used, along with a region encoding of

nodes to represent XML content. Another important aspect of XML storage in relational databases, re-building either partial or complete XML documents from their relational representation, is also discussed in (Yoshikawa et al. 2001). The method used for handling this requirement is to keep the entire text of XML content along with its encoded form.

The approach based on keeping both XML content and its relational representation in the database is reported to be implemented by a well known commercial relational database server product which also uses specialised numerical encoding of XML nodes (Pal et al. 2004). The path based representation of nodes approach is used as the basis of a fast indexing method for XML content in (Cooper et al. 2001) which is reported to outperform indexing mechanisms of a commercial relational database server.

(Haifeng Jiang et al. 2002) introduces a relational design that is based on a generic schema similar to (Yoshikawa et al. 2001) but its representation can support both parent-child and ancestor-descendant information explicitly. This strategy can improve query performance for ancestor-descendant queries in a trade-off with increased storage costs. In the context of AQL query implementation via TPQ matching, this specialised representation, used by (Haifeng Jiang et al. 2002), offers a significant advantage since AQL uses ancestor-descendant constraints heavily. (Harding, Li, and Moon 2003) provides another node encoding scheme that can support parent-child and ascendant-descendant queries.

Indexing mechanisms that rely on path information, along with numeric region encodings, are not solely of interest to XML persistence based on relational databases. They have also been studied extensively in other contexts such as implementation of query processors for XML. The findings of such studies are relevant and important in the context of a relational database implementation of XINO. This is because of the extension mechanisms to SQL, which are supported by all major relational database servers.

Therefore, assuming that these extension mechanisms are available to implementers, many potential improvements to TPQ matching based AQL implementation can be accomplished through the use of such research findings, which are usually classified as native XML processing and indexing approaches.

For example, (Han, Xi, and Le 2005) develops a hybrid index that uses both structure and value information of nodes. (Barbay 2005) introduces a specialized index structure for descendant elements queries. (Haifeng Jiang et al. 2003) develops another specialized index structure which offers optimized I/O performance for the same type of queries.

Aside from the relational database focused and native XML processing approaches, a third category of research for XML processing links native XML processing methods with relational ones, presenting a family of hybrid approaches (H. Wu et al. 2012), (Weigel, Schulz, and Meuss 2005), (Weigel et al. 2003).

Some of the results of this large body of research that focuses on XML query processing are relevant to the design principles of XINO. These are mainly the results of studies that focus on persisting XML content in relational databases, which exhibits features similar to the EAV model of relational persistence. Highly granular representation of XML content in a relational database is prone to performance issues, yet there has been significant effort to use relational databases for XML querying, similar to widespread adoption of EAV model despite its well-known performance issues. This is attributable to the maturity of relational databases and amount of research that has gone into improving their performance. The benefits of a generic database schema approach are recognised, with the alternative being XML schema driven relational representations. Aside from content representation, transformations from XML query languages to SQL and the relationship between database schema design and query performance achieved from these transformations, have been extensively explored.

Query processing and indexing approaches for a native XML processing context - i.e. XML query processors and native XML databases - provide highly specialised algorithms for particular aspects of queries such as finding all descendants of a node. These approaches do not assume the use of SQL, and are therefore free to assume more flexible execution environments. Hybrid approaches have been followed, aiming to leverage outcomes of research from this group of studies in the context of relational databases.

The findings of these studies have been used in the development of XINO, considering both relational and non-relational approaches to implementation of openEHR persistence. The details of this process are discussed next.

The relational implementation of XINO architecture is based on Postgresql and will be referred to as **XINO-P**. The primary goal of this implementation is to provide data access for the BN based CDS scenario discussed in Chapter 9. This does not mean that clinical information systems development has been disregarded. Even though this clinical care scenario has not been comprehensively tested, it has been included in the design, and key aspects have been implemented at the proof of concept level. XINO-P aims to comply with the previously stated design principles, while making use of the results of relevant research.

There exists a significant number of studies in the XML processing domain that are relevant to methods used for the implementation of XINO-P (which is based on AQL processing via TPQ matching, as described in Chapter 7). Comprehensive reviews of these studies are provided in (Hachicha and Darmont 2013), (Gou and Chirkova 2007) and (Tahraoui et al. 2013).

The timescale for completion of this thesis has made it impractical to adopt and experiment with all the algorithms and architectures from the literature. Instead, these findings are employed in two ways. First, to implement a persistence layer for openEHR, using a relational database that is capable of supporting the holistic approach to clinical information systems and CDS based on openEHR, at least at a proof of concept (POC) level. Second, to build a research roadmap based on the use of these findings from the literature for the construction of a large scale data processing platform based on openEHR.

8.3: Implementing the XINO Architecture with a Relational Database

XINO-P is an openEHR persistence implementation that is based on DAGs encoded as rows in a single table and TPQ matching implemented via SQL. The interactions of main components of persistence and the overall process are depicted in Figure 64.

The central component for persisting openEHR data in XINO-P is the Eclipse Modelling Framework (EMF) based analysis. This step takes an openEHR Composition instance in XML form as input and loads its content as an instance of an EMF ECore model. This ECore model is created via EMF's support for transforming XML schemas to ECore models, which has also been used to process XML schemas published as part of the published openEHR specifications. Once XML content is loaded as an instance of the ECore model, capabilities of EMF are used to analyse this data in order to create a DAG representation of it. This DAG representation is then persisted to a single table in a Postgresql database.

Access to data, after it is persisted to Postgresql, is performed via AQL. Following the approach developed in Chapter 7, an AQL query is represented as a TPQ, which is then expressed as an SQL query. Therefore, the process of building the SQL query can be defined as compiling AQL to SQL based on an intermediate representation, which is TPQ.

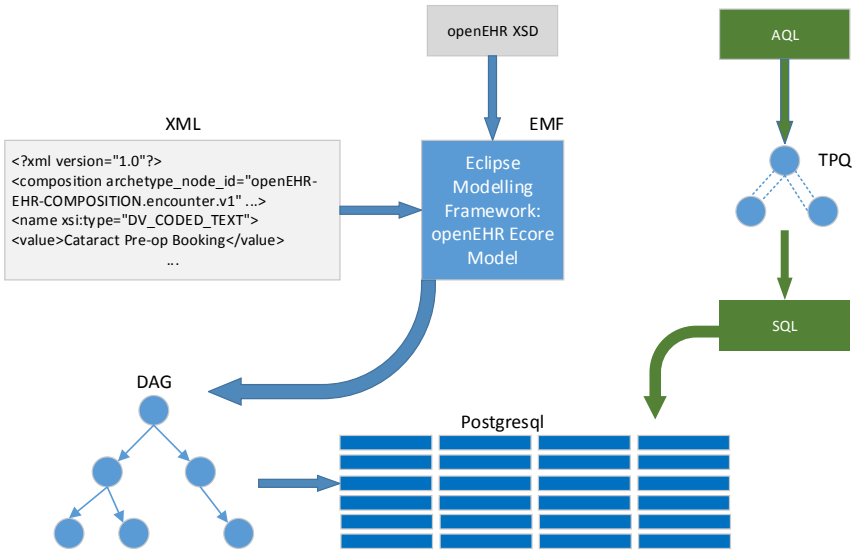


Figure 64: XINO-P: main components

During initial experiments on this architecture, an AQL parser and an SQL generator that takes a TPQ as input has been developed and tested to a limited extent. Despite the components working as expected, a short cut had to be adopted, due to the time it was taking to reflect changes in AQL to TPQ, or TPQ to SQL transformation, into code. Instead, mappings between different representations across subsequent steps were used for manual implementation of the AQL query used for data access in Chapter 9.

A key aspect of the architecture in Figure 64 is that all the components aside from Postgresql and the SQL query are platform independent. Therefore, both the relational database representation and SQL based data access can be replaced with other persistence systems.

The EMF based analysis treats every element of XML content as a node, and the output of this process is a DAG, which consists of a list of nodes. All nodes are then persisted into a single database table. Every node of the resulting DAG has the following six attributes:

- Pathstring: a string value that contains the archetype path of a node starting from the root of the COMPOSITION instance.
- Valstring: a string value that contains the actual value that a node may point at. These values are actual numeric or literal values which would normally map to primitive types such as strings or numbers. Therefore, not all nodes necessarily have this attribute set.
- ArchetypeNodeId: the archetype node id of the data item, if there is one defined in the openEHR archetype.

- ActualRmTypeName: the openEHR RM type name of the data item.
- Left: Left value of node based on DAG's region encoding.
- Right: Right value of node based on DAG's region encoding.

Figure 65 depicts the high level transformation from a Composition XML file to a DAG.

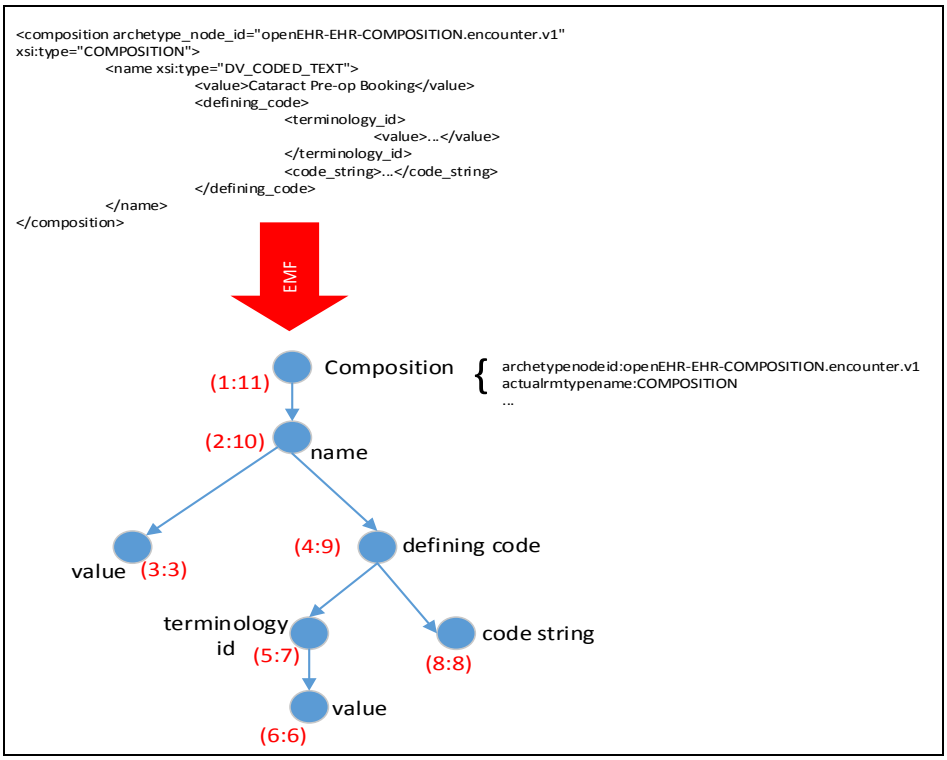


Figure 65: XML to DAG transformation with region encoding

Figure 65 shows how XML elements are transformed into an in memory DAG, using EMF's capabilities. The "actualrmtypename" attribute of all nodes that represent an element with an RM type is set to the corresponding type. The "archetypenodeid" attribute is assigned the corresponding value from the XML element.

The pair of numbers next to each node of the DAG provide position information based on the location of the elements in the XML file. These values are the "left" and "right" attributes of nodes, displayed separately in the diagram for clarity. The position information is based on a depth-first traversal of the DAG starting from the root node. The left attribute of each node is found by incrementing the left position of its parent. The right attribute is found by incrementing the last child of a node during depth-first traversal. Leaf nodes have equal left and right attributes.

This is a simplified version of the positional representation used by (Bruno, Koudas, and Srivastava 2002). The advantage of this representation, as discussed in (Bruno, Koudas, and Srivastava 2002) and (Al-Khalifa et al. 2002) is that it allows easy identification of structural relationships between DAG nodes.

The (*Left:Right*) encoding is sufficient for checking the descendant status of a node given another one. A DAG node N_1 with (L_1:R_1) is an ancestor of a node N_2 if $L_1 < L_2$ and $R_1 > R_2$. A more comprehensive encoding that includes node level alongside left and right enumerations can be used to test parent child relationship as described in (Bruno, Koudas, and Srivastava 2002) but this type of structural relationship is not explicitly expressed and therefore not needed in AQL, and therefore only left and right values are encoded.

If the processed XML element contains a value represented with a primitive type, as shown in the value of the name element in the XML snippet in Figure 65, which is “Cataract Pre-Op Booking”, this value is assigned to the *valstring* attribute of the DAG node. The end result of the process in Figure 65 is a set of nodes, each containing six attributes, with values assigned to them whenever necessary. This set is then persisted into a table in Postgresql, which has a column for each node attribute along with some extra columns. The screenshot below shows how DAG nodes are represented at the database level:

	id integer	ehr_id character varying(38)	instance_id character varying(38)	pathstring text	valstring text	archetypenodeid character varying(512)	actualrmtypename text	left integer	right integer
1	91540	1	886d4ff1-dfba-44	XMLDocun	token		string	58	58
2	91541	1	886d4ff1-dfba-44	XMLDocun	1.0.1		string	71	71
3	63323	1	7cbd1958-cfcc-49	XMLDocun	2014-02-20T11:35:49Z		string	37	37
4	74504	1	097d1eaf-326e-49	XMLDocun	token		string	144	144
5	74604	1	097d1eaf-326e-49	XMLDocun	2164b2c4-0465-4b53-a1		string	139	139
6	63125	1	7cbd1958-cfcc-49	XMLDocun			CODEPHRASE	41	46
7	63126	1	7cbd1958-cfcc-49	XMLDocun			CODEPHRASE	28	33
8	63127	1	7cbd1958-cfcc-49	XMLDocun			TERMMAPPING	379	396
9	63317	1	7cbd1958-cfcc-49	XMLDocun	AMD		string	351	351
10	63128	1	7cbd1958-cfcc-49	XMLDocun			ARCHETYPED	11	19
11	63129	1	7cbd1958-cfcc-49	XMLDocun		at0001	ITEMTREE	21	26
12	63130	1	7cbd1958-cfcc-49	XMLDocun			DVEHRURI	87	89
13	63131	1	7cbd1958-cfcc-49	XMLDocun			PARTYSELF	166	174
14	63132	1	7cbd1958-cfcc-49	XMLDocun			DVTEXT	319	321
15	63133	1	7cbd1958-cfcc-49	XMLDocun			EVENTCONTEXT	20	39
16	63134	1	7cbd1958-cfcc-49	XMLDocun			DVTEXT	460	462
17	63135	1	7cbd1958-cfcc-49	XMLDocun			TERMINOLOGYID	477	479
18	63136	1	7cbd1958-cfcc-49	XMLDocun			DVCODEDTEXT	47	73

Figure 66: Database representation of DAGs

As seen in Figure 66, the table that contains DAG nodes has three columns added to the six columns that are based on DAG node attributes. These are id, ehr_id and instance_id columns. The id column is an automatically generated primary key value for each row, which also serves as the unique identifier of a DAG node. The ehr_id is the id of the openEHR EHR which contains the Composition the

DAG is created from, and finally the instance_id is an identifier shared by all rows generated by processing a DAG. The instance_id , therefore, corresponds to the document id used as part of the XML element encoding in (Bruno, Koudas, and Srivastava 2002). The remaining columns represent values of DAG node attributes generated during EMF based processing, and they can have null values when a DAG attribute has no value assigned to it, as seen in Figure 66.

The ehr_id column represents a critical aspect of the XINO architecture: the requirement for a persistence system to process openEHR data as DAG does not mean that all data will be represented or treated in the same way. Various nodes and aspects of a DAG may be represented and processed in different ways to benefit from specific advantages of a persistence system or to avoid specific disadvantages of it.

Even though the EHR is the top-level concept in openEHR RM, it is not represented as a node in XINO-P. This is due to performance reasons. As discussed below, the number of joins required to implement TPQ matching is a critical determinant of the query performance. Directly associating every node with its highest level ascendant, which is the EHR node instance, saves XINO-P from having to perform join operations whenever the TPQ contains an EHR node. Since most data access during clinical care is driven by the identity of the patient, which consequently implies use of an EHR that belongs to the patient, encoding EHR id at the relational table level for all rows (DAG nodes) improves performance significantly for clinical care use cases. It should be noted that this and further specialisations for DAG representation and TPQ matching are all driven by the tree-based approach to the openEHR persistence of Chapter 7. Therefore, these specialisations do not modify this fundamental approach. They are just performance driven optimisations specific to a persistence system. The DAG representation of openEHR stays intact and consistent.

The nine columns used for the relational persistence of DAG nodes are all that are needed to implement the key operations of TPQ matching defined in Chapter 7. The result of these operations is a transformation from DAG instances to rows of a result set, as depicted in the following Figure 67.

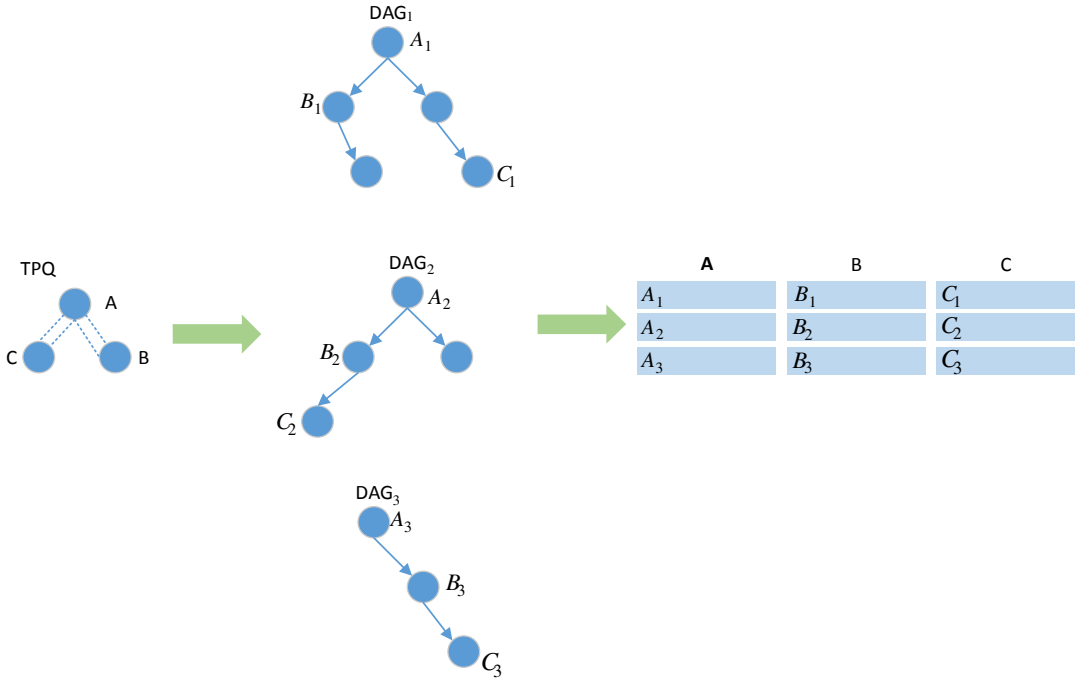


Figure 67: DAG to tuple transformation via TPQ matching

The TPQ in Figure 67 is simplified in the interests of clarity. It represents a structural constraint where an instance of node type A has two descendants of type C and B. When this TPQ is matched against the three DAGs in the diagram, the result set transforms all matches into rows. Each DAG that satisfies the TPQ is returned as a row. The identifiers of TPQ nodes (which, in this simplified form correspond to their types) become the columns of the result set.
Even though the TPQ representation of AQL and constraints expressed by it can become significantly more complicated, this fundamental transform does not change.

The following sections discuss the implementations of various TPQ matching operations implemented via SQL, results of which are then joined together to create the result set structure in Figure 66.

8.3.1: TPQ Matching for the FROM Section of AQL Queries

The FROM section of an AQL query defines a list of nodes which are later referenced from either SELECT or WHERE sections, along with hierarchical constraints on the members of this list. Therefore, matches for the TPQ defined by the FROM section of an AQL query can be considered as a precondition for matches based on other sections; both SELECT and WHERE sections define their constraints based on nodes from the FROM section.

A simplified AQL query with an emphasis on the FROM section is mapped to a TPQ as depicted in Figure 68.

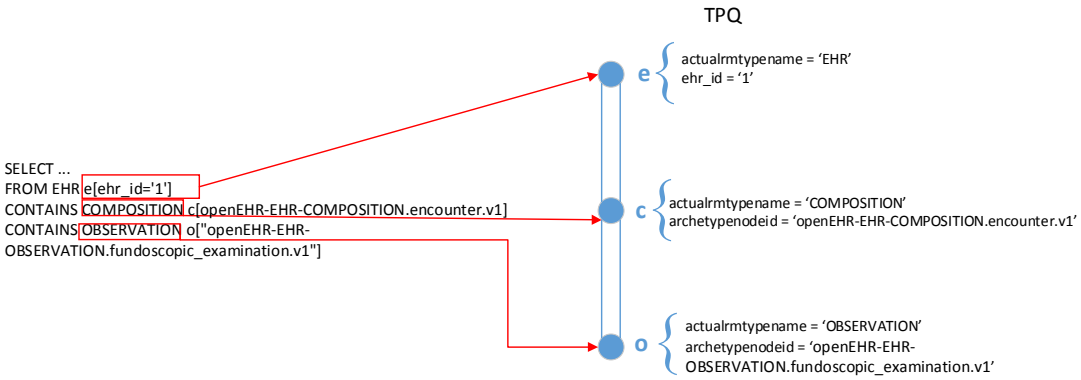


Figure 68: TPQ for FROM section of AQL

This TPQ has three nodes arranged in a structure that defines the Observation instance *o* as a descendant of Composition instance *c,* which in turn is a descendant of EHR node *e*. There are also value constraints on attributes of nodes. This TPQ can be translated into an SQL query that assumes the nine column table schema described above, as follows:

1) *Define SQL subqueries that match individual nodes based on node attribute constraints*

   A convenient feature of SQL is common table expressions (CTE) that define a temporary result set which can be used easily as a subquery. Using this feature, the *c* and *o* nodes of the TPQ can be selected as shown in Figure 69.

```
WITH
ehr AS
   ( SELECT distinct(node.ehr_id) as id FROM DAG node
     WHERE node.ehr_id = '3'),
c AS
   ( SELECT node.* FROM eav node
     WHERE node.archetypenodeid = 'openEHR-EHR-
COMPOSITION.encounter.v1'
     AND node.actualrmtypename = 'COMPOSITION'
     AND node.ehr_id = '1' )
,o AS
   ( SELECT node.* FROM eav node
     WHERE node.archetypenodeid = 'openEHR-EHR-
OBSERVATION.fundoscopic_examination.v1'
     AND node.actualrmtypename = 'OBSERVATION'
     AND node.ehr_id = '1'   )
....
```

Figure 69: CTEs for matching TPQ nodes

The c and o subqueries match all nodes (rows) of DAGs with the archetype node ids, and RM types expressed in the TPQ, which in turn is based on the AQL. The semantics of EHR with id ‘3’ is expressed in SQL via a constraint on the ‘ehr_id’ column of the DAG nodes table. Once these CTEs are in place, the rest of the SQL query can use them.

2) *Enforce structural constraints of nodes using positional encoding*
The structural constraints are enforced through the use of positional encoding of nodes. The constraint on the “ehr_id “ column value automatically establishes the structural constraint that all nodes that are selected are descendants of the EHR node that the TPQ targets. This is due to the EHR root position, by RM design. The other structural constraint, *o* being a descendant of *c*, can be enforced within another CTE as in Figure 70.

```
....
,from_nodes AS
  ( SELECT --FROM NODES SUB-QUERY
     '3' ehr_id,

     obs_fund_exam.id obs_fund_exam_id,
     obs_fund_exam.instance_id obs_fund_exam_ins_id,
     obs_fund_exam."left" obs_fund_exam_left,
     obs_fund_exam."right" obs_fund_exam_right,
     obs_fund_exam.pathstring obs_fund_exam_pstring

    FROM comp_encounter c_root_cl_exam
    INNER JOIN obs_fundoscopic_exam obs_fund_exam
          ON c_root_cl_exam.instance_id =
obs_fund_exam.instance_id
          AND c_root_cl_exam."left" < obs_fund_exam."left" AND
c_root_cl_exam."right" > obs_fund_exam."right"
          AND obs_fund_exam.ehr_id = '3'
    WHERE  c_root_cl_exam.ehr_id = '3'
  )
....
```

Figure 70: CTE that enforces ‘descendant of’ constraint

The CTE named as “from_nodes” in the query in Figure 70 selects *c* and *o* node instances which satisfy the following properties: every *o* is a descendant of *c* (using positional encoding), both *o* and *c* are from the same DAG (using “instanceId” equality) and finally both *o* and *c* are under the same EHR whose id is known. The “from_nodes” CTE uses previously defined CTEs, and it returns values of columns of the table row that contains *o* node.

The “from_nodes” CTE in the example in Figure 70 returns information related to the Observation node *o*, for clarity of the example. Using a reference to *o* node, the SELECT section of AQL query may define the value that the AQL query is supposed to return, as discussed next.

8.3.2: TPQ Matching for the SELECT Section of AQL Queries

The SELECT section of an AQL query defines the result set using data items defined in the FROM section, either directly or as the root of a relative path that points at the data item that should be returned. In order to demonstrate how TPQ matching is implemented via SQL for this purpose, the AQL query in Figure 68 is expanded to include a fully defined SELECT section that returns diagnosis of diabetic retinopathy in Figure 71.

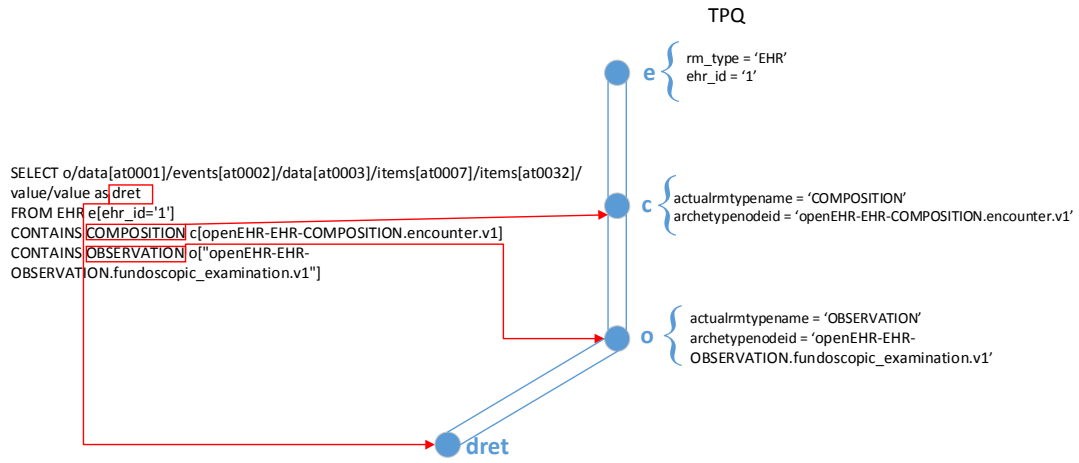


Figure 71: TPQ matching for SELECT clause of AQL

The SELECT section of the AQL query in Figure 71 points at the ‘*dret*’ node via a relative path that starts from the *o* node. This relationship between *o* and *dret* is represented in the extended TPQ. This hierarchical relationship requires selecting all DAG nodes that are reachable via the relative path from *o*. This operation is performed as follows through use of another CTE:

```
....
,dret AS
  (SELECT fn.obs_fund_exam_id, T.valstring as valstring
   FROM DAG T INNER JOIN from_nodes fn
              ON T.instance_id = fn.obs_fund_exam_ins_id
              AND  T."left" > fn.obs_fund_exam_left AND T."right" <
fn.obs_fund_exam_right
              AND  T.pathstring = fn.obs_fund_exam_pstring || '/' ||
'data[at0001]/events[at0002]/data[at0003]/items[at0007]/items[at0032]/val
ue/value'
              AND T.ehr_id = '3'
  )
....
```

Figure 72: Enforcing relative path in a CTE

The CTE defined as '*dret*' uses the nodes returned by the '*from_nodes*' CTE and it leverages both instance id and positional encoding columns along with ehr_id column since any node reachable through a path relative to a DAG node *o* is guaranteed to be a descendant of it.

The relative path that is defined in the AQL query is used to identify the '*dret*' node through the use of the pathstring column of the DAG nodes table. Since all nodes have their absolute paths from the COMPOSITION root encoded in the 'pathstring' column any node A and its descendant B have the following relationship:

Path of A + relative path of B from Path of A = Path of B

Therefore, '*dret*' CTE allows selection of the 'dret' node in the TPQ in Figure 71, using SQL. However, for a result set of a TPQ matching operation to be returned, the CTEs created for the SELECT and FROM sections must be joined with consideration for nodes that may not exist.

8.3.3: Linking Matches for Different TPQ Hierarchical Relationships

TPQ matching based on SQL Subqueries handle different semantics that can be represented in the TPQ approach developed in Chapter 7 and for the whole TPQ matching to provide a result set, the results from subqueries must be brought together.

In doing so, other TPQ semantics become relevant when queries with more data items and constraints must be processed. An extension of the TPQ used so far with another data item in the SELECT section is provided in Figure 73 as an example:

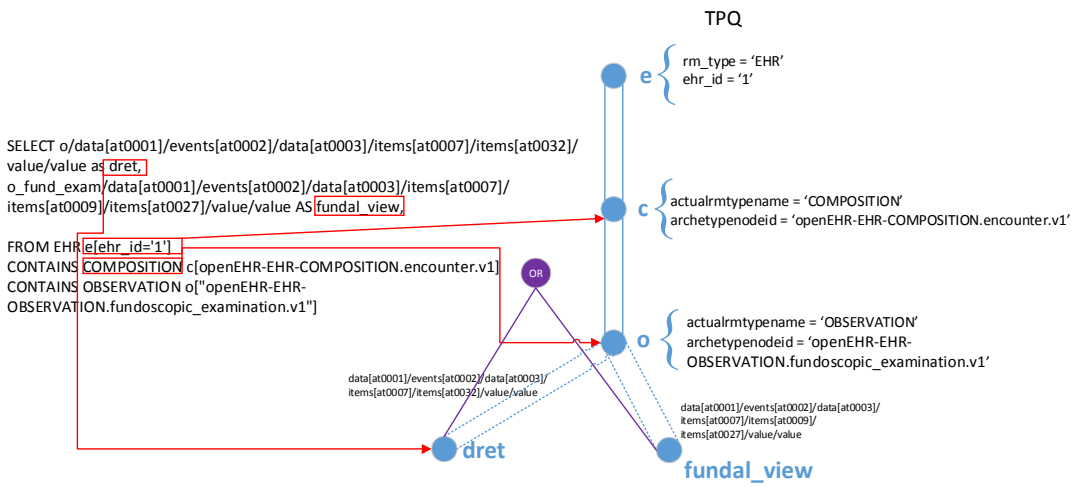


Figure 73: Optional containment for data items in SELECT clause

Figure 73 extends the previous TPQ with another node: 'fundal_view'. With this node under *o*, the relationship between *dret* and *fundal_view* requires clarification. As discussed in Chapter 7, an intuitive expectation for matching this TPQ would be that if diabetic retinopathy does not exist, but fundal view has been recorded, the results should represent this. This implies optional containment for both nodes under *o* and when CTEs that perform the TPQs are joined, this behavior must be preserved.

This is established via using the relevant SQL join operations as shown in Figure 74. First, the CTE for *fundal_view*, following the same approach with *dret*:

```
....
,fundal_view as
  (SELECT  fn.obs_fund_exam_id,T.valstring as valstring
       FROM DAG T INNER JOIN from_nodes fn
       ON T.instance_id = fn.obs_fund_exam_ins_id
       AND T.pathstring = fun.obs_fund_exam_pstring || '/' ||
'data[at0001]/events[at0002]/data[at0003]/items[at0007]/items[at0009]/ite
ms[at0027]/value/value'
       AND t."left" > fn.obs_fund_exam_left AND t."right" <
fn.obs_fund_exam_right
       AND t.ehr_id = '3'
  )
....
```

Figure 74: CTE for fundal view node of TPQ

Then the SQL query that uses previous CTEs to build the result set:

```
WITH
  comp_encounter AS
  (SELECT ....)

  ,obs_fundoscopic_exam AS
  (SELECT ....)

  ,from_nodes AS
  (SELECT  .... )

  ,dret   AS
  (SELECT ....)

  ,fundal_view   AS
  (SELECT ....)

SELECT
  '3' as ehr_id,
  fn.obs_fund_exam_ins_id,
  ,dret.valstring as dret,
  ,fundal_view.valstring as fview

FROM from_nodes fn
  FULL OUTER JOIN dret ON dret.obs_fund_exam_id = fn.obs_fund_exam_id,
  FULL OUTER JOIN fundal_view ON fundal_view.obs_fund_exam_id =
fn.obs_fund_exam_id
```

Figure 75: SQL for the complete TPQ matching

The main WITH…SELECT…FROM SQL query matches the TPQ on DAG nodes table using previously defined CTEs. The optional containment is implemented via use of FULL OUTER JOINs, which ensures that if dret or fundal_view nodes are missing, the TPQ matching results include existing nodes. When run on an EHR which has no diabetic retinopathy, this query returns the results in Figure 76.

	ehr_id unknown	obs_fund_exam_ins_id integer	dret text	dret_instid text	fview text
1	1	207			No Fundal View

Figure 76: Query results when no diabetic retinopathy exists

Whereas, when run on an EHR with diabetic retinopathy diagnosis, both nodes are returned in the results as depicted in Figure 77.

	ehr_id unknown	obs_fund_exam_ins_id integer	dret text	dret_instid text	fview text
1	3	111	Dia	111	No Fundal View

Figure 77: Query results when diabetic retinopathy exists

If an INNER JOIN were used in the main query for joining nodes returned by the CTEs, the results for the EHR which does not have diabetic retinopathy would incorrectly exclude the fundal_view node as depicted in Figure 78.

	ehr_id unknown	obs_fund_exam_ins_id integer	dret text	dret_instid text	fview text

Figure 78: Query results: unintended exclusion of fundal view node

The FROM statement of the main SQL query for TPQ matching uses nodes defined in the FROM section of AQL for connecting relevant CTEs together. The inclusion of more variables in the AQL SELECT section is therefore simply a matter of adding CTEs for nodes and including them in the FROM section of main SQL query using FULL OUTER JOINs.

The use of different types of SQL JOIN operations for expressing TPQ semantics is not limited to optional containment. Another fundamental aspect of TPQ matching, applying Boolean operators, is also implemented via use of different SQL join operations.

8.3.4: Representing Boolean Operator Semantics for TPQ Node Relationships

Explicit use of Boolean operators is required by AQL grammar when multiple data items share a parent data item. Figure 79 shows an extension of the previous example to include another node in the TPQ: an Observation which can be used to select extra information.

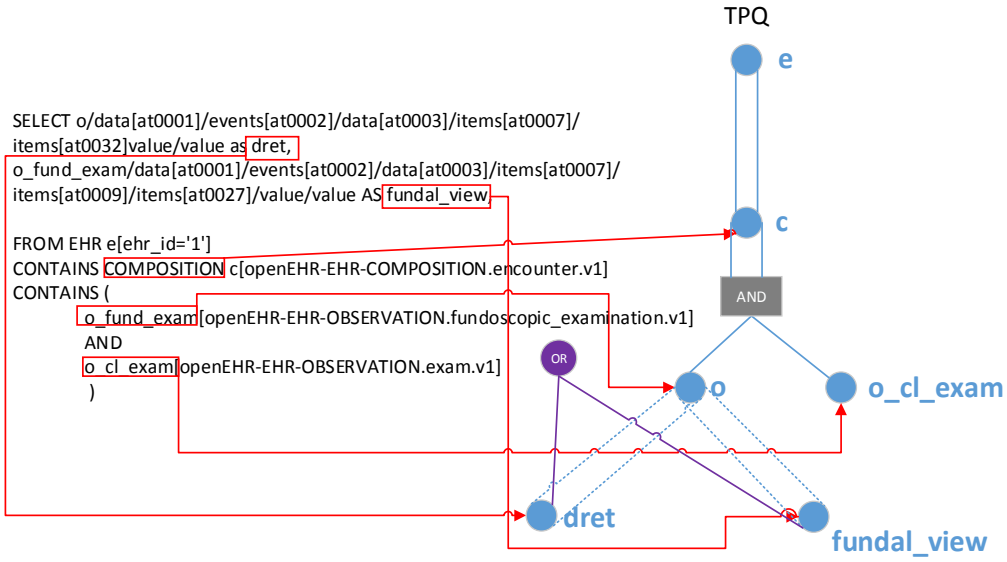


Figure 79: TPQ for AQL with an AND operator in the FROM clause

The AND operator in Figure 79 requires selection of *c* nodes only if the DAG contains both *o* and *o_cl_exam* nodes as c's descendants. AQL also supports nesting of Boolean operators in the FROM section so these complex scenarios must be handled as well. To deal with this requirement the AND Boolean operator is implemented with INNER JOINs and the OR Boolean operator is implemented with LEFT OUTER JOINs along with the use of nested SQL subqueries. An INNER JOIN between a parent node and a number of its descendants requires that all the descendant nodes from TPQ have at least one instance in the DAG, or the CTE for the parent node would return zero rows. In case of a LEFT OUTER JOIN expressed as:

'parent_node_CTE' LEFT OUTER JOIN 'child_node_CTE' ON…

As long as the parent exists in the DAG and instance, the parent and any existing children would be returned. Handling nested Boolean operators then becomes repeated INNER JOINs or LEFT OUTER JOINs through subqueries. Use of Boolean operators for the WHERE section of AQL is handled differently as discussed next

8.3.5: TPQ Matching for the WHERE Section of AQL Queries

The WHERE section of AQL can refer to any node from the FROM section in order to introduce constraints, either on them or on their descendants accessible via relative paths. The AQL WHERE section can also employ Boolean operators. The constraints defined in the WHERE section are implemented via WHERE keyword of SQL. The TPQ representation of the AQL WHERE section can take two forms. The first form introduces a single constraint which can be expressed with a value node and a value check on the value node's attribute, as depicted in Figure 80.

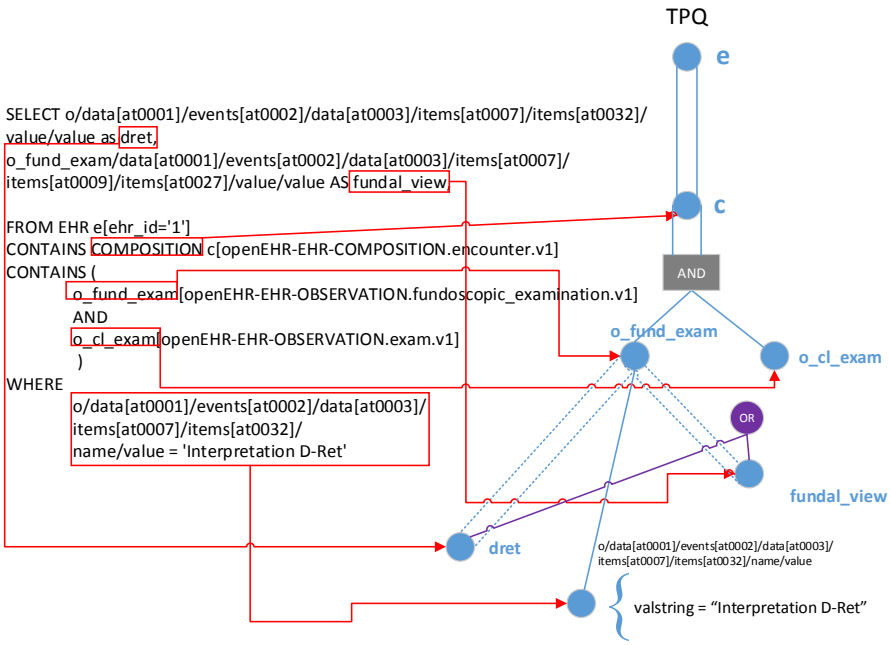


Figure 80: TPQ for AQL with a WHERE clause

The TPQ in Figure 80 has a node that is a descendant of *o* and valstring attribute of this node should be equal to ‘Interpretation D-Ret’. This node is introduced to TPQ by the WHERE section of AQL, and its existence and value can be enforced by introducing an INNER JOIN into CTE for *o* as shown in Figure 81.

```
....
,obs_fundoscopic_exam AS
(SELECT node.* FROM DAG node
    INNER JOIN DAG T ON    T.instance_id = node.instance_id
    AND  T."left" > node."left" AND T."right" < node."right"
    AND  T.pathstring = node.pathstring || '/' ||
'data[at0001]/events[at0002]/data[at0003]/items[at0007]/items[at0032]/nam
e/value'
    and T.valstring = 'Interpretation D-Ret'
    AND T.ehr_id = '3'
    WHERE node.archetypenodeid = 'openEHR-EHR-
OBSERVATION.fundoscopic_examination.v1'
    AND node.ehr_id = '3'  )
....
```

Figure 81: Enforcing AND operator with INNER JOIN

This inner join forces all *obs_fundoscopic* nodes that are returned to have a descendant that corresponds to a node that satisfies the WHERE constraint of AQL.

When more than one constraint is specified in the AQL WHERE clause, their relationship must be predicated with a Boolean operator, whether or not the constraints are placed on the same data item. If all the constraints are placed on the

same node, then they can be included in the CTE for the node using the previously defined mappings to INNER JOIN and LEFT OUTER JOIN operations along with nesting of subqueries.

If all constraints in the WHERE section of AQL are not placed on the same data item, then Boolean operators across these constraints can only be applied at the main SQL query level since a CTE for a node can only refer to its ascendant and its descendants. This makes it impossible for a single CTE to enforce constraints on other CTEs if they are not on the same ascendant-descendant axis of the DAG.

Therefore, constraints on multiple nodes are implemented through the use of CTEs for these constraints that are included in the main SQL query. An example of this scenario is depicted in Figure 82.

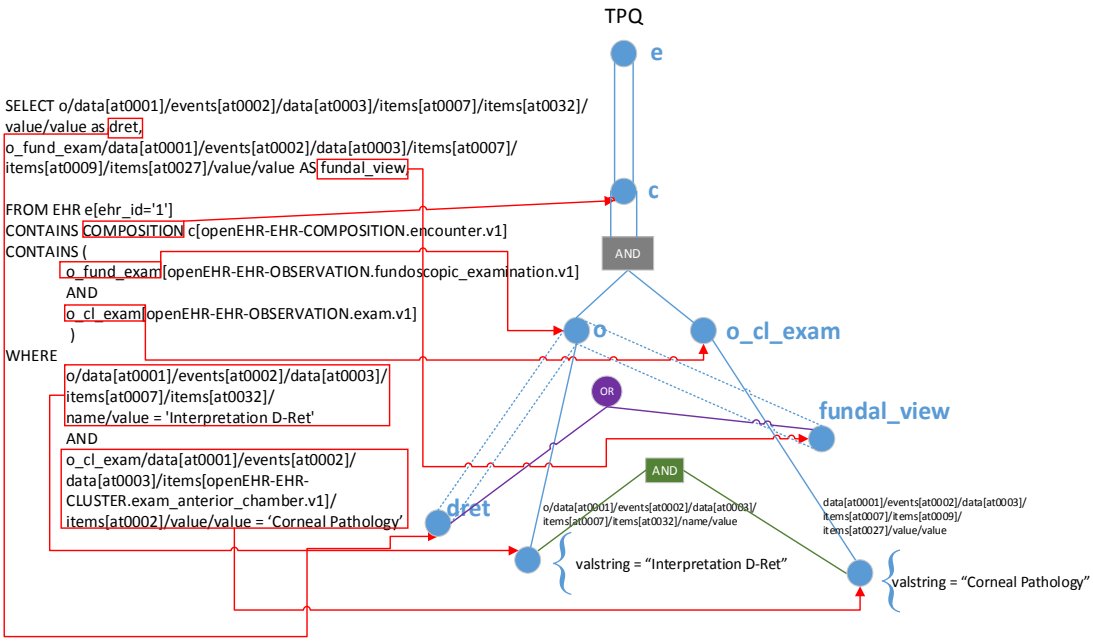


Figure 82: TPQ for AQL: multiple AND operators in WHERE clause

The AQL query and corresponding TPQ in the diagram enforce the existence of diabetic retinopathy along with corneal pathology. The AND operator in the diagram can be implemented as in Figure 83.

```
WITH
  comp_encounter AS
  (....)
  ,obs_fundoscopic_exam AS
  (....)
  ,o_cl_exam AS
  (....)
  --  THIS IS WHERE FROM NODES HIERARCHY IS ENFORCED
  ,from_nodes AS
  (....)
  ,dret   AS
  (....)
  ,dret_where   AS
  (....)
  ,fundal_view as
  (....)
  ,corneal_pathology_where as
  (....)

SELECT
  ....
FROM from_nodes fn
  FULL OUTER JOIN dret ON dret.obs_fund_exam_id = fn.obs_fund_exam_id
  FULL OUTER JOIN fundal_view ON fundal_view.obs_fund_exam_id =
fn.obs_fund_exam_id
  FULL OUTER JOIN dret_where ON dret_where.obs_fund_exam_id = fn.obs_fund_exam_id
  FULL OUTER JOIN corneal_pathology_where ON corneal_pathology_where.o_cl_exam_id
= fn.o_cl_exam_id
WHERE dret_where.valstring = 'Interpretation D-Ret' AND
corneal_pathology_where.valstring = 'Corneal pathology'
```

Figure 83: Enforcing multiple Boolean Operators in TPQ

The ‘dret_where’ and ‘corneal_pathology_where’ CTEs select the nodes from the TPQ attributes of which (valstring in this case) must have the values specified in the AQL WHERE criteria. The outermost WHERE clause of TPQ matching SQL uses SQL AND operator to implement the TPQ AND operator. The individual CTEs for ‘dret_where’ and ‘corneal_pathology’ that enforce the constraints for filtering out TPQ matches are provided in Figure 84.

These CTEs individually ensure that the constraint nodes that are operands of the AND operator in the TPQ exist. Their results are appended to other CTE results that join nodes created by the FROM and SELECT sections of AQL with a FULL OUTER JOIN. This approach treats results returned from the CTEs of AQL WHERE clause as if they’re subject to optional containment, in the same way the nodes introduced by the SELECT clause of AQL: their existence is optional and would not affect query results due to the FULL OUTER JOIN. The logical AND operator is then applied at the main SQL query level, turning these nodes into a filter mechanism for all the results.

This approach provides support for other Boolean operators such as OR or NOT with nesting, if necessary, since the SQL WHERE clause enables nesting of logical operators.

```
....
,corneal_pathology_where as
  (SELECT  fn.o_cl_exam_id,T.valstring as valstring
    FROM EAV T INNER JOIN from_nodes fn
       ON T.instance_id_int = fn.o_cl_exam_ins_id
          AND T.pathstring = fn.o_cl_exam_pstring || '/' ||
'data[at0001]/events[at0002]/data[at0003]/items[openEHR-EHR-
CLUSTER.exam_anterior_chamber.v1]/items[at0002]/value/value'
          AND t."left" > fn.o_cl_exam_left AND t."right" <
fn.o_cl_exam_right
          AND t.ehr_id = '3'
          AND t.valstring = 'Corneal pathology'
  )
....
,dret_where   AS
  ( SELECT fn.obs_fund_exam_id, T.valstring as valstring
    FROM eav T INNER JOIN from_nodes fn
      ON    T.instance_id_int = fn.obs_fund_exam_ins_id
      AND  T."left" > fn.obs_fund_exam_left AND T."right" <
fn.obs_fund_exam_right
      AND  T.pathstring = fn.obs_fund_exam_pstring || '/' ||
'data[at0001]/events[at0002]/data[at0003]/items[at0007]/items[at0032]/name
/value'
      AND T.valstring = 'Interpretation D-Ret'
      AND T.ehr_id = '3'
  )
....
```

Figure 84: Individual CTEs for AQL WHERE clause constraints

8.3.6: Discussion of the Relational Modelling Approach

XINO-P consists of a small number of fundamental operations and the use of a single database table that provides a TPQ matching based implementation of AQL, using SQL. The most significant and gratifying aspect of this design is its simplicity, which is the key to its applicability to architectures that may not necessarily use a relational database. XINO-P leverages SQL features for querying, but its data representation makes little use of the relational approach to data modelling. Instead of relying on a relational representation of the openEHR RM, a single table is used that is repeatedly joined on itself (self-join). This design can be classified as a highly specialized form of EAV where only the value table exists, and entity and attribute definitions are ignored. RM types and archetype node ids are included as columns without formally defining these attributes with an attribute table. This simplified representation, accompanied by structural attributes such as positional encoding of DAG nodes and their absolute paths from the DAG root (similar to (Yoshikawa et al. 2001)) is built on a few key properties of openEHR.

The most important aspect of openEHR in the context of implementing XINO-P is that openEHR RM and archetypes provide both structural and value constraints ahead of the creation of actual data. When this information is accessed

through a framework such as EMF, it becomes possible to process and encode all clinical data based on openEHR RM type information. Even though an essentially infinite number of clinical models for an infinite number of clinical concepts can be created with openEHR, instances of all these models conform to the structures of a small number of types brought together in a flexible way.

AQL queries on the clinical data have the same characteristic: a large number of clinical models can be queried based on the same RM. These iteratively evolved and hard-won features of openEHR allow all information about the clinical data to be available outside of the actual persistence implementation. XINO-P leverages this approach by not representing any openEHR related information at the database level unless that information is necessary to get the data back. Furthermore, the only assumed method for retrieving data from the persistence implementation is AQL; no data access characteristics other than those used by AQL are considered.

As a result of this specialization, both the data representation and TPQ matching operations used in XINO-P can easily be used in non-relational settings, thereby delivering the persistence system portability goal of the XINO architecture. For example, removing the EHR id condition from the SQL queries that were used as examples in the XINO-P implementation of TPQ turns these queries into population queries instead of queries specific to a single EHR. This approach, used to build the data set that was used in Chapter 9, conveniently lends itself to parallelization when the underlying persistence system supports it. Using a big data framework such as Apache Hive (Thusoo et al. 2009) that supports parallel processing of table structures with a query language that is highly compatible with SQL, the approach used by XINO-P can be reused with minimum change with the benefit of running the join operations across hundreds or even thousands of servers with very large data sets.

It should be noted that neither the availability of SQL nor table based representation is a requirement for leveraging other platforms for TPQ processing, even though various big data platforms already support SQL. The operations on nodes that are implemented via SQL, such as finding descendant nodes, finding nodes at a relative path or applying logical operators, can be implemented by other means. In fact, XINO-P uses extensions of the fundamental model described thus far in order to improve performance as discussed next.

8.4: Extensions of the Purely Relational Model and Other Improvements

The TPQ matching implementation of XINO-P allows the fundamental semantics of AQL to be expressed using SQL. The use of SQL can be improved by use of extension mechanisms provided by the relational database servers, for better query performance. These extension mechanisms can also support the addition of new features to AQL and the enablement of a richer set of types in query results.

From a performance point of view, the use of repeated self joins to select DAG nodes based on TPQs presents a challenge for XINO-P. Repeated self-joins on a table, size of which is bound to grow as new data arrives, leads to a large number of index scans being performed by Postgresql. As the data size grows, index scans need to cover a larger number of rows to identify the ones defined by the CTEs. The analysis of the CTE performed on a test database, using Postgresql's EXPLAIN ANALYZE feature, demonstrates this problem as shown in Figure 85.

```
EXPLAIN ANALYZE SELECT node.* FROM temp_eav_table_global node
WHERE node.archetypenodeid = 'openEHR-EHR-COMPOSITION.encounter.v1'
      AND node.ehr_id = '3'

Bitmap Heap Scan on temp_eav_table_global node  (cost=2353.66..2377.73
rows=6 width=399) (actual time=33.307..33.309 rows=2 loops=1)
  Recheck Cond: (((ehr_id)::text = '3'::text) AND ((archetypenodeid)::text
= 'openEHR-EHR-COMPOSITION.encounter.v1'::text))

  ->  BitmapAnd  (cost=2353.66..2353.66 rows=6 width=0) (actual
time=33.299..33.299 rows=0 loops=1)

        ->  Bitmap Index Scan on temp_eav_table_global_ehrid
(cost=0.00..45.07 rows=2201 width=0) (actual time=0.176..0.176 rows=931
loops=1)
              Index Cond: ((ehr_id)::text = '3'::text)

        ->  Bitmap Index Scan on temp_eav_table_global_archndId
(cost=0.00..2308.34 rows=122636 width=0) (actual time=32.718..32.718
rows=100000 loops=1)
              Index Cond: ((archetypenodeid)::text = 'openEHR-EHR-
COMPOSITION.encounter.v1'::text)
Total runtime: 33.348 ms
```

Figure 85: Postgresql query plan and execution for simple CTE

Since the constraints expressed in the WHERE clause of SQL query require the Postgresql engine to perform separate index scans, the number of rows the index scans must process becomes the primary determinant of the query performance. As Figure 85 shows, 32.7 milliseconds of the 33.348 millisecond total query runtime is spent on the archetype node id index scan that must process 100K

rows. As more data is added, based on an archetype that has this archetype id, this index scan would have to process the growing number of rows that satisfy the same criteria. This behaviour is there by design and cannot be avoided.

As a result, the more joins a TPQ structure leads to, the more query performance for the same TPQ drops, as the total data size stored by XINO-P grows. This drop in performance is bound to happen even though the increase in data volume is due to the arrival of new data that would not be matched by this TPQ. Therefore, processing as few rows as possible during querying, or even keeping the number of rows in the DAG table as few as possible, can improve performance and provide resilience against this performance drop.

Two of Postgresql's features have been used to this end: JSON (Crockford 2006) type support and custom functions that process JSON content. JSON stands for JavaScript Object Notation, a text based data representation format that has been replacing XML for many use cases in recent years. Postgresql allows the storage of JSON content alongside other data types it supports, and it provides a number of functions and operators that operate on JSON data.

XINO-P uses JSON to represent some nodes of the DAG during pre-persistence processing, and these nodes are inserted into a column with JSON type in the row that represents their parent, as depicted in Figure 86.

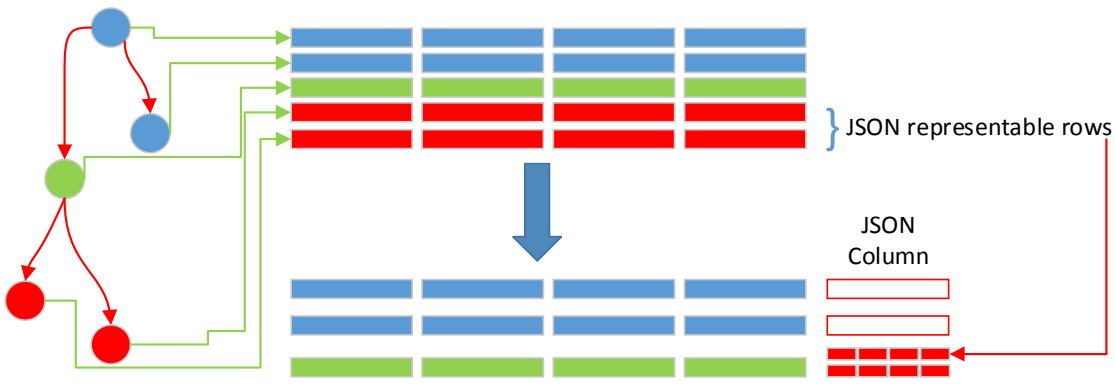


Figure 86: Node transformation from tuples to column via JSON

As shown in Figure 86, this representation decreases the number of rows by moving multiple rows into the row of their parent as a column. The JSON content is exactly the same as row content, but it is a textual representation that can compactly represent child nodes of a DAG node as an attribute of its parent. The CTEs for matching nodes can access this compact representation using functions

that process the JSON content. Figure 87 contains an example of a CTE before and after use of JSON representation for some rows.

The use of JSON processing functions replaces two joins that are required to select two child nodes. Instead, JSON content is dynamically extracted to an array of values and queried. Since JSON content is created from child nodes, some of the checks such as node coordinates or instance id checking is no longer necessary. Moreover, JSON processing functions are provided directly with the contents of the row that has been selected by the previous table reference in the same FROM clause, saving more index scans.

This approach both decreases the number of rows in the table that holds all nodes and avoids costly joins and scans. The criteria for selecting child nodes to be represented as JSON is an interesting topic, arising here, for future research. In the XINO-P implementation that has been used to provide data to the BN implementation, as described in Chapter 9, all child nodes that do not have an archetype node id have been collapsed into JSON content, but this is a criterion that has been used for the data set at hand without any claim or expectation of its general applicability.

In the context of the applicability of the TPQ matching approach to persistence systems other than relational databases, use of JSON and custom functions may not necessarily achieve the same performance benefits as were found for Postgresql. The text based nature of JSON allows its storage in both relational and non-relational persistence systems. The capability to introduce user-defined functions into SQL queries is implemented by most major databases and even in big data frameworks such as Apache Hive (Thusoo et al. 2009). Therefore, this approach allows the extension of TPQ matching based on SQL, with a significant level of applicability to other platforms.

The use of JSON or alternative serialisation formats allow the persistence layer to perform AQL path access on small amounts of content, without having to return this content as interim query results, which must be further processed for path extraction. This approach avoids the need to return and process the whole content of an openEHR COMPOSITION instance to access only a few fields, and can provide significant performance improvements for a class of queries that require a small number of simple values, especially if the number of the COMPOSITION instances that must be processed is large.

The use of user-defined functions can allow implementation of many advanced features on top of AQL. This is another topic suggested for future research. One such feature that can be built on user- defined functions is the use of

BN inference directly from within AQL for CDS. This extension, provisionally named Probabilistic AQL, is based on the idea of extending AQL syntax to use an existing BN definition as a filter for AQL results. Figure 88 depicts the extended AQL query along with the relationship with the corresponding TPQ.

```
....
,dret AS
(SELECT fn.obs_fund_exam_id, Z.instance_id_int AS  instanceid, Z.valstring as
valstring
 FROM temp_eav_table_global T INNER JOIN ehr ON T.ehr_id = ehr.id
   INNER JOIN from_nodes fn ON T.instance_id_int = fn.obs_fund_exam_ins_id
   AND  T."left" > fn.obs_fund_exam_left AND T."right" < fn.obs_fund_exam_right
   AND  T.pathstring = fn.obs_fund_exam_pstring || '/' ||
'data[at0001]/events[at0002]/data[at0003]/items[at0007]/items[at0032]'

   INNER JOIN temp_eav_table_global Y ON Y.instance_id_int = T.instance_id_int
   AND Y."left" > T."left" AND Y."right" < T."right"
   AND Y.pathstring = T.pathstring || '/' || 'name/value'
   AND Y.valstring = 'Interpretation D-Ret'
   AND  Y.ehr_id = ehr.id

   INNER JOIN temp_eav_table_global Z ON Z.instance_id_int = T.instance_id_int
   AND Z."left" > T."left" AND Z."right" < T."right"
   AND Z.pathstring = T.pathstring || '/' || 'value/value'
   AND  Z.ehr_id = ehr.id
)
....
,dret   AS
(SELECT  fn.obs_fund_exam_id, Z->>'instance_id_int' AS instanceid,Z->>'valstring'
AS valstring
 FROM ehr, temp_eav_table_global  t,from_nodes fn,json_array_elements(t.jsndata) AS
Y,json_array_elements(t.jsndata) AS Z

 WHERE t.instance_id_int = fn.obs_fund_exam_ins_id
       AND t.pathstring = fn.obs_fund_exam_pstring || '/' ||
'data[at0001]/events[at0002]/data[at0003]/items[at0007]/items[at0032]'
       AND t."left" > fn.obs_fund_exam_left AND t."right" < fn.obs_fund_exam_right
       AND Y->>'pathstring' = t.pathstring || '/' || 'name/value'
       AND Y->>'valstring' = 'Interpretation D-Ret'
       AND Z ->>'pathstring' = t.pathstring || '/' || 'value/value'
       AND t.ehr_id = ehr.id
)
....
```

Figure 87: Using JSON and functions in a CTE

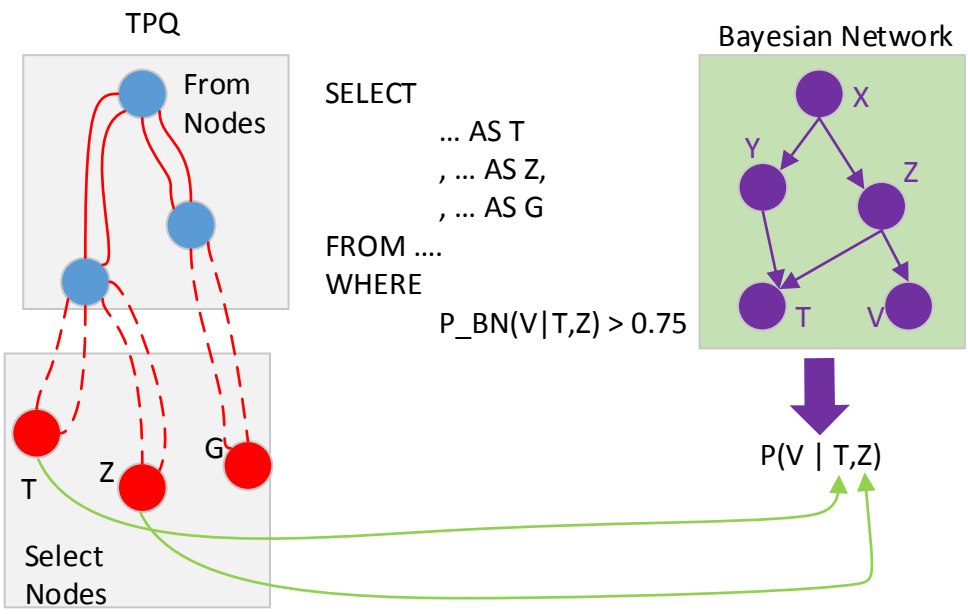


Figure 88: TPQ matching and BN inference integration for probabilistic AQL

The AQL query in Figure 88 defines three data items for selection, which have the aliases T, Z and G. These are represented in the corresponding TPQ and grouped together as Select Nodes for clarification. A subset of these nodes, T and Z, are used as observations in a BN, through a function call from AQL named P_BN. This syntax means that the AQL query should return results that produce a probability larger than 0.75 when the T and Z nodes are used as observations. This approach can be extended with a more expressive syntax for referring to probabilities of multiple nodes from the BN. The implementation of such an extension to AQL syntax can readily be implemented using the mechanisms used in XINO-P. Figure 89 contains the pseudo code of a possible implementation.

This pseudo code uses a user-defined function called INFER_BN which takes a string representation of the observations for a particular BN structure, the semantics of which is assumed known. Since user-defined functions in Postgresql can be implemented in a number of programming languages, including Python and even R, this function call would effectively call a BN implementation, infer the probabilities of the network for each row of the result set and produce a column named BN_RESULT, which can be used as a filter to select only rows that satisfy the probabilistic constraint.

It is of note that, even though this thesis has focused on BNs as the CDS mechanism, this approach could be used to add probabilistic querying capabilities to AQL through use of other CDS methodologies, using the user-defined function method for filtering TPQ results.

The use of user-defined functions along with JSON representation presents another opportunity for future research around integrating various TPQ processing algorithms into XINO-P and other implementations. The current implementation of XINO-P uses the same logic for processing both nodes represented as rows and rows represented as JSON.

Since the strategy for collapsing some nodes of the DAG into JSON is implemented in the pre-persistence analysis phase of the methodology, multiple representations for groups of nodes can be used in the form of sub-trees which can be processed via different TPQ matching algorithms that would address these sub-trees. This approach would allow the use of algorithms that offer clear advantages for specific tree structures under a set of conditions which would not necessarily apply to all of the DAG. Examples of research relevant to this approach is covered in stack based twig matching algorithms (L. Chen, Gupta, and Kurul 2005) and improvements over them such as TwigList (Qin, Yu, and Ding 2007) as well as extensions of basic TPQs (Xiaoying Wu et al. Dec.). Handling pattern matching with logical operators (H. Jiang, Lu, and Wang 2004), (Izadi, Haghjoo, and Härder 2012), (Zeng, Jiang, and Zhuge 2011), improving performance of descendant only TPQs (Götz, Koch, and Martens 2009), leveraging parallel processing for TPQ matching (Machdi, Amagasa, and Kitagawa 2009) are examples of research outcomes that can be adopted in the context of this future research.

```
WITH
  ...

  ,from_nodes AS
  (....)

  ,T  AS
  (....)

  ,Z  AS
  (....)


  ,G AS
  (....)

SELECT
.... AS Z_value,
.... AS T_value,
.... AS G_value,
INFER_BN('BayesianNetwork1', '{T:'|| T_Value || ',Z:' || Z_Value ||' }',
        '{V}') AS BN_RESULT
FROM from_nodes fn
  LEFT OUTER JOIN T ON ...
  LEFT OUTER JOIN Z ON ...
  LEFT OUTER JOIN G ON ...
WHERE  BN_RESULT.V > 0.75
....
```

Figure 89: Implementation of probabilistic AQL in SQL via user-defined function call

An important design requirement of XINO-P, the ability to support both building of clinical information systems and CDS systems with the same architecture, is another important future direction for research which has been partly explored and experimented with. The SQL based TPQ matching that has been used for learning the parameters of the BN that is described Chapter 9 returns a result set that consists of values only. This is the most likely result set content for machine learning scenarios: the input to machine learning frameworks consists of values instead of data that represents complex objects.

Using the same query mechanism for clinical information systems implementation usually requires a higher level data representation in the query results that is closer in nature to the concepts of mainstream, object oriented programming languages, as modelled by the openEHR RM. The DAG based representation of the RM in XINO-P presents a higher level granularity: instances of RM types are modelled as a number of nodes. Therefore, if the SELECT section of AQL defines a node that corresponds to a complex object such as an Observation or an Entry, selecting the row that represents the root of this object is not enough to return the requested content. Inserting all rows that represent a complex object provides one option but this approach breaks the consistency of the semantics of the result set, due to the inclusion of a complex object in the SELECT section of the AQL query. Also, the process to construct a complex object from its DAG representation must still be dealt with.

In order to overcome this problem, EMF based pre-processing has been extended with the capability to link each node to its corresponding location in the XML content that is transformed to DAG. EMF supports navigation to a particular object in an EMF model instance, using a unique path that identifies that object. The EMF pre-processing builds this unique path for all nodes along with its openEHR path, and this path is assigned to an attribute of DAG nodes. The XML content that is processed via EMF is then persisted in a separate table along with the instance id that associates it to all the DAG nodes created from it. Figure 90 depicts the relationship between nodes and the original XML content.

The extended table schema for nodes allows TPQ matches to return EMF URIs (Uniform Resource Locator) for nodes. Therefore, if a node introduced by the SELECT section of AQL is required to be an RM object, this URI can be used to access the EMF representation of the object by loading the XML content as an EMF resource.

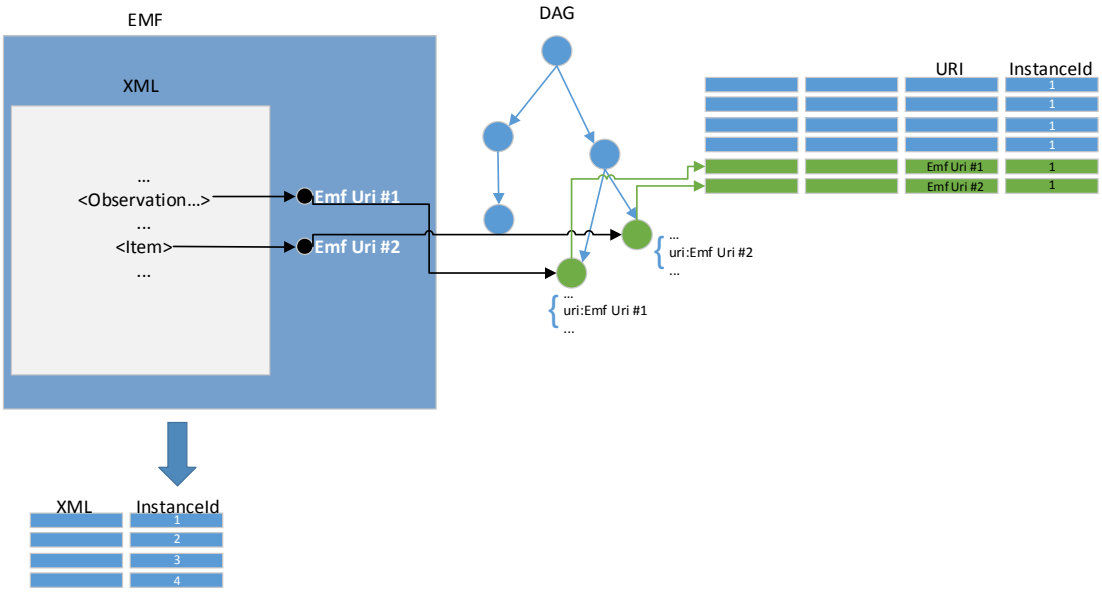


Figure 90: Representing references to XML nodes as a DAG node attribute

This approach requires an iteration over query results returned by SQL to identify XML payload(s) that should be loaded, therefore it implies a multi-step process for constructing the result set for TPQ matching, with potential performance problems that may arise in cases where large result sets are returned.

It is envisaged that access to an object form of AQL results is a requirement that is associated with use cases for clinical information systems , which usually require much smaller size data that will be used for human interpretation, compared to the model building and model training use cases for machine learning. Therefore, the dual content representation is considered a viable approach, allowing SQL based TPQ matching to address significantly different use cases with the same architecture.

8.5: Summary

The architecture and implementation discussed in this chapter has been the result of many iterative experiments, addressing persistence requirements with both clinical information systems and CDS natures. It is a specific implementation of a generic approach that uses the Postgresql relational database and SQL as the underlying persistence system.

Despite being at the proof of concept level, the Postgresql based implementation has allowed an openEHR driven CDS implementation to be tested. Although less time has been allocated for testing the clinical information system

implementation scenarios and requirements, limited experiments have confirmed feasibility of using this implementation for this purpose.

The TPQ matching approach has been implemented with SQL, and this implementation is used to support the fundamental semantics of AQL functionality in a CDS setting. At the time of writing, the AQL specification has a proposal status and is not part of the officially released openEHR specification. Therefore, some of its features are selectively implemented by software vendors, and there is not yet a clear definition of the functionality that an implementation must support to claim having a complete implementation of AQL. Two extensions to SQL based TPQ matching – user-defined functions and JSON representation of sub-trees – have significantly improved the capabilities and performance of XINO-P. These extensions provide a robust mechanism to deal with both fundamental and less frequently implemented features of AQL as well as being the basis of new features such as Probabilistic AQL.

Despite SQL providing the required expressiveness for TPQ matching, albeit with some extensions, the Postgresql based implementation is not immune to various problems. Some aspects of the current design, such as using a single database table for all clinical data, would benefit from further optimizations. The effect of growing data size on performance of queries on existing EHRs, difficulty of returning representations of complex objects instead of atomic values, and storage requirements of holding full paths for all nodes as well as complete XML documents for easy construction of complex objects, are all topics that require attention for the current proof of concept implementation to address real life requirements. These problems have been identified as key next steps for research, and early solutions are being identified at the time of writing of this thesis.

The existence of these various issues does not diminish the promising aspects of XINO. The extension of current work, both in terms of design and implementations based on new persistence technologies and data access scenarios presents a large set of options for future research.

The implementation discussed in this chapter is used for the experiments performed in the next one, to explore an end to end CDS setting based on openEHR

Chapter 9: An Experimental openEHR Based Clinical Decision Support Implementation for Ophthalmology: Risk Estimation for Cataract Operations

This chapter presents the development of CDS functionality based on an implementation of the persistence abstraction method developed in Chapter 7 on a relational database, as discussed in Chapter 8. This implementation, named XINO-P, is integrated to a BN that provides the decision-making capability for risk analysis prior to a cataract operation.

The persistence abstraction for openEHR described in Chapter 7 links openEHR methodology coherently with the different characteristics of implementations required for both clinical information systems and CDS.

Even though the use of different persistence systems can address issues of data size and parallel processing, a more detailed look at an actual openEHR-based CDS is required to identify other issues, independent of the underlying persistence implementation, in the use of BNs and other machine learning approaches for CDS in combination with an openEHR-based clinical record.

A number of issues arise:

First, ability to process larger amounts of openEHR standardised data does not in itself guarantee solutions to the barriers discussed in Section 2.2, such as the difficulty of integrating clinical information system implementations to CDS implementations. The extent to which openEHR's approach to modelling clinical information can support CDS requirements must be determined. The logical architecture for integrating openEHR and BNs set out in Section 4.7 assumes that the openEHR specifications can be used during the design of BNs , and this assumption must be tested.

Second, the clinical care process is the dominant approach in building openEHR models, and the implications of this approach in CDS integration need to be examined. As observed in the Opereffa experiment described in Chapter 6, a strong focus on a particular subset of use cases may fail to support others.

Third, there will be an inevitable interaction between the above two factors in any actual software implementation context, which is likely to introduce new complexities due to the specific underlying technology platform.

Therefore, testing the assumptions around openEHR's potential improvement of CDS requires a setting that includes the factors identified above, which is provided by the CDS implementation discussed in this chapter. This CDS

implementation calculates the probability of a complication that can be encountered during cataract surgery, based on the values of relevant clinical variables.

The clinical scenario for decision support was identified with help from Mr. Bill Aylward from the Moorfields Eye Hospital, who was actively involved in openEHR based clinical model development for ophthalmology in addition to leading a software development team tasked with development of an open source EHR implementation for Moorfields Eye Hospital. Mr. Aylward has also suggested the use of an existing study (Narendran et al. 2008) as the gold standard for the decision-making model. This study uses data from 55,567 cataract operations to build a logistic regression model (Agresti 2007), which, in turn is used to develop a risk assessment tool.

The clinical decision-making scenario from the ophthalmology domain, as defined by (Narendran et al. 2008), was used to define and implement a BN, which is then integrated to openEHR data hosted in the XINO-P. A data extraction pipeline was developed to load already existing patient data for cataract care from a legacy information system at Moorfields Eye Hospital into the XINO-P. However, the complexity of the relational database design used by this legacy software and its retired status at the time of the writing of this thesis produced an unreliable data set despite vigorous efforts. As a result, a synthetic data generation method was used to create clinical data, which was then persisted in XINO-P, using some components of the data extraction pipeline.

The openEHR based CDS setup was tested via use of ROC Curves (Metz 1978), however, comparison of the performance of the BN based CDS with (Narendran et al. 2008), was not possible due to lack of similar performance evaluation for (Narendran et al. 2008), accompanied by lack of access to data used by this study.

9.1: Relevant Research

The openEHR based CDS experiment (referred to as ‘experiment’ from now on) is built on the approach developed in (Narendran et al. 2008) which presents a predictive model for posterior capsular rupture (PCR) (Howard Vance Gimbel 1990), (Howard V Gimbel et al. 2001) and vitreous loss (VL) (Astbury et al. 2008). Both conditions are complications that can be encountered in a cataract operation. (Narendran et al. 2008) uses a data set that was collected from a single ophthalmology software to identify variables that are relevant to PCR and VL. The variables that are found to be statistically significant is then used to build a logistic

regression model for predicting the probability of PCR, VL or both for a patient. Clinicians are provided with the predicted probability of a complication during cataract surgery based on this logistic regression model, which allows taking precautionary actions such as assigning a more experienced surgeon to the operation.

Identifying a clinical problem that could benefit from CDS, building the CDS approach and identifying the initial set of relevant clinical variables require significant clinician input. Moreover, accessing clinical data that would allow development and testing of CDS is notoriously complicated due to the obvious sensitivity of clinical data. (Narendran et al. 2008) is a study that has fulfilled all of these critical tasks. Taking it as a template for an openEHR based CDS implementation allows this thesis to focus on openEHR and BN related aspects of CDS for a well-defined CDS setting.

For the purposes of distinguishing between variables that are included in the CDS model and data items in clinical models such as openEHR archetypes, the variables of the CDS model will be referred to as CDS variables below.

9.2: Setup of the Experiment

The experiment aims to develop a CDS implementation that serves as a workbench for openEHR and BN integration experiments, which, at the same time, makes it possible to observe the relative advantages and disadvantages of such an approach compared to logistic regression based method adopted in(Narendran et al. 2008). A comparison between the implementation discussed in this chapter and (Narendran et al. 2008) requires some extensions to logistic regression based approach.

First of all, (Narendran et al. 2008) uses logistic regression to develop a probability chart. This probability chart is meant to be used by a clinician to find out the probability of a complication for a particular patient based on two factors. The first factor is the baseline probability of a complication discovered from data, which belongs to a group of patients whose clinical and demographic variables have a certain set of values. This baseline probability of risk is multiplied by a ratio, which is the second factor for finding the probability of a complication. This ratio is obtained via a calculation based on the combinations of the values of clinical variables for a particular patient. The probability chart maps values of the ratio to the probability of a complication and the chart is the decision-making mechanism that is meant to be used by the clinician.

This approach requires the clinician to take action based on his or her subjective threshold for risk level, which is the estimated probability found by consulting the probability chart. (Narendran et al. 2008) does not include a performance evaluation of this decision-making method that shows the relationship between a threshold value for probability that classifies a patient as likely to have a complication or not and the overall successful prediction rate based on that threshold.

A meaningful comparison between BN based risk assessment and the logistic regression method developed in (Narendran et al. 2008) requires that they perform the same task for CDS, so that their performance can be compared. This task is defined as the prediction of a complication during surgery in the context of the experiment. This prediction is used to implement a classifier, which classifies patients into complications or no complications category prior to surgery, using a threshold value. When both logistic regression and BN classifiers are used for this task, their performance can be compared by visualising their classification performances via ROC curves, which shows their correct classification rates in response to changing the classification threshold.

Therefore, drawing ROC curves for both the logistic regression would be the first extension to (Narendran et al. 2008) that would be required for a healthy comparison. ROC curves would also allow the results of modifications to both approaches, such as changing the number of intervals for categorical variables as well as changes to openEHR models to be observed as well.

Another useful extension to (Narendran et al. 2008) would be the analysis of correlations between covariates, which corresponds to conditional dependencies expressed as parent-child relationships in a BN. (Narendran et al. 2008) does not include any interaction terms between covariates used in the logistic regression, which, translates to a rather simple topology for a BN with no conditional dependencies between clinical variable nodes.

These extensions to (Narendran et al. 2008), required to perform a meaningful and informative comparison with the BN based approach implemented in this chapter depend on access to the clinical data set used in (Narendran et al. 2008). In order to observe the results of adopting an openEHR and BN based approach, the same data set should be transformed into RM based data and persisted to XINO-P.

The most significant issue that has made extensions to (Narendran et al. 2008) ,and consequently the ideal comparison with the implementation discussed in

this chapter, infeasible is that the dataset that has been used in (Narendran et al. 2008) has not been available for access during the writing of this thesis.

Failing access to the original data set that was used, a second best option is to use a data set from another source, which must at least contain same data elements. In order to adopt this option, a data analysis and extraction project was initiated at the Moorfields Eye Hospital in London with the aim of building a data set that could provide the same clinical data as with the unavailable data set. This data set would then be used to repeat the development of logistic regression of (Narendran et al. 2008) from scratch.

The analysis step of the project aimed mapping the relational database design of a legacy information system, which contained cataract data, to openEHR clinical models that define cataract care. The extraction step aimed building a data transformation pipeline, which would allow exporting data from relational database to flat files and to XML files compatible with published XML schemas from the openEHR foundation, accomplishing a transform from the information model of the existing information system for cataract care to openEHR RM.

At the time of the data analysis, Moorfields Eye Hospital was in a state of transition from a retired information system, which contained almost all the historic cataract care data, to the OpenEyes system. This situation complicated the analysis. The retired status of the clinical software containing the cataract care data meant that enquiring about the design of the relational database tables and how data was laid out across them was not possible since the software vendor no longer provided support. Despite significant efforts from Moorfields IT staff, attempts to discover the location and semantics of the cataract care data led to a rather fuzzy data source for the data extraction and transformation pipeline.

The transformation pipeline was implemented in the Python programming language and it was used to apply transformations to results of SQL queries which were persisted into an HDF5 file, a scientific data persistence format developed by the HDF group (Folk et al. 2011, 5). This flat file was used as input to further transformations that generated XML files compatible with the published openEHR XML schemas. Therefore, the transformation pipeline had the capability to generate both flat files and openEHR XML files from the relational database.

Despite the flexibility of the pipeline, the lack of insight into how the data is kept in the relational database by the original clinical information system has led to unsatisfactory results. It was also confirmed by Mr. Bill Aylward that some of the clinical data included in the analysis of (Narendran et al. 2008) was not collected by this system. However, this issue was of secondary importance compared to the

difficulty of being sure that the data extraction mechanism was pulling the data it was thought to be pulling from the relational database. The following findings from the data analysis are therefore provided here with that underlying uncertainty:

The total number of cataract operations found in the relational database tables is 79656. These data were analysed with the view that each operation marks the end of a care episode that consists of one or more clinical examinations, a booking for the operation and the operation itself. This view was adopted as the data related to risk estimation for surgery is distributed across these care steps and for an operation to be included in the data set for a CDS, it must be complemented by other relevant steps in the care episode.

When records that either could not be linked to an episode or that did not include key data were excluded, only 16070 care episodes remained from the 79656 operations. Of these 16070 cataract care episodes, only 163 were found to have the clinical complication that was to be predicted. The ratio of complication (0.010) in the data set with complete episodic data is similar to the overall ratio for the whole data set (0.012), for which episodic relationships are discarded and only frequency of complications is considered.

The existence of only 163 incidents from 16070 observations shows the rather rare nature of the event, which compounds the data quality and uncertainty issues already mentioned.

Even though Moorfields Eye Hospital staff have been successfully providing reports from this data source for other purposes, the uncertainty of the data representation for this particular data set, and the amount of time that would be required to eliminate this uncertainty, meant that the analysis possible was insufficient for the purposes of this thesis. This was especially the case given that no guidance was available from the software vendor.

Given this situation, synthetic data generation was chosen as the means of generating a dataset with the properties required for the controlled implementation experiment being sought. The details of the synthetic data generation method developed are discussed in Appendix I, in full detail.

The primary benefit of using synthetic data is full control over data set size and clinical characteristics. As discussed in Appendix I, synthetic data generation can produce any number of observations with adjustable characteristics for subsequent analysis with the experimental CDS. The low prevalence of complications that the CDS implementation aims to predict means that a large number of observations are required in order for a sufficient number of occurrences of the event that needs to be predicted, to be observed. A study based on the data

set used in (Narendran et al. 2008) finds the overall prevalence of complications (either posterior capsular rupture, vitreous loss or both) to be %1.92 (Jaycock et al. 2007), which shows the low prevalence of the condition.

Another advantage of synthetic data set generation is the capability to generate patient data with target characteristics, for example in terms of numbers of high-risk patients. This capability allows detailed profiling of various CDS components such as the BN, not only at different data scales but also across changing prevalence characteristics of the outcome being predicted.

Although the synthetic data generation method makes it impossible to perform a detailed comparison between the predictions of (Narendran et al. 2008) and the experimental openEHR based CDS implementation, it nonetheless allows the CDS design to be based on clinical variables that were found to be significant for decision-making in a study based on real patient data.

The collaboration at Moorfields Eye Hospital on cataract data analysis and modelling has led to significant secondary benefits. Even though the initial focus of the analysis has been on cataract surgery data, significant interest in openEHR from the Moorfields Eye Hospital contributed to its use in another project.
The primary health IT project that had been under active development at the Moorfields Eye Hospital when cataract data analysis began is called OpenEyes. OpenEyes is a project that is closely associated with some of the key concepts discussed in this study.

First, its origins lie in the senior clinicians' awareness of data quality issues that stem from shortcomings of healthcare IT. Second, Mr. Bill Aylward who initiated the OpenEyes project to deal with these shortcomings has been aware of and interested in openEHR approach since before the initiation of OpenEyes.
This alignment in principles and methods has enabled an openEHR centric collaboration with the Moorfields Eye Hospital team. The use of openEHR archetypes in cataract data analysis contributed to discussions about using openEHR methodology in OpenEyes. A small scale project has been implemented with support from NHS to connect practices in Wales to OpenEyes for better glaucoma care to this end. This interoperability focused project used openEHR models for data exchange between clinical information systems and concluded with a successful pilot implementation.

The use of openEHR archetypes has also led to development of openEHR templates that were created by Dr. Ian McNicoll, a senior clinical modeller, in response to requirements derived through detailed study of the Moorfields cataract care records and the clinical decision support approach followed in (Narendran et al

2008). These templates were used as the basis of a data transformation pipeline that was required for analysis of cataract care data. They were also used as the basis of the experimental CDS implementation based on BNs discussed in this chapter.

9.3: Components of the openEHR Based CDS Experiment

The openEHR based experiment for CDS consist of the following primary components:

- Clinical models
- openEHR persistence implementation
- Predictive model
- Predictive model implementation

As discussed in 9.2, the openEHR based CDS was built on the approach developed in (Narendran et al. 2008) and therefore some of these components have counterparts in that study and some are specific to the openEHR based approach. Figure 43 depicts the relationship between the components of the openEHR based CDS experiment:

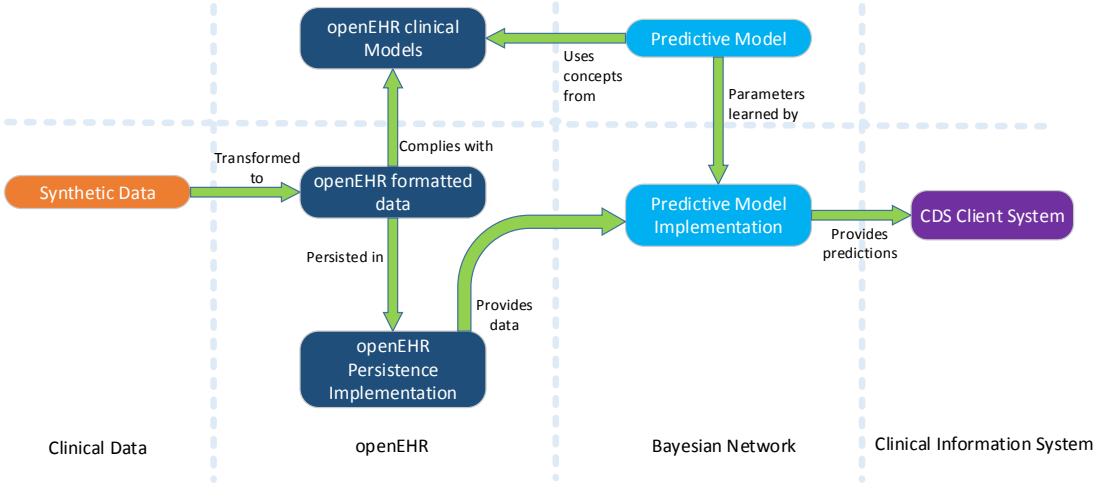


Figure 43: Components of the openEHR based CDS experiment

The starting point of the experiment is the synthetic data generation. The data generated through the mechanisms discussed in Appendix I was transformed into openEHR compliant XML files, which are represented in Figure 43 as 'openEHR formatted data'. This transformation to XML represents a step that would be necessary if clinical data were to be provided by any of the large number of

legacy EHR systems currently in use. Even though the data used in the experiment was synthetic, the process for moving the data into the openEHR persistence implementation followed a valid, reusable approach.

The structure of the openEHR formatted data complies with the openEHR clinical models in the form of openEHR templates. This data was persisted into the openEHR Persistence Implementation as discussed in detail in Chapter 8. This persistence implementation is based on the novel persistence abstraction of RM based data, which is discussed in Chapter 7.

The persistence implementation is used by the Predictive Model Implementation for its required data access, as Figure 43 shows. The Predictive Model Implementation in Figure 43 corresponds to a BN inference algorithm that calculates the probability of a complication during the cataract operation. The Predictive Model is a BN, which provides a decision-making model for probabilistic reasoning on the clinical data via defining clinical variables and relationships between them. These clinical variables are based on the data items defined in the openEHR clinical models, as depicted in Figure 43.

The nodes of the BN that represent the clinical variables, which are used to calculate the probability of a complication during cataract surgery, are based on data items defined in the openEHR templates. Therefore, a mapping from these RM based data items to a BN structure creates an implementation independent mapping from clinical concepts to probabilistic reasoning.

The horizontal dashed line in Figure 43 emphasises the technology agnostic nature of this mapping by separating clinical model definitions and predictive model (BN definitions) from components that are actual software implementations. Particular implementations of the technology agnostic CDS components that are above this line are discussed in detail through the rest of this chapter, but each component can potentially be implemented on many different platforms with different technology options. The openEHR XML format that was used as an intermediate data representation for clinical data could be replaced with other representations such as custom, binary encoded data, to achieve faster performance. The openEHR Persistence Implementation, which used a relational database for this experiment, can be replaced by alternative persistence systems, based on the approach developed in Chapter 7 and discussed in detail in Chapter 8. Likewise, the Predictive Model Implementation can be built with many different languages and algorithms.

Therefore, this experiment explores both the feasibility of generalising an openEHR based CDS architecture and the efficacy of a particular configuration of its

components. As discussed in Chapter 7, a principal assumption that such feasibility rests on, here, is the successful abstraction of RM based data persistence and access. This assumption was tested through a relational database based implementation.

Finally, the rightmost component of Figure 43 represents any client system that might use the predictive model implementation to calculate actual probabilities of a complication before a cataract operation.

The components in Figure 43 were built and integrated via a number of steps, which consist of:

- The development of openEHR archetypes and templates that provide formal models for the relevant ophthalmology data.
- The creation of the clinical data set based on the openEHR models and openEHR persistence implementation
- The development of an AQL query that provides access to the clinical data for the CDS.
- The definition of the structure of a BN.
- Building the connection between AQL query execution and BN implementation, for parameter learning and inference.

The following sections provide the details of these steps, followed by a discussion of the findings and a comparison of the experiment with the pilot BN implementation experiment discussed in Chapter 5.

9.4: Development of the Clinical Models

All clinical data used in the CDS represented via openEHR templates. Therefore, the first step for a CDS implementation based on openEHR is to build the clinical models. The models need to cover all the clinical data that has been classified as significant by (Narendran et al. 2008).

The openEHR archetypes that were used for the experiment were previously developed by an openEHR modelling expert Dr. Ian McNicoll who worked with Mr. Bill Aylward from Moorfields Eye Hospital. These archetypes are publicly available from an instance of Clinical Knowledge Manager (CKM) (Ocean Informatics 2015) software run under openEHR foundation's website. An initial review of archetypes with input from Dr McNicoll revealed that their scope covered most of the clinical scenario that the CDS experiment focused on. A number of additions and changes were implemented by Dr McNicoll based on the input provided by Mr Aylward and

the author. A number of new openEHR templates were produced as a result of these changes.

It was observed that previously developed openEHR archetypes, which were used as the basis of these templates, provided a significant amount of coverage for the clinical concepts that define the cataract care scenario, including the cataract operation. However, despite their relevant content, these archetypes did not include data items which would be required to represent some of the variables used in the logistic regression in (Narendran et al. 2008).

The missing data items in these archetypes consisted of both clinical and non-clinical variables. Clinical variables such as “patient can lie flat”, “Pseudoexfoliation” or “doxazosin” had not been included in the original archetype design because they were not part of the data that is collected at the Moorfields Eye Hospital. The remaining missing data items were non-clinical variables such as the age of the patient and the category of the surgeon performing the operation. Therefore, it was normal that they would not be included in the clinical model design process.

Despite the justification for their exclusion, these variables were still required for the predictive model. Therefore, a final set of changes to the openEHR models was implemented by the author. The inclusion of non-clinical variables in archetypes is not considered as a generic and widely acceptable solution, but this approach was considered as an acceptable workaround for the proof of concept implementation in the current experiment.

The resulting openEHR templates, therefore, represented a large superset of clinical information that was gathered across the three key stages of cataract treatment at Moorfields Eye Hospital.

The care process that leads to a cataract operation begins with an examination of the patient by the clinician. In case there is a need and also consent for a cataract operation, the operation is booked, and the cataract operation is performed, during which the complications PCR and VL may arise.

The three templates that were developed to model these steps in the care process are discussed next, with their structure depicted in figures.

9.4.1: Clinical Examination

The clinical examination of the patient was modelled by the template depicted in Figure 44. Figure 44 presents the openEHR template that consists of a top level COMPOSITION archetype (Cataract Clinic Note) that contains three

archetypes which in turn represent clinical concepts. The archetypes are marked in the figure. Data items within the archetypes are associated with CDS variables that are relevant to PCR and VL, as identified by (Narendran et al. 2008).

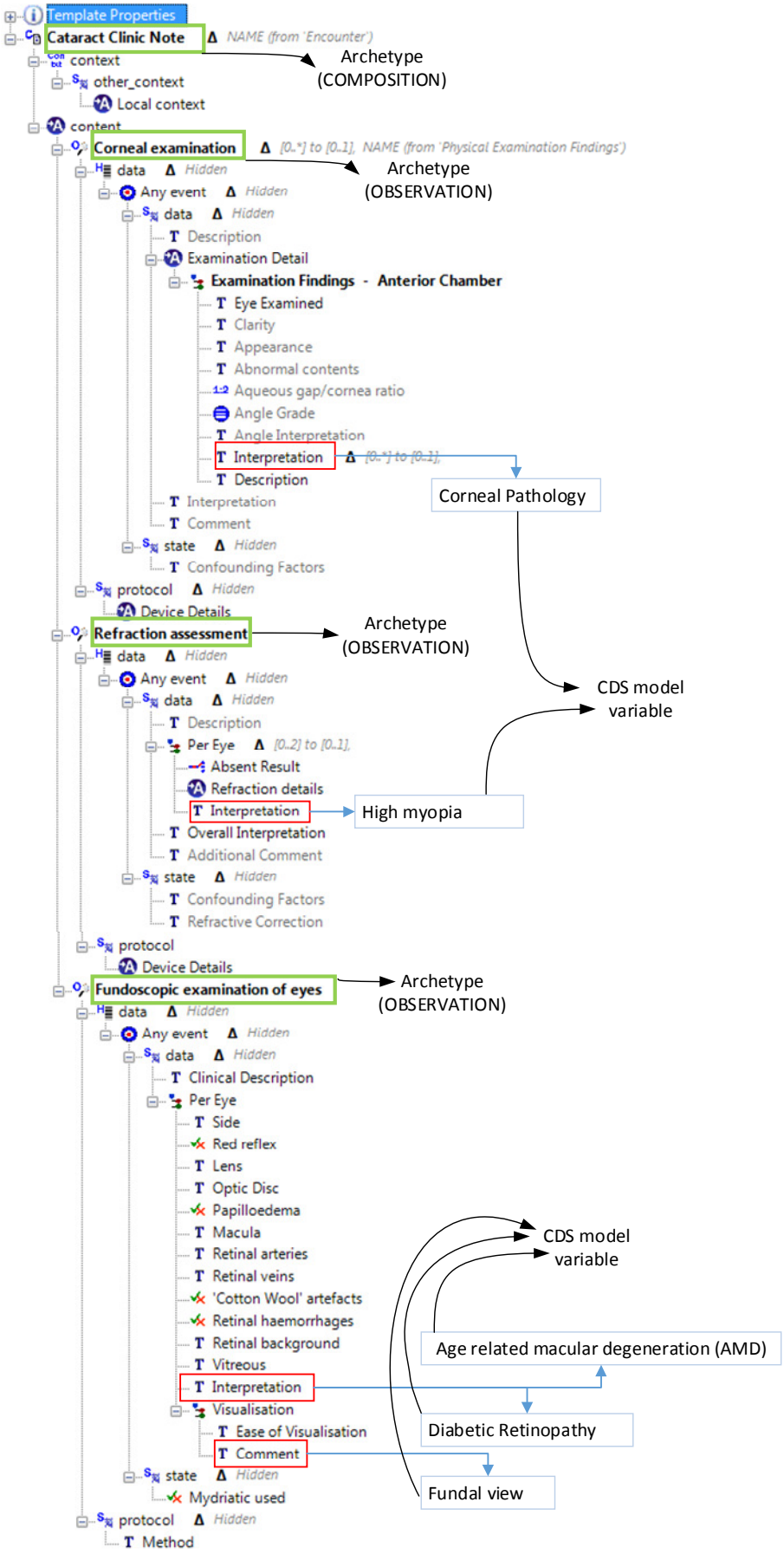


Figure 44: Clinical examination template

The fundoscopic examination archetype used in the cataract clinic node template contains a particular modelling approach that is likely to arise in clinical systems development: *Interpretation* data item corresponds to two different CDS variables: Age Related Macular Degeneration (AMD) and Diabetic Retinopathy. The *Interpretation* data item is capable of representing different content based on its value. The chosen modelling approach allows multiple interpretations of a clinical condition to be recorded at the same point in the RM instance. This is possible because the archetype is a model, and actual data instances may have a number of *Interpretation* data item instances as long as the archetype allows for a cardinality that is greater than one.

From a clinical modelling point of view, the advantage of using *Interpretation* to represent multiple data items is that a potentially very large number of clinical interpretations can be recorded against this field. If the modeller specifies AMD and Diabetic Retinopathy as separate fields instead of using *Interpretation* alone then this modelling approach leads to adding a new field for every problem that the modeller thinks the clinicians may record for a single eye. This is likely to be a problematic modelling approach since it requires the definition of a potentially large number of data items.

The side effect of using *Interpretation* field for representing multiple clinical problems is that the archetype path to multiple problems would be the same. That is both the AMD and Diabetic Retinopathy variables in Figure 44 would have the following path:

/composition[openEHR-EHR-COMPOSITION.encounter.v1]/content[openEHR-EHR-OBSERVATION.fundoscopic_examination.v1]/data[at0001]/events[at0002]/data[at0003]/items[at0007]/items[at0032]/value/value

This path would also point at any other clinical problems that could be recorded by the clinician. The actual semantics of the clinical data for this path is established through the *name* property of the *Identification* data item, which is also accessible via the path:

/composition[openEHR-EHR-COMPOSITION.encounter.v1]/content[openEHR-EHR-OBSERVATION.fundoscopic_examination.v1]/data[at0001]/events[at0002]/data[at0003]/items[at0007]/items[at0032]/name/value

Therefore, the semantics of particular data becomes dependent on a textual description, which is a significantly weaker form of computable healthcare data representation than openEHR's features can support. The weakness stems from the modelling approach used. Considering the fact that the *…/name/value* path may change based on the actual human language the implementation can use (English, Spanish, etc.) the problem may be elevated further when data sets from different sources that use different languages need to be merged for CDS.

It is important to underline that the modelling approach taken in this particular archetype by Dr. Ian McNicoll does not point at a weakness of the openEHR formalism. The clinical modeller who created the archetypes, suggested using openEHR's support for clinical terminologies to strengthen the semantics of the *Interpretation* field when notified about this potential problem, but this thesis kept the initial modelling approach in order to highlight and discuss the potential implications of modelling choices made.

The template in Figure 44 shows that clinical modellers can stay within the bounds of openEHR formalism to express clinical concepts and still face problematic issues of semantic interoperability. This is an important finding: openEHR formalism cannot guarantee the realisation of all of its potential benefits without consideration of how the models are going to be used.

For this particular modelling issue, using openEHR's support for terminologies would solve the problem. Through the use of a code from a widely used terminology such as SNOMED-CT (IHTSDO 2015), via the DV_CODED_TEXT openEHR type, the archetype and resulting RM data would become resilient to these semantic interoperability issues.

The remaining CDS variables that are based on data points in the template are similar to AMD and Diabetic Retinopathy: they are represented with text fields and the discussion about the use of the *Interpretation* field is valid for them as well.

9.4.2 Pre-Operation Booking

During the clinical care process, a clinical examination may lead to a decision to perform a cataract operation. In this case, the operation needs to be booked and clinical data expressed in some of the CDS variables from (Narendran et al. 2008) is created at this step.

Figure 45 depicts the association of data items from the openEHR template for pre-operation booking to CDS variables in the same way as in Figure 44.

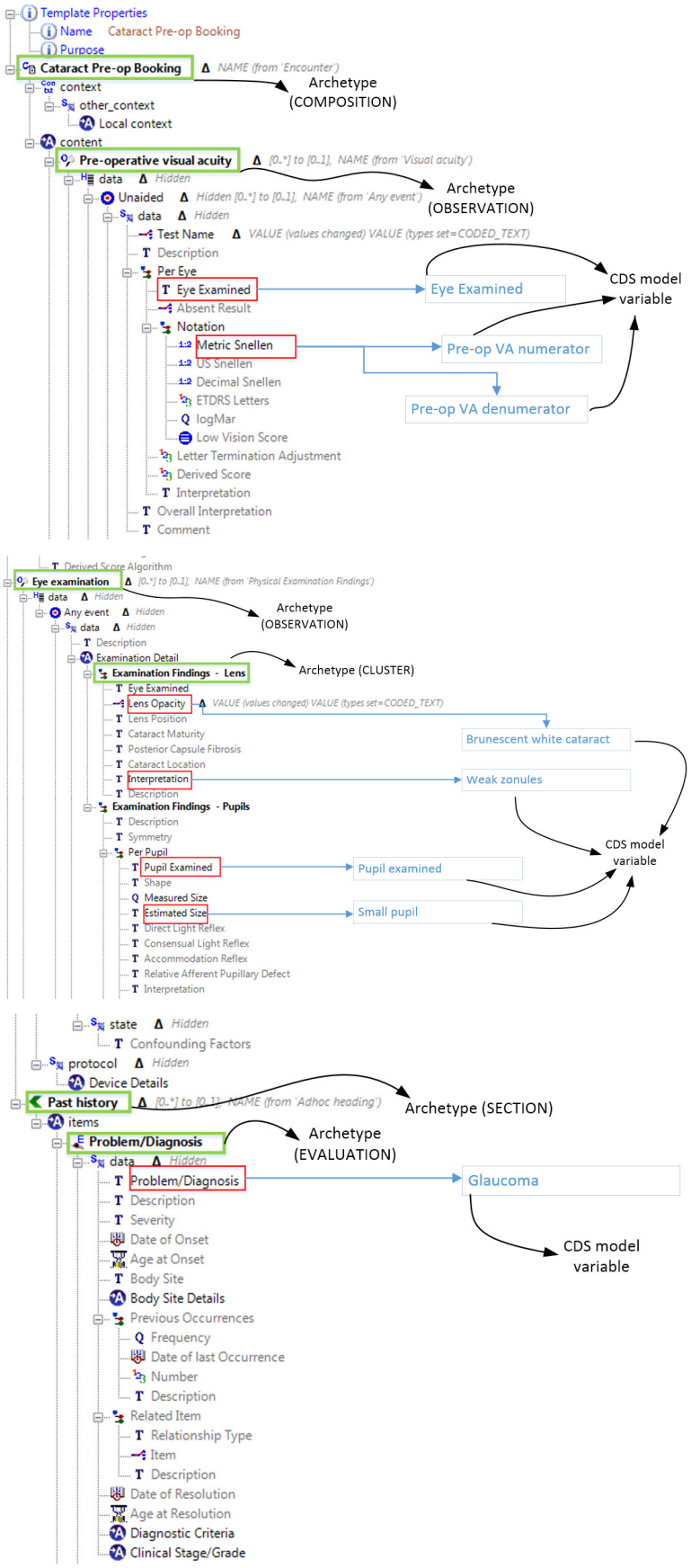


Figure 45: Pre-Operation booking template

This figure includes only sections of the template that contain CDS variables for the sake of clarity.

The associations in Figure 45 are similar to ones in Figure 44. The numerator and denominator in the diagram are different fields of the *Metric Snellen* data item and not two different variables with the same archetype path as discussed before.

The Cataract Pre-Op Booking template used a key clinical modelling practice supported by openEHR: an exclusion archetype. Even though it is not associated with a CDS variable, and not used in the implementation, the *Exclusion of a Problem/Diagnosis* archetype included in the template is related to the semantics of the CDS variable Glaucoma. It explicitly expresses the semantics of a patient having no glaucoma. This is a modelling approach that the clinical modeller can use to clarify the meaning of lack of a data item.

An example case that requires this clarification may be a change in the clinical record keeping practice: eg a new piece of clinical data that was not recorded previously now needs to be recorded routinely during the care process. After a while, there would be two groups of patients with missing data for this clinical variable. The first group would have their data recorded when the care process did not record this clinical variable. The second group would have missing data due to their clinical condition not requiring its creation (as in no glaucoma) even though the clinical variable is included in the care process and recorded.

If lack of this data item in data is interpreted as the condition does not exist, this interpretation would incorrectly classify patients who had the condition but whose treatment took place during a time when the condition was not recorded. Using the exclusion archetype as in the Cataract Pre-Op Booking template allows a clear interpretation of missing data. However, using this modelling approach does not guarantee that actual data will have the required semantics. A clinician may not record the existence of a condition such as glaucoma to express a lack of it and may ignore recording its exclusion, which may thereby lead to an ambiguity between missing and not recorded data.

To avoid this situation ADL's support for invariants may be used to force recording of an exclusion when a particular condition is not observed, but this scenario may complicate the modelling process for the cases where the number of conditions that may require this check is large.

The use of exclusion archetype provides a good example of the situation in which clinical models are developed with a focus on clinical care and even though there are powerful mechanisms available to clinical modellers, they may not be

employed because they are primarily needed for secondary use cases such as building a data set from population queries.

If the clinical modeller focuses on the clinical care process, handling the ambiguity of the type discussed above is not a significant problem for the users of a clinical information system. A clinician may spot the lack of a particular condition and request clarification. However, secondary uses of clinical data mostly happen in contexts other than direct clinical care, with little or no possibility of confirmation or clarification of clinical data.

9.4.3 Cataract Operation

The final step in the clinical care that is included in the scope of the clinical modelling is the cataract operation. Figure 46 depicts the openEHR template that was used to represent the cataract operation.

Figure 46 associates the *Clinical Interpretation* data item to PCR and VR CDS variables, which represent the clinical complications that (Narendran et al. 2008) aims to predict.

The PCR and VR complications are represented through the same data item, and therefore the ambiguity when selecting their values using archetype paths must be removed using the name attribute of the data item as in the case of the clinical examination template in Figure 44.

The operative report template is the last of the three templates that were developed for managing data for cataract care using openEHR. These templates correspond to the openEHR clinical model component of the CDS architecture in Figure 43, and they are the mechanism through with actual clinical data was created. As discussed Section in 9.2, synthetic data generation was chosen as the method for building the data set for CDS implementation. Due to the openEHR driven nature of the CDS, this approach required that synthetic data was transformed into openEHR RM based data using these openEHR templates.

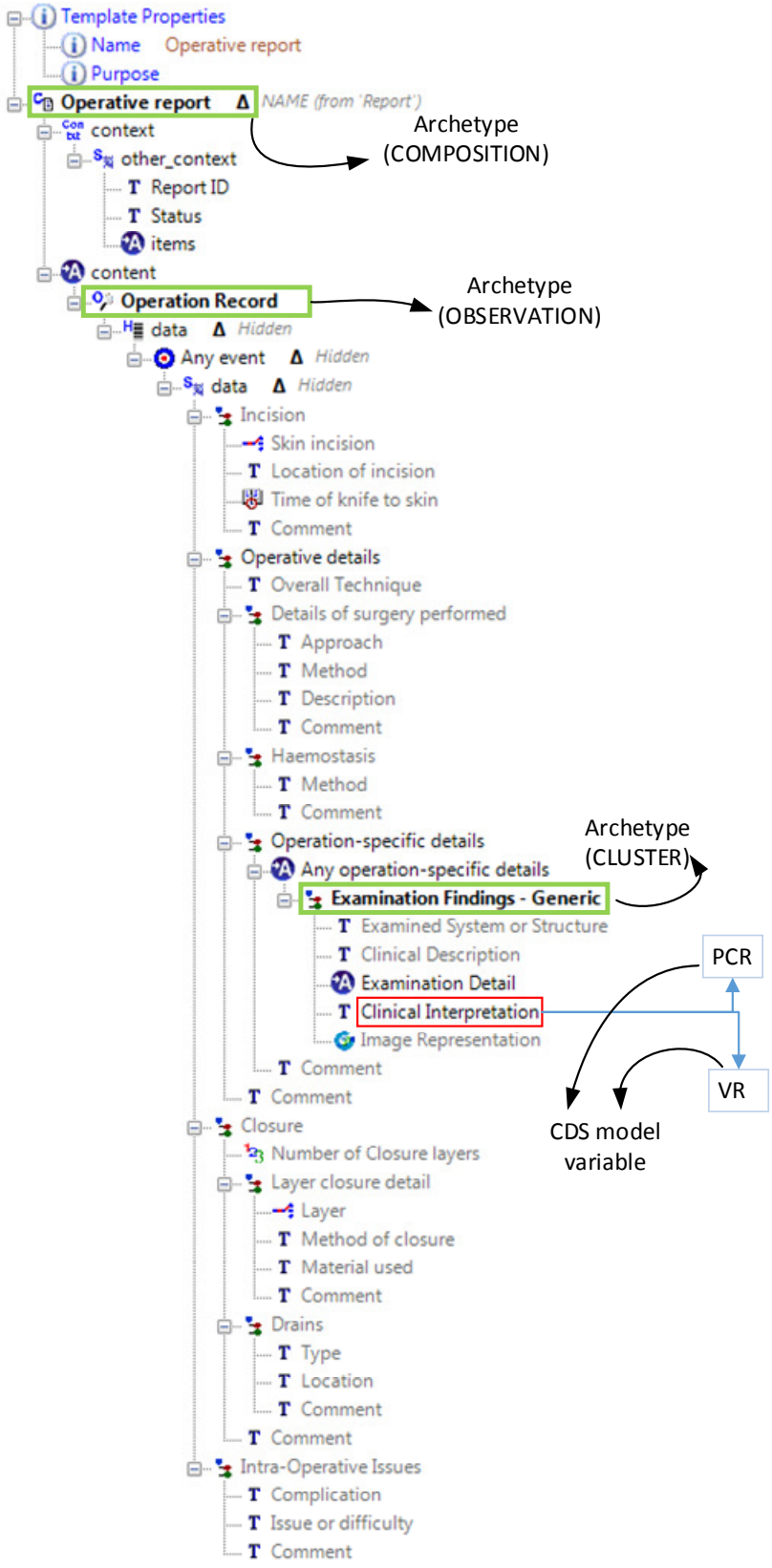


Figure 46: Cataract operation template

9.5 Data Transformation to openEHR RM

Aside from the synthetic nature of the data, transforming the simulated data to RM based data represents a realistic requirement for making use of any existing clinical data for an openEHR based CDS implementation. This requirement originates from the existence of large quantities of clinical data that is kept in legacy systems, which could be used as a data source for a CDS based on openEHR.

The methods used for the transformation are independent of the actual values of the clinical data and therefore the synthetic nature of the data used is not an issue in the following discussion of the transformation.

The transformation used the openEHR XML schemas, which are published as part of the openEHR specifications, as the target, which is a common approach in openEHR implementations.

Synthetic data was transformed into XML form using XSLT, producing valid XML files according to XML schemas, which were automatically generated from openEHR templates. These automatically generated schemas are called Template Data Schemas (TDS) and XML files that are valid according to these schemas are called Template Data Documents (TDD).

The first component of the transformation implementation is the generation of XML files (TDDs) that are used as a placeholder for data items. Figure 47 depicts how these XML files were generated.

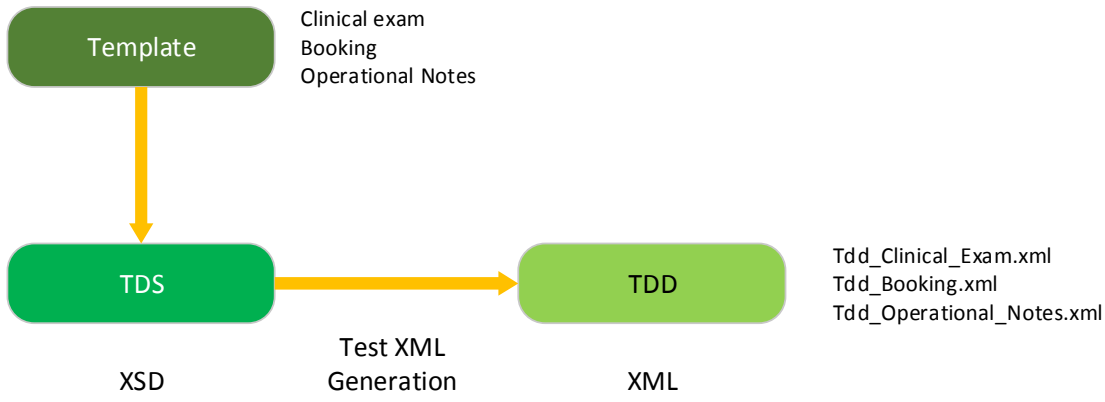


Figure 47: Test XML document generation from XSD

The openEHR templates discussed in Section 9.4 were exported as TDSs from the freely available Template Designer tool. This tool allows development of openEHR templates using archetypes and it also supports the capability to generate TDSs from templates.

Since a TDS is an XSD, it is possible to generate test XML document instances based on it using widely available XML tools, as shown in Figure 47. In this case, the XML data generation capability of the Eclipse Web Tools Platform (WTP) project was used to generate individual XML files for each of the three openEHR templates created for the experiment.

From an openEHR perspective, these test files are TDDs. Three XML (TDD) files were generated as a result of this step. The initial content of the data elements in these files were assigned by the XML tool, based on the information in the XSDs (TDSs). These files were used as placeholders for clinical data for a single patient's cataract care. Synthetic clinical data was injected into these files, replacing values generated by the XML tool, leaving the rest of the content the same, at their automatically generated values. Figure 48 visualises this process.

Figure 48 shows how synthetic data related to a single patient episode was injected into three TDD files through the use of XSLT. The XSLT processing injects values from synthetic data to relevant points in the TDD, and other content in the TDDs is left untouched. Since these values are generated based on the XSD (TDS), they may not always be clinically meaningful. The experiment left these values untouched for the following reasons. The data contents of the CDS data set is a subset of the clinical data defined by the three openEHR templates, so any data outside of this data set were not used in the CDS implementation. However, it is necessary to have a realistic content structure in the XML files (TDDs), so that the persistence implementation based on the approach developed in Chapter 7 can be tested with as realistic content as possible.

Therefore, having XML files (TDDs) with realistic sizes were considered important as well, since this factor is likely to become significant from a performance point of view for the persistence implementation as the data size grows. Since validation of data values based on their definitions in the clinical models is not within the scope of this study, automatically generated XML data that is not replaced by synthetic data values was left as it was.

The process depicted in Figure 48 was performed for each row of synthetic data, which represents a patient episode consisting of a clinical examination, operation booking and cataract operation. The resulting XML files (TDDs) required another transformation. The need for this transformation stems from the nature of TDS. A TDS defines a template specific type system based on types from RM. These types enforce further modifications to archetype data items that are modified in the openEHR templates so that XML data based on the TDS always conforms to these specialisations. At any point following its creation, the contents of the XML file

with data (TDD) can always be translated back to canonical, unmodified RM types. Therefore, the TDS mechanism can be considered as a small, and template specific type system based on openEHR. Its outputs can always be translated to the canonical XML form. This means that openEHR implementations can work on the basis of canonical XML definitions, based on the fact that openEHR template based data can always be transformed into this form.

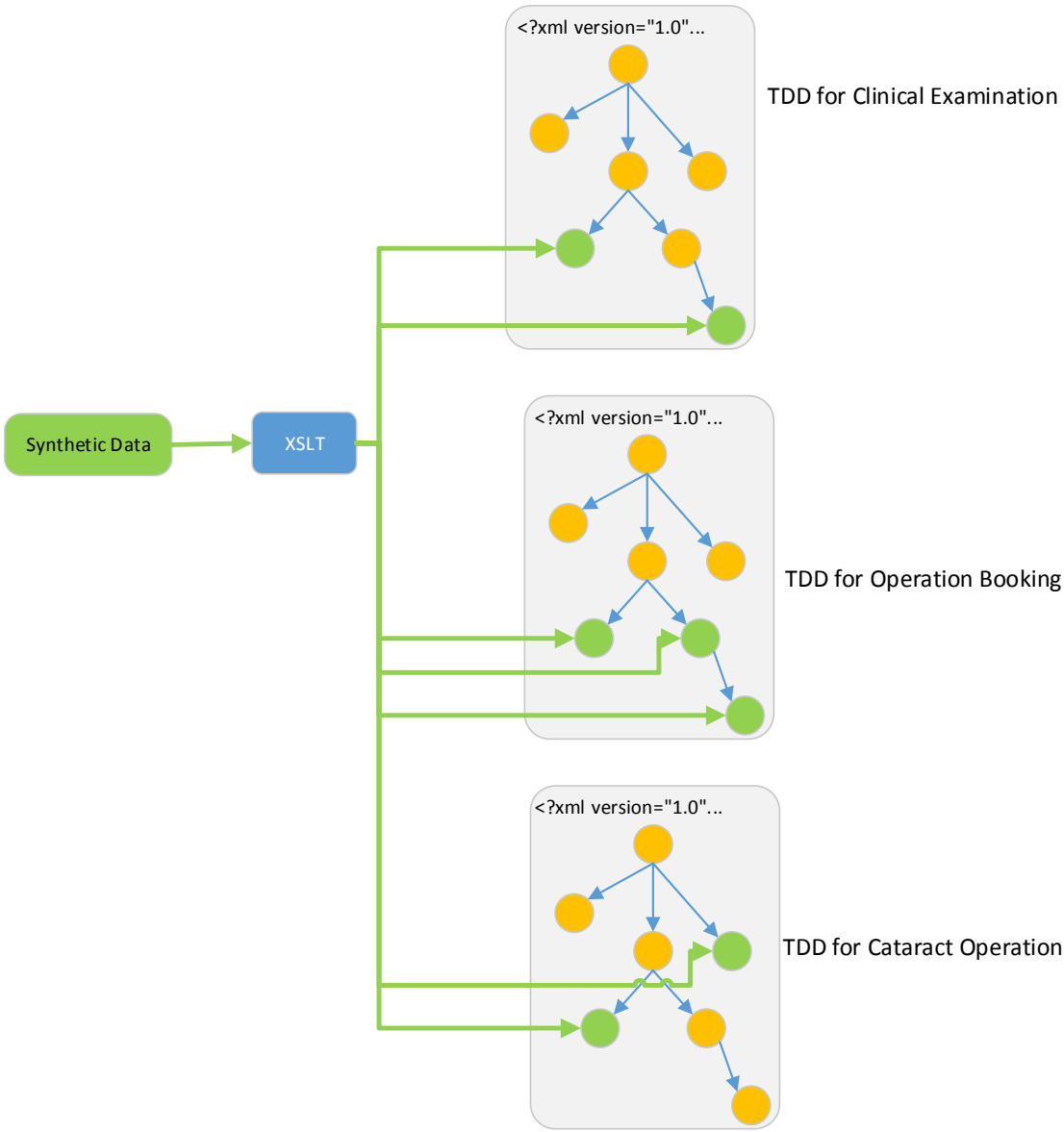


Figure 48: Inserting synthetic data to TDDs

The (TDDs) with content based on synthetic data were finally transformed to canonical XML files, which are valid according to canonical XML Schema documents published by the openEHR foundation, using this approach. Following this transformation, XML data was processed by the persistence layer

implementation. The details of this step are discussed in Chapter 8. Figure 49 complements Figure 48 and describes the whole process.

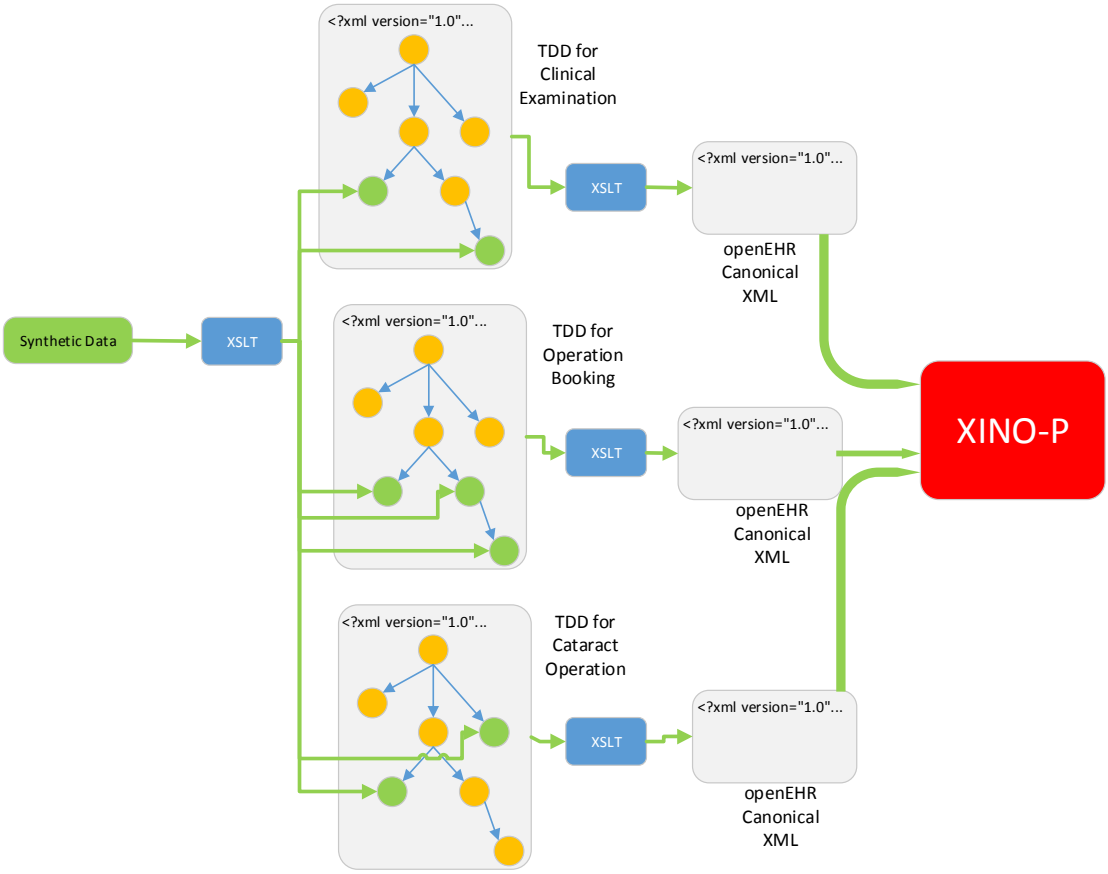


Figure 49: Persisting openEHR data to XINO-P

The process depicted in Figure 49 provides a good approximation of a real-life data import method from a legacy system for an openEHR implementation. The data import process implemented for the experiment concludes with the population of clinical data in the openEHR persistence implementation, called XINO-P, which establishes a workbench for performing experiments with the openEHR based CDS. Based on the approach developed in Chapter 7, this persistence implementation was accessed via AQL queries from the CDS.

9.6: AQL Based Data Access for CDS

9.6.1: Using AQL for Use Cases involving Non-Clinical Care Data

AQL has a pivotal role in openEHR data access due to its capability to provide a standard access method to data, independent of the underlying

persistence implementation. Therefore, an AQL query, as a means of implementation independent openEHR data access, was required to map the data items in the cataract care templates to CDS variables. The results of this query provided the clinical data for the CDS implementation.

The CDS data set that was used in the experiment does not focus on a single patient or a particular step in the care process, but as observed from the associations between data items across multiple templates and CDS variables, aggregates data from a number of steps in the care process.

A common approach to this type of data aggregation requirement is to export data from a clinical information system to another form, which can be used for analysis purposes and other secondary uses, including development of CDS mechanisms. Based on this common practice, extracting data from an openEHR implementation to a format that can be further transformed and modified so that it becomes native to the tools used for CDS model development, would be a valid approach for developing a CDS based on openEHR.

However, moving data out of an openEHR context for the purposes of CDS model building limits the use of openEHR to clinical care only. An extended use of RM based data and AQL is required to observe how openEHR methodology and its implementation aspects perform in settings beyond clinical care such as CDS system development. Using AQL for population queries to build a data set for a BN based CDS served this goal.

9.6.2: Data Aggregation

AQL was used in the experiment to aggregate data from multiple steps in the care of a patient. This data was used for building the CDS model, which is a BN. Since the care steps were modelled via openEHR templates, the aggregation used them as a definition of the clinical data source, as depicted in Figure 50.

The AQL query in Figure 50 uses data items from the three templates discussed in Section 9.4. The full AQL query from Figure 50 is provided in Figure 51.

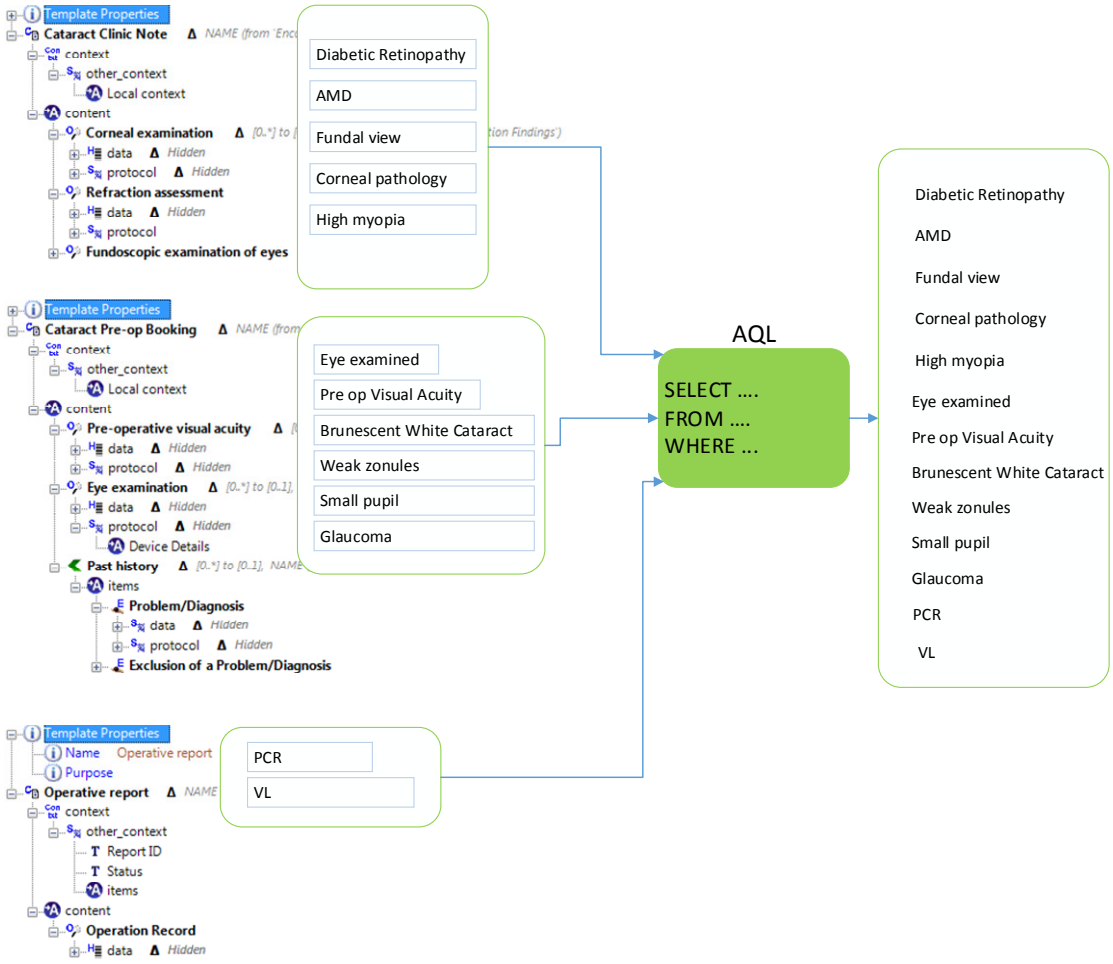


Figure 50: AQL query for CDS: relation to openEHR templates

The AQL query in Figure 51 aggregates data points from the three templates. For the sake of clarity, paths in the *SELECT* clause have been shortened in the diagram. The query defines instances of COMPOSITION RM types that are the roots of their respective templates and uses the CONTAINS AQL statement to define OBSERVATION instances that are under the COMPOSITION instances.

These definitions take place in the *FROM* clause of the AQL query. The *SELECT* clause then uses the references to OBSERVATIONs as the root of a number of archetype paths that define the data items to return as the query result.

The variable defined with the alias ‘e’ in the *FROM* clause has no constraints on its ehr_id attribute since this query is meant to process all clinical data that fits the criteria regardless of whose EHR contains it.

The conditions defined in the *WHERE* clause ensure that using the same archetype as the root of a number of templates does not lead to ambiguity.

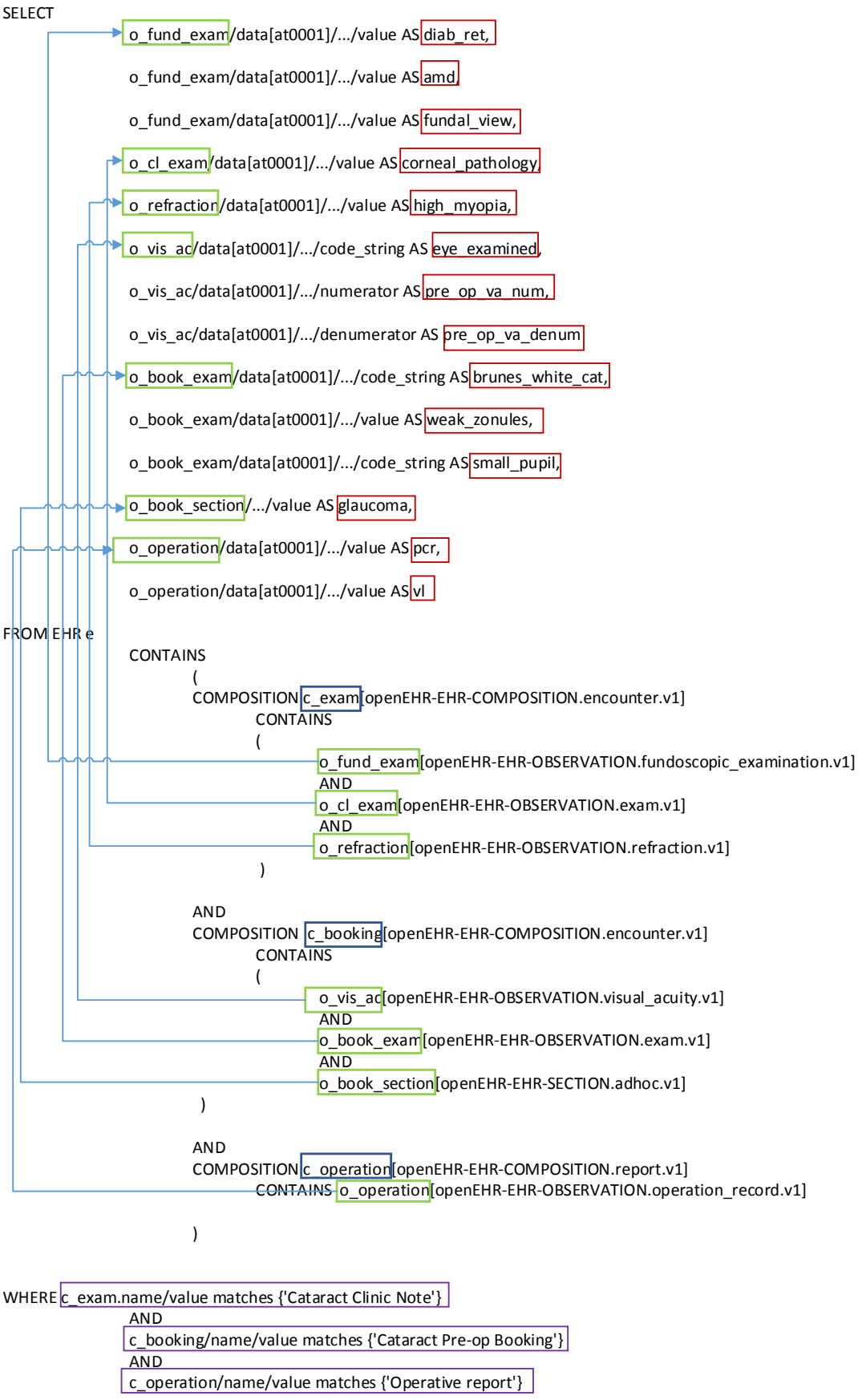


Figure 51: AQL query for CDS

All the templates that are used by the AQL query have the same root archetype as the root of the template: a COMPOSITION archetype with the archetype id ‘openEHR-EHR-COMPOSITION.encounter.v1’.

Since archetypes are meant to be reused within templates, there is nothing problematic in this setting, but the templates are defining different clinical concepts and therefore they must be clearly identified in the AQL query. Their names are used as constraints in the *WHERE* clause to distinguish the root COMPOSITIONs of templates

Following the persistence abstraction approach of Chapter 7, the AQL query in Figure 51 can be represented in the TPQ form as depicted in Figure 52.

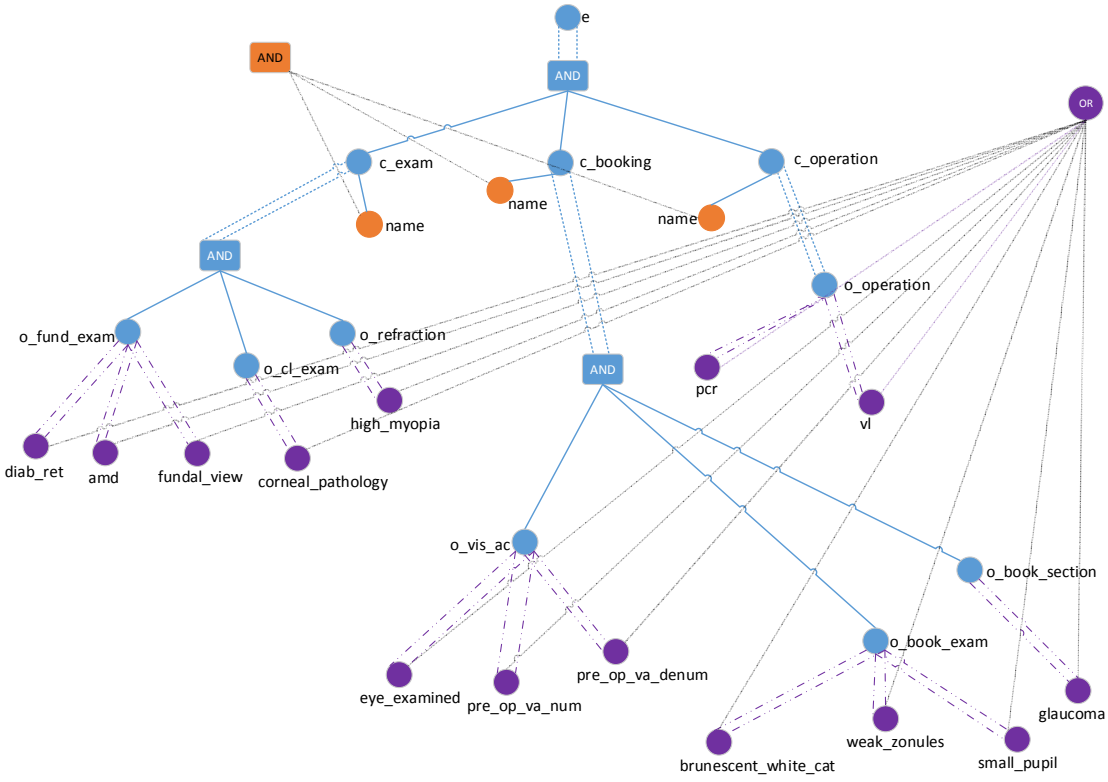


Figure 52: AQL query for CDS as a TPQ

Figure 52 shows how the AQL query aggregates information from different compositions that are created during the care process. It also shows how CDS variables are distributed across these compositions.

The existence of the EHR typed ‘e’ node is required for two reasons. First, the syntax of AQL requires that the *FROM* clause has a single root item which acts as a parent for other items which can be expressed via the CONTAINS statement. Therefore, it would not be possible to group the three key COMPOSITION data

items (c_exam, c_booking, c_operation) without using a shared container (e). Second, even though the ‘e’ node has no EHR id constraints, it still implicitly forces the c_exam, c_booking and c_operation nodes to exist under the same EHR, through containment constraint.

The TPQ form of the AQL query employs a logical OR interpretation for the data items defined by the *SELECT* clause, which allows the building of a data set for the CDS that may contain missing values. This flexibility leaves the option of making use of information about missing data in various steps of BN development for the CDS.

9.6.3: Issues Encountered

The AQL query developed in Section 9.6.2 was used to build a data set from a simulated patient cohort. This approach led to significant findings regarding the use of AQL for defining and creating a population data set.

9.6.3.1: Non-clinical CDS Variables

Both the TPQ in Figure 52 and its underlying AQL query define a list of data items distributed across three COMPOSITION instances, which are based on three templates. This list leaves out some of the CDS variables identified by (Narendran et al. 2008) because these variables do not contain clinical data. Two such CDS variables are the age of the patient and surgeon grade.

Both variables were found significant by (Narendran et al. 2008)in terms of their contribution to the probability of complications related to cataract surgery and were, therefore, included in the logistic regression. However, these variables were not included in the openEHR templates, because age would usually be a data item associated with patient demographics, and surgeon grade is likely to be classified as administrative data, based on the classification of surgeons in a hospital.

The openEHR RM allows representation of such variables, but as this experiment shows, they are not necessarily considered relevant when modelling clinical data with a focus on the particular steps of a care process.

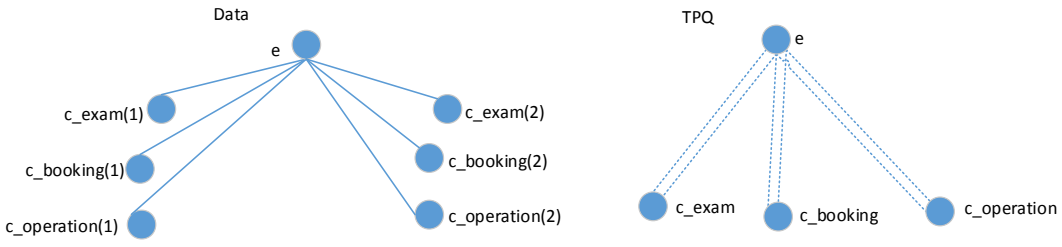
9.6.3.2: Lack of a Care Episode Identifier

The steps of the care process of a patient: clinical examination, operation booking and cataract operation, were modelled via separate openEHR templates. A clinician accessing information in these templates can use the dates associated with

data entry for each of these steps to build an ad-hoc view of the care process even if the three steps do not have an explicitly defined data item that identifies a care episode.

openEHR RM allows for tracking of such care episodes without explicitly including this information in the clinical models. This capability is based on its support for making use of external systems for various tasks, such as using an external workflow engine to associate clinical data with workflow steps. But even when information about a care episode is not tracked, the users of the clinical information system that is based on the templates can still construct the temporal sequence of aspects of the care process, intuitively, using date information which is likely to be provided.

In the context of the experiment, it was observed that not having episodic data for the care process may cause problems due to the nature of the clinical condition CDS focuses on. In case of cataract treatment, a patient can have more than one operation if the problem exists in both eyes. This means that a patient's EHR may have more than one instance for each of the steps that make up the care process. This situation may lead to duplicate data in the AQL query results. Figure 53 depicts a simplified tree representation of RM based data in a patient's EHR, along with the relevant part of the TPQ, again, simplified, from Figure 52. The hypothetical patient's EHR contains two episodes of care that concludes with a cataract operation.



Matches for TPQ

c_exam(1)	c_booking(1)	c_operation(1)
c_exam(1)	c_booking(2)	c_operation(1)
c_exam(1)	c_booking(1)	c_operation(2)
c_exam(1)	c_booking(2)	c_operation(2)
c_exam(2)	c_booking(1)	c_operation(1)
c_exam(2)	c_booking(2)	c_operation(1)
c_exam(2)	c_booking(1)	c_operation(2)
c_exam(2)	c_booking(2)	c_operation(2)

Figure 53: Unintended, duplicate TPQ matches

Both episodes include a clinical exam, booking for an operation and the operation. The TPQ that is a simplified representation of the one in Figure 52 has no consideration for the episodic nature of the data, leading to an unintended number of matches. The problem is due to TPQ defining a tree pattern based on archetype ids, which can be satisfied in eight different combinations of results by the data tree, as depicted in Figure 53.

The intended representation of the clinical data for this patient consists of the first and last rows of the table in Figure 53, grouping COMPOSITION instances based on their episodes. The AQL used for the CDS data set needs to include a constraint that would allow the steps of a cataract care episode to be grouped together to express this intention, explicitly.

When the episode identifier is not included in the modelling, it is not possible to introduce a condition to the AQL query based on this identifier. Moreover, if the episode identifier were to be included in the modelling phase, its use in AQL would require features that are not explicitly defined by the current AQL specifications. The condition that must be expressed in AQL in order to avoid the unintended duplicate results depicted in Figure 53 is the equivalence of episode identifiers of compositions included in the query. This equivalence condition does not require expressing the actual value of episode identifiers, it only requires that three COMPOSITION instances that represent the care steps have the same identifier value. Expressing this condition in AQL requires referencing values of data items within the query without explicitly providing values. A natural way of doing this would be extending the *WHERE* clause. Pseudo AQL code that expresses the equivalence of episode id data item values, based on this approach would be:

"*…WHERE c_exam/…path_to_episode_id…/value = c_booking/…path_to_episode_id…/value AND…*"

Figure 54 shows how equivalence of episode ids for COMPOSITION instances recorded under episodes 1 and 2 can be expressed without referring to actual values.

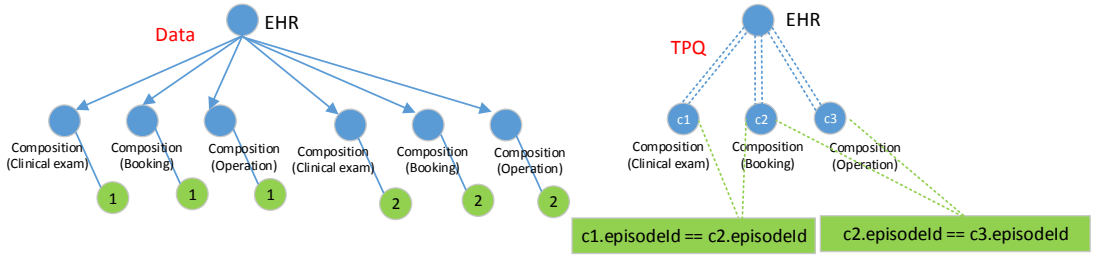


Figure 54: Episode id in data and query

The current specification of AQL does not clarify if a constraint based on the equivalence of values of different data items can be used in the *WHERE* clause. The experimental implementation discussed in Chapter 8 supports this capability via the use of SQL's support for expressing these types of conditions.
The experiment did not encounter this particular problem since the synthetically generated data set included only one episode per patient EHR. However, the requirement to support constraints on relative values is obvious.

9.7: The Bayesian Network

The BN for risk estimation is the main clinical decision support component of the experiment. It replaces the logistic regression used in (Narendran et al. 2008) as the means of predicting the probability of a complication during cataract surgery. The structure of the BN is provided in Figure 55 in the form of a screenshot taken from the BN tool GENIE (Druzdzel 1999)

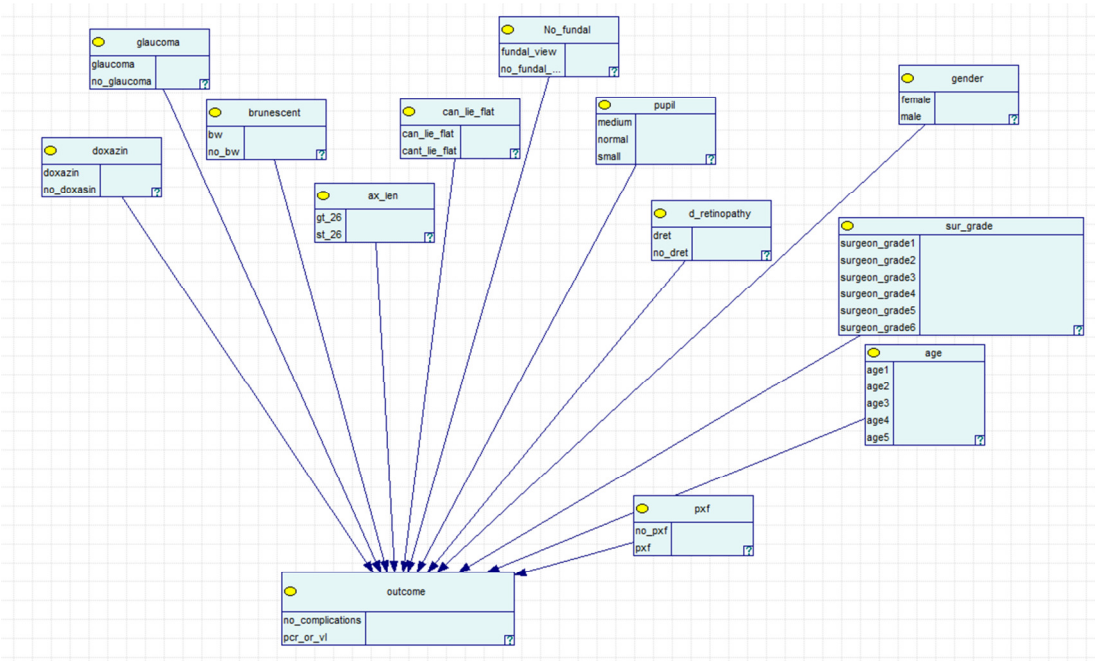


Figure 55: BN for CDS

9.7.1: Network Structure

The structure of the BN in Figure 55 is based on the logistic regression model developed in (Narendran et al. 2008). The logistic regression model has an outcome variable with two possible results and 12 covariates, and this model is

encoded directly in the structure of the BN. The outcome variable has a dependency on all factors identified by (Narendran et al. 2008), and there is no dependency relationship between any of these factors. Only the structure of the BN is based on the semantics of the logistic regression derived by (Narendran et al. 2008). The actual probability distributions of the nodes of the BN were learned from the CDS data set.

Defining the structure of the BN in this way has some disadvantages compared to alternative structures that could have expressed dependency relationships between clinical variables, such as the BN learned from data in Section 5.3. Since a BN encodes a joint probability distribution of categorical variables, all the nodes with a high number of parents end up having large conditional probability tables. A key advantage of a BN is that it allows significant computational savings based on the conditional independence properties of variables. When there are very few conditional independence relationships in the network, both storage and computation requirements of variables increase, which has been the case for the network in Figure 55. The outcome variable has 12 parent variables leading to a conditional probability table with 92160 entries.

It is possible that the covariates of the logistic model may have some degree of interaction between them, especially between the age variable and some clinical conditions. But the model developed in (Narendran et al. 2008) does not include such interaction variables. Since this experiment did not have access to the dataset underlying the logistic regression model in this study, checking for correlations in the data set for a more expressive BN Network structure has not been possible. Therefore, the BN used in the experiment has a structure that mimics the relationships introduced by the logistic regression model.

All covariates in the logistic regression model in (Narendran et al. 2008) are categorical. This allows the BN nodes to be parameterized based on the outcome categories of the corresponding covariates in the logistic regression model. For example, a continuous value such as the age of the patient is represented with a discrete covariant with values that represent five age categories in the logistic regression, and the corresponding node in the BN that represents the distribution of the age has five outcomes which correspond to these categories.

9.7.2 Network Parameters

The values of the BN parameters were learned from synthetic data through the use of bnlearn (Marco Scutari 2009), a package developed for the statistical

programming language R (R Development Core Team 2008). The parameters of the network, i.e., the distributions expressed by conditional probability tables were learned through the maximum likelihood method (Scholz 2004) implemented in bnlearn.

It was observed that learning network parameters in this setting led to missing values for some conditional probabilities. The maximum likelihood approach in bnlearn assigns “not available” to some conditional probabilities in the network if the data set does not contain a sufficient number of observations for those configurations of variables. Therefore, not having access to conditional probabilities for some configurations of variables in the network leads to not being able to predict outcomes for these configurations.

In the context of the network used in the experiment, the high number of entries in the conditional probability table of the outcome variable (indicating whether or not a complication is expected) and the rather low prevalence of the complications during a cataract surgery, exacerbates this problem. 92160 conditional probability entries and the infrequent occurrence of the event the model aims to predict (complication during surgery) means that a high number of observations would be required to learn the network parameters. Since an event with a low prevalence will require a large number of test cases to be observed, learning all the conditional probabilities with relatively low prevalence requires a large data set.

9.7.3 Inference Performance and Relation to Data Size

The implications of the rare event nature of the predicted outcome along with the change in the classification performance of the BN in response to change in data size were tested via building ROC curves (Metz 1978).

ROC curves provide an informative, yet compact representation of the performance of the BN. A significant determinant of the classification performance of the BN that is used to predict the occurrence of a complication during a patient’s cataract surgery is the decision threshold value for the estimated probability of the complication. The decision threshold is used to classify the patient as high risk or not high risk. If the predicted probability is higher than the threshold, then the occurrence of the complication becomes the classifier output. Therefore, the performance of a classifier depends on the selected threshold and a good metric for performance is the true positive(Tp) and false positive (Fp) rates as discussed in Section 5.5. The ROC curve is a plot of the Tp and Fp values across different

decision threshold values. It enables an evaluation to be made of the decision-making method at hand and provides a visual clue for deciding the best threshold value for a particular decision-making strategy. The ROC curve in Figure 56 shows how the BN performed with a data set of ten thousand cataract operations, based on synthetic data.

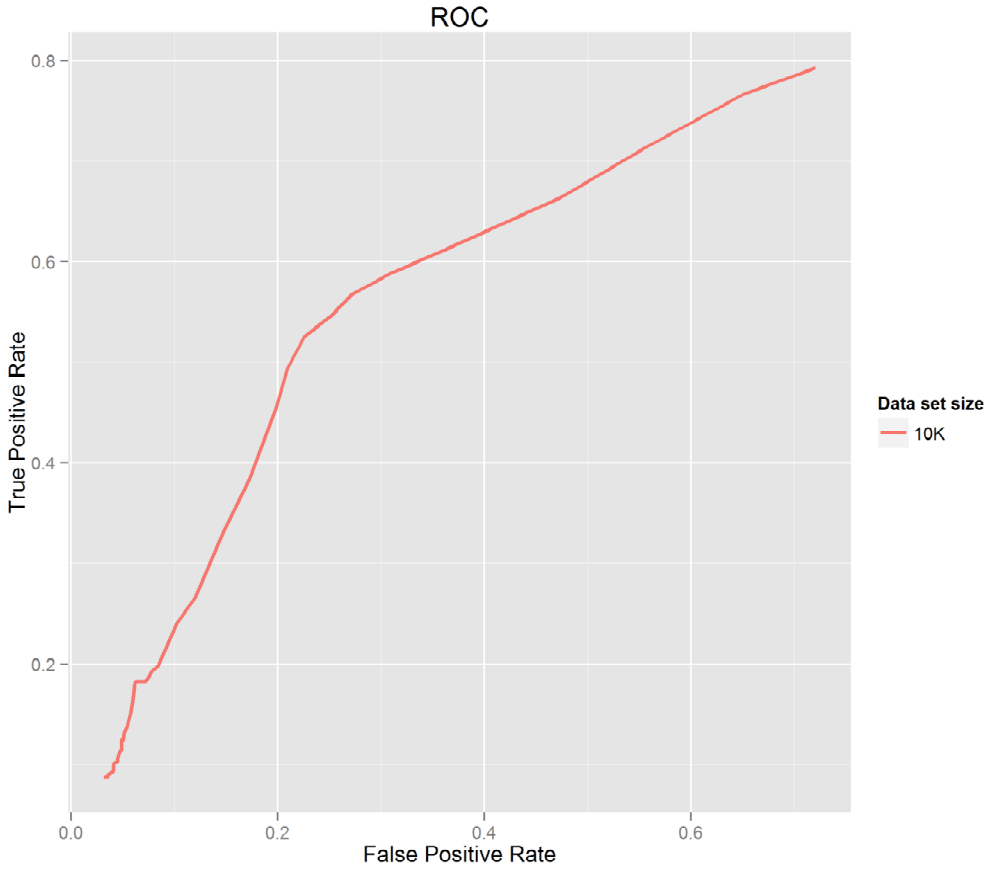


Figure 56: ROC curve for BN performance. 10K data instances

The starting value for the decision-making threshold is 0.001. The threshold was incremented by 0.005 until 0.496. Therefore, the ROC curve in Figure 56 is based on 100 different threshold values. Each threshold value was used to perform k-fold cross validation (Kohavi 1995). K was set as 10 for all steps. During k-fold cross validation, 90% of the available data was used to learn the parameters of the BN with the same given network structure, and the resulting BN was used to classify the remaining 10% of data. This learning-testing process was repeated for all folds, 10 times in total for any threshold value.

The mean values of sensitivity and specificity from each k-fold cross validation were used to arrive at Tp and Fp values. Therefore, the ROC curve in Figure 56, and the following figures that repeat the same process with more data,

are based on 100 applications of k-fold cross validation with a single threshold value for each application.

The classification behaviour of the BN in response to choosing a particular threshold value can also be observed from the plots of sensitivity and specificity. Figure 57 provides these plots for the same 10K data set:

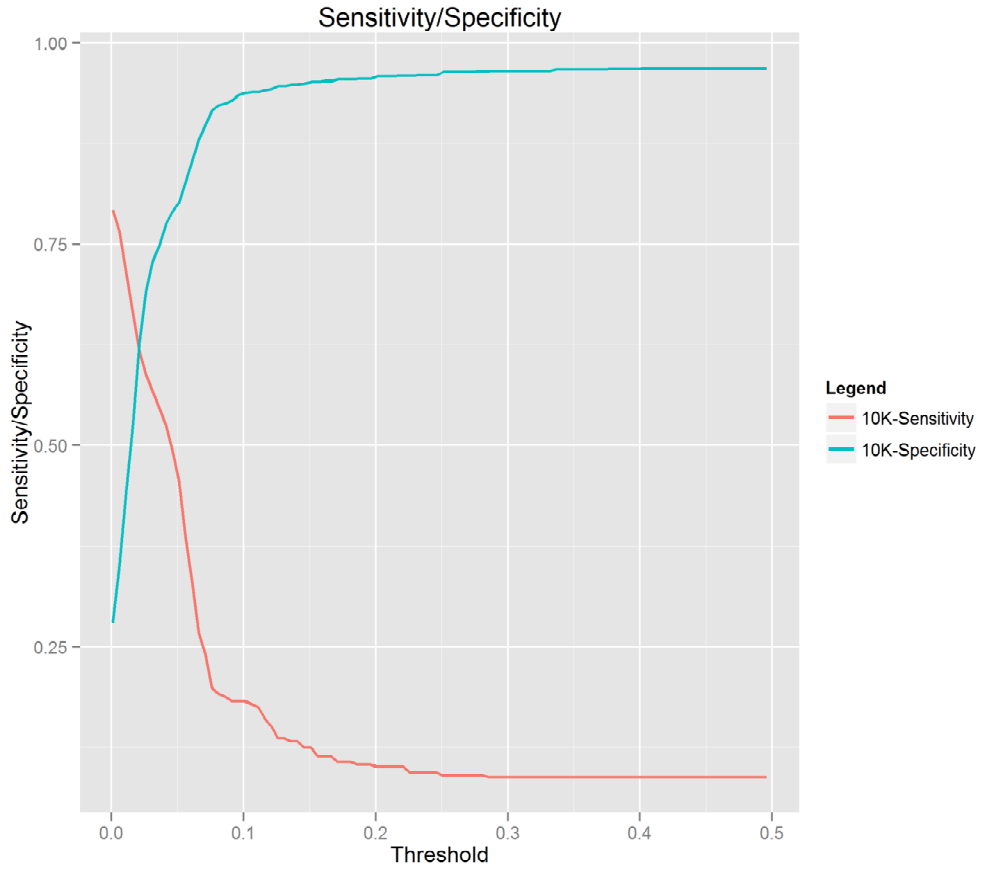


Figure 57: Sensitivity/Specificity for BN performance. 10K data instances

The sensitivity and specificity curves in Figure 57 do not shift significantly in response to changing the threshold beyond the value of 0.1. Threshold values above 0.1 result in classifier performance that departs from the desired scenario of high Tp and low Fp.

The ROC curve in Figure 56 shows that the classifier in the main stays above the diagonal (which would represent a completely random decision). The effect of the data set size on classifier performance can be observed by increasing the amount of data while keeping every other factor the same. Figure 58 compares the results of following the same procedure using 100K instances of synthetic data with the previously used 10K instance data set.

As Figure 58 shows, an increase in the data set size leads to better classifier performance, achieving better Tp rates given a Fp rate. Increasing the data set size to 200K and 500K preserved the same trend as shown in Figure 59.

The ROC curves discussed above were all generated using clinical data retrieved from the openEHR persistence implementation discussed in Chapter 8, achieving the thesis goal of an integration of openEHR methodology with a BN for CDS.

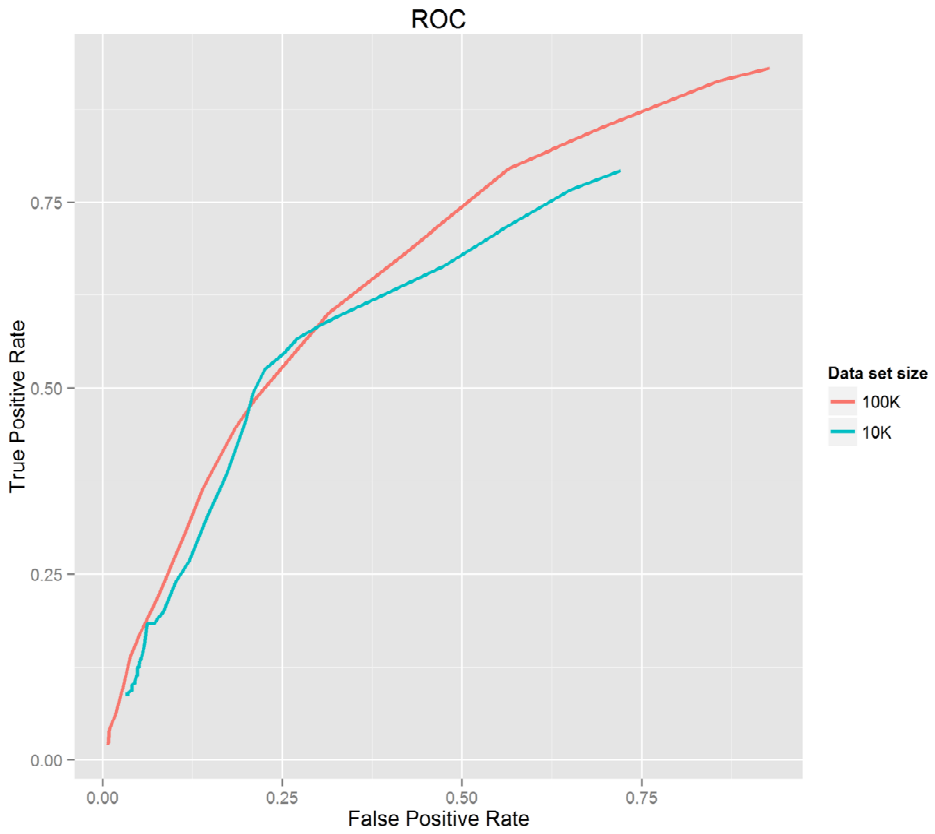


Figure 58: ROC curve for BN performance. 10K and 100K data instances

The performance of the BN classifier demonstrated by the ROC curves in Figure 59 is not close to the ideal performance a ROC curve could represent. The best performance a ROC curve can represent is high true positive rate accompanied by a low false positive rate, which means a ROC curve that comes close to upper left corner of the diagram in Figure 59. Even though the BN implementation did not achieve a remarkable performance for classification, the use of ROC curve to observe its behaviour introduced a useful instrument for observing the results of changes to components of the CDS setup, including the data volume.

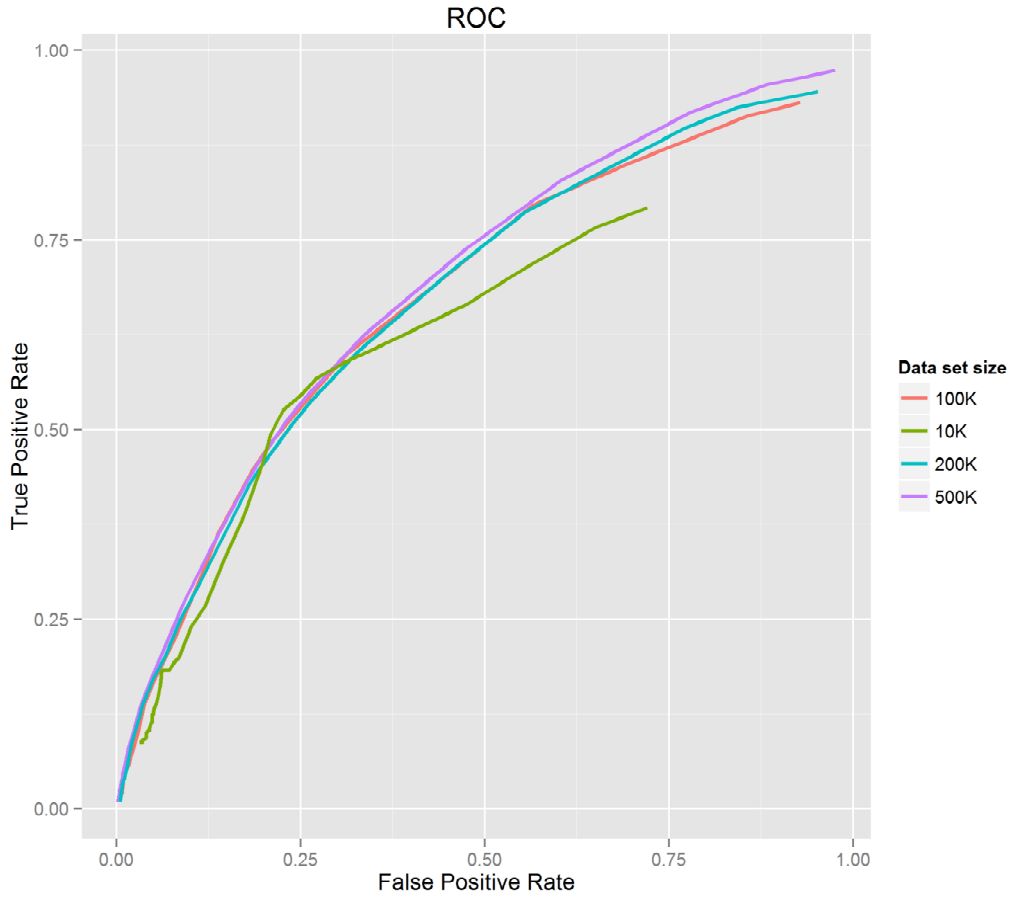


Figure 59: ROC curve for BN performance. 10K to 500K data instances

9.8: Discussion of the CDS Approach

9.8.1: High Level Architecture

The first step in developing a BN as a CDS mechanism is the identification of variables which are considered relevant to the clinical condition at hand. This step usually includes input from a domain expert. Even though the clinical variables used in the experiment were based on the covariates of the logistic regression model in (Narendran et al. 2008), a significant number of these variables were already included as data items in the openEHR clinical models, which were initially developed independently of this thesis. Therefore, these models provided an initial set of clinical variables that could have been used by a domain expert as a candidate set from which nodes of the BN could be selected. The usability of openEHR clinical models for the development of BNs for estimating the risk of a cataract operation suggest that these models have the potential to serve knowledge engineering requirements beyond clinical information systems development.

In the experiment, the clinical concepts defined in the openEHR archetypes were used to connect clinical data and BN development. A single clinical concept such as `glaucoma` (meaning that the patient has glaucoma) represents the clinical condition, a variable in a joint probability distribution (coded by the BN) and the actual value that resides in the persistence implementation and its access using AQL. Figure 60 depicts the relationship between these different uses of the same concept.

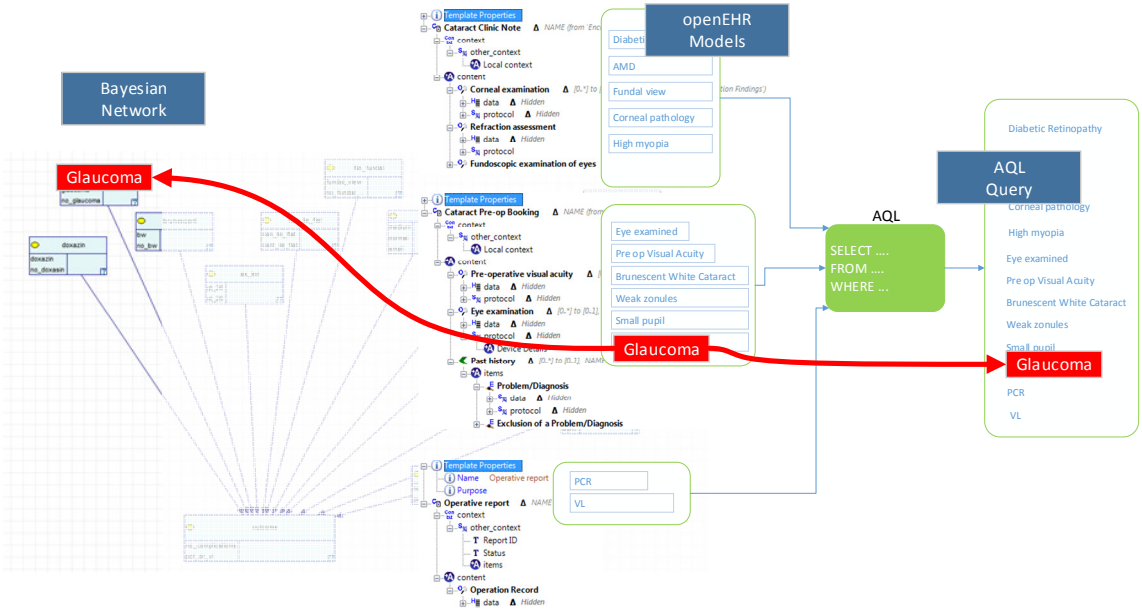


Figure 60: Using openEHR data item for CDS

Figure 60 shows how the glaucoma variable that originates from the clinical model was used in both the BN definition and in the AQL query. The relationship between the BN, the openEHR models and the AQL model is not depicted for all variables in the diagram, for purposes of clarity.

The AQL query enables data that is associated with the glaucoma concept from the openEHR model to be fetched from the underlying persistence implementation. The simplified relationship in Figure 60 is the basis of a scalable approach that can be implemented on various platforms. The BN implementation and openEHR persistence layer that supports AQL can be based on any technology. The templates in Figure 60 can also support clinical information systems development even though this is not included in the figure.

9.8.2: Implementation Details

The software implementation of the integration of the high level concepts in Figure 60 has not been completely seamless. Integration of these concepts requires integration of various software tools and frameworks at the implementation level. Key software and artefacts that were used to provide this integration are as follows:

- The clinical models consisted of openEHR archetypes and templates that were created with the freely available Archetype Editor and Template Designer software from Ocean Informatics. These models were transformed to their XML representation (XSD) via the export mechanisms of the modelling tools. Synthetic data was generated in the form of comma separated value files, and these files were used to create openEHR data in XML format, compatible with the XML schemas, which were based on the clinical models.
- The clinical data in XML form was stored into an openEHR persistence implementation, which supports Archetype Query Language through a transformation from AQL to SQL. Therefore, the actual means of getting access to clinical data is to use the SQL implementation of the associated relational database.
- The BN that was used for decision-making was defined and deployed using the R package Bnlearn. (M. Scutari 2010)

This integration shows that data creation, openEHR persistence and BN implementation required the use of a number of programming languages and technologies. Python (Van Rossum 2007), Java (Arnold et al. 1996), Scala (Odersky et al. 2004), Eclipse Modelling Framework (Steinberg et al. 2008), Postgresql relational database (Momjian 2001) with its SQL (Date and Darwen 1987) implementation and extensions, R (R Development Core Team 2008) are the tools used to build this openEHR-BN integration. The use of these tools, despite their substantial learning curves and complexity, was necessary because no single technology provided all the functionality required to implement the integration in Figure 60. The complexity of individual components of the implementation led to the concurrent use of existing tools and frameworks, even though they were built on different software technologies. The alternative of implementing all the functionality on a single platform would have been impossibly inefficient, due to the work that would have had to be done from scratch, and would mean dismissing findings and results of a vast amount of published research and development.

Despite the large number of technologies required, in practice, the maturity and large user base of these technologies provided rather smooth and well-established means of connectivity between them. The SQL based implementation of AQL exposes the results of an AQL query as a regular SQL query over standard database access mechanisms, which can be accessed from R through an R package such as rpg (Keitt 2015). All major programming languages have libraries and frameworks for processing CSV and XML files and XML schemas. Therefore, even though learning curves of a number of technologies had to be tackled to implement the main components of an openEHR based CDS implementation, using well-established technologies enabled an efficient integration.

9.8.3: Findings Related to Implementation

The low frequency of complications during a cataract surgery makes data set size a critical component of parameter learning for the BN. A 0.2% rate of complications as found by (Jaycock et al. 2007) means that a clinical data set with 50,000 cataract operations would be expected to contain about a 100 events with the outcome we would like to detect.

The maximum likelihood estimation used by the bnlearn R package requires observations for a particular combination of values of BN nodes to assign probabilities to that combination. When the structure of the BN leads to a conditional probability table for the clinical complications node that has 92160 entries, obtaining probabilities based on observations requires large amounts of data. Therefore, the number of possible observations defined by the structure of the model and the rare event nature of the complications both elevate the amount of data required.

The ROC curves in Figure 59 demonstrate this point. An acceptable and consistent performance from the BN requires tens of thousands of data instances and increasing data set size helps improve the classification performance.

This data size requirement of the chosen CDS method leads to the requirement for accumulating data for a large number of operations. Use of openEHR for both clinical systems implementation and data interchange between systems will help fulfil this requirement. Cataract operation data from various systems and locations can be pooled with little effort. However, the use of openEHR data in a machine learning context requires persistence implementation for such a pool of data to perform sufficiently well to feed data to machine learning frameworks. There is thus a requirement for an openEHR persistence

implementation that can perform large volume queries with a high level of performance.

From a knowledge engineering perspective, the use of the openEHR models helps define the BN structure by providing a set of clinical variables for easily identifying and defining the nodes of the network. The data access mechanism based on AQL also benefits from the openEHR approach since it can support data access based on the same clinical variables, in a platform independent way.

However, the parameter learning performance of the BN, especially for approximate inference methods, is dependent on the data volume. Satisfying the requirements related to the performance of the openEHR persistence implementation is not sufficient to ensure performant parameter learning. The BN implementation such as the bnlearn package used in this experiment has its own scalability requirements. Without support for parallel structure and parameter learning implementations for BNs, the computing power of a single CPU becomes the performance bottleneck for these operations. Therefore, scalability of the persistence layer for openEHR based CDS does not imply scalability of the integrated architecture.

The iterative nature of the BN development is likely to require many CPU intensive tasks to be performed repeatedly. A CDS implementer may consider changing the intervals for discretization of the continuous variables such as age to achieve better performance by following different discretization approaches (Dougherty, Kohavi, and Sahami 1995), (Irani 1993). In this case, the ROC curves must be rebuilt to observe the results of these changes.

The process of building the multiple ROC curves in Figure 59 is another example of the iterative nature of BN development, since the data set size changes and consequently the whole computation of the ROC curve is performed from scratch. Building the ROC curves require inference task to be performed by the BN, which is dependent on parameter learning. Parameter learning is performed via bnlearn package, and the inference was performed using the gRain R package (Højsgaard 2014) which employs the Junction Tree inference algorithm (Nagarajan, Scutari, and Lèbre 2013). Both the bnlearn and gRain packages use a single CPU core for computation and as more data is used, mostly to deal with the low prevalence of the clinical outcome of interest, the time to learn the parameters of the network and perform inference on a data set to measure classifier performance grows significantly. The bnlearn package is not limited to implementation of sequential algorithms. It supports parallel structure learning (Marco Scutari 2014) but in this particular experiment the structure of the network is based on the logistic

regression of (Narendran et al. 2008), and therefore performance benefits from parallel computation were not realized within the workflow that produced the ROC curves in Figure 59.

The implementation limitations mentioned here are case-specific and should not be seen as a limitation of BNs in general. Parallel algorithms for structure and parameter learning, as well as approximate and exact inference, are active fields of research with potentially useful outcomes for dealing with large data volumes such as (X.-L. Wu et al. 2012), (Neiswanger, Wang, and Xing 2013), (Xia and Prasanna 2008), (Xia and Prasanna 2007) and (V.K. Namasivayam, Pathak, and Prasanna 2006).

The last significant finding from the experiment relates to data transformations. It was observed that data transformations were required on the AQL query results for continuous values of some AQL variables to be used by the BN implementation as values of discrete variables. This transformation is required due to the discrete nature of conditional probability tables used to represent the nodes of the BN. One such variable is the age of the patients going through the cataract surgery. It was observed that transformations that are required to import legacy data to openEHR persistence and later provide it to the BN implementation, create a data transformation pipeline that is susceptible to information loss.

The term information loss refers to conditions in which various characteristics of data become unavailable due to a transformation, such as discretisation of continuous values. Once a set of continuous values that fall into the same category are grouped together and assigned the same category identifier, the original values can no longer be recovered in further steps of the transformation pipeline. For example, if systolic blood pressure of a patient is imported from a legacy system based on an openEHR model, which defines a data item for systolic blood pressure that only has values low or high, further access to openEHR data cannot introduce three categories such as low, normal and high. The members of both low and high groups are indistinguishable from each other, and without access to original values, it is impossible to know which data instances would be classified as normal for the new step in the computation pipeline.

9.9: Comparison of the Thyroid and Ophthalmology Experiments

The experiment in Chapter 5, based on thyroid data, provides an example of a simplified machine learning implementation, which can be compared with the

implementation discussed in this chapter, to observe the requirements for building a more realistic BN setup built on openEHR methodology. The pilot implementation in Chapter 5 was intentionally kept simple to explore requirements and implementation characteristics of a BN based CDS without an EHR platform.

The use of a BN with its discrete conditional probability tables introduces the same issues in both experiments: the lack of observations that correspond to one or more combinations of variables of the network can lead to biased models. A rather problematic case arises when lack of certain combinations of observations in the analysed population dataset leads to the learned parameters of the network computing the probability of that set of observations as zero, implying that they are impossible.

Even when synthetic data generation is used to produce large amounts of data, some combinations of values may not be observed, due to both network parameterization and the nature of the events. There are well-established methods for dealing with missing data, so remedies exist for this issue, but the discrete nature of the BN is likely to require their frequent use for BN based CDS.

The experiment in Chapter 5 follows what is a quite common approach to building a data set for machine learning: clinical data is transformed into a comma separated value file for direct consumption by any tool that can consume CSV files. This approach requires that data from different clinical information systems has consistent semantics, which must be checked and ensured by rigorous data analysis and cleaning. Adding new data to the existing data set is likely to require new mappings in addition to the effort required to implement data export functionality from the source systems.

The openEHR based approach followed by the experimental CDS implementation develops a model driven representation of data, which supports a number of use cases. The openEHR models, which are central to clinical data representation, can also support clinical systems development, CDS design and development, and data interoperability.

Despite the significantly more complicated infrastructure that is required to support the openEHR based approach, the platform provided by this infrastructure eliminates the need for repeated, error prone data cleaning and mapping tasks that are required if a data export approach is adopted.

9.10 Summary

The CDS implementation discussed in this chapter attempted to identify key aspects of developing a CDS system based on the integration of openEHR and BNs. The amount of effort that was required to build the data processing infrastructure has been significantly greater than the effort that was required to use BNs for CDS functionality.

This finding confirms the well-known problem of most CDS development efforts: most of the time and available resources is spent on the data infrastructure or data cleaning. However, the openEHR based architecture and approach delivers an output that can be reused and extended. Even though it was implemented at a proof of concept level, the SQL based AQL support produced a promising way of eliminating the well-known practice of data extraction from the clinical systems to build a separate data set for CDS development. openEHR clinical models that have been initially designed for clinical care scenarios provided sufficient support for defining data items for the CDS models (BNs), albeit with various workarounds. These workarounds, such as adding demographic data (age of patients) into the clinical models, are valuable observations that are used as the basis of suggested improvements to the openEHR specifications to better support CDS integration scenarios, as discussed in Chapter 10.

Overall, the pilot implementation discussed in this chapter demonstrates the feasibility of the integrated architecture defined in Chapter 4.

Chapter 10: Conclusions and Future Research

The primary objective of this thesis is to place openEHR into the heart of a clinical decision support setting and to observe the outcomes of this approach on all the components of the resulting architecture, both at the specification (abstract) and implementation (concrete) levels.

Establishing this objective with an experimental approach that includes as many aspects of a realistic openEHR implementation as possible requires the use of a number of technologies. The results of the experiments based on the development of such an implementation show that software standards, frameworks and tools reveal their strengths and weaknesses according to the use cases at hand, and the complex interactions between them depend on the functionality supported.

This functionality can be classified into two groups: related to clinical information system and CDS.

The openEHR specifications define functionality required to support clinical care: the existence of the EHR as a core concept, the fundamental units of clinical data such as COMPOSITION instances, and other design characteristics of openEHR imply a set of operations on clinical data for clinical care.

The functionality related to CDS is not currently explicitly identified in the openEHR specifications. This is a perfectly natural outcome of openEHR's primary goal: delivering a computable representation of healthcare data that focuses on the concept of electronic health record, which in turn implies a patient whose clinical data is kept in the EHR. From a data processing point of view, this is a patient-centric design, which does not include patient populations as a first class concept. On the other hand, implementations of the CDS concept has a strong dependence on the concept of patient population: a patient's diagnosis or prognosis can be evaluated based on the degree of deviation from the characteristics of the relevant patient population.

This dependency on the population characteristics introduces different patterns of data access to clinical data than the patient-centric ones. This thesis has explored the feasibility of an architecture that can support both sets of patterns, based on a set of implementation driven experiments. From an openEHR point of view, the most significant research challenge this thesis has tackled is that of introducing openEHR methodology as the basis of both clinical care and CDS system implementation, without resorting to completely different architectures.

openEHR's two-level modelling approach allows clinical concepts to be reused in different components in this unified architecture. Therefore, concepts defined by the openEHR RM provide a significant level of robustness by supporting various CDS specific tasks, which were not necessarily included in the initial design of openEHR as functional requirements. However, this robustness has its limits.

The design and implementation of openEHR persistence emerges as an immensely important research topic, central to both overcoming openEHR methodology's current limits to robustness and enabling innovation by supporting new capabilities for AQL. Extensions to AQL, such as the Probabilistic AQL idea discussed in Chapter 8, have the potential to integrate results of cutting edge machine learning research with a clinical query language in novel ways. Further research into openEHR persistence, based on persistence abstraction and big data frameworks can support this integration, as data volume grows at a rapid rate.

Therefore, the findings based on the work done for this thesis, which are discussed in the following sections, are considered as starting points for future research based on the four key components: openEHR specifications, parallel, large scale data processing, AQL and machine learning.

10.1: openEHR Models for Computable Healthcare Data

The development of archetypes for the ophthalmology domain, which are used in the CDS implementation, have been initiated independent of this thesis, and their scope includes the data items that would be required to implement a clinical information system. These openEHR models have been developed by a highly experienced clinical modeller, with input from a senior clinician from the ophthalmology domain.

The implementation based on these models reveals some important findings related to their use in a CDS context. First of all, there are multiple ways of expressing the same clinical content in a model and modellers can not necessarily predict the outcomes of their modelling choices in downstream contexts. An example of this case is designing a clinical model that allows multiple clinical findings to be included at the same point in the model. The underlying assumption for this approach is that a clinician may add any number of data items as he or she sees appropriate during the care process. The modeller cannot easily constrain the list of clinical findings that can be added; doing so may lead to clinicians not being able to record an observation if it was not considered by the modeller.

When this clinical model is used for implementing a clinical information system, the modelling approach may not lead to any problems. The clinicians looking at the list of findings can easily interpret the information. When the same model is used in a CDS setting, existence or lack of a particular observation may become a key determinant of both CDS model learning and decision-making . The existence of any number of data items under the same container in the clinical model may also require attention. If there are multiple data items at a single location in the openEHR model, then the semantics of corresponding data items must be differentiated by some means other than their path. If this scenario is not considered in advance by the clinical modeller, openEHR models that present no problems when used by a clinical information system may end up causing ambiguity in a CDS driven use case.

Another problem which would not necessarily reveal itself in a clinical care setting is an interpretation of the lack of a condition. A human interpretation of a list of problems for a patient is manageable for a clinician: he or she can reason about the lack of a particular condition or simply ignore it. When a CDS implementation uses the existence of a variable as a significant variable for calculating an outcome, it cannot mimic the human reasoning of the clinician that is performed at the time of care.

These potential issues are not related to openEHR's capability for expressing clinical data. They are results of the specific focus the clinical modeller has on a limited set of use cases during the model development phase. However, when a particular use case is known, its consideration in the modelling process may not be without any trade-offs. For example, when a modelling approach that explicitly records lack of a condition is adopted so that this setting can be clearly identified in a clinical decision support scenario, use of this model in a clinical care setting requires the clinicians to record this information. From a clinical information system end user perspective, this is an extra step that would take valuable time, which would not be required if the clinical model allowed simply not recording a condition instead of explicitly recording lack of it.

Therefore, a critical finding of the thesis based on the CDS implementation process is that flexibility and capabilities of openEHR modelling formalism do not provide models that can support different uses of clinical data without any effort. Claims of better CDS based on the expressive power of openEHR should contemplate this finding for a more insightful approach.

The scope of clinical data in openEHR models that are developed with a focus on clinical care is another significant issue. The CDS model used in Chapter 9

include variables such as age and surgeon experience. Age of a patient is most likely to be considered as part of demographics data and despite the existence of a demographics information model under openEHR specifications, this data may be provided by a shared service such as a master patient index or other specialised software. The experience of the surgeon is unlikely to be considered as part of a clinical model so this information may also reside in an external system that manages administrative data.

Both age and surgeon level variables used in the CDS model are therefore unlikely to be part of clinical models for clinical care, and their actual values are likely to reside in a system outside of the openEHR implementation. This situation, which can be generalized as dependencies on non-clinical variables in CDS models, is problematic in an openEHR based CDS setting at multiple levels.

Modifying clinical models to include data for potentially non-clinical concepts, as done in this thesis as a workaround, is a misuse of openEHR's capabilities at the modelling level. Not including these variables in the models means that AQL can no longer be used to define all the data that the CDS implementation would require. It can be argued that the potentially non-clinical nature of CDS model variables is simply a matter of scope; that non-clinical data is not relevant to openEHR. However, if associating openEHR models and data based on these models to non-clinical concepts and data is considered as a frequently encountered requirement, at least in a CDS setting, then this requirement deserves attention as a research topic.

10.2: Using AQL for Clinical Data Access

AQL allows access to clinical data via use of the concepts defined by openEHR specifications, independent of the underlying persistence system that openEHR persistence is implemented on. Given the recent advances in large scale, parallel data processing frameworks such as Hadoop (Borthakur 2007) or Apache Spark (Zaharia et al. 2010), AQL becomes the strongest candidate for means of data scale independent clinical data access .

The persistence abstraction approach developed in Chapter 7 strengthens the argument for adopting AQL for both clinical information system and CDS system development by providing a consistent approach to persistence implementation.

However, the maturity of AQL as a specification is not on par with its suggested advantages at the time of the writing of this thesis. As it stands, AQL is not documented along with the rest of the openEHR specifications. The current

documentation available to implementers is a web page (Ma, Frankel, and Beale 2014) Moreover, the contents of this webpage focuses on the syntax and grammar of AQL, leaving some behaviour undefined.

The implementation of the tree based persistence abstraction adopts intuitive interpretations of this type of undefined behaviour when necessary. One such behaviour is the treatment of missing values for data items listed in the SELECT clause of AQL.

The need to provide an interpretation for this behaviour was observed in the CDS experiment discussed in Chapter 9. The diagnosis of diabetic retinopathy for a cataract patient is a variable, the value of which is required in both parameter learning for BN and inference tasks. When the clinical model does not include an explicit data item that represents a lack of this diagnosis, all patients without this condition would be missing the diagnosis element. If the AQL implementation were to leave out all query results that do not have this diagnosis, this would lead to large number of patients being excluded, and only a subset of the patients with the diagnosis would be included.

Therefore, the intuitive approach, which is followed by the implementation discussed in Chapter 8, is to allow empty values in query results and leave interpretation of them to later phases of data processing but this is still unspecified behaviour from the standards based data access point of view. The need to select COMPOSITION instances with the same care episode id, discussed in Section 9.6.3, also requires clarification of AQL specifications, regarding the possibility of defining conditions based on the equivalence of values of data items, without expressing the actual values.

Another finding of this thesis is the importance of AQL-first design for the performance and flexibility of persistence implementation, even though AQL is defined independent of any implementation methods or technologies.

openEHR data can be persisted in many ways and initial experiments that use a relational database as a persistence layer have delivered satisfactory results for clinical care use cases. These use cases, such as accessing a list of compositions for a patient can be implemented with custom application programming interfaces (APIs) without significant difficulty, especially due to the availability of high level software development frameworks. However, these custom APIs provide a less than an optimum solution for a platform approach based on openEHR: data exchange is possible, but moving individual applications such as CDS implementations across openEHR implementations becomes complicated. AQL solves this problem, but it comes at a price; its data access semantics that

heavily rely on constraints on the hierarchy of data elements must be supported by the persistence layer.

The Opereffa implementation described in Chapter 6 showed that building custom data access APIs to support clinical applications first is likely to produce a persistence architecture that cannot easily support AQL semantics. Since development of statistical models for CDS significantly benefits from large data sets, the most common solution to this design problem is a data export mechanism to another persistence system, which can support large volume queries with better performance. This is indeed a widely adopted industry and research practice leading to a data warehouse approach.

Even though such a solution would be possible for Opereffa, it would still fail to benefit from the advantage of a unified data access method based on openEHR concepts, as provided by AQL. The persistence abstraction method developed in this thesis provides the means to implement this unified data access method across a variety of persistence systems. However, some transformations on data are inevitable, when AQL is used to integrate openEHR to BNs for CDS. The reason for this was discovered to be the nature of the AQL result set and openEHR data types.

The AQL result set could contain empty values, which are suitable for representation via use of the relevant types of the implementation technology: such as empty values in SQL query results or null values in an in-memory Java object. The semantics of lack of a value must be expressed as a specific numeric value for a machine learning framework, such as 0, where other numeric values would have other meanings.

The AQL result can also return values based on the reference model of openEHR. For example, if existence or lack of a diagnosis was expressed in the model with codes from a terminology (either specific to that model or an external one), the results would contain one of the two terminology codes which would again require a transformation to either numeric values used by the CDS related frameworks.

These transformations are usually required, aside from the rare cases where an actual numeric value is to be fetched from openEHR data and used directly in a CDS implementation. This is because of the difference between the highly specialised type system introduced by the RM and the much simpler and fully numeric nature of data that the machine learning algorithms require.

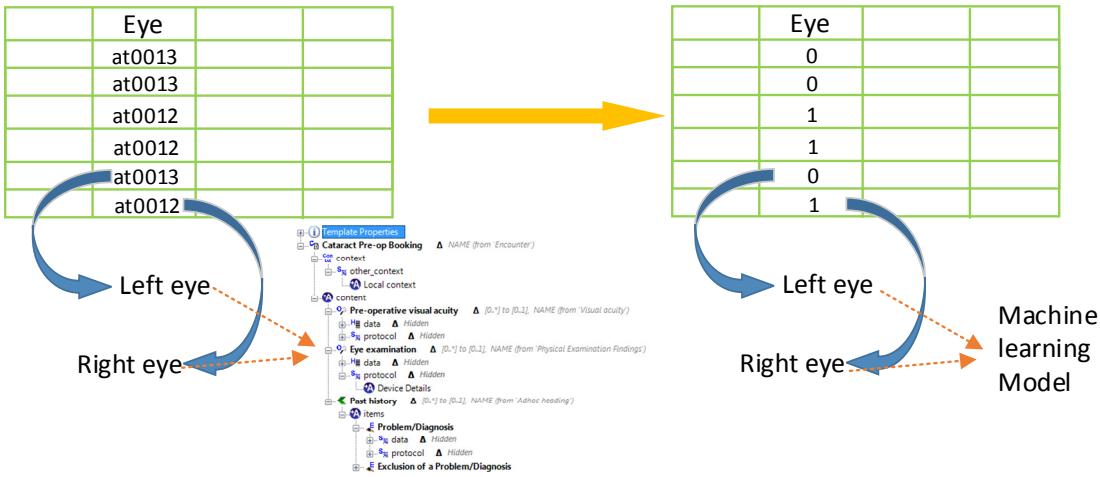


Figure 61: openEHR model data vs. machine learning model data

At a high level, this is a transformation from a matrix in which values belong to openEHR's type system, to another one where a much simpler, numeric type system is used. Figure 61 illustrates this transformation.

Figure 61 shows how openEHR models define the semantics of actual values and how these semantics end up in openEHR data retrieved by the AQL query ("at…." codes). Even though the semantics assigned to this variable stays the same in the machine learning model, the transformation to the numeric value used in the machine learning model is inevitable.

Therefore, providing an abstraction over persistence systems via the use of AQL does not guarantee clinical data can be used directly in machine learning contexts. There exists an extra computational step, which must be performed for some values in the AQL query results, consequently having significance from a performance point of view.

10.3: Using Bayesian Networks for Clinical Decision Support

The term "Bayesian Network" has been used in this thesis to refer consistently to a particular type of probabilistic graphical model, which is based on a directed acyclic graph, nodes of which consist of conditional probability tables. Other kinds of probabilistic graphical models (Koller and Friedman 2009), which are referred to as continuous Bayesian Networks, hybrid Bayesian Networks, etc. have been described, as extensions of the term Bayesian Network. The classification of these graphical models as extensions of the BN as defined by (Pearl 1988) is specific to this thesis, based on their increased expressiveness in terms of semantics of nodes as well as supported topologies.

The primary advantage of a BN as a CDS tool is its high level conceptual representation. Even though established statistical methods are available for regression or classification tasks, graphical models offer a unique way for domain experts to contribute to the construction of a probabilistic model. This contribution can become even more efficient when the openEHR clinical models are used as the underlying knowledge repository, which allows data items from the clinical models to be used to define the nodes of BNs, as discussed in Chapter 9.

CDS implementations have an iterative nature, with performance improvements achieved experimentally, regardless of the underlying mechanism for decision-making.

Analyses, such as calculation of ROC curves, allow observation of the effects of changes to components of the CDS implementation, on its performance, such as the assumptions of the decision-making model, data set size or threshold values. The iterative process is hampered if increasing data volume introduces a performance bottleneck. Such growth in data volume can stem from increased adoption of clinical information systems, or the actual CDS scenario at hand, such as analysis of rare events that require a large number of observations for the CDS to be characterised. Therefore, the robustness of the BN approach to CDS, in the face of growing volume and complexity of clinical data, is an important determinant of its usability in addition to the use of openEHR methodology - which in itself enables data sharing and consequently data pooling by design. The increasing adoption of parallel programming methods and their inclusion in popular programming language runtimes and frameworks, provides a potentially reliable solution to this problem.

Implementation of parallel learning and inference algorithms for BNs must be complemented by generic parallel computing frameworks so that key tasks in the model development lifecycle, such as the k-fold cross validation (Kohavi 1995) used in Chapter 9, can be performed on large data sets. The scope of future research on this topic should also include extensions of BNs such as continuous and hybrid networks.

10.4: Future Directions for openEHR Based CDS

Both clinical application development and CDS implementation based on openEHR have been explored in detail in this thesis. Actual software implementations with mainstream technologies have demonstrated experimentally that openEHR provides a robust and implementable platform definition.

A very significant amount of time was required for this essential software development and for research on tools and technologies that could be used to implement various functionalities. This exhaustive approach has been justified by the findings from the pilot experiments and the implementations described in the thesis.

Without implementation driven experiments, research on electronic health records is bound to ignore some critical requirements for better CDS since these requirements are related to complex interactions between different aspects of the components of the chosen CDS approach. Observing these interactions requires the implementations of the CDS components in place. One understandable obstacle that makes it hard for EHR research to explore these requirements is a lack of freely available platform implementations based on standardisation frameworks like openEHR. The feedback that was provided in response to public and open source release of some of the components developed for this thesis is evidence of significant interest, from both industry and academia, in a platform that could support future research and development.

The potential improvements to the openEHR specifications suggested in this thesis are based on requirements that were identified through software implementations. These were not, though, solely implementation tasks; they are components of an integrated architecture for openEHR and BN integration, and are essential research and development contributions in the ongoing mission of openEHR.

Extending the scope of openEHR models and data with concepts external to clinical models is one such requirement. Modifying clinical models to include data items that are not directly related to the clinical concepts represented by the models is not an acceptable method for extending openEHR's benefits to CDS. This approach carries the risk of introducing data elements that could confuse clinicians concerned with non-CDS uses of the models. As commonly shared openEHR archetypes such as medications, allergies or blood pressure, are associated with more CDS scenarios, extra data items would clutter openEHR models with concerns not relevant to clinicians.

Support for metadata at the openEHR RM level could help express data items for these separate concerns, in a flexible way. In the case of including the age of a patient in an existing archetype, as discussed in Section 9.6.3.1, the problems introduced by adding this extra data item at the clinical model level can be avoided by expressing this variable as metadata. This approach would handle the data as an optional value associated with an instance of an RM type. Despite the flexibility this

offers, the integration of metadata within the RM, and its use in various new scenarios, require careful evaluation.

The openEHR object-oriented reference model presents an opportunity for metadata related properties to be defined at the level of abstract types and therefore to become available to RM types that inherit from them. Both the representation of metadata and its integration to the object oriented design of RM are significant future research topics.

Figure 62 visualises how successful outcomes of research on these topics might hypothetically support separation of concerns for multiple CDS implementations that use the same clinical model.

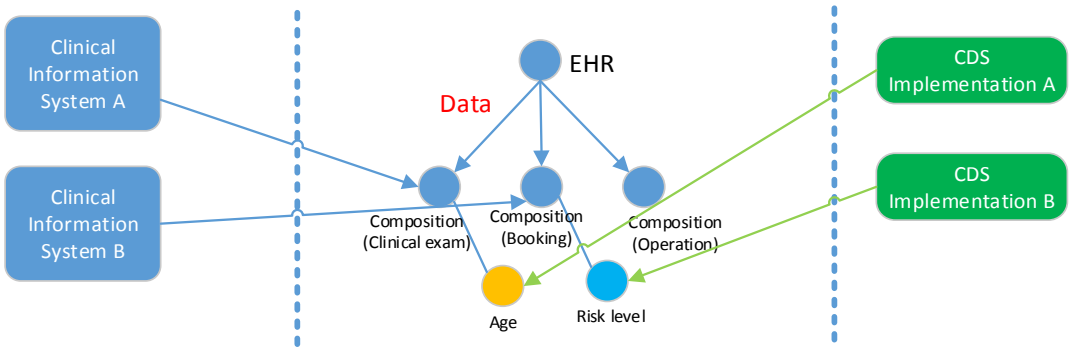


Figure 62: openEHR metadata for different CDS implementations

The clinical information systems in Figure 62 operate on data items that are defined through openEHR archetypes and templates. Since metadata is optional content for a reference model type instance, its definition can be completely omitted by the clinical modellers. When other uses for clinical data arise such as various CDS systems that use these models, these systems can use existing data created by clinical information systems without any extensions to the models, via use of additional metadata inserted alongside clinical data.

Operations that create metadata as part of the CDS life cycle, such as assigning the value of age of patients to a metadata path based on the underlying clinical model, saving the outcome of a risk assessment for a patient before the operation, or providing an estimate for prognosis can all take place without affecting the operation of existing clinical systems.

Metadata support for openEHR would also require the openEHR specifications to clearly define how metadata should be managed in various

scenarios that make use of RM data. These scenarios include, but are not limited to, clinical data versioning, clinical data exchange and AQL based data access.

One of the key features of the openEHR methodology is its recognition of the importance of tracking changes to clinical data due to both clinical and legal requirements. The openEHR specifications fulfil this requirement through strong support for versioning. Use cases such as correcting a valid but incorrect data entry, or adding a new allergy to an existing list of allergies, can introduce new versions of openEHR RM instances. When an existing instance has metadata attached to it, how this data should be treated in case of a version increment must be defined by the specification, considering various use cases.

Clinical data exchange scenarios also need to clarify how metadata is to be treated. Moving metadata across information system boundaries along with actual RM instances can help receiving systems use CDS implementations that rely on particular metadata. This scenario requires that the metadata itself is clearly defined so that CDS implementations can consistently use it. The requirement can potentially be fulfilled through the use of openEHR's data types. The privacy implications of sharing metadata would also require consideration: since the use of metadata is suggested for data that is not necessarily clinical in nature, sharing this data across system boundaries may introduce further problems, such as age, gender or geographic location of patient unintentionally moving to other systems, compromising anonymity; after all, it is likely that the clinical modellers did not include it in the scope of openEHR archetypes in the first place.

Introducing metadata in the above suggested manner should take care not to introduce new and custom methods for access to this data – that would detract seriously from the benefits of using AQL, as widely discussed in the thesis. Therefore, extending AQL's syntax and semantics to accommodate metadata support in this way, is another important future task.

Each of these suggested extensions is likely to require significant efforts, with input from clinicians, clinical system implementers and CDS implementers. Therefore, they are suggested future research topics for openEHR.

Another future line of research, the scope of which arises from observation of the behaviour and operations required on RM data in a CDS integration scenario, as well as performance requirements for processing RM data, is the extension of capabilities of AQL.

The experimental setup discussed in Chapter 9 included various processing steps which are likely to emerge in CDS implementations based on both BNs and other machine learning methods. Transformation of openEHR RM types to numeric

values and discretisation of continuous values, are data transformation tasks which are likely to be performed in almost every CDS implementation that uses AQL. Supporting some of these tasks at the AQL level could allow AQL to support CDS implementation better by shifting frequently required capability from machine learning frameworks to AQL implementation.

Given AQL's syntax and semantics, which resemble both XPath (Clark and DeRose 1999) and SQL (Date and Darwen 1987), various extension mechanisms used in these languages might usefully be adopted by AQL to support data transformation tasks and other requirements to process data.

A potential approach to achieving this goal would be support for function calls. This approach is part of the standard for both SQL (ISO 2015) and XPath (Clark and DeRose 1999). Various relational database servers and XPath processors support user-defined function definitions in SQL and XPath queries. The advantage of this approach is that it keeps the core language simple. User-defined functions not only provide support for extending the capabilities of these query languages, but they also allow access to more expressive, general purpose programming languages for customising behaviour.

This is a flexible and powerful approach to developing functionality which may be inefficient or simply impossible to deliver directly with SQL or XPath. For example, Postgresql (Momjian 2001) supports user-defined functions developed in languages such as Java (Arnold et al. 1996) or Python (Van Rossum 2007). Similarly, XPath processors allow calls to functions implemented with host languages such as C# (Hejlsberg, Wiltamuth, and Golde 2003) and Java (Arnold et al. 1996). Successful use of this approach across mainstream relational database servers and programming languages is evidence of its versatility.

The extension of AQL to support user-defined functions should follow the same careful approach discussed above for metadata extensions to the openEHR specifications. While custom functions could allow implementers of AQL to provide advanced data processing capabilities, they could also introduce dependencies on the availability of particular functions for CDS implementation. As with the suggested metadata extensions, this could potentially diminish the re-usability of CDS implementations that utilise specific user-defined functions within AQL.

Both SQL and XPath attempt to solve this problem through the introduction of standard functions. These core functions are gradually introduced to new releases of the standard, so that designers of new systems can choose to rely only on functionality that they know any standard compliant platform would provide. Following a similar approach, based on input from implementers as part of the

openEHR specifications development process, is suggested to improve AQL's support for CDS.

Given that both metadata and function call extensions for AQL are suggested as part of openEHR specifications development, an underlying assumption is that the AQL specification is due to become part of openEHR specifications, which is not yet the case at the time of the writing of this thesis.

Defining AQL as the sole query language for RM based data access for the purposes of CDS, means that its implementation becomes a key determinant of the efficacy of openEHR and CDS integration, as discussed in the context of BN integration in this thesis. However, the benefits of a consistent data access method are likely to be cancelled out by the implementation efforts required of persistence systems implementers, to address the requirements of new CDS implementations such as to introduce large scale, parallel data processing.

Therefore, AQL implementation should be based on an approach that is technology agnostic and formally consistent, but not necessarily included as a part of the openEHR specifications.

The reasoning behind this suggestion is as follows. First, openEHR's technology agnostic approach to developing specifications should not be compromised by references to particular persistence systems, so any approaches related to AQL implementation must have the same technology agnostic nature.

Leaving persistence aspects completely out of the specification, which is the case at the time of the writing of this thesis, causes two problems: first, implementers find it hard to deal with AQL semantics, especially when using relational databases as the basis of implementation. Second, each platform for implementation requires design from scratch, making it costly for implementers to employ different platforms tuned to different use cases. A technology agnostic persistence methodology would offer a solution to these problems, without compromising openEHR's platform independent nature.

The new persistence abstraction approach developed in thesis fulfils these criteria, but is suggested as a methodology, not as a future addition to the openEHR specifications. Despite the fact that it is based on a technology agnostic tree representation and associated tree operations, this thesis would not recommend including persistence concepts within the openEHR specifications. Instead, introduction of optional, well defined methodologies for guiding and assisting key implementation tasks facing openEHR adopters, such as persistence, should be considered, thereby establishing a middle ground between extending the

specification with implementation concepts and leaving crucial and inevitable implementation tasks completely out of scope.

The tree based persistence abstraction of Chapter 7 achieves this balance, in addition to establishing a rich topic for future research. Large numbers of algorithms and architectures, which are already available from research in XML processing, can be adapted to numerous implementations involving different persistence systems. This new approach establishes openEHR persistence as a field of research rather than its currently accepted simply as an implementation task. The scope of this newly defined research topic is important and vast, built on the intersection of concepts from computer science, information retrieval, medicine, knowledge engineering, and also statistics.

Recent research in concurrent computing has produced results that offer significant capabilities for future research on tree based persistence abstraction for openEHR. These results, which are now known as big-data frameworks, have the potential to unify all aspects of the integration architecture defined in Section 4.7, by simultaneously supporting mainstream programming languages and statistical programming languages. The Apache Spark (Zaharia et al. 2010) framework is an example of this new holistic platform approach, based on its support for the Java (Arnold et al. 1996), Scala (Odersky et al. 2004), Python (Van Rossum 2007) and R (R Development Core Team 2008) programming languages, with seamless access to large scale distributed data.

10.5: Concluding Remarks

The first and foremost aim of this thesis has been to test the idea of better CDS being made possible through the direct incorporation of standardised electronic health records, with openEHR and BNs chosen as the particular representatives of these two key concepts.

Adopting an experimental approach, and attempting to develop an architecture that tests this idea in practice, by implementing and applying the components of this architecture, has proven to be a challenging task. The nature of the challenge lies both in the vast scope of both of the key concepts involved, and in the skill set and learning required to build a workbench that can be used to experiment with the number of complex, interacting components required.

This thesis does not claim that the architecture and methods that it describes are definitive for achieving the purposes set out in the scope of the work. However, it does claim that they are original, realistic, open for further research and

development, and are built with due consideration of key requirements. These requirements cover the building of clinical information systems and CDS systems on the same infrastructure, considering the performance requirements of both types of systems, and eliminating the need for designing and implementing multiple software architectures from scratch. Healthcare informatics needs proven methodology to this end – the data analytics of health care will collapse under the weight of the current inconsistencies and lack of standardisation in clinically meaningful ways that it currently battles. Seemingly small human actions, for example by patients in invoking their rights to withdraw consent in relation to their records, or parts of them, can currently lead to well-nigh impossible complexity and workload.

The implementation of both clinical information systems and CDS functionality based on openEHR clinical models has proven openEHR to be a rich and robust formalism. However, some of the problems discovered during implementation of the CDS system prototype require attention and consideration. These findings show that the capabilities of openEHR should not be taken as a guarantee for improved CDS adoption and implementation. Generalising the versatility of its model-driven approach beyond clinical care, without careful experiments and observation, is wrong.

openEHR's capabilities and potential uses need to be tested with a realistic workbench. Lack of an easily accessible implementation for research purposes makes it hard for researchers to follow this approach. However, the suggested changes to the openEHR specifications show that experiments on such a testbed allow otherwise unachievable bottom up contributions, which justify the implementation efforts involved.

Despite a significant amount of software development using a number of programming languages, open source tools and frameworks, this thesis does not explore the complete scope of the openEHR specifications. This was an intentional choice, made inevitable by the limited time available for completion of the study. Other aspects of an openEHR implementation, such as versioning of data, message exchange with external systems or relationship to terminologies and terminology servers have been left out of scope.

Other aspects of openEHR that were left out of the scope of the thesis are considered within the scope of future research, and to be included in a planned more comprehensive implementation of the openEHR-CDS integration architecture defined in Chapter 4. Therefore, this thesis concludes with the hope that its findings will assist CDS implementations based on openEHR move forward, and that the new methods and changes to the openEHR specifications that it proposes can be

extended in future research, based on the fundamental and crucial components defined at the beginning of this chapter.

Appendix I: Synthetic Data Generation

Although the primary focus of this thesis is not the use of a BN for predicting the outcome in the cataract surgery scenario discussed by (Narendran et al. 2008), the use of a test data set for evaluation of the methods it has developed should reflect the characteristics of patient population as fully as possible. To achieve the scale and consistency of test data required to explore the methods, experimentally, synthetic data generation approach was adopted.

To this end, a small scale literature search for patient population simulation was performed. Research related to Clinical Trial Simulation (CTS) was identified as particularly relevant.

CTS complements the drug development process based on clinical trials, and its adoption has been increasing (N. H. G. Holford et al. 2000), (N. Holford, Ma, and Ploeger 2010), (Mould and Upton 2012). It consists of three types of simulations: system models (input/output models), covariate models and execution models (Perez-Ruixo et al. 2007), (Kimko and Duffull 2002). Of these, the covariate model is relevant to synthetic data generation for a virtual patient population.

The virtual patient population is built on a model that uses covariates such as age, weight and gender. The virtual patient population can be generated in different ways, depending on availability of real population data and knowledge of relationships between covariates (Kimko and Duffull 2002). The approach used in this thesis is based on the sampling a vector of covariates with the assumption that they are independent. The covariants are taken from (Narendran et al. 2008). The assumption of independence is potentially problematic since covariates may be correlated. Therefore covariances should be included in the simulation (Kimko and Duffull 2002). However, covariance information for the variables is not provided by (Narendran et al. 2008) or (Jaycock et al. 2007) which uses the same data set. Therefore, individual distributions of covariates, where available, were used to generate a covariate vector for each patient.

A data generation script was written in R that samples a provided number of vectors from the individually defined probability distributions for covariates. For continuous variables, first a sample from a normal distribution was taken and transformed to a discrete variable, following the same rules for discretization given in (Narendran et al. 2008). The data set created with this approach was then processed to generate an extra column to represent the existence of the clinical problem that the CDS implementation in Chapter 9 focuses on. Data values in every row in the data set (which represents the data in the patient's EHR based on

sampled values) is used as input to the logistic regression equation from (Narendran et al. 2008) and the resulting value is used as the parameter of the Binomial distribution, generating either 1 (problem exists) or 0. This binary outcome is then appended to every row, creating the (existence of) clinical problem column. The assumption behind this approach was that the logistic regression learned from the original data set encapsulated the relationship between the covariates and prevalence of the problem, and therefore that using it with sampled values would simulate a patient population in which the prevalence of the problem satisfies the constraints enforced by the regression equation.

Finally, the simulated data set with covariates and clinical problem column is persisted as a comma separated value file for further processing by the pipeline, as explained in Chapter 9.

REFERENCES

1. Agresti, A. 2007. *An Introduction to Categorical Data Analysis*. Wiley-Blackwell.
2. Aikins, J. S. 1980. "Representation of Control Knowledge in Expert Systems." In *Proceedings of the First AAAI*, 121–23,198.
3. Al-Khalifa, S., H. V. Jagadish, N. Koudas, J. M. Patel, D. Srivastava, and Y. Wu. 2002. "Structural Joins: A Primitive for Efficient XML Query Pattern Matching." In *Data Engineering, 2002. Proceedings. 18th International Conference on*, 141–52. IEEE.
4. Allen, Christian, Darius Jazayeri, Justin Miranda, Paul G. Biondich, Burke W. Mamlin, Ben A. Wolfe, Chris Seebregts, Neal Lesh, William M. Tierney, and Hamish S. F. Fraser. 2007. "Experience in Implementing the OpenMRS Medical Record System to Support HIV Treatment in Rwanda." *Studies in Health Technology and Informatics* 129 (Pt 1): 382–86.
5. Alvarez, Sonia M., Beverly A. Poelstra, and Randall S. Burd. 2006. "Evaluation of a Bayesian Decision Network for Diagnosing Pyloric Stenosis." *Journal of Pediatric Surgery* 41 (1): 155–61. doi:10.1016/j.jpedsurg.2005.10.019.
6. Amer-Yahia, Sihem, SungRan Cho, Laks VS Lakshmanan, and Divesh Srivastava. 2001. "Minimization of Tree Pattern Queries." In *ACM SIGMOD Record*, 30:497–508. ACM.
7. Andersen, S. K, K. G Olesen, F. V Jensen, and F. Jensen. 1989. "HUGIN—a Shell for Building Bayesian Belief Universes for Expert Systems." In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 2:1080–85.
8. Andreassen, Steen. 1992. "Planning of Therapy and Tests in Causal Probabilistic Networks." *Artificial Intelligence in Medicine* 4 (3): 227–41. doi:10.1016/0933-3657(92)90029-O.
9. Andreassen, Steen, Christian Riekehr, Brian Kristensen, Henrik C. Schønheyder, and Leonard Leibovici. 1999. "Using Probabilistic and Decision–theoretic Methods in Treatment and Prognosis Modeling." *Artificial Intelligence in Medicine* 15 (2): 121–34. doi:10.1016/S0933-3657(98)00048-7.
10. Antal, Peter, Geert Fannes, Dirk Timmerman, Yves Moreau, and Bart De Moor. 2003. "Bayesian Applications of Belief Networks and Multilayer Perceptrons for Ovarian Tumor Classification with Rejection." *Artificial Intelligence in Medicine* 29 (1–2): 39–60. doi:10.1016/S0933-3657(03)00053-8.
11. Antal, Peter, Geert Fannes, Dirk Timmerman, Yves Moreau, and Bart De Moor. 2004. "Using Literature and Data to Learn Bayesian Networks as Clinical Models of Ovarian Tumors." *Artificial Intelligence in Medicine* 30 (3): 257–81. doi:10.1016/j.artmed.2003.11.007.
12. Arion, Andrei, Véronique Benzaken, Ioana Manolescu, and Yannis Papakonstantinou. 2007. "Structured Materialized Views for XML Queries." In *Proceedings of the 33rd International Conference on Very Large Data Bases*, 87–98. VLDB '07. VLDB Endowment. http://dl.acm.org/citation.cfm?id=1325851.1325865.
13. Arion, Andrei, Véronique Benzaken, Ioana Manolescu, Yannis Papakonstantinou, and Ravi Vijay. 2006. "Algebra-Based Identification of Tree Patterns in XQuery." In *Flexible Query Answering Systems*, edited by Henrik Larsen, Gabriella Pasi, Daniel Ortiz-Arroyo, Troels Andreasen, and Henning Christiansen, 4027:13–25. Lecture Notes in Computer Science. Springer Berlin / Heidelberg. http://www.springerlink.com/content/n66j6v486423026x/abstract/.
14. Arnold, Ken, James Gosling, David Holmes, and David Holmes. 1996. *The Java Programming Language*. Vol. 2. Addison-wesley Reading.

15. Astbury, Nick, Mark Wood, Uday Gajiwala, Rajesh Patel, Yi Chen, Larry Benjamin, and Sunday O Abuh. 2008. "Management of Capsular Rupture and Vitreous Loss in Cataract Surgery." *Community Eye Health* 21 (65): 6–8.
16. Austin, Tony, Yin Lim, David Nguyen, and Dipak Kalra. 2011. "Design of an Electronic Healthcare Record Server Based on Part 1 of ISO EN 13606." *Journal of Healthcare Engineering* 2 (2): 143–60.
17. Austin, Tony, Shanghua Sun, Taher Hassan, and Dipak Kalra. 2013. "Evaluation of ISO EN 13606 as a Result of Its Implementation in XML." *Health Informatics Journal* 19 (4): 264–80. doi:10.1177/1460458212473993.
18. Aziz, Ayesha, Salvador Rodriguez, and Chris Chatwin. 2014. "From Guidelines to Practice: Improving Clinical Care through Rule-Based Clinical Decision Support at the Point of Care." In *Rules on the Web. From Theory to Applications*, edited by Antonis Bikakis, Paul Fodor, and Dumitru Roman, 178–85. Lecture Notes in Computer Science 8620. Springer International Publishing. http://link.springer.com/chapter/10.1007/978-3-319-09870-8_13.
19. Bache, Kevin, and Moshe Lichman. 2013. *UCI Machine Learning Repository*.
20. Bahga, A., and V.K. Madisetti. 2013. "A Cloud-Based Approach for Interoperable Electronic Health Records (EHRs)." *IEEE Journal of Biomedical and Health Informatics* 17 (5): 894–906. doi:10.1109/JBHI.2013.2257818.
21. Bahga, A., and V.K. Madisetti. 2015. "Healthcare Data Integration and Informatics in the Cloud." *Computer* 48 (2): 50–57. doi:10.1109/MC.2015.46.
22. Barbay, Jérémy. 2005. "Index-Trees for Descendant Tree Queries on XML Documents." *University of Watreloo Technical Reports*.
23. Barclay Adams, J. 1976. "A Probability Model of Medical Reasoning and the MYCIN Model." *Mathematical Biosciences* 32 (1-2): 177–86.
24. Bates, David W., and Atul A. Gawande. 2003. "Improving Safety with Information Technology." *New England Journal of Medicine* 348 (25): 2526–34. doi:10.1056/NEJMsa020847.
25. Batra, Shivani, Shelly Sachdeva, Pulkit Mehndiratta, and Hem Jyotsana Parashar. 2014. "Mining Standardized Semantic Interoperable Electronic Healthcare Records." *Pham, TD, Ichikawa, K., Oyama-Higa, M., Coomans, D., Jiang, X. Eds*, 179–93.
26. Bauer, Christian, and Gavin King. 2005. "Hibernate in Action."
27. Beale, T., and Sam Heard. 2007a. "The openEHR Archetype System." openEHR Foundation. http://www.openehr.org/releases/1.0.2/architecture/am/archetype_system.pdf.
28. Beale, T., and Sam Heard. 2007b. "Archetype Definitions and Principles." openEHR Foundation.
29. Beale, T., and Sam Heard. 2008a. "openEHR Architecture Overview." openEHR Foundation. http://www.openehr.org/releases/1.0.2/architecture/overview.pdf.
30. Beale, T., and Sam Heard. 2008b. "Archetype Definition Language." openEHR Foundation.
31. Beale, T., Sam Heard, D Kalra, and D Lloyd. 2008a. "The openEHR Reference Model Demographic Information Model." openEHR Foundation.
32. Beale, T., Sam Heard, D Kalra, and D Lloyd. 2008b. "The openEHR Reference Model Data Types Information Model." openEHR Foundation.
33. Beale, T., Sam Heard, D Kalra, and D Lloyd. 2008c. "The openEHR Reference Model Common Information Model." openEHR Foundation.

34. Beale, T., Sam Heard, D Kalra, and D Lloyd. 2008d. "The openEHR Reference Model Data Structures Information Model." openEHR Foundation.
35. Beale, T., Sam Heard, D Kalra, and Kalra Lloyd. 2008e. "The openEHR Reference Model EHR Information Model." openEHR Foundation.
36. Beale, T., S. Heard, D. Kalra, and D. Lloyd. 2006. "OpenEHR Architecture Overview." *The OpenEHR Foundation*. http://www.openehr.org/releases/1.0.1/html/architecture/overview/Output/front.html.
37. Beeler, George W. 1998. "HL7 Version 3—An Object-Oriented Methodology for Collaborative Standards development1." *International Journal of Medical Informatics* 48 (1–3): 151–61. doi:10.1016/S1386-5056(97)00121-4.
38. Bender, D., and K. Sartipi. 2013. "HL7 FHIR: An Agile and RESTful Approach to Healthcare Information Exchange." In *2013 IEEE 26th International Symposium on Computer-Based Medical Systems (CBMS)*, 326–31. doi:10.1109/CBMS.2013.6627810.
39. Benson, Tim. 2012. *Principles of Health Interoperability HL7 and SNOMED*. Springer Science & Business Media.
40. Benzaken, Véronique, Giuseppe Castagna, and Cédric Miachon. 2005. "A Full Pattern-Based Paradigm for XML Query Processing." In *Practical Aspects of Declarative Languages*, edited by Manuel Hermenegildo and Daniel Cabeza, 3350:235–52. Lecture Notes in Computer Science. Springer Berlin / Heidelberg. http://www.springerlink.com/content/9xlntv1me75nd790/abstract/.
41. Berner, Eta S. 2009. "Clinical Decision Support Systems: State of the Art." *AHRQ Publication*, no. 09-0069: 4–26.
42. Berners-Lee, Tim, and Dan Connolly. 1995. *Hypertext Markup Language-2.0*. RFC 1866, November.
43. Bisbal, Jesús, Gaye Stephens, and Jane Grimson. "Generic Access to Synapses EHCR Data." In .
44. Bishop, Christopher M. 2007. *Pattern Recognition and Machine Learning*. 1st ed. 2006. Corr. 2nd printing. Springer.
45. Blake, C., and C. J Merz. 1998. "{UCI} Repository of Machine Learning Databases."
46. Boag, Scott, Don Chamberlin, Mary F. Fernández, Daniela Florescu, Jonathan Robie, Jérôme Siméon, and Mugur Stefanescu. 2002. *XQuery 1.0: An XML Query Language*.
47. Bodon, F., and L. Rónyai. 2003. "Trie: An Alternative Data Structure for Data Mining Algorithms." *Mathematical and Computer Modelling*, Hungarian Applied Mathematics, 38 (7–9): 739–51. doi:10.1016/0895-7177(03)90058-6.
48. Bohannon, P., J. Freire, P. Roy, and J. Simeon. 2002. "From XML Schema to Relations: A Cost-Based Approach to XML Storage." In *18th International Conference on Data Engineering, 2002. Proceedings*, 64–75. doi:10.1109/ICDE.2002.994698.
49. Bolstad, W. M. 2004. *Introduction to Bayesian Statistics*. Wiley-Ieee.
50. Borthakur, D. 2007. "The Hadoop Distributed File System: Architecture and Design." *Hadoop Project Website*.
51. Bowie, Jack, and G.Octo Barnett. 1976. "MUMPS — An Economical and Efficient Time-Sharing System for Information Management." *Computer Programs in Biomedicine* 6 (1): 11–22. doi:10.1016/0010-468X(76)90048-9.
52. Boxwala, A. A, M. Peleg, S. Tu, O. Ogunyemi, Q. T Zeng, D. Wang, V. L Patel, R. A Greenes, and E. H Shortliffe. 2004. "GLIF3: A Representation Format for Sharable Computer-Interpretable Clinical Practice Guidelines." *Journal of Biomedical Informatics* 37 (3): 147–61.

53. Bradley, A. P. 1997. "The Use of the Area under the ROC Curve in the Evaluation of Machine Learning Algorithms." *Pattern Recognition* 30 (7): 1145–59.
54. Bray, T., J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. 1997. "Extensible Markup Language (XML)." *World Wide Web Journal* 2 (4): 27–66.
55. Brenes, S., Y. Wu, D. Van Gucht, and P. Santa Cruz. 2008. "Trie Indexes for Efficient Xml Query Evaluation." *WebDB, Vancouver, Canada*.
56. Brewer, M. J., C. G. G. Aitken, and M. Talbot. 1996. "A Comparison of Hybrid Strategies for Gibbs Sampling in Mixed Graphical Models." *Computational Statistics & Data Analysis* 21 (3): 343–65.
57. Brown, Steven H., Michael J. Lincoln, Peter J. Groen, and Robert M. Kolodner. 2003. "VistA—U.S. Department of Veterans Affairs National-Scale HIS." *International Journal of Medical Informatics*, Working Conference on Health Information Systems, 69 (2–3): 135–56. doi:10.1016/S1386-5056(02)00131-4.
58. Bruno, Nicolas, Nick Koudas, and Divesh Srivastava. 2002. "Holistic Twig Joins: Optimal XML Pattern Matching." In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, 310–21. SIGMOD '02. New York, NY, USA: ACM. doi:10.1145/564691.564727.
59. Buntine, W. 1996. "A Guide to the Literature on Learning Probabilistic Networks from Data." *Knowledge and Data Engineering, IEEE Transactions on* 8 (2): 195–210.
60. Cattell, Rick. 2011. "Scalable SQL and NoSQL Data Stores." *SIGMOD Rec.* 39 (4): 12–27. doi:10.1145/1978915.1978919.
61. Cd, Kohl, Garde S, and Knaup P. 2009. "Facilitating Secondary Use of Medical Data by Using openEHR Archetypes." *Studies in Health Technology and Informatics* 160 (Pt 2): 1117–21.
62. Celko, Joe. 2012. *Joe Celko's Trees and Hierarchies in SQL for Smarties*. Elsevier.
63. CEN/ISO 13606 Association. 2015. "The CEN/ISO 13606 Association Site." December 12. http://www.en13606.org/.
64. Charitos, Theodore, Linda C. van der Gaag, Stefan Visscher, Karin A.M. Schurink, and Peter J.F. Lucas. 2009. "A Dynamic Bayesian Network for Diagnosing Ventilator-Associated Pneumonia in ICU Patients." *Expert Systems with Applications* 36 (2, Part 1): 1249–58. doi:10.1016/j.eswa.2007.11.065.
65. Cheng, J., and M. J Druzdzel. 2000. "AIS-BN: An Adaptive Importance Sampling Algorithm for Evidential Reasoning in Large Bayesian Networks." *J. Artif. Intell. Res. (JAIR)* 13: 155–88.
66. Cheng, P.H., C.H. Yang, H.S. Chen, S.J. Chen, and J.S. Lai. 2004. "Application of HL7 in a Collaborative Healthcare Information System." In *26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2004. IEMBS '04*, 2:3354–57. doi:10.1109/IEMBS.2004.1403942.
67. Chen, Li, Amarnath Gupta, and M. Erdem Kurul. 2005. "Stack-Based Algorithms for Pattern Matching on DAGs." In *Proceedings of the 31st International Conference on Very Large Data Bases*, 493–504. VLDB '05. VLDB Endowment. http://dl.acm.org/citation.cfm?id=1083592.1083651.
68. Chen, Roland S., Prakash Nadkarni, Luis Marenco, Forrest Levin, Joseph Erdos, and Perry L. Miller. 2000. "Exploring Performance Issues for a Clinical Database Organized Using an Entity-Attribute-Value Representation." *Journal of the American Medical Informatics Association* 7 (5): 475–87. doi:10.1136/jamia.2000.0070475.
69. Chen, Rong, and Iago Corbal. 2015. "Guideline Definition Language (GDL)." openEHR Foundation.

70. Chen, Rong, Susan M. Resnick, Christos Davatzikos, and Edward H. Herskovits. 2012. "Dynamic Bayesian Network Modeling for Longitudinal Brain Morphometry." *NeuroImage* 59 (3): 2330–38. doi:10.1016/j.neuroimage.2011.09.023.
71. Chen, T., J. Lu, and T. W. Ling. 2005. "On Boosting Holism in XML Twig Pattern Matching Using Structural Indexing Techniques." In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, 455–66. ACM.
72. Chevrolat, Jean-Paul, Jean-Louis Golmard, Salomon Ammar, Roland Jouvent, and Jean-François Boisvieux. 1998. "Modelling Behavioral Syndromes Using Bayesian Networks." *Artificial Intelligence in Medicine* 14 (3): 259–77. doi:10.1016/S0933-3657(98)00037-2.
73. CIMI. 2015. "Mission and Goals | Www.opencimi.org." December 30. http://www.opencimi.org/.
74. Cimino, James J. 1996. "Review Paper: Coding Systems in Health Care." *Methods of Information in Medicine-Methodik Der Information in Der Medizin* 35 (4): 273–84.
75. Cimino, J. J. 2011. "High-Quality, Standard, Controlled Healthcare Terminologies Come of Age." *Methods of Information in Medicine* 50 (2): 101.
76. Clancey, W. J, and E. H Shortliffe. 1984. *Readings in Medical Artificial Intelligence: The First Decade*. Addison-Wesley Longman Publishing Co., Inc.
77. Clark, James, and Steve DeRose. 1999. "XML Path Language (XPath)." http://www.w3.org/TR/xpath/.
78. Codd, E. F. 1970. "A Relational Model of Data for Large Shared Data Banks." *Commun. ACM* 13 (6): 377–87. doi:10.1145/362384.362685.
79. Cooper, Brian F., Neal Sample, Michael J. Franklin, Gisli R. Hjaltason, and Moshe Shadmon. 2001. "A Fast Index for Semistructured Data." In *VLDB*, 1:341–50.
80. Cormen, Thomas H. 2009. *Introduction to Algorithms*. MIT press.
81. Cornet, R. 2015. "ISO 13606 Based System for Biomedical Parameter Storage, Querying and Alarm Detection."
82. Cowan, John, and Richard Tobin. 2004. *XML Information Set*. W3C REC REC-xml-infoset-20040204.
83. Cowell, R.G., A.P. Dawid, T. Hutchinson, and D.J. Spiegelhalter. 1991. "A Bayesian Expert System for the Analysis of an Adverse Drug Reaction." *Artificial Intelligence in Medicine* 3 (5): 257–70. doi:10.1016/0933-3657(91)90031-6.
84. Crockford, Douglas. 2006. "The Application/json Media Type for Javascript Object Notation (json)."
85. Date, Chris J., and Hugh Darwen. 1987. *A Guide to the SQL Standard*. Vol. 3. Addison-Wesley New York.
86. De Dombal, F. T., D. J. Leaper, J. R. Staniland, A. P. McCann, and J. C. Horrocks. 1972. "Computer-Aided Diagnosis of Acute Abdominal Pain." *British Medical Journal* 2 (5804): 9.
87. de Lusignan, Simon, Christopher Minmagh, John Kennedy, Marco Zeimet, Hans Bommezijn, and John Bryant. 2001. "A Survey to Identify the Clinical Coding and Classification Systems Currently in Use across Europe." *Studies in Health Technology and Informatics*, no. 1: 86–89.
88. Dempster, A. P, N. M Laird, and D. B Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B (Methodological)* 39 (1): 1–38.
89. desRivieres, J., and J. Wiegand. 2004. "Eclipse: A Platform for Integrating Development Tools." *IBM Systems Journal* 43 (2): 371–83.

90. Díez, F.J., J. Mira, E. Iturralde, and S. Zubillaga. 1997. "DIAVAL, a Bayesian Expert System for Echocardiography." *Artificial Intelligence in Medicine* 10 (1): 59–73. doi:10.1016/S0933-3657(97)00384-9.
91. Dinu, Valentin, and Prakash Nadkarni. 2007. "Guidelines for the Effective Use of Entity–attribute–value Modeling for Biomedical Databases." *International Journal of Medical Informatics* 76 (11–12): 769–79. doi:10.1016/j.ijmedinf.2006.09.023.
92. Dinu, Valentin, Hongyu Zhao, and Perry L. Miller. 2007. "Integrating Domain Knowledge with Statistical and Data Mining Methods for High-Density Genomic SNP Disease Association Analysis." *Journal of Biomedical Informatics*, Intelligent Data Analysis in Biomedicine, 40 (6): 750–60. doi:10.1016/j.jbi.2007.06.002.
93. Di Tomaso, E., and J. F. Baldwin. 2008. "An Approach to Hybrid Probabilistic Models." *International Journal of Approximate Reasoning* 47 (2): 202–18.
94. Doctor, Jason N., and Greg Strylewicz. 2010. "Detecting 'wrong Blood in Tube' Errors: Evaluation of a Bayesian Network Approach." *Artificial Intelligence in Medicine* 50 (2): 75–82. doi:10.1016/j.artmed.2010.05.008.
95. Dougherty, James, Ron Kohavi, and Mehran Sahami. 1995. "Supervised and Unsupervised Discretization of Continuous Features." In *Machine Learning: Proceedings of the Twelfth International Conference*, 12:194–202.
96. Druzdzel, M. J. 1996. "Qualitiative Verbal Explanations in Bayesian Belief Networks." *AISB QUARTERLY*, 43–54.
97. Druzdzel, M. J. 1999. "SMILE: Structural Modeling, Inference, and Learning Engine and GeNIe: A Development Environment for Graphical Decision-Theoretic Models." In *Proceedings of the National Conference on Artificial Intelligence*, 902–3. JOHN WILEY & SONS LTD.
98. Duda, R. O, and E. H Shortliffe. 1983. "Expert Systems Research." *Science* 220 (4594): 261–68.
99. Du, Fang, Sihem Amer-Yahia, and Juliana Freire. 2004. "ShreX: Managing XML Documents in Relational Databases." In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, 1297–1300. VLDB '04. Toronto, Canada: VLDB Endowment. http://dl.acm.org/citation.cfm?id=1316689.1316818.
100. Duftschmid, Georg, Judith Chaloupka, and Christoph Rinner. 2013. "Towards Plug-and-Play Integration of Archetypes into Legacy Electronic Health Record Systems: The ArchiMed Experience." *BMC Medical Informatics and Decision Making* 13 (1): 11. doi:10.1186/1472-6947-13-11.
101. Duftschmid, Georg, Thomas Wrba, and Christoph Rinner. 2010. "Extraction of Standardized Archetyped Data from Electronic Health Record Systems Based on the Entity-Attribute-Value Model." *International Journal of Medical Informatics* 79 (8): 585–97. doi:10.1016/j.ijmedinf.2010.04.007.
102. Elvira, Consortium. 2002. "Elvira: An Environment for Creating and Using Probabilistic Graphical Models." In *Proceedings of the First European Workshop on Probabilistic Graphical Models*, 222–30.
103. Fan, Wenfei, Jeffrey Xu Yu, Hongjun Lu, Jianhua Lu, and Rajeev Rastogi. 2005. "Query Translation from XPath to SQL in the Presence of Recursive DTDs." In *Proceedings of the 31st International Conference on Very Large Data Bases*, 337–48. VLDB Endowment.
104. Fenz, Stefan. 2012. "An Ontology-Based Approach for Constructing Bayesian Networks." *Data & Knowledge Engineering* 73 (March): 73–88. doi:10.1016/j.datak.2011.12.001.

105. Fernández, Mary, Ashok Malhotra, Jonathan Marsh, Marton Nagy, and Norman Walsh. 2002. "XQuery 1.0 and XPath 2.0 Data Model." *W3C Working Draft* 15.
106. Florescu, Daniela, and Donald Kossmann. 1999a. "A Performance Evaluation of Alternative Mapping Schemes for Storing XML Data in a Relational Database."
107. Florescu, Daniela, and Donald Kossmann. 1999b. "Storing and Querying XML Data Using an RDMBS." *IEEE Data Engineering Bulletin, Special Issue on* 1060 (22): 3.
108. Folk, Mike, Gerd Heber, Quincey Koziol, Elena Pourmal, and Dana Robinson. 2011. "An Overview of the HDF5 Technology Suite and Its Applications." In *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*, 36–47. AD '11. New York, NY, USA: ACM. doi:10.1145/1966895.1966900.
109. Fraccaro, Paolo, Mercedes Arguello Castelerio, John Ainsworth, and Iain Buchan. 2015. "Adoption of Clinical Decision Support in Multimorbidity: A Systematic Review." *JMIR Medical Informatics* 3 (1): e4. doi:10.2196/medinform.3503.
110. Fraser, Hamish SF, Ali Habib, Mark Goodrich, David Thomas, Joaquin A. Blaya, Joseph Reginald Fils-Aime, Darius Jazayeri, Michael Seaton, Aamir J. Khan, and Sharon S. Choi. 2013. "E-Health Systems for Management of MDR-TB in Resource-Poor Environments: A Decade of Experience and Recommendations for Future Work." In *MedInfo*, 627–31.
111. Fraser, Hamish SF, David Thomas, Juan Tomaylla, Nadia Garcia, Leonid Lecca, Megan Murray, and Mercedes C Becerra. 2012. "Adaptation of a Web-Based, Open Source Electronic Medical Record System Platform to Support a Large Study of Tuberculosis Epidemiology." *BMC Medical Informatics and Decision Making* 12 (1): 125. doi:10.1186/1472-6947-12-125.
112. Freire, Sergio Miranda, Erik Sundvall, Daniel Karlsson, and Patrick Lambrix. 2012. "Performance of XML Databases for Epidemiological Queries in Archetype-Based EHRs." In , 51–57. Linköping University Electronic Press. http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A558990&dswid=2323.
113. Galán, S.F., F. Aguado, F.J. Díez, and J. Mira. 2002. "NasoNet, Modeling the Spread of Nasopharyngeal Cancer with Networks of Probabilistic Events in Discrete Time." *Artificial Intelligence in Medicine* 25 (3): 247–64. doi:10.1016/S0933-3657(02)00027-1.
114. Gappa, Ute, Frank Puppe, and Stefan Schewe. 1993. "Graphical Knowledge Acquisition for Medical Diagnostic Expert Systems." *Artificial Intelligence in Medicine* 5 (3): 185–211. doi:10.1016/0933-3657(93)90024-W.
115. Geiger, D., T. Verma, and J. Pearl. 1990. "Identifying Independence in Bayesian Networks." *Networks* 20 (5): 507–34.
116. Gelman, A., J. B Carlin, H. S Stern, and D. B Rubin. 2004. *Bayesian Data Analysis*. CRC press.
117. Getoor, Lise, Jeanne T Rhee, Daphne Koller, and Peter Small. 2004. "Understanding Tuberculosis Epidemiology Using Structured Statistical Models." *Artificial Intelligence in Medicine* 30 (3): 233–56. doi:10.1016/j.artmed.2003.11.003.
118. Gevaert, O., F. De Smet, D. Timmerman, Y. Moreau, and B. De Moor. 2006. "Predicting the Prognosis of Breast Cancer by Integrating Clinical and Microarray Data with Bayesian Networks." *Bioinformatics* 22 (14): e184–90.
119. Gilks, W. R, S. Richardson, and D. J Spiegelhalter. 1996. *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC.

120. Gimbel, Howard Vance. 1990. "Posterior Capsule Tears Using Phaco-Emulsification Causes, Prevention and Management." *European Journal of Implant and Refractive Surgery*, Capsular Surgery, 2 (1): 63–69. doi:10.1016/S0955-3681(13)80127-X.
121. Gimbel, Howard V, Ran Sun, Maria Ferensowicz, Ellen Anderson Penno, and Aasim Kamal. 2001. "Intraoperative Management of Posterior Capsule Tears in Phacoemulsification and Intraocular Lens implantation1." *Ophthalmology* 108 (12): 2186–89. doi:10.1016/S0161-6420(01)00716-3.
122. González-Ferrer, Arturo, Mor Peleg, Bert Verhees, Jan-Marc Verlinden, and Carlos Marcos. 2013. "Data Integration for Clinical Decision Support Based on openEHR Archetypes and HL7 Virtual Medical Record." In *Process Support and Knowledge Representation in Health Care*, edited by Richard Lenz, Silvia Miksch, Mor Peleg, Manfred Reichert, David Riaño, and Annette ten Teije, 71–84. Lecture Notes in Computer Science 7738. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-642-36438-9_5.
123. Gosling, J., B. Joy, G. Steele, and G. Bracha. 2005. *Java (TM) Language Specification, The (Java (Addison-Wesley))*. Addison-Wesley Professional.
124. Götz, Michaela, Christoph Koch, and Wim Martens. 2009. "Efficient Algorithms for Descendant-Only Tree Pattern Queries." *Information Systems* 34 (7): 602–23. doi:10.1016/j.is.2009.03.010.
125. Gou, Gang, and R. Chirkova. 2007. "Efficiently Querying Large XML Data Repositories: A Survey." *IEEE Transactions on Knowledge and Data Engineering* 19 (10): 1381–1403. doi:10.1109/TKDE.2007.1060.
126. Greenes, R. A. 2007. *Clinical Decision Support: The Road Ahead*. Academic Press.
127. Greenes, Robert A. 2014. *Clinical Decision Support (Second Edition)*. Edited by Robert A. Greenes. Oxford: Academic Press. http://www.sciencedirect.com/science/article/pii/B978012398476000014.
128. Grimson, J., E. Felton, G. Stephens, W. Grimson, and D. Berry. 1997. "Interoperability Issues in Sharing Electronic Healthcare Records-the Synapses Approach." In *, Third IEEE International Conference on Engineering of Complex Computer Systems, 1997. Proceedings*, 180–85. doi:10.1109/ICECCS.1997.622309.
129. Grimson, J., W. Grimson, D. Berry, G. Stephens, E. Felton, D. Kalra, P. Toussaint, and O.W. Weier. 1998. "A CORBA-Based Integration of Distributed Electronic Healthcare Records Using the Synapses Approach." *IEEE Transactions on Information Technology in Biomedicine* 2 (3): 124–38. doi:10.1109/4233.735777.
130. Hachicha, M., and J. Darmont. 2013. "A Survey of XML Tree Patterns." *IEEE Transactions on Knowledge and Data Engineering* 25 (1): 29–46. doi:10.1109/TKDE.2011.209.
131. Haddawy, Peter, Joel Jacobson, and Charles E Kahn Jr. 1997. "BANTER: A Bayesian Network Tutoring Shell." *Artificial Intelligence in Medicine* 10 (2): 177–200. doi:10.1016/S0933-3657(96)00374-0.
132. Halland, Ken, Katarina Britz, and Aurona Gerber. 2011. "Investigations into the Use of SNOMED CT to Enhance an OpenMRS Health Information System." *South African Computer Journal* 47: 33–45.
133. Hall, M., E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H Witten. 2009. "The WEKA Data Mining Software: An Update." *ACM SIGKDD Explorations Newsletter* 11 (1): 10–18.
134. Hand, D. J, and K. Yu. 2001. "Idiot's Bayes—Not So Stupid after All?" *International Statistical Review* 69 (3): 385–98.
135. Han, Zhongming, Congting Xi, and Jiajin Le. 2005. "Efficiently Coding and Indexing XML Document." In *Database Systems for Advanced*

Applications, edited by Lizhu Zhou, Beng Chin Ooi, and Xiaofeng Meng, 138–50. Lecture Notes in Computer Science 3453. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/11408079_14.
136. Harding, Philip J., Quanzhong Li, and Bongki Moon. 2003. "XISS/R: XML Indexing and Storage System Using RDBMS." In *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, 1073–76. VLDB '03. Berlin, Germany: VLDB Endowment. http://dl.acm.org/citation.cfm?id=1315451.1315552.
137. Hayward, John, Sergio A. Alvarez, Carolina Ruiz, Mary Sullivan, Jennifer Tseng, and Giles Whalen. 2010. "Machine Learning of Clinical Performance in a Pancreatic Cancer Database." *Artificial Intelligence in Medicine* 49 (3): 187–95. doi:10.1016/j.artmed.2010.04.009.
138. Heckerman, David E., and Edward H. Shortliffe. 1992. "From Certainty Factors to Belief Networks." *Artificial Intelligence in Medicine* 4 (1): 35–52. doi:10.1016/0933-3657(92)90036-O.
139. Hejlsberg, Anders, Scott Wiltamuth, and Peter Golde. 2003. *C# Language Specification*. Addison-Wesley Longman Publishing Co., Inc.
140. Helms, Ronald W., and Imogene McCanless. 1990. "The Conflict between Relational Databases and the Hierarchical Structure of Clinical Trials Data." *Controlled Clinical Trials* 11 (1): 7–23. doi:10.1016/0197-2456(90)90028-Z.
141. HL7. 2005. "GELLO." http://wiki.hl7.org/index.php?title=Product_GELLO.
142. HL7. 2015a. "HL7 Clinical Genomics Group." December 12. http://www.hl7.org/special/committees/clingenomics/.
143. HL7. 2015b. "Profiling - FHIR v0.5.0." December 12. https://www.hl7.org/FHIR/2015May/profiling.html.
144. HL7. 2015c. "Reference Information Model (RIM)." December 12. http://www.hl7.org/implement/standards/rim.cfm.
145. HL7. 2015d. "Summary - FHIR v1.0.2." December 12. https://www.hl7.org/fhir/summary.html.
146. Højsgaard, Søren. 2014. *gRain: Graphical Independence Networks*. http://cran.r-project.org/web/packages/gRain/index.html.
147. Holford, N. H. G., H. C. Kimko, J. P. R. Monteleone, and C. C. Peck. 2000. "Simulation of Clinical Trials." *Annual Review of Pharmacology and Toxicology* 40 (1): 209–34. doi:10.1146/annurev.pharmtox.40.1.209.
148. Holford, N., S. C. Ma, and B. A. Ploeger. 2010. "Clinical Trial Simulation: A Review." *Clinical Pharmacology & Therapeutics* 88 (2): 166–82. doi:10.1038/clpt.2010.114.
149. Hripcsak, George. 1994. "Writing Arden Syntax Medical Logic Modules." *Computers in Biology and Medicine* 24 (5): 331–63. doi:10.1016/0010-4825(94)90002-7.
150. Hripcsak, G., P. Ludemann, T. A Pryor, O. B Wigertz, and P. D Clayton. 1994. "Rationale for the Arden Syntax." *Computers and Biomedical Research* 27 (4): 291–324.
151. IHTSDO. 2015. "Snomed CT." IHTSDO. Accessed August 13. http://www.ihtsdo.org/snomed-ct/.
152. Ingram, D. 1995. "The Good European Health Record." *Health in the New Communication Age, MF Laires, MF Ladeira and JP Christensen (Eds), IOS*, 66–74.
153. Ingram, David. 2002. "openEHR - Origins of openEHR." October. http://www.openehr.org/about/origins.
154. Irani, Keki B. 1993. "Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning."
155. ISO. 2015. "ISO/IEC DIS 9075-1 Information Technology -- Database Languages -- SQL -- Part 1: Framework (SQL/Framework)." Accessed January 2.

http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=63555.
156. ISO/EN 13606. 2012. "ISO/EN 13606." Accessed July 16. http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=40784.
157. Izadi, Sayyed Kamyar, Mostafa S. Haghjoo, and Theo Härder. 2012. "S3: Processing Tree-Pattern XML Queries with All Logical Operators." *Data & Knowledge Engineering* 72 (February): 31–62. doi:10.1016/j.datak.2011.09.003.
158. Izadi, Sayyed Kamyar, Theo Härder, and Mostafa S. Haghjoo. 2009. "S3: Evaluation of Tree-Pattern XML Queries Supported by Structural Summaries." *Data & Knowledge Engineering* 68 (1): 126–45. doi:10.1016/j.datak.2008.09.001.
159. Jaakkola, T., and M. Jordan. 1997. "A Variational Approach to Bayesian Logistic Regression Models and Their Extensions." In *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*. Citeseer.
160. Jagadish, H. V., S. Al-Khalifa, A. Chapman, L. V. S. Lakshmanan, A. Nierman, S. Paparizos, J. M. Patel, et al. 2002. "TIMBER: A Native XML Database." *The VLDB Journal* 11 (4): 274–91. doi:10.1007/s00778-002-0081-x.
161. Jaycock, P., R. L. Johnston, H. Taylor, M. Adams, D. M. Tole, P. Galloway, C. Canning, and J. M. Sparrow. 2007. "The Cataract National Dataset Electronic Multi-Centre Audit of 55 567 Operations: Updating Benchmark Standards of Care in the United Kingdom and Internationally." *Eye* 23 (1): 38–49.
162. Jenders, R. A, R. Corman, and B. Dasgupta. 2003. "Making the Standard More Standard: A Data and Query Model for Knowledge Representation in the Arden Syntax." In *AMIA Annual Symposium Proceedings*, 2003:323. American Medical Informatics Association.
163. Jenders, R. A., G. Hripcsak, R. V. Sideli, W. DuMouchel, H. Zhang, J. J. Cimino, S. B. Johnson, E. H. Sherman, and P. D. Clayton. 1995. "Medical Decision Support: Experience with Implementing the Arden Syntax at the Columbia-Presbyterian Medical Center." *Proceedings of the Annual Symposium on Computer Application in Medical Care*, 169–73.
164. Jensen, C. S., and A. Kong. "Blocking-Gibbs Sampling in Very Large Probabilistic Expert Systems." *International Journal of Human-Computer Studies* 42: 647–66.
165. Jensen, Finn V. 2002. *Bayesian Networks and Decision Graphs*. Corrected. Springer.
166. Jiang, Haifeng, Hongjun Lu, Wei Wang, and Beng-Chin Ooi. 2003. "XR-Tree: Indexing XML Data for Efficient Structural Joins." In *19th International Conference on Data Engineering, 2003. Proceedings*, 253–64. doi:10.1109/ICDE.2003.1260797.
167. Jiang, Haifeng, Hongjun Lu, Wei Wang, and Jeffrey Xu Yu. 2002. "Path Materialization Revisited: An Efficient Storage Model for XML Data." In *Proceedings of the 13th Australasian Database Conference - Volume 5*, 85–94. ADC '02. Darlinghurst, Australia, Australia: Australian Computer Society, Inc. http://dl.acm.org/citation.cfm?id=563906.563916.
168. Jiang, H., H. Lu, and W. Wang. 2004. "Efficient Processing of XML Twig Queries with OR-Predicates." In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, 59–70. ACM.
169. Johnson, Stephen B. 1996. "Generic Data Modeling for Clinical Repositories." *Journal of the American Medical Informatics Association* 3 (5): 328–39. doi:10.1136/jamia.1996.97035024.
170. Julia Flores, M., Ann E. Nicholson, Andrew Brunskill, Kevin B. Korb, and Steven Mascaro. 2011. "Incorporating Expert Knowledge When Learning

Bayesian Network Structure: A Medical Case Study." *Artificial Intelligence in Medicine* 53 (3): 181–204. doi:10.1016/j.artmed.2011.08.004.
171. Kahn Jr, Charles E., Linda M. Roberts, Katherine A. Shaffer, and Peter Haddawy. 1997. "Construction of a Bayesian Network for Mammographic Diagnosis of Breast Cancer." *Computers in Biology and Medicine* 27 (1): 19–29. doi:10.1016/S0010-4825(96)00039-X.
172. Kalra, D., and D. Ingram. 2006. "Electronic Health Records." *Information Technology Solutions for Healthcare*, 135–81.
173. Kashfi, Hajar, and Robledo Jairo Jr. 2011. "Towards a Case-Based Reasoning Method for openEHR-Based Clinical Decision Support." In *Proceedings of The 3rd International Workshop on Knowledge Representation for Health Care (KR4HC'11)*.
174. Kasthurirathne, Suranga N., Burke Mamlin, Harsha Kumara, Grahame Grieve, and Paul Biondich. 2015. "Enabling Better Interoperability for HealthCare: Lessons in Developing a Standards Based Application Programing Interface for Electronic Medical Record Systems." *Journal of Medical Systems* 39 (11): 1–8. doi:10.1007/s10916-015-0356-6.
175. Kawamoto, Kensaku. 2010. "Standards for Scalable Clinical Decision Support: Need, Current and Emerging Standards, Gaps, and Proposal for Progress." *The Open Medical Informatics Journal* 4 (1): 235–44. doi:10.2174/1874431101004010235.
176. Keitt, H. Timothy. 2015. *Rpg: Easy Interface to Advanced PostgreSQL Features* (version 1.4). R. Accessed April 17. http://cran.r-project.org/web/packages/rpg/index.html.
177. Khalifa, Mohamed. 2014. "Clinical Decision Support: Strategies for Success." *Procedia Computer Science* 37: 422–27.
178. Kimko, Hui, and Stephen B. Duffull. 2002. *Simulation for Designing Clinical Trials: A Pharmacokinetic-Pharmacodynamic Modeling Perspective*. CRC Press.
179. Kloek, T., and H. K. Van Dijk. 1978. "Bayesian Estimates of Equation System Parameters: An Application of Integration by Monte Carlo." *Econometrica: Journal of the Econometric Society*, 1–19.
180. Knuth, D. E. 1968. "The Art of Computer Programming, Volume 1: Fundamental Algorithms Addison-Wesley." *Reading, Mass*.
181. Kohavi, Ron. 1995. "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection." In *Ijcai*, 14:1137–45.
182. Kohl, Christian Dominik. 2012. "Patientenübergreifende, Multiple Verwendung von Patientendaten Für Die Klinische Forschung Unter Nutzung von Archetypen."
183. Kohler, Michael, Christoph Rinner, Gudrun Hübner-Bloder, Samrend Saboor, Elske Ammenwerth, and Georg Duftschmid. 2011. "The Archetype-Enabled EHR System ZK-ARCHE-Integrating the ISO/EN 13606 Standard and IHE XDS Profile." In *MIE*, 799–803.
184. Koller, D., and N. Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press.
185. Koller, D., U. Lerner, and D. Angelov. 1999. "A General Algorithm for Approximate Inference and Its Application to Hybrid Bayes Nets." In *Proc. UAI*, 15:324–33.
186. Kondylakis, H., L. Koumakis, E. Genitsaridi, M. Tsiknakis, K. Marias, G. Pravettoni, A. Gorini, and K. Mazzocco. 2012. "IEmS: A Collaborative Environment for Patient Empowerment." In *2012 IEEE 12th International Conference on Bioinformatics Bioengineering (BIBE)*, 535–40. doi:10.1109/BIBE.2012.6399770.
187. Kopanitsa, G., C. Hildebrand, J. Stausberg, and K. H. Englmeier. 2013. "Visualization of Medical Data Based on EHR Standards." *Methods Inf Med* 52 (1): 43–50.

188. Korb, Kevin B., and Ann E. Nicholson. 2003. *Bayesian Artificial Intelligence*. 1st ed. Chapman and Hall/CRC.
189. Kozlov, A. V, and D. Koller. 1997. "Nonuniform Dynamic Discretization in Hybrid Networks." In *Uncertainty in Artificial Intelligence*, 13:314–25. Citeseer.
190. Krauthausen, P., and U. D Hanebeck. 2010. "Parameter Learning for Hybrid Bayesian Networks with Gaussian Mixture and Dirac Mixture Conditional Densities." In *American Control Conference (ACC), 2010*, 480–85. IEEE.
191. Kuhn, K. 2007. "Model-Centric Approaches for the Development of Health Information Systems." *Medinfo 2007*, 28.
192. Kuperman, G. J, A. Bobb, T. H Payne, A. J Avery, T. K Gandhi, G. Burns, D. C Classen, and D. W Bates. 2007. "Medication-Related Clinical Decision Support in Computerized Provider Order Entry Systems: A Review." *Journal of the American Medical Informatics Association* 14 (1): 29–40.
193. Lacave, Carmen, Agnieszka Oniśko, and Francisco J. Díez. 2006. "Use of Elvira's Explanation Facility for Debugging Probabilistic Expert Systems." *Knowledge-Based Systems* 19 (8): 730–38. doi:10.1016/j.knosys.2006.05.010.
194. Lacave, C., and F. J Díez. 2002. "A Review of Explanation Methods for Bayesian Networks." *The Knowledge Engineering Review* 17 (2): 107–27.
195. Lacave, C., M. Luque, and F. J Díez. 2007. "Explanation of Bayesian Networks and Influence Diagrams in Elvira." *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 37 (4): 952–65.
196. Lakshmanan, Laks V. S., Hui Wang, and Zheng Zhao. 2006. "Answering Tree Pattern Queries Using Views." In *Proceedings of the 32Nd International Conference on Very Large Data Bases*, 571–82. VLDB '06. Seoul, Korea: VLDB Endowment. http://dl.acm.org/citation.cfm?id=1182635.1164177.
197. Lalkhen, Abdul Ghaaliq, and Anthony McCluskey. 2008. "Clinical Tests: Sensitivity and Specificity." *Continuing Education in Anaesthesia, Critical Care & Pain* 8 (6): 221–23. doi:10.1093/bjaceaccp/mkn041.
198. Langseth, H., T. D Nielsen, R. Rumí, and A. Salmerón. 2009. "Inference in Hybrid Bayesian Networks." *Reliability Engineering & System Safety* 94 (10): 1499–1509.
199. Larrañaga, P., and S. Moral. 2011. "Probabilistic Graphical Models in Artificial Intelligence." *Applied Soft Computing*, The Impact of Soft Computing for the Progress of Artificial Intelligence, 11 (2): 1511–28. doi:10.1016/j.asoc.2008.01.003.
200. Lauritzen, S. L, and F. Jensen. 2001. "Stable Local Computation with Conditional Gaussian Distributions." *Statistics and Computing* 11 (2): 191–203.
201. Lauritzen, S. L, and D. J Spiegelhalter. 1988. "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems." *Journal of the Royal Statistical Society. Series B (Methodological)* 50 (2): 157–224.
202. Leaper, D. J., J. C Horrocks, J. R. Staniland, and F. T. De Dombal. 1972. "Computer-Assisted Diagnosis of Abdominal Pain Using 'estimates' Provided by Clinicians." *British Medical Journal* 4 (5836): 350–54.
203. Ledley, R. S., and L. B. Lusted. 1959a. "Reasoning Foundations of Medical Diagnosis." *Science* 130 (3366): 9–21.
204. Ledley, R.S., and L.B. Lusted. 1959b. "The Use of Electronic Computers to Aid in Medical Diagnosis." *Proceedings of the IRE* 47 (11): 1970–77. doi:10.1109/JRPROC.1959.287213.
205. Lee, Dennis, Nicolette de Keizer, Francis Lau, and Ronald Cornet. 2014. "Literature Review of SNOMED CT Use." *Journal of the American Medical*

Informatics Association 21 (e1): e11–19. doi:10.1136/amiajnl-2013-001636.
206. Lepar, V. 1998. "A Comparison of Lauritzen-Spiegelhalter, Hugin, and Shenoy-Shafer Architectures for Computing Marginals of Probability Distributions." In *Uncertainty in Artificial Intelligence*, 14:328–37. Citeseer.
207. Linder, J. A, J. Ma, D. W Bates, B. Middleton, and R. S Stafford. 2007. "Electronic Health Record Use and the Quality of Ambulatory Care in the United States." *Archives of Internal Medicine* 167 (13): 1400.
208. Lloyd, D., D. Kalra, T. Beale, A. Maskens, R. Dixon, J. Ellis, D. Camplin, P. Grubb, and D. Ingram. *The GEHR Final Architecture Description. European Commission, Brussels; 1995; The Good European Health Record Project: Deliverable 19. 11 Chapters; 250 Pages*.
209. Long, William J. 2001. "Medical Informatics: Reasoning Methods." *Artificial Intelligence in Medicine* 23 (1): 71–87. doi:10.1016/S0933-3657(01)00076-8.
210. Long, William J., Hamish Fraser, and Shapur Naimi. 1997. "Reasoning Requirements for Diagnosis of Heart Disease." *Artificial Intelligence in Medicine* 10 (1): 5–24. doi:10.1016/S0933-3657(97)00381-3.
211. Loong, T. W. 2003. "Understanding Sensitivity and Specificity with the Right Side of the Brain." *Bmj* 327 (7417): 716.
212. Löper, Dortje, Meike Klettke, Ilvio Bruder, and Andreas Heuer. 2012. "Integrating Healthcare-Related Information Using the Entity-Attribute-Value Storage Model." In *Health Information Science*, edited by Jing He, Xiaohui Liu, Elizabeth A. Krupinski, and Guandong Xu, 13–24. Lecture Notes in Computer Science 7231. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-642-29361-0_4.
213. Lopez, Diego M., and Bernd G. M. E. Blobel. 2009. "A Development Framework for Semantically Interoperable Health Information Systems." *International Journal of Medical Informatics* 78 (2): 83–103. doi:10.1016/j.ijmedinf.2008.05.009.
214. Lucas, Peter J.F., Nicolette C. de Bruijn, Karin Schurink, and Andy Hoepelman. 2000. "A Probabilistic and Decision-Theoretic Approach to the Management of Infectious Disease at the ICU." *Artificial Intelligence in Medicine* 19 (3): 251–79. doi:10.1016/S0933-3657(00)00048-8.
215. Luciani, Davide, and Federico M. Stefanini. 2012. "Automated Interviews on Clinical Case Reports to Elicit Directed Acyclic Graphs." *Artificial Intelligence in Medicine* 55 (1): 1–11. doi:10.1016/j.artmed.2011.11.007.
216. Lu, Jiaheng, Ting Chen, and Tok Wang Ling. 2004. "Efficient Processing of XML Twig Patterns with Parent Child Edges: A Look-Ahead Approach." In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, 533–42. CIKM '04. New York, NY, USA: ACM. doi:10.1145/1031171.1031272.
217. Lu, Jiaheng, Tok Wang Ling, Chee-Yong Chan, and Ting Chen. 2005. "From Region Encoding to Extended Dewey: On Efficient Processing of XML Twig Pattern Matching." In *Proceedings of the 31st International Conference on Very Large Data Bases*, 193–204. VLDB '05. Trondheim, Norway: VLDB Endowment. http://dl.acm.org/citation.cfm?id=1083592.1083618.
218. Lu, Jiaheng, Xiaofeng Meng, and Tok Wang Ling. 2011. "Indexing and Querying XML Using Extended Dewey Labeling Scheme." *Data & Knowledge Engineering* 70 (1): 35–59. doi:10.1016/j.datak.2010.08.001.
219. Lunn, D. J, A. Thomas, N. Best, and D. Spiegelhalter. 2000. "WinBUGS-a Bayesian Modelling Framework: Concepts, Structure, and Extensibility." *Statistics and Computing* 10 (4): 325–37.

220. Machdi, Imam, Toshiyuki Amagasa, and Hiroyuki Kitagawa. 2009. "Executing Parallel TwigStack Algorithm on a Multi-Core System." In *Proceedings of the 11th International Conference on Information Integration and Web-Based Applications & Services*, 176–84. iiWAS '09. New York, NY, USA: ACM. doi:10.1145/1806338.1806376.
221. Ma, Chunlan, Heath Frankel, and Thomas Beale. 2014. "Archetype Query Language Description - Specifications - openEHR Wiki." Accessed December 30. https://openehr.atlassian.net/wiki/display/spec/Archetype+Query+Language+Description.
222. MacLeod, Bruce Bradford, James Phillips, Allison Stone, Aliya Walji, and John Koku Awoonor-Williams. 2012. "The Architecture of a Software System for Supporting Community-Based Primary Health Care with Mobile Technology: The Mobile Technology for Community Health (MoTeCH) Initiative in Ghana." *Online Journal of Public Health Informatics* 4 (1). doi:10.5210/ojphi.v4i1.3910.
223. Madaan, Aastha, and Subhash Bhalla. 2014. "Usability Measures for Large Scale Adoption of the Standardized Electronic Health Record Databases." *Journal of Information Processing* 22 (3): 508–26. doi:10.2197/ipsjjip.22.508.
224. Madaan, Aastha, Wanming Chu, Yaginuma Daigo, and Subhash Bhalla. 2013. "Quasi-Relational Query Language Interface for Persistent Standardized EHRs: Using NoSQL Databases." In *Databases in Networked Information Systems*, edited by Aastha Madaan, Shinji Kikuchi, and Subhash Bhalla, 182–96. Lecture Notes in Computer Science 7813. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-642-37134-9_15.
225. Madigan, D., K. Mosurski, and R. G Almond. 1997. "Graphical Explanation in Belief Networks." *Journal of Computational and Graphical Statistics*, 160–81.
226. Mamlin, Burke W., Paul G. Biondich, Ben A. Wolfe, Hamish Fraser, Darius Jazayeri, Christian Allen, Justin Miranda, and William M. Tierney. 2006. "Cooking Up An Open Source EMR For Developing Countries: OpenMRS – A Recipe For Successful Collaboration." *AMIA Annual Symposium Proceedings* 2006: 529–33.
227. Mann, Kito D. 2005. *Java Server Faces in Action*. Dreamtech Press.
228. Markwell, D., L. Sato, and E. Cheetham. 2008. "Representing Clinical Information Using SNOMED Clinical Terms with Different Structural Information Models." *Proceedings of KR-MED 2008*.
229. Martínez-Costa, Catalina, Ronald Cornet, Daniel Karlsson, Stefan Schulz, and Dipak Kalra. 2015. "Semantic Enrichment of Clinical Models towards Semantic Interoperability. The Heart Failure Summary Use Case." *Journal of the American Medical Informatics Association* 22 (3): 565–76. doi:10.1093/jamia/ocu013.
230. McGuinness, D. L., and F. Van Harmelen. 2004. "OWL Web Ontology Language Overview." *W3C Recommendation* 10: 2004–03.
231. Menarguez, Marcos. 2013. "Digitum : Depósito de La Universidad de Murcia: Modelos de Representación de Arquetipos En Sistemas de Información Sanitarios." May 30. https://digitum.um.es/jspui/handle/10201/35157.
232. Menárguez-Tortosa, Marcos, Catalina Martínez-Costa, and Jesualdo Tomás Fernández-Breis. 2011. "A Generative Tool for Building Health Applications Driven by ISO 13606 Archetypes." *Journal of Medical Systems* 36 (5): 3063–75. doi:10.1007/s10916-011-9783-1.
233. Metz, Charles E. 1978. "Basic Principles of ROC Analysis." *Seminars in Nuclear Medicine* 8 (4): 283–98. doi:10.1016/S0001-2998(78)80014-2.
234. Meyer, Bertrand. 1988. "Object Oriented Software Construction."

235. Mohammed-Rajput, Nareesa A., Nyoman W. Ribeka, Sylvester Kimaiyo, and Martin C. Were. 2010. "Creating and Evaluating a Dynamic Study Randomization and Enrollment Tool within a Robust EHRs." *AMIA Annual Symposium Proceedings* 2010: 517–21.
236. Mohammed-Rajput, Nareesa A., Dawn C. Smith, Burke Mamlin, Paul Biondich, and Brad N. Doebbeling. 2011. "OpenMRS, A Global Medical Records System Collaborative: Factors Influencing Successful Implementation." *AMIA Annual Symposium Proceedings* 2011: 960–68.
237. Momjian, Bruce. 2001. *PostgreSQL: Introduction and Concepts*. Vol. 192. Addison-Wesley New York.
238. Montironi, Rodolfo, Peter H Bartels, Peter W Hamilton, and Deborah Thompson. 1996. "Atypical Adenomatous Hyperplasia (adenosis) of the Prostate: Development of a Bayesian Belief Network for Its Distinction from Well-Differentiated Adenocarcinoma." *Human Pathology* 27 (4): 396–407. doi:10.1016/S0046-8177(96)90114-8.
239. Montironi, Rodolfo, Roberta Mazzucchelli, Paola Colanzi, Marco Streccioni, Marina Scarpelli, Deborah Thompson, and Peter H. Bartels. 2002. "Improving Inter-Observer Agreement and Certainty Level in Diagnosing and Grading Papillary Urothelial Neoplasms:: Usefulness of a Bayesian Belief Network." *European Urology* 41 (4): 449–57. doi:10.1016/S0302-2838(02)00028-3.
240. Moral, S., R. Rumí, and A. Salmerón. 2001. "Mixtures of Truncated Exponentials in Hybrid Bayesian Networks." *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, 156–67.
241. Mori, Angelo Rossi. 1995. "Coding Systems and Controlled Vocabularies for Hospital Information Systems." *International Journal of Bio-Medical Computing*, Information System with Fadin Boundaries, 39 (1): 93–98. doi:10.1016/0020-7101(94)01085-F.
242. Mould, Dr, and Rn Upton. 2012. "Basic Concepts in Population Modeling, Simulation, and Model-Based Drug Development." *CPT: Pharmacometrics & Systems Pharmacology* 1 (9): 1–14. doi:10.1038/psp.2012.4.
243. Müller, M. L, T. Ganslandt, H. P Eich, K. Lang, C. Ohmann, and H. U Prokosch. 2001. "Towards Integration of Clinical Decision Support in Commercial Hospital Information Systems Using Distributed, Reusable Software and Knowledge Components." *International Journal of Medical Informatics* 64 (2-3): 369–77.
244. Murphy, K. P. 1999. "A Variational Approximation for Bayesian Networks with Discrete and Continuous Latent Variables." In *Proc. UAI*, 99:457–66.
245. Murphy, K. P. 2002. "Dynamic Bayesian Networks." *Probabilistic Graphical Models, M. Jordan*.
246. Musen, M. A. 1992. "Dimensions of Knowledge Sharing and Reuse." *Computers and Biomedical Research* 25 (5): 435–67.
247. Musen, Mark A., Blackford Middleton, and Robert A. Greenes. 2014. "Clinical Decision-Support Systems." In *Biomedical Informatics*, 643–74. Springer.
248. Musen, M. A, Y. Shahar, and E. H Shortliffe. 2006. "Clinical Decision-Support Systems." *Biomedical Informatics*, 698–736.
249. Nadkarni, Prakash M. 1997. "QAV: Querying Entity-Attribute-Value Metadata in a Biomedical Database." *Computer Methods and Programs in Biomedicine* 53 (2): 93–103. doi:10.1016/S0169-2607(97)01815-4.
250. Nadkarni, Prakash M. 1998. "Data Extraction and Ad Hoc Query of an Entity—Attribute— Value Database." *Journal of the American Medical Informatics Association* 5 (6): 511–27. doi:10.1136/jamia.1998.0050511.
251. Nadkarni, Prakash M., Luis Marenco, Roland Chen, Emmanouil Skoufos, Gordon Shepherd, and Perry Miller. 1999. "Organization of Heterogeneous Scientific Data Using the EAV/CR Representation." *Journal*

of the American Medical Informatics Association 6 (6): 478–93. doi:10.1136/jamia.1999.0060478.
252. Nagarajan, Radhakrishnan, Marco Scutari, and Sophie Lèbre. 2013. "Bayesian Network Inference Algorithms." In *Bayesian Networks in R*, 85–101. Use R! 48. Springer New York. http://link.springer.com/chapter/10.1007/978-1-4614-6446-4_4.
253. Namasivayam, Vasanth Krishna, Animesh Pathak, and Viktor K. Prasanna. "Parallelizing Exact Inference in Bayesian Networks."
254. Namasivayam, V.K., A Pathak, and V.K. Prasanna. 2006. "Scalable Parallel Implementation of Bayesian Network to Junction Tree Conversion for Exact Inference." In *18TH International Symposium on Computer Architecture and High Performance Computing, 2006. SBAC-PAD '06*, 167–76. doi:10.1109/SBAC-PAD.2006.26.
255. Narendran, N., P. Jaycock, R. L. Johnston, H. Taylor, M. Adams, D. M. Tole, R. H. Asaria, P. Galloway, and J. M. Sparrow. 2008. "The Cataract National Dataset Electronic Multicentre Audit of 55 567 Operations: Risk Stratification for Posterior Capsule Rupture and Vitreous Loss." *Eye* 23 (1): 31–37. doi:10.1038/sj.eye.6703049.
256. Neal, R. M. 1993. "Probabilistic Inference Using Markov Chain Monte Carlo Methods (Technical Report CRG-TR-93-1)." *Department of Computer Science, University of Toronto*.
257. Neapolitan, R. E. 2004. *Learning Bayesian Networks*. Pearson Prentice Hall Upper Saddle River, NJ.
258. Neiswanger, Willie, Chong Wang, and Eric Xing. 2013. "Asymptotically Exact, Embarrassingly Parallel MCMC." *arXiv Preprint arXiv:1311.4780*.
259. Nicol, Gavin, Lauren Wood, Mike Champion, and Steve Byrne. 2001. "Document Object Model (DOM) Level 3 Core Specification." *W3C Working Draft* 13: 1–146.
260. Nogueira Reis, Zilma Silveira, Marcelo Rodrigues dos Santos Junior, Juliano de Souza Gaspar, Thais Abreu Maia, Andreia Cristina de Souza, and Marcelo Rodrigues dos Santos. 2015. "Electronic Systems Interoperability Study: Based on the Interchange of Hospital Obstetrical Information." In *Computer-Based Medical Systems (CBMS), 2015 IEEE 28th International Symposium on*, 201–4. IEEE.
261. Ocean Informatics. 2015. "Clinical Knowledge Manager." Accessed August 9. http://www.openehr.org/ckm/.
262. Odersky, Martin, Philippe Altherr, Vincent Cremet, Burak Emir, Sebastian Maneth, Stéphane Micheloud, Nikolay Mihaylov, Michel Schinz, Erik Stenman, and Matthias Zenger. 2004. "An Overview of the Scala Programming Language."
263. Onisko, A. 2003. "Probabilistic Causal Models in Medicine: Application to Diagnosis of Liver Disorders." Ph. D. thesis, Institute of Biocybernetics and Biomedical Engineering, Polish Academy of Science, Warsaw.
264. openEHR Foundation. 2015. "openEHR - Deployed Solutions." July 31. http://www.openehr.org/who_is_using_openehr/healthcare_providers_and_authorities.
265. OpenMRS Inc. 2015. "OpenMRS." December 12. http://openmrs.org/.
266. Overby, Casey Lynnette, Isaac Kohane, Joseph L. Kannry, Marc S. Williams, Justin Starren, Erwin Bottinger, Omri Gottesman, et al. 2013. "Opportunities for Genomic Clinical Decision Support Interventions." *Genetics in Medicine* 15 (10): 817–23. doi:10.1038/gim.2013.128.
267. Pal, Shankar, Istvan Cseri, Oliver Seeliger, Gideon Schaller, Leo Giakoumakis, and Vasili Zolotov. 2004. "Indexing XML Data Stored in a Relational Database." In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, 1146–57. VLDB '04. Toronto, Canada: VLDB Endowment. http://dl.acm.org/citation.cfm?id=1316689.1316787.

268. Parashar, Hem Jyotsana, Shelly Sachdeva, and Shivani Batra. 2013. "Enhancing Access to Standardized Clinical Application for Mobile Interfaces." In *Databases in Networked Information Systems*, edited by Aastha Madaan, Shinji Kikuchi, and Subhash Bhalla, 212–29. Lecture Notes in Computer Science 7813. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-642-37134-9_17.
269. Parashar, H.J., S. Sachdeva, S. Batra, and P. Mehndiratta. 2013. "Usability and Information Retrieval Issues for Electronic Healthcare Record Databases." In *2013 Sixth International Conference on Contemporary Computing (IC3)*, 410–14. doi:10.1109/IC3.2013.6612230.
270. Parikh, Rajul, Annie Mathai, Shefali Parikh, G Chandra Sekhar, and Ravi Thomas. 2008. "Understanding and Using Sensitivity, Specificity and Predictive Values." *Indian Journal of Ophthalmology* 56 (1): 45–50.
271. Pathak, Jyotishman, Harold R. Solbrig, James D. Buntrock, Thomas M. Johnson, and Christopher G. Chute. 2009. "LexGrid: A Framework for Representing, Storing, and Querying Biomedical Terminologies from Simple to Sublime." *Journal of the American Medical Informatics Association* 16 (3): 305–15. doi:10.1197/jamia.M3006.
272. Paul, Razan, and Abu Sayed Md. Latiful Hoque. 2011. "Optimized Entity Attribute Value Model: A Search Efficient Representation of High Dimensional and Sparse Data." *Interdisciplinary Bio Central* 3 (3): 1–5. doi:10.4051/ibc.2011.3.3.0009.
273. Pearl, J. 1986. "Fusion, Propagation, and Structuring in Belief Networks." *Artificial Intelligence* 29 (3): 241–88.
274. Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.
275. Peelen, Linda, Nicolette F. de Keizer, Evert de Jonge, Robert-Jan Bosman, Ameen Abu-Hanna, and Niels Peek. 2010. "Using Hierarchical Dynamic Bayesian Networks to Investigate Dynamics of Organ Failure in Patients in the Intensive Care Unit." *Journal of Biomedical Informatics* 43 (2): 273–86. doi:10.1016/j.jbi.2009.10.002.
276. Peleg, M., O. Ogunyemi, S. Tu, A. A Boxwala, Q. Zeng, R. A Greenes, and E. H Shortliffe. 2001. "Using Features of Arden Syntax with Object-Oriented Medical Data Models for Guideline Modeling." In *Proceedings of the AMIA Symposium*, 523. American Medical Informatics Association.
277. Peleg, M., S. Tu, J. Bury, P. Ciccarese, J. Fox, R. A Greenes, R. Hall, P. D Johnson, N. Jones, and A. Kumar. 2003. "Comparing Computer-Interpretable Guideline Models: A Case-Study Approach." *Journal of the American Medical Informatics Association* 10 (1): 52.
278. Perez-Ruixo, Juan Jose, Filip De Ridder, Hui Kimko, Mahesh Samtani, Eugene Cox, Surya Mohanty, and An Vermeulen. 2007. "Simulation in Clinical Drug Development." In *Biosimulation in Drug Development*, edited by rtin Bertau, Erik Mosekilde, and Hans V. Westerhoff, 1–26. Wiley-VCH Verlag GmbH & Co. KGaA. http://onlinelibrary.wiley.com/doi/10.1002/9783527622672.ch1/summary.
279. Plummer, M. 2003. "JAGS: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling." In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003). March*, 20–22.
280. Pourret, Olivier, Patrick Naïm, and Bruce Marcot, eds. 2008. *Bayesian Networks: A Practical Guide to Applications*. 1st ed. Wiley.
281. Pradhan, Malcolm, Gregory Provan, Blackford Middleton, and Max Henrion. 1994. "Knowledge Engineering for Large Belief Networks." In *Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence*, 484–90. UAI'94. San Francisco, CA, USA: Morgan

Kaufmann Publishers Inc. http://dl.acm.org/citation.cfm?id=2074394.2074456.
282. Qin, L., J. X. Yu, and B. Ding. 2007. "TwigList: Make Twig Pattern Matching Fast." In *Proceedings of the 12th International Conference on Database Systems for Advanced Applications*, 850–62. Springer-Verlag.
283. Ramoni, Marco, Alberto Riva, Mario Stefanelli, and Vimla Patel. 1995. "An Ignorant Belief Network to Forecast Glucose Concentration from Clinical Databases." *Artificial Intelligence in Medicine* 7 (6): 541–59. doi:10.1016/0933-3657(95)00026-1.
284. Ranjan, R. 2014. "Streaming Big Data Processing in Datacenter Clouds." *IEEE Cloud Computing* 1 (1): 78–83. doi:10.1109/MCC.2014.22.
285. Rao, P., and B. Moon. 2004. "PRIX: Indexing and Querying XML Using Prufer Sequences." In *20th International Conference on Data Engineering, 2004. Proceedings*, 288–99. doi:10.1109/ICDE.2004.1320005.
286. R Development Core Team. 2008. "R: A Language and Environment for Statistical Computing." *R Foundation for Statistical Computing Vienna Austria ISBN* 3 (10).
287. Rector, A. L. 2001. "The Interface between Information, Terminology, and Inference Models." *Studies in Health Technology and Informatics*, no. 1: 246–50.
288. Rector, Alan, and Luigi Iannone. 2012. "Lexically Suggest, Logically Define: Quality Assurance of the Use of Qualifiers and Expected Results of Post-Coordination in SNOMED CT." *Journal of Biomedical Informatics* 45 (2): 199–209. doi:10.1016/j.jbi.2011.10.002.
289. Rector, Al, Md Phd, R. Qamar Msc, and T. Marley Msc. 2006. "Binding Ontologies & Coding Systems to Electronic Health Records and Messages." In *Proc of the Second International Workshop on Formal Biomedical Knowledge Representation (KR-MED 2006); 2006; 2006. P. 11-9. The SAGE Guideline Model Page 30 of 35*, 11–19.
290. Riley, Gary. 2015. "CLIPS: A Tool for Building Expert Systems." December 12. http://clipsrules.sourceforge.net/.
291. Rinner, Christoph, Michael Kohler, Gudrun Hübner-Bloder, Samrend Saboor, Elske Ammenwerth, and Georg Duftschmid. 2011. "Creating ISO/EN 13606 Archetypes Based on Clinical Information Needs." In *Proceedings of EFMI Special Topic Conference" E-Health Across Borders Without Boundaries*, 14–15.
292. Rinner, C., T. Wrba, and G. Duftschmid. 2007. *Publishing Relational Medical Data as Cen 13606 Archetype Compliant Ehr Extracts Using Xml Technologies*. Citeseer.
293. Riva, Alberto, and Riccardo Bellazzi. 1996. "Learning Temporal Probabilistic Causal Models from Longitudinal Data." *Artificial Intelligence in Medicine* 8 (3): 217–34. doi:10.1016/0933-3657(95)00034-8.
294. Russell, Stuart, and Peter Norvig. 2002. *Artificial Intelligence: A Modern Approach*. 2nd ed. Prentice Hall.
295. Sacha, Jarosław P., Lucy S. Goodenday, and Krzysztof J. Cios. 2002. "Bayesian Learning for Cardiac SPECT Image Interpretation." *Artificial Intelligence in Medicine* 26 (1–2): 109–43. doi:10.1016/S0933-3657(02)00055-6.
296. Sachdeva, Shelly, Daigo Yaginuma, Wanming Chu, and Subhash Bhalla. 2011. "Dynamic Generation of Archetype-Based User Interfaces for Queries on Electronic Health Record Databases." In *Databases in Networked Information Systems*, edited by Shinji Kikuchi, Aastha Madaan, Shelly Sachdeva, and Subhash Bhalla, 109–25. Lecture Notes in Computer Science 7108. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-642-25731-5_10.

297. Sakellaropoulos, G.C, and G.C Nikiforidis. 2000. "Prognostic Performance of Two Expert Systems Based on Bayesian Belief Networks." *Decision Support Systems* 27 (4): 431–42. doi:10.1016/S0167-9236(99)00059-7.
298. Samwald, Matthias, Karsten Fehre, Jeroen de Bruin, and Klaus-Peter Adlassnig. 2012. "The Arden Syntax Standard for Clinical Decision Support: Experiences and Directions." *Journal of Biomedical Informatics*, Translating Standards into Practice: Experiences and Lessons Learned in Biomedicine and Health Care, 45 (4): 711–18. doi:10.1016/j.jbi.2012.02.001.
299. Saxena, Upaang, Shelly Sachdeva, and Shivani Batra. 2015. "Moving from Relational Data Storage to Decentralized Structured Storage System." In *Databases in Networked Information Systems*, edited by Wanming Chu, Shinji Kikuchi, and Subhash Bhalla, 180–94. Lecture Notes in Computer Science 8999. Springer International Publishing. http://link.springer.com/chapter/10.1007/978-3-319-16313-0_13.
300. Schadow, G., D. C Russler, and C. J McDonald. 2001. "Conceptual Alignment of Electronic Health Record Data with Guideline and Workflow Knowledge." *International Journal of Medical Informatics* 64 (2-3): 259–74.
301. Schloeffel, P., T. Beale, G. Hayworth, S. Heard, and H. Leslie. 2006. "The Relationship between CEN 13606, HL7, and openEHR." *HIC 2006 and HINZ 2006: Proceedings* 24.
302. Scholz, F. W. 2004. "Maximum Likelihood Estimation." In *Encyclopedia of Statistical Sciences*. John Wiley & Sons, Inc. http://onlinelibrary.wiley.com/doi/10.1002/0471667196.ess1571.pub2/abstract.
303. Schrempf, O. C, and U. D Hanebeck. 2004. "A New Approach for Hybrid Bayesian Networks Using Full Densities." In *Proceedings of 6th Workshop on Computer Science and Information Technologies, CSIT 2004*. Citeseer.
304. Scutari, M. 2010. "Bnlearn: Bayesian Network Structure Learning." *R Package*.
305. Scutari, Marco. 2009. "Learning Bayesian Networks with the Bnlearn R Package." *arXiv Preprint arXiv:0908.3817*.
306. Scutari, Marco. 2014. "Bayesian Network Constraint-Based Structure Learning Algorithms: Parallel and Optimised Implementations in the Bnlearn R Package." *arXiv Preprint arXiv:1406.7648*.
307. Shabot, M. M. 2004. "Ten Commandments for Implementing Clinical Information Systems." *Proceedings (Baylor University. Medical Center)* 17 (3): 265.
308. Shachter, Ross D. 1998. "Bayes-Ball: Rational Pastime (for Determining Irrelevance and Requisite Information in Belief Networks and Influence Diagrams)." In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 480–87. UAI'98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. http://dl.acm.org/citation.cfm?id=2074094.2074151.
309. Shanmugasundaram, Jayavel, Kristin Tufte, Chun Zhang, Gang He, David J. DeWitt, and Jeffrey F. Naughton. 1999. "Relational Databases for Querying XML Documents: Limitations and Opportunities." In *Proceedings of the 25th International Conference on Very Large Data Bases*, 302–14. VLDB '99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. http://dl.acm.org/citation.cfm?id=645925.671499.
310. Shenoy, P. P. 2006. "Inference in Hybrid Bayesian Networks Using Mixtures of Gaussians." In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, 428–36.
311. Shiffman, R. N. 1994. "Towards Effective Implementation of a Pediatric Asthma Guideline: Integration of Decision Support and Clinical Workflow Support." In *Proceedings of the Annual Symposium on Computer*

Application in Medical Care, 797. American Medical Informatics Association.
312. Shiffman, R. N, B. T Karras, A. Agrawal, R. Chen, L. Marenco, and S. Nath. 2000. "GEM: A Proposal for a More Comprehensive Guideline Document Model Using XML." *Journal of the American Medical Informatics Association* 7 (5): 488.
313. Shortliffe, Edward H., Randall Davis, Stanton G. Axline, Bruce G. Buchanan, C.Cordell Green, and Stanley N. Cohen. 1975. "Computer-Based Consultations in Clinical Therapeutics: Explanation and Rule Acquisition Capabilities of the MYCIN System." *Computers and Biomedical Research* 8 (4): 303–20. doi:10.1016/0010-4809(75)90009-9.
314. Shortliffe, E. H. 1987. "Computer Programs to Support Clinical Decision Making." *JAMA: The Journal of the American Medical Association* 258 (1): 61–66.
315. Shortliffe, E. H. 1993. "The Adolescence of AI in Medicine: Will the Field Come of Age in the'90s?" *Artificial Intelligence in Medicine* 5 (2): 93–106.
316. Shortliffe, E. H, and B. G Buchanan. 1975. "A Model of Inexact Reasoning in Medicine." *Mathematical Biosciences* 23 (3): 351–79.
317. Shortliffe, E. H, B. G Buchanan, and E. A Feigenbaum. 1979. "Knowledge Engineering for Medical Decision Making: A Review of Computer-Based Clinical Decision Aids." *Proceedings of the IEEE* 67 (9): 1207–24.
318. Shortliffe EH, Perreault LE, Wiederhold G, and Fagan LM. eds. 1990. *Medical Informatics. Computer Applications in Health Care. Reading, MA: Addison-Wesley Publishing Company*.
319. Sierra, Basilio, and Pedro Larrañaga. 1998. "Predicting Survival in Malignant Skin Melanoma Using Bayesian Networks Automatically Induced by Genetic Algorithms. An Empirical Comparison between Different Approaches." *Artificial Intelligence in Medicine* 14 (1–2): 215–30. doi:10.1016/S0933-3657(98)00024-4.
320. Sim, I., P. Gorman, R. A Greenes, R. B Haynes, B. Kaplan, H. Lehmann, and P. C Tang. 2001. "Clinical Decision Support Systems for the Practice of Evidence-Based Medicine." *Journal of the American Medical Informatics Association* 8 (6): 527–34.
321. Sloan, I. H. 2000. "Multiple Integration Is Intractable but Not Hopeless." *Journal of the Australian Mathematical Society-Series B* 42 (1): 3–8.
322. Smith, Wade P., Jason Doctor, Jürgen Meyer, Ira J. Kalet, and Mark H. Phillips. 2009. "A Decision Aid for Intensity-Modulated Radiation-Therapy Plan Selection in Prostate Cancer Based on a Prognostic Bayesian Network and a Markov Model." *Artificial Intelligence in Medicine* 46 (2): 119–30. doi:10.1016/j.artmed.2008.12.002.
323. Spiegelhalter, D. J, and R. P Knill-Jones. 1984. "Statistical and Knowledge-Based Approaches to Clinical Decision-Support Systems, with an Application in Gastroenterology." *Journal of the Royal Statistical Society. Series A (General)*, 35–77.
324. Spirtes, P., C. N Glymour, and R. Scheines. 2000. *Causation, Prediction, and Search*. The MIT Press.
325. Steinberg, D., F. Budinsky, E. Merks, and M. Paternostro. 2008. *EMF: Eclipse Modeling Framework*. Addison-Wesley Professional.
326. Stonebraker, Michael, and Ariel Weisberg. 2013. "The VoltDB Main Memory DBMS." *IEEE Data Eng. Bull.* 36 (2): 21–27.
327. Su, J., and H. Zhang. 2005. "Representing Conditional Independence Using Decision Trees." In *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, 20:874. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.
328. Sundvall, Erik, Mikael Nyström, Daniel Karlsson, Martin Eneling, Rong Chen, and Håkan Örman. 2013. "Applying Representational State Transfer (REST) Architecture to Archetype-Based Electronic Health

Record Systems." *BMC Medical Informatics and Decision Making* 13 (1): 57.
329. Tahraoui, Mohammed Amin, Karen Pinel-Sauvagnat, Cyril Laitang, Mohand Boughanem, Hamamache Kheddouci, and Lei Ning. 2013. "A Survey on Tree Matching and XML Retrieval." *Computer Science Review* 8 (May): 1–23. doi:10.1016/j.cosrev.2013.02.001.
330. Tao, Cui, Guoqian Jiang, Thomas A. Oniki, Robert R. Freimuth, Qian Zhu, Deepak Sharma, Jyotishman Pathak, Stanley M. Huff, and Christopher G. Chute. 2013. "A Semantic-Web Oriented Representation of the Clinical Element Model for Secondary Use of Electronic Health Records Data." *Journal of the American Medical Informatics Association* 20 (3): 554–62. doi:10.1136/amiajnl-2012-001326.
331. Tarczy-Hornoch, Peter, Laura Amendola, Samuel J. Aronson, Levi Garraway, Stacy Gray, Robert W. Grundmeier, Lucia A. Hindorff, et al. 2013. "A Survey of Informatics Approaches to Whole-Exome and Whole-Genome Clinical Reporting in the Electronic Health Record." *Genetics in Medicine* 15 (10): 824–32. doi:10.1038/gim.2013.120.
332. Tatarinov, Igor, Stratis D. Viglas, Kevin Beyer, Jayavel Shanmugasundaram, Eugene Shekita, and Chun Zhang. 2002. "Storing and Querying Ordered XML Using a Relational Database System." In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, 204–15. SIGMOD '02. New York, NY, USA: ACM. doi:10.1145/564691.564715.
333. Theodoratos, D., S. Souldatos, T. Dalamagas, P. Placek, and T. Sellis. 2006. "Heuristic Containment Check of Partial Tree-Pattern Queries in the Presence of Index Graphs." In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, 445–54. ACM.
334. Thusoo, Ashish, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, and Raghotham Murthy. 2009. "Hive: A Warehousing Solution over a Map-Reduce Framework." *Proc. VLDB Endow.* 2 (2): 1626–29. doi:10.14778/1687553.1687609.
335. Tolven Institute. 2015a. "Templates - Tolven.org." December 12. http://home.tolven.org/?page_id=149.
336. Tolven Institute. 2015b. "The Tolven Open Source Project." December 12. http://home.tolven.org/.
337. Tropashko, Vadim, and Donald Burleson. 2007. *SQL Design Patterns: Expert Guide to SQL Programming*. Rampant Techpress.
338. Tucker, Allan, Veronica Vinciotti, Xiaohui Liu, and David Garway-Heath. 2005. "A Spatio-Temporal Bayesian Network Classifier for Understanding Visual Field Deterioration." *Artificial Intelligence in Medicine* 34 (2): 163–77. doi:10.1016/j.artmed.2004.07.004.
339. van der Gaag, L.C., S. Renooij, C.L.M. Witteman, B.M.P. Aleman, and B.G. Taal. 2002. "Probabilities for a Probabilistic Network: A Case Study in Oesophageal Cancer." *Artificial Intelligence in Medicine* 25 (2): 123–48. doi:10.1016/S0933-3657(02)00012-X.
340. Van Rossum, Guido. 2007. "Python Programming Language." In *USENIX Annual Technical Conference*. Vol. 41.
341. Velte, Linda, Tiago Pedrosa, Carlos Costa, and José Luís Oliveira. 2012. "An OpenEHR Repository Based on a Native XML Database." http://bibliotecadigital.ipb.pt/handle/10198/8804.
342. Verduijn, Marion, Peter M.J. Rosseel, Niels Peek, Evert de Jonge, and Bas A.J.M. de Mol. 2007. "Prognostic Bayesian Networks: II: An Application in the Domain of Cardiac Surgery." *Journal of Biomedical Informatics* 40 (6): 619–30. doi:10.1016/j.jbi.2007.07.004.
343. Vogels, Werner. 2009. "Eventually Consistent." *Commun. ACM* 52 (1): 40–44. doi:10.1145/1435417.1435432.

344. Wabersich, Dominik, and Joachim Vandekerckhove. 2013. "Extending JAGS: A Tutorial on Adding Custom Distributions to JAGS (with a Diffusion Model Example)." *Behavior Research Methods* 46 (1): 15–28. doi:10.3758/s13428-013-0369-3.
345. Wang, Haixun, and Xiaofeng Meng. 2005. "On the Sequencing of Tree Structures for XML Indexing." In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, 372–83. IEEE.
346. Wang, Haixun, Sanghyun Park, Wei Fan, and Philip S. Yu. 2003. "ViST: A Dynamic Index Method for Querying XML Data by Tree Structures." In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, 110–21. SIGMOD '03. New York, NY, USA: ACM. doi:10.1145/872757.872774.
347. Wang, Xiao-Hui, Bin Zheng, Walter F. Good, Jill L. King, and Yuan-Hsiang Chang. 1999. "Computer-Assisted Diagnosis of Breast Cancer Using a Data-Driven Bayesian Belief Network." *International Journal of Medical Informatics* 54 (2): 115–26. doi:10.1016/S1386-5056(98)00174-9.
348. Waters, Evan, Jeff Rafter, Gerald P. Douglas, Mwatha Bwanali, Darius Jazayeri, and H. S. Fraser. 2010. "Experience Implementing a Point-of-Care Electronic Medical Record System for Primary Care in Malawi." *Stud Health Technol Inform* 160 (Pt 1): 96–100.
349. Weigel, Felix, Holger Meuss, François Bry, and Klaus U. Schulz. 2003. *Content-Aware DataGuides: Interleaving IR and DB Indexing Techniques for Efficient Retrieval of Textual XML Data*.
350. Weigel, Felix, Klaus U. Schulz, and Holger Meuss. 2005. "Exploiting Native XML Indexing Techniques for XML Retrieval in Relational Database Systems." In *Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management*, 23–30. WIDM '05. New York, NY, USA: ACM. doi:10.1145/1097047.1097054.
351. Welch, Brandon M., Salvador Rodriguez-Loya, Karen Eilbeck, and Kensaku Kawamoto. 2014. "Clinical Decision Support for Whole Genome Sequence Information Leveraging a Service-Oriented Architecture: A Prototype." *AMIA Annual Symposium Proceedings* 2014 (November): 1188–97.
352. WHO. 2015. "WHO | International Classification of Diseases (ICD) Information Sheet." *WHO*. Accessed August 6. http://www.who.int/classifications/icd/factsheet/en/.
353. Wollersheim, Dennis, Anny Sari, and Wenny Rahayu. 2009. "Archetype-Based Electronic Health Records: A Literature Review and Evaluation of Their Applicability to Health Data Interoperability and Access." *Health Information Management Journal* 38 (2): 7.
354. Wood, Lauren, Arnaud Le Hors, Vidur Apparao, Steve Byrne, Mike Champion, Scott Isaacs, Ian Jacobs, Gavin Nicol, Jonathan Robie, and Robert Sutor. 1998. "Document Object Model (DOM) Level 1 Specification." *W3C Recommendation* 1.
355. World Health Organization. 1992. *ICD-10: International Statistical Classification of Diseases and Related Health Problems: 10th Revision*. World Health Organization.
356. Wright, Adam, Dean F. Sittig, Joan S. Ash, Jessica L. Erickson, Trang T. Hickman, Marilyn Paterno, Eric Gebhardt, et al. 2015. "Lessons Learned from Implementing Service-Oriented Clinical Decision Support at Four Sites: A Qualitative Study." *International Journal of Medical Informatics* 84 (11): 901–11. doi:10.1016/j.ijmedinf.2015.08.008.
357. Wu, Huayu, Ruiming Tang, Tok Wang Ling, Yong Zeng, and Stéphane Bressan. 2012. "A Hybrid Approach for General XML Query Processing." In *Database and Expert Systems Applications*, edited by Stephen W. Liddle, Klaus-Dieter Schewe, A. Min Tjoa, and Xiaofang Zhou, 10–25.

Lecture Notes in Computer Science 7446. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-642-32600-4_3.
358. Wu, Xiao-Lin, Chuanyu Sun, Timothy M Beissinger, Guilherme JM Rosa, Kent A Weigel, Natalia de Leon Gatti, and Daniel Gianola. 2012. "Parallel Markov Chain Monte Carlo - Bridging the Gap to High-Performance Bayesian Computation in Animal Breeding and Genetics." *Genetics, Selection, Evolution : GSE* 44 (1): 29. doi:10.1186/1297-9686-44-29.
359. Wu, Xiaoying, S. Souldatos, D. Theodoratos, T. Dalamagas, Y. Vassiliou, and T. Sellis. Dec. "Processing and Evaluating Partial Tree Pattern Queries on XML Data." *IEEE Transactions on Knowledge and Data Engineering* 24 (12): 2244–59. doi:10.1109/TKDE.2011.137.
360. Wu, X., S. Souldatos, D. Theodoratos, T. Dalamagas, Y. Vassiliou, and T. Sellis. 2011. "Processing and Evaluating Partial Tree Pattern Queries on XML Data." *IEEE Transactions on Knowledge and Data Engineering* PP (99): 1. doi:10.1109/TKDE.2011.137.
361. Xiang, Yang, B. Pant, A. Eisen, M.P. Beddoes, and D. Poole. 1993. "Multiply Sectioned Bayesian Networks for Neuromuscular Diagnosis." *Artificial Intelligence in Medicine* 5 (4): 293–314. doi:10.1016/0933-3657(93)90019-Y.
362. Xia, Yinglong, and Viktor K. Prasanna. 2008. "Junction Tree Decomposition for Parallel Exact Inference." In *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, 1–12. IEEE.
363. Xia, Yinglong, and V.K. Prasanna. 2007. "Node Level Primitives for Parallel Exact Inference." In *19th International Symposium on Computer Architecture and High Performance Computing, 2007. SBAC-PAD 2007*, 221–28. doi:10.1109/SBAC-PAD.2007.18.
364. Yoshikawa, Masatoshi, Toshiyuki Amagasa, Takeyuki Shimura, and Shunsuke Uemura. 2001. "XRel: A Path-Based Approach to Storage and Retrieval of XML Documents Using Relational Databases." *ACM Transactions on Internet Technology* 1 (1): 110–41.
365. Yuan, C., and M. J Druzdzel. 2003. "An Importance Sampling Algorithm Based on Evidence Pre-Propagation." In *Proceedings of the 19th Annual Conference on Uncertainty on Artificial Intelligence*, 624–31.
366. Yu, Bo, and Duminda Wijesekera. 2013. "Building Dialysis Workflows into EMRs." *Procedia Technology*, CENTERIS 2013 - Conference on ENTERprise Information Systems / ProjMAN 2013 - International Conference on Project MANagement/ HCIST 2013 - International Conference on Health and Social Care Information Systems and Technologies, 9: 985–95. doi:10.1016/j.protcy.2013.12.110.
367. Zaharia, Matei, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2010. "Spark: Cluster Computing with Working Sets." In *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*, 10–10.
368. Zanstra, P.E, A.L Rector, W Ceusters, and P.F de Vries Robbé. 1998. "Coding Systems and Classifications in Healthcare: The Link to the Record." *International Journal of Medical Informatics* 48 (1–3): 103–9. doi:10.1016/S1386-5056(97)00115-9.
369. Zeng, Qiang, Xiaorui Jiang, and Hai Zhuge. 2011. "Adding Logical Operators to Tree Pattern Queries on Graph-Structured Data." *arXiv:1109.4288*, September. http://arxiv.org/abs/1109.4288.
370. Zhang, C., J. Naughton, D. DeWitt, Q. Luo, and G. Lohman. 2001. "On Supporting Containment Queries in Relational Database Management Systems." In *Acm Sigmod Record*, 30:425–36. ACM.