University of Minho
School of Engineering

Miguel André Rocha Dias

A novel system for managing OpenEHR structures

may 2023

University of Minho
School of Engineering

Miguel André Rocha Dias

A novel system for managing OpenEHR structures

Masters Dissertation Master's Degree in Bioinformatics
Master's in Bioinformatics

Dissertation supervised by
Professor José Manuel Ferreira Machado

may 2023

Copyright and Terms of Use for Third Party Work

License granted to users of this work:

Acknowledgements

I would like to start out by acknowledging my tutor, José Machado, for allowing me to carry out this dissertation under his guidance. I feel extremely privileged to have had the opportunity to learn from him and his feedback was crucial in putting the final touches of this work.

Secondly I'm thanking Daniela Oliveira, my co-tutor, for her dedicated help with all the programming work carried out. She always made herself readily available to answer any questions and give helpful feedback related to the research and writing of this dissertation. Without her help this project wouldn't be possible.

To my friends, Samuel, José and Gil, for their unconditional support and encouragement throughout all the ups and down while making this dissertation.

Lastly my family, namely my mom, dad, and sister for their constant love and support, as well as for their patience and understanding during the long hours spent working on this dissertation. Would also like to make a special dedication to both of my grandmothers who have sadly passed away recently. My memories of them will be cherished for many years to come.

To everyone who has contributed to this work in big and small ways, I offer my most sincere gratitude. Your support and encouragement have been instrumental in making this dissertation possible and It couldn't have ever been possible without your help.

Statement of Integrity

I hereby declare having conducted this academic work with integrity.

I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

University of Minho, Braga, may 2023

Miguel André Rocha Dias

Assinado por: Miguel André Rocha Dias
Identificação: BI14690110
Data: 2023-05-25 às 23:38:18

Abstract

The term e-Health has been increasingly used in research projects in the health industry. This concept encompasses the development and application of software and hardware solutions for the collection, storage, manipulation, and communication of data in an efficient way, with an objective of continuously improving the provision of health care.

The development of Health Information Systems has increasingly lacked the establishment of high levels of interoperability in its semantic, syntactic, and technical aspects. The development of systems that allow promoting interoperability between HIS within the same institution or even between HIS from different institutions is on the global agenda as a common concern for all countries.

The use of globally recognized standards has been increasingly common, ensuring that both the information structure and its meaning remain intact, regardless of the flow they follow. Thus arises the motivation to develop a system that contributes to the continuous improvement of interoperability in health, through the use of the openEHR standard.

The following dissertation presents a novel way to handle clinical data by creating an artifact enabling the conversion of openEHR standardized data into a JSON object. The web application showcases a new way for a user to check for openEHR data while the API can be utilized by developers to work with openEHR data in a more accessible and supported manner with other programming tools. To carry out the work a thorough examination of web development tools to build both the backend and frontend of the app was essential, as well as coming up with the most accurate regex expressions that are able to extract data from openEHR files. The research and engineering effort put through the project was successful in showcasing this novel approach implementing yet another tool to help out healthcare professionals and biomedical software engineers.

Keywords: openEHR, e-Health, Interoperability, Health Information Systems, JSON, API

Resumo

O termo *e-Health* tem sido cada vez mais utilizado em projetos de pesquisa na área da saúde. Este conceito engloba o desenvolvimento e aplicação de soluções de *software* e *hardware* para recolha, armazenamento, manipulação e comunicação de dados de forma eficiente, com o objetivo de melhorar continuamente a prestação de cuidados de saúde.

O desenvolvimento dos Sistemas de Informação em Saúde (SIS) tem falta de estabelecimento de elevados níveis de interoperabilidade nas suas vertentes semântica, sintática e técnica. O desenvolvimento de sistemas que permitam promover a interoperabilidade entre SIS dentro da mesma instituição ou mesmo entre SIS de diferentes instituições é uma preocupação comum a todos os países.

A utilização de padrões mundialmente reconhecidos tem sido cada vez mais comum, garantindo que tanto a estrutura da informação assim como o seu significado permaneçam intactos, independentemente do fluxo que seguem. Surge assim a motivação para desenvolver um sistema que contribua para a melhoria contínua da interoperabilidade na saúde, através da utilização da norma *openEHR*.

Esta dissertação apresenta uma nova forma de tratar dados clínicos criando um artefato que permite a conversão de dados padronizados *openEHR* em um objeto JSON. A aplicação *Web* apresenta uma nova maneira de um usuário verificar dados *openEHR* enquanto a API pode ser utilizada por desenvolvedores para trabalhar com dados *openEHR* de uma maneira mais acessível e compatível com outras ferramentas de programação. Para realizar o trabalho, foi essencial uma análise das ferramentas de desenvolvimento *web* para construir o *backend* e o *frontend* do aplicativo, além de encontrar as expressões *regex* mais precisas que são capazes de extrair dados de arquivos *openEHR*. O esforço de pesquisa e engenharia realizado no projeto foi bem-sucedido em mostrar esta nova abordagem, implementando mais uma ferramenta para ajudar profissionais de saúde e engenheiros de *software* biomédico.

Palavras-chave: openEHR, e-Health, Interoperabilidade, Sistemas de Informação em Saúde, JSON, API

Contents

List of Figures

List of Tables

Acronyms

ADL Archetype Definition Language.

API Application Programming Interface.

cADL Constrain Archetype Definition Language.

CDS Clinical Data sets.

CIRP Clinical Investigator Recording Process.

CKM Clinical Knowledge Manager.

CT Computerized Tomography.

dADL Data Archetype Definition Language.

DICOM Digital Imaging and Communications in Medicine.

DSR Design Science Research.

EHR Electronic Health Record.

HIS Hospital Information Systems.

HL7 Health Level 7.

HTTP Hypertext Transfer Protocol.

IHTSDO International Health Terminology Standards Development Organisation.

JSON JavaScript Object Notation.

MRI Magnetic Resonance Imaging.

PACS Picture Archiving and Communication System.

REST Representational state transfer.

RM Reference Model.

SNOMED Systematized Nomenclature of Medicine Clinical Terms.

XML Extensible Markup Language.

Part I

Introductory material

Chapter 1

Introduction

The research and engineering work covered in this paper was done for a dissertation final year project under the name "A novel system for managing OpenEHR structures" as a part of the University of Minho's Bioinformatics Masters's degree. This project consists of a web application built using the front end library React that connects to a GitHub repository to manage OpenEHR related files while parsing said files to valid JSON structures, essentially creating a working API using OpenEHR files. The work was carried out in the Algoritmi Research Center, a research division of the School of Engineering at the University of Minho. This introductory chapter includes a description of the project's scope and context (Section 1.1), an explanation of its motivation (Section 1.2), a list of goals that must be achieved in order for it to be completed (Section 1.3), and a description of the document structure (Section 1.4).

1.1 Context and Scope

Nowadays, the use of information technologies (IT) is critical in a wide range of industries, particularly the health care industry. The need for ongoing patient care improvement is evident in the health space, along with this requirement, clinical decision-making must be effective and efficient or there can be serious issues Oliveira et al. [2018]. For this to be achievable, an increasing number of new auxiliary instruments have appeared to help carry out clinical actions with a higher degree of success. Assuring the patient's safety and well being is the first step in providing high-quality healthcare and, subsequently, effective outcomes to treat said patient. All healthcare personnel have a responsibility to ensure the patient's safety, but given their constantly evolving and chaotic environment in the day-to-day life of their hospital work carrying out said responsibility proves to be a very difficult task Oliveira et al. [2020].

To accommodate the dynamic changes in data requirements, a framework that is open, semantically shareable, and collaboratively modelled is required. As a multilevel modeling framework, openEHR specifications can be used to develop standards and build information and interoperability solutions for the healthcare industry Lamine et al. [2017]. In the method recommended by openEHR, the reference model (RM) concentrates on the properties and logic structures needed to describe data, making it stable and providing the fundamental building blocks for creating actual medical information models. Templates and archetypes make up the archetypal model. Archetypes can be created based on the RM to define all the characteristics of particular therapeutic notions. It is possible to group several archetypes into context-specific data sets, or templates, which are typically created and applied locally.

1.2 Motivation

With this project, the intention is to appeal to the reuse of openEHR standardized data, and consequently to their interpretation between systems in a direct and efficient way. This project appeals to help the ever emerging interoperability problem by suggesting a new more efficient way to collect and distribute electronic health records and the showcasing of a prototype application to aid in the observation of said data. The developed artefact should have it's focal point the forward improvement of health data management for health care professionals and the contribution of a new tool for IT developers in the biomedical space hoping to improve on the distribution of healthcare data. This initial artefact can be used as an entry point and be further improved or updated later on while remaining consistent with the standards defined by openEHR systems.

1.3 Objectives

The main goal for this master's thesis is to create a novel HIS artifact that helps both healthcare professionals as well as developers at managing openEHR records. The research question for a project of this magnitude could be defined as:

"Will this new approach provide a solution for the significant challenges faced by professionals in the medical industry with the increasing complexity of propagation of healthcare record data?"

To demonstrate how the new clinical system can be readily and swiftly constructed in a secure and transparent manner multiple objectives need to be accomplished and the final developed artefact needs to be critically analysed in order to deduct if it trully is fit to solve the research question. As such the objectives required to achieve throughout the development of this thesis are as follows:

1. A research effort over the main topics of health data on answering relevant questions for the work

such as the importance of data standards in healthcare, what is interoperability in health data and how openEHR functions.

2. Identification of the most suitable research methods and technologies to be used to develop the artefact. In order to bring the Healthcare Information System artefact to life there's a need to explore a range of research methodologies that are suited for the development of the artefact, as well as identifying the specific technologies that will enable its creation.

3. Artefact development. A description of the steps taken to achieve the artefact plus any challenges encountered or measures taken in order to build the best version of the artefact.

4. An artefact critical evaluation going over the aspects that it actually improved over previous work done in the area as well as any limitations that can be improved in the future.

The built artefact comes in the form of software with the functionality to connect to an existing repository storing openEHR related files (Archetypes and Template files) with a way to parse them into a more reliable and supported data structure, the JSON format. A graphical interface for the user to manage these newly converted files is also one of the main features developed.

1.4 Dissertation Organization

The following dissertation is divided into two parts. Part I, comprises of the introduction, a research methodologies used section a state of the art going over background and important theoretical concepts of the openEHR system and finally a materials and methods section going over any tools used to make this project. Part II is the core of the dissertation that goes over the entire development process ending off with a conclusions and future work discussion over the buit artefact.

Chapter 2

Research methodologies

The development of new technologies is usually accompanied by an extensive research effort in order to determine efficient methods or tools for a project to come to fruition.

The starting point of this project was the identification of the problem that translates into the high dissemination of relevant clinical information, of each patient, upon admission, through different systems and in different forms. A critical analysis of similar past projects in the subject area of health data was performed namely the extensive reading of engineering journal reports, articles, thesis and dissertation work done in biomedical health IT. This initial research was crucial in fully understanding the problem of interoperability and implementing a new solution. Later, the implementation of the solution required an investigation of the most suitable programming tools and techniques for the job. This process required extensive reading of documentation and trial and error during it's implementation in the design of the artefact.

On account of this, the following chapter exemplifies and describes proven research methods that were used to manage the case study defined and later design a solution for the proposed investigation question.

2.1 Design Science Research

The DSR (Design Science Research) methodology was selected as the optimal approach for addressing the problem in the case study. This methodology involves the creation and evaluation of practical solutions or artefacts within a real-world context. It is commonly used in IT product development, offering a structured framework for addressing specific issues and implementing targeted actions.

In a DSR project aimed at developing a new health application, the artefact could be the actual application itself, along with any accompanying design documents, user guides, and usability testing results. The application is the tangible output of the design process and is intended to address a specific health-related problem or need. The artefact is evaluated based on its effectiveness in addressing the identified problem and meeting the needs of the intended user group Weigand et al. [2021].

The methodology was chosen because it aligned well with the case study, which required a deep understanding of the problem before proposing a supported and justified solution. With this initial awareness, development could proceed confidently, knowing that the proposed solution was feasible. After obtaining the first version of the artefact, it must be evaluated and reviewed by users to ensure it meets the initial requirements and identify areas for improvement and future goals. The DSR methodology is acknowledged as a cyclical process, initiated by problem identification and case study formulation by the Designer, followed by solution prediction based on research and investigation.

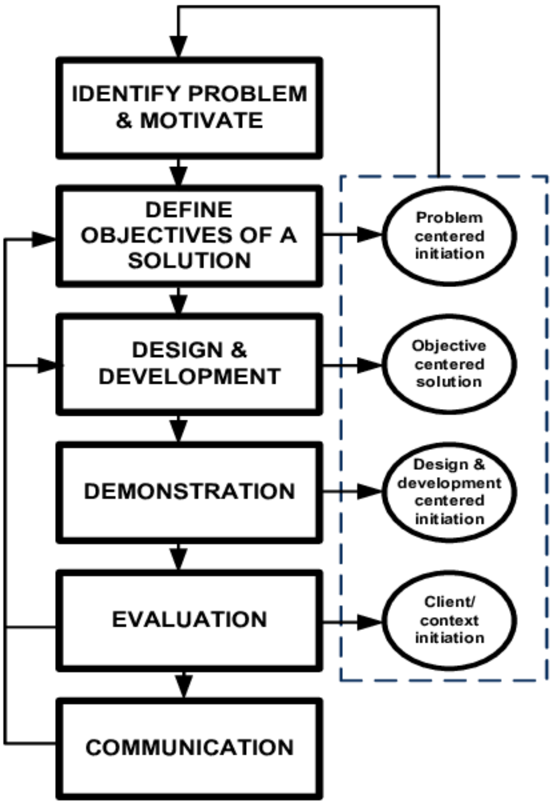

Figure 1: An overview of the Design Science methodology taken from Azasoo and Boateng [2015].

The main investigation question was first identifying the challenge concerning the dispersion of crucial clinical information of patients across various systems. The aim was to create a solution that facilitates swift and intuitive access to necessary information for healthcare professionals as well as developers. Accounting for that question, an artefact was built which consists of a prototype of an automatic platform devised to present updated electronic record data information stored in a database, by converting the data into a new more flexible and easily accessible format. Following implementation, the prototype's impact will be assessed by users, and adjustments will be made as necessary.

2.2 Waterfall Development Methodology

Software development encompasses diverse methodologies that developers can utilize to conceptualize, plan, create, and test software. Selecting the optimal methodology for a specific need and project requirements is vital. Being well-versed in the prevalent software development methodologies is essential for software practitioners to make informed choices on the most appropriate one to implement.

A reliable approach for software engineering is the Waterfall method. As it's name might imply it's a linear and sequential approach that is dependant on each phase being fully planed out and developed before moving on to the next phase. Any requirements or documentation are gathered upfront, with the design and development stages following in a sequentially manner, with no overlap or parallel work. Finally, testing and deployment are carried out before the final software is delivered to the end users Adenowo and Adenowo [2013].

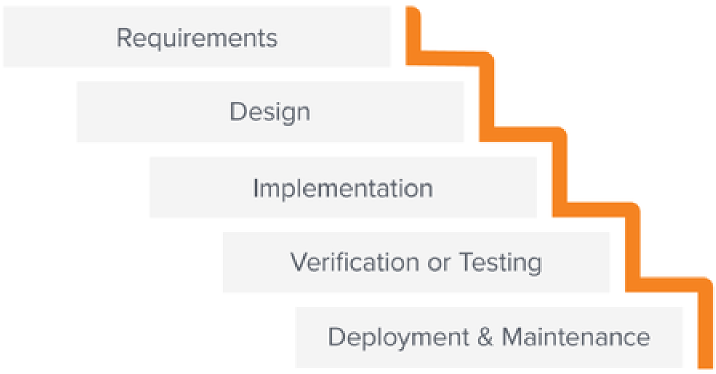

Figure 2: The general steps of Waterfall Development Methodology in software engineering taken from Adobe Communications Team [2022].

Each step can be summarized as follows:

- Requirements: Initial planning phase where information is compiled and requirements for the software are identified and documented. This involves gathering inputs from the client, understanding their needs, and defining the functional and non-functional requirements of the software.
- Design: The software architecture, system design, and detailed design are created in this phase. This includes defining the overall structure of the software, designing interfaces, data models, and algorithms.

- Implementation: Phase where the code writing of the software takes place. Software engineers write the code according to the design specifications defined previously. During this stage, software engineers convert the design into executable software, ultimately creating the final product.

- Verification or Testing: After the development phase is finished, the software undergoes deep testing to detect and fix programming bugs. This stage encompasses unit testing, integration testing, system testing, and acceptance testing, to verify that the software performs as intended and fulfills the specified requirements.

- Deployment & Maintenance: Once the software has passed the testing phase with success, it is deployed to the production environment, making it accessible to end users. Any new problem that may arise from the software is then taken care off by a team of developers that fix it by releasing newer versions of the software.

Generally this approach to software development might feel more restrictive than others. The step by step approach works for the scope of this project since it isn't being developed by a large team of software engineers nor doesn't have a client whose input might change the development after the developed artefact. All the design choices were developed based off the initial given input and for the most part have remained unchanged. In the future, assuming the implementation of this novel way to represent data is more widespread, a different methodology could be implemented to account for the limitations of the Waterfall method.

Chapter 3

State of the art

This chapter starts out with a brief history of how Hospital Information Systems emerged and the creation of Electronic Health Records due to an increasing need to organize information. Subsequently, the state of the art delves into the openEHR foundation, exploring key concepts needed for the development of the work in this dissertation namely the general architecture of archetypes and reference models and the file syntax of archetypes. This section also provides an in-depth exploration of how web applications work.

3.1 Hospital information systems

The amount of clinical data has increased significantly in recent years, and the need for more organized and automated information management has in consequence become increasingly important. Nowadays there's the use of computerized systems designed to manage and integrate all of the administrative, financial, and clinical functions of a hospital. All this is being done by collecting, organizing, and analyzing clinical data in an efficient manner, allowing hospitals to easily access and utilize this information as needed Islam et al. [2018].

The use of technology in conjunction with medical assistance is not a recent phenomenon however. In the 1960s the first forms of telemedicine, a term used to describe the delivery of medical services and knowledge via electronic communication and information technology, were implemented in a Nebraska Psychiatric Institute Wittson and Benschoter [1972]. Due to the internet still being in its infancy telemedicine at the time was primarily supported through video broadcasting and telephone Jagarapu and Savani [2021].

The 1990s saw a huge growth in telehealth. In a co-funded project by the European Commission, numerous large-scale initiatives and supporting policies were put in place to encourage the development of e-health applications across Europe, including remote imaging, teleconsultation, e-learning, surgery-hospital communication, telesurgery, virtual staff meetings, emergency telemedicine, and more Beuscart et al. [2014]. Many initiatives were created by scientists, medical experts, and institutions, but only a small number of them have persisted to this day. This decade also saw the development of electronic health records systems as well as the national and international health data standards that facilitated the interoperability of said electronic records, allowing for the sharing of health information between different systems. Electronic health records EHR are essentially digital versions of a patient's medical history, including their medical conditions, medications, and test results. These records are created and maintained by healthcare providers and can be accessed by authorized healthcare professionals to provide better, more coordinated care for patients. This surge in popularity for e-health eventually also led to the development of more sophisticated EHR systems, which were able to support the management of larger and more complex datasets.

EHRs brought several advantages over traditional paper-based health records. Some of the key benefits of EHRs include:

- Improved accuracy and completeness of health information: Because EHRs are digital, they can support the capture of more detailed and accurate health information than what is possible with paper records. For example, EHRs can automatically populate fields with standardized data, and can support the use of data validation rules to ensure the accuracy of the information entered Chan et al. [2010]. This can help to reduce errors and omissions in the health record, and improve the quality of care.

- Easier access and sharing of health information: EHRs can be accessed and shared more easily than paper records, which can facilitate care coordination among different healthcare providers Staroselsky et al. [2006]. They can be accessed remotely by authorized users, and can support the sharing of clinical information through secure networks or web-based portals. This can help to support more effective communication and decision-making among healthcare teams, and can improve the availability of health information to support patient care.

- Increased efficiency and productivity: EHRs can support the automation of many administrative and clinical tasks, which can help to reduce the time and effort required to manage health records Atreja et al. [2008]. For example, EHRs can support the use of decision support tools and clinical guidelines, which can help to streamline the care process and reduce the time and effort required to document and manage patient information Romano and Stafford [2011]. This approach can enhance the efficiency of healthcare organizations.

- Better support for population health management: EHRs can support the aggregation and analysis of health data from multiple patients and sources, which can help to identify trends, patterns, and opportunities for improvement in population health. For example, EHRs can support the identification of high-risk patients or populations, and can help to facilitate the implementation of targeted interventions to improve health outcomes Corey et al. [2018].

- Enhanced patient engagement and participation: EHRs can support the involvement of patients in their own care, and can provide patients with access to their own health information Caligtan and Dykes [2011]. For example, EHRs can support the use of patient portals, which can allow patients to view their health records, communicate with their providers, and manage their own health information. This can help to improve patient engagement and participation in their care, and can support the development of more collaborative and patient-centered care models.

- Improved public health surveillance and emergency preparedness: EHRs can support the real-time monitoring and tracking of health data, which can help to identify potential outbreaks or public health threats. For example, EHRs can support the rapid identification and notification of cases of infectious diseases, and can help to facilitate the implementation of appropriate public health interventions Birkhead et al. [2015]. This can improve public health surveillance and emergency preparedness, and can help to protect the health of the population.

- Reduced risk of medical errors and adverse events: EHRs can support the use of clinical decision support tools and alerts, which can help to prevent medical errors and adverse events. For example, EHRs can alert providers to potential drug interactions or contraindications, and can support the implementation of evidence-based best practices and guidelines Kuperman et al. [2007]. This can help to reduce the risk of medical errors and adverse events, and can improve the safety and quality of care.

- Improved patient outcomes and satisfaction: EHRs can support the delivery of more effective and efficient care, which can lead to improved patient outcomes and satisfaction. For example, EHRs can support the use of evidence-based care protocols and decision support tools, which can help to optimize treatment and management plans Moja et al. [2014]. This can help to improve patient outcomes, and can enhance patient satisfaction with the care they receive.

- Enhanced research and data analytics capabilities: EHRs can support the collection and analysis of large and complex datasets, which can help to advance research and support data-driven decision-making Luo et al. [2016]. For example, EHRs can support the identification of trends, patterns, and associations in health data, and can facilitate the development of new knowledge and insights. This can enhance the research and data analytics capabilities of healthcare organizations, and can support the development of more effective and personalized care approaches.

This sizable number of advantages over traditional paper-based health records has proven the value that EHRs offer in clinical care which is why they are increasingly being used in healthcare settings to support the delivery of high-quality, efficient, and patient-centered care.

The first ten years of the new millennium saw further advancements of e-health. The implementation of incentives and penalties related to the use of EHR by the US government, through programs such as the Meaningful Use initiative Sherer et al. [2016], helped to accelerate the adoption of EHR by healthcare organizations. This led to a significant increase in the number of EHR systems in use, and the development of a range of new and innovative EHR technologies and approaches. Today, some applications are utilized often, including encrypted texting, emergency telemedicine, remote imaging, electronic health records, and interhospital communication.

The availability of EHRs solutions continues to evolve with further developments that try to improve interoperability and standardization of data. New ways to represent data more efficiently that integrate machine learning or artificial intelligence are being developed to support the sharing of health data Yu et al. [2018]. The history of EHRs is one of rapid evolution and growth, and the future of EHRs is likely to be even more efficient.

3.2 Interoperability

Medical information may often be displayed differently depending on the type of data stored. As a result, health professionals may face difficulties in comprehending the patient's status. To ensure smooth operations, prevent communication breakdowns, and uphold consistency, it is essential to achieve seamless communication between multiple HIS. Hence the concept of interoperability is crucial to achieve this. This involves the ability of different systems, devices, or applications to communicate and share information efficiently.

A general lack of interoperability can cause major inefficiencies in a hospital environment. There can be patients that may end up receiving care from multiple providers and each provider may have their own EHR system. If these systems are not interoperable, they can not exchange clinical information about the patient efficiently, and consequently, supporting the advancement of a comprehensive understanding of the patient's overall health proves to be a more challenging task. An efficient interoperable system however not only helps improve the quality and effectiveness of data sharing, it also supports more effective decision-making and care planning. The concept of interoperability in HIS can also play a crucial role in enabling the consolidation and examination of health data on a broader scope. This capability assists in identifying trends, patterns, and areas for enhancing population health, thereby supporting the creation of more precise and impactful interventions Kukafka et al. [2007].

Three primary categories of interoperability are typically recognized: semantic interoperability, syntactic interoperability, and structural interoperability.

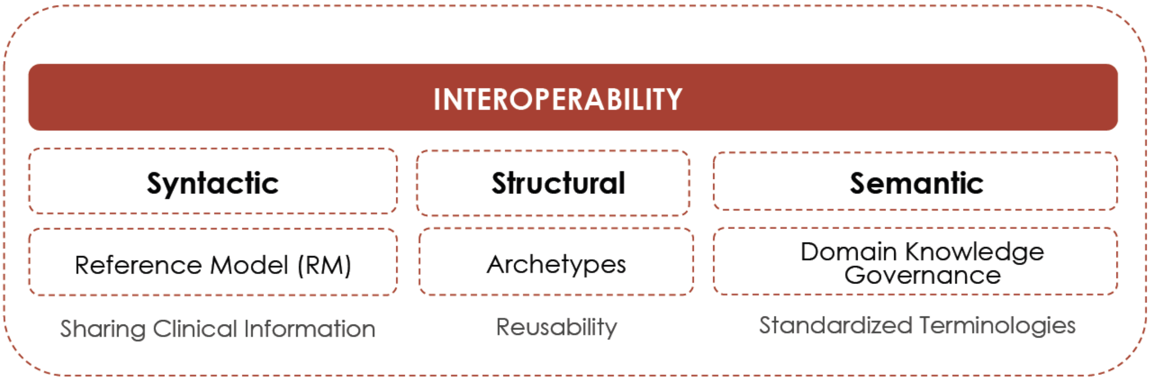

Figure 3: The three main types of interoperability and their association with EHRs.

- **Semantic interoperability**: The ability of different systems to interpret, understand and share the meaning of the data that is exchanged between them Heiler [1995]. Systems must use a common set of concepts, definitions, and terminologies to represent clinical data and be able to map and transform data between different representation formats.

- **Syntactic interoperability**:: The exchange of data in a consistent and structured format. In this case, systems should use a common set of rules and conventions for representing and transmitting data and must be able to validate and interpret the data that is exchanged between them Hosseini and Dixon [2016].

- **Structural interoperability**: The process of integration and linking data from multiple sources. Interoperable systems should have the ability to collect and analyze data from multiple patients and sources which may include different HIS Delgado [2013].

3.3 Healthcare data standards

Healthcare data standards are a set of guidelines and specifications that came to existence due to agreements between various international organizations, in an effort to have a single standardized way to define health records across various different systems, organizations, and individuals. This is used to ensure that health-related information is accurately and consistently collected, stored, and shared. These standards enable interoperability, which as previously said is the ability of different systems and devices to work together seamlessly allowing patient information to be easily shared among healthcare providers.

Various organizations have dedicated to developing international frameworks and standards for electronic health information, these include specifications about EHRs, terminologies, and models that support the exchange, integration, and retrieval of information. Achieving semantic interoperability, the ability for different systems to understand and accurately interpret the meaning of the data being shared is considered the most challenging level to achieve Intely [2022]. The European Commission (EC) suggests that semantic interoperability can only be attained through agreements on standards, information models, terminologies, and semantic definitions for data sharing. Factors such as cultural, legal, and social considerations within organizations, regions, or countries may also impact the implementation of semantically interoperable systems Guijarro [2009].

For this reason, it is suggested that this high level of semantic interoperability should be initially implemented and practiced only in specific, high-priority clinical areas with significant relevance to patient safety.

One of the earliest examples of healthcare data standards is the Health Level Seven International HL7 standard but there many more examples of data standards created for medicine like the DICOM standard, SNOMED or the OpenEHR standard, all of which will be described in the followings sections.

3.3.1 HL7

Health level 7 is a US-based organization founded in 1987 whose main objective is in creating data standards in messaging as well as electronic records for HIS Rene Spronk [2014]. This is done by producing electronic records whose document structure supports system interoperability and emphases various important services in hospital care like administration and clinical practice. The HL7 standards have come a long way starting with the initial development in 1987. Version 2 of HL7 came out in 1989 and was made to create interoperability in data exchange between hospital systems however the major downside of this version is the lack of scalability in integrating other information systems not directly related to healthcare like jurisdictional information systems Goossen and Langford [2014]. Another limitation of this version is not having an organized system for understanding and defining the shared concepts used in various communication and interaction methods. Version 3 started development in 1995 and one of the improvements was the creation of an ontology section that was lacking in the previous version as well as completely remodeling the entire document structure by using Extensible Markup Language (XML) syntax, based on a new specification framework developed called the Reference Information Model (RIM) Goossen and Langford [2014].

Figure 4: An example of a HL7 V2 (blue square) file and a HL7 V3 file (red square).

A new version of HL7 called Fast Health Interoperability Resources (FHIR) that implements a RESTful API approach was eventually released. This version supports both XML and JSON which makes it easier to implement server-wide communications and create web applications Hussain et al. [2018].

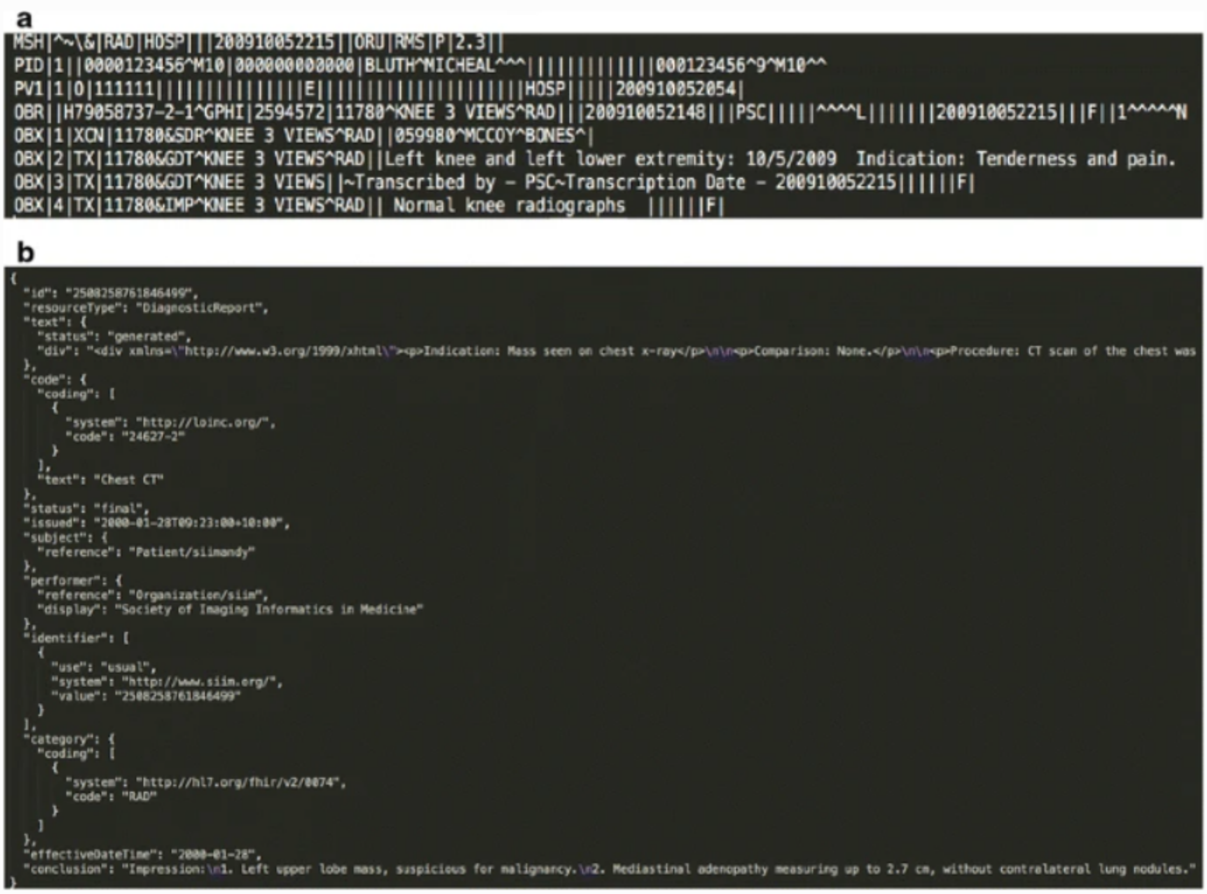

Figure 5: Comparison between a V2 HL7 file and a FHIR HL7 file.

3.3.2 DICOM

The DICOM acronym stands for Digital Imaging and Communications in Medicine, a data standard created in 1983 as a joint effort between the National Electrical Manufacturers Association and the American College of Radiology Mustra et al. [2008]. The sole purpose of this standard was for creating interoperability between any medical imaging device and servers that capture and archive said generated imaging. These servers are formally called Picture Archiving and Communication Systems PACS.

It serves to standardize images of all types of exams such as computerized tomography CT scans, magnetic resonance imaging MRI, X-rays, and mammograms, but unlike other image file formats such as JPEG or TIFF, files in this standard are not recognized by image reading software that is installed in most users operating systems. To view this type of file, it is necessary to use a DICOM viewer, which interprets the file information and displays it as an image Altexsoft [2022].

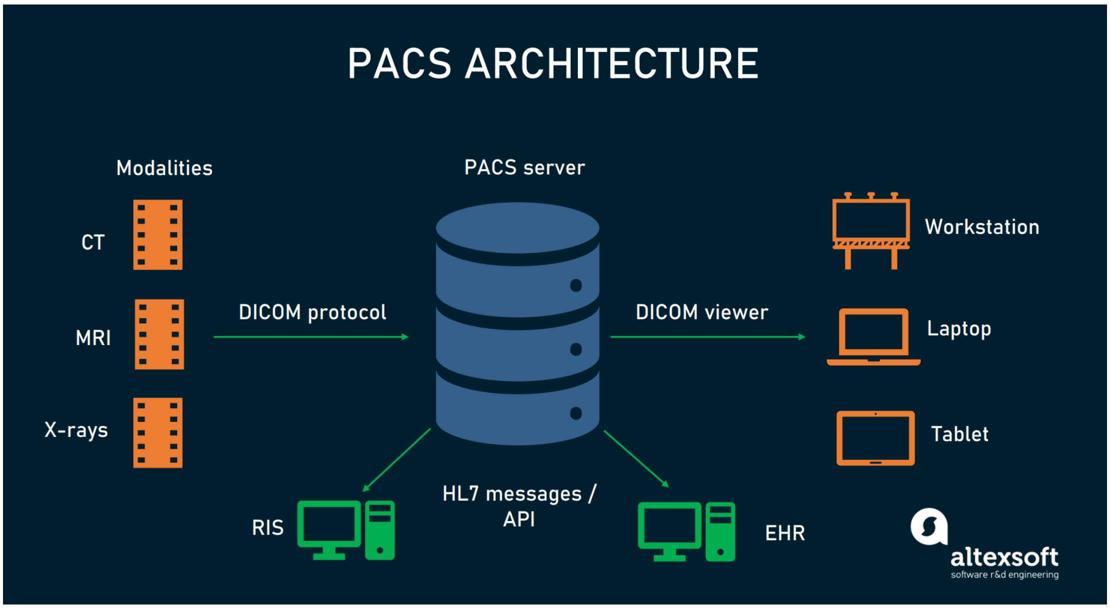

Figure 6: Diagram showing how the DICOM-PACS architecture works.

DICOM is a good standard for medical imaging due to its interoperability, comprehensiveness, wide adoption, and continuous updates, allowing for easy sharing and accessing of medical images. The support of a wide range of imaging modalities and information systems and integration with web technologies and cloud services makes it a format that is continuously updated to stay current with the latest medical imaging technology.

3.3.3 SNOMED

SNOMED CT (Systematized Nomenclature of Medicine - Clinical Terms) is a comprehensive and standardized clinical terminology system developed and maintained by the International Health Terminology Standards Development Organisation IHTSDO NIH [2016]. It is used as a reference terminology for EHR and other health information systems, and it is widely adopted in many countries worldwide. It is a formal ontology that represents biomedical concepts, such as diseases, symptoms, procedures, and medications, using a hierarchical structure of concepts, relationships, and attributes. Each concept is assigned a unique identifier, called a SNOMED CT Concept ID, which is used to represent the concept in EHRs and other systems IHTSDO [2013].

SNOMED CT is available in various formats, such as a relational database, a set of flat files, or an RDF (Resource Description Framework) representation. The most commonly used formats are RF2 (Release Format 2), which is a set of flat files, and RF1 (Release Format 1), which is a relational database format. RF2 is the latest format of SNOMED CT and it is the most widely used format in the current implementation.

The RF2 format represents the SNOMED CT concepts, relationships, and other metadata in a number of different files, such as a concepts file, a descriptions file, and a relationships file. The files are organized by release, and each file contains a set of records, where each record corresponds to a single concept, description, or relationship, usually provided in a compressed .zip or .gz format.

SNOMED CT also provides a number of additional resources and tools to support the implementation and use of the terminology, such as a browser, a mapping tool, and a subsetting tool. It also provides the SNOMED CT International Release, which is the main version of SNOMED CT, as well as a number of national extensions, which are versions of SNOMED CT that have been adapted to the specific needs of a particular country or region.

3.3.4 openEHR

OpenEHR is an organization that creates open-source community-driven electronic health record (EHR) systems based on internationally agreed standards, supporting the interoperability of any application in the clinical setting. It was founded in the early 2000s by a group of researchers and clinicians who were interested in creating an open-source, standards-based approach to EHR systems. openEHR Foundation [2022c]

A big advantage of openEHR over traditional EHR systems is its ability to support the sharing and exchange of clinical information between different systems. Because openEHR uses a standardized data model and a consistent approach to representing clinical data, it is easier for different systems to communicate and exchange information. In the context of openEHR, semantic interoperability is supported by the use of archetype reference models, which provide standardized definitions of clinical data that can be used to specify the allowable data for a particular clinical concept Min et al. [2018]. Syntactic interoperability is supported by the use of open standards and specifications, which define the rules and conventions for representing and exchanging health data using openEHR. Finally, structural interoperability is supported by the use of domain knowledge governance, which is the process of defining, managing, and maintaining the clinical data models and terminologies that are used in openEHR systems.

This can help to improve the quality and availability of health data and support more effective care coordination and collaboration among healthcare providers.

OpenEHR is designed to be a flexible system, allowing it to support the representation of a wide range of clinical data and to adapt to changing requirements and standards, helping to ensure that openEHR systems remain relevant and useful over time to support the ongoing evolution of healthcare practice and technology.

Because the user interface of an openEHR system is separate from the underlying data model (dual model approach), it is easier to customize and modify the system to meet the specific needs of a particular healthcare organization or clinical specialty. This can help to improve the usability and effectiveness of the system and support the capture of more comprehensive and accurate clinical information.

3.4 openEHR Architecture

3.4.1 openEHR Reference model

Arguably one of the most important features of the openEHR approach to interoperability is the use of archetype reference models. These are standardized definitions of clinical data that can be used to specify the allowable data for a particular clinical concept. By using archetype reference models, different openEHR systems can represent clinical data in a consistent and standardized manner, which can support the sharing and exchange of health data between different systems.

This utilizes a very characteristic architecture based on a dual-level approach. The first level is commonly referred to as the Reference Model, it's in this section that various medical definitions, data types/-data structures are defined. This RM serves as the main ontology for every class that gets formerly defined in the EHR. The Archetype model is located on level 2, it's here that all the information about a patient is located.

Here there can be more than one archetype, or even archetypes that include other archetypes inside them, and each of them can provide information specific to certain medical exams, or patient characteristics like body weight and blood pressure.

Level 2 uses the information provided by the RM in level 1 to conjugate all the knowledge about a certain patient. Using the body weight characteristic as an example the archetype will have 2 attributes, one for "Weight" which will have a numeric Datatype in the RM and another for "Comment" which in turn will have a text datatype.

According to Batra et al. [2017] a reference model can be represented with the following schema.

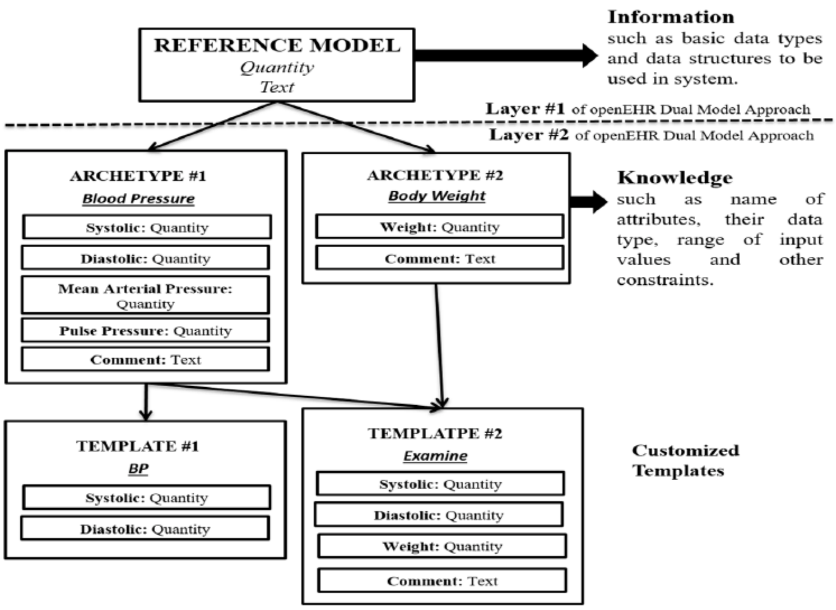

Figure 7: Representation of a dual model approach. Taken from Batra et.al.

The use of two-level modeling in electronic health records changes the engineering of the software significantly. The core system is based on stable reference and archetype models, with domain semantics delegated to domain specialists who build archetypes, templates, and terminology rather than relying on ”ad hoc” conversations with users to gather information. This approach results in archetypes becoming a technology-independent, single-source expression of domain semantics used to drive all technical expressions of the semantics. While some applications require custom engineering, all can now rely on an archetype and template-driven computing platform, resulting in a more consistent and efficient development process. openEHR Foundation [b]

This dual-layer approach is a practical way to create EHRs because one could completely change the Reference Model or come up with new data structures/types and none of the archetypes would get deconstructed or have any significant loss of information.

3.4.2 Classes

In order to accomplish interoperable data sharing across clinical systems it's important to distinguish between the various types of archetypes and the information they carry. For that reason, each archetype is assigned a specific class which serves as a blueprint for a given EHR. Depending on the class type they have properties that are unique to them. A complete EHR is essentially made up of various Archetypes that are organized depending on the type or how specific the information they carry is. The openEHR Reference Model currently defines the following 4 categories of component archetype classes those being the Composition, Section, Entry, and the Cluster class.

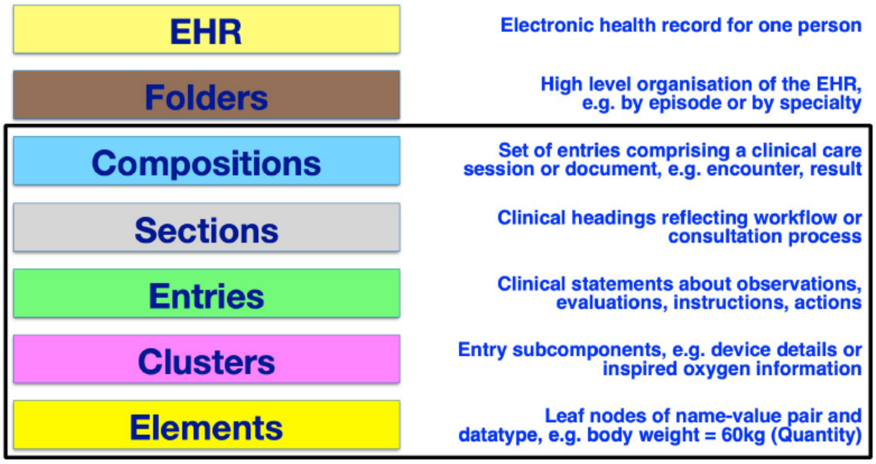

Figure 8: All the classes defined in the openEHR reference model.

Composition

Composition is essentially the main container of an EHR. All recorded patient data is saved inside this class of archetype. Some reference model attributes that are exclusive include the clinical author identifier (composer), start and end times of clinical encounters, and healthcare location. This type of archetype represents generic documents including the discharge summary of a patient or lab reports.

Some of the RM attributes for a typical composition archetype report include the Identifier for the creator of the Composition archetype (Composer) any other clinical professionals that may have been involved in the medical appointment (Participations), the start and end time of the clinical session and location of the Health care facility openEHR Foundation.

Section

The next class contained inside Composition. A Section is used to divide complex compositions into manageable pieces. They are just used for human navigation to place other smaller and more detailed clinical data provided by the Entry and Cluster classes. Examples of Section classes include reports about vital signs or physical examination history.

Entry

In the reference model an Entry is on the lower branch of a composition and inside a section archetype component while at the same time, it's possible for more than one Entry component to be inside a section. The information within an Entry retains its meaning, regardless of where it is used, true to openEHR's fundamental design feature of being able to reuse information for the automatic processing of EHR data. These types of archetypes are used to store data for the Clinical Investigator Recording Process, an ontology model for representing clinical information proposed by Thomas Beale† and Sam Heard as a way to address the lack of standardization and consistency in clinical documentation. Beale et al. [2007].

A diagram of the Clinical Investigator Recording Process can be seen in the following figure:

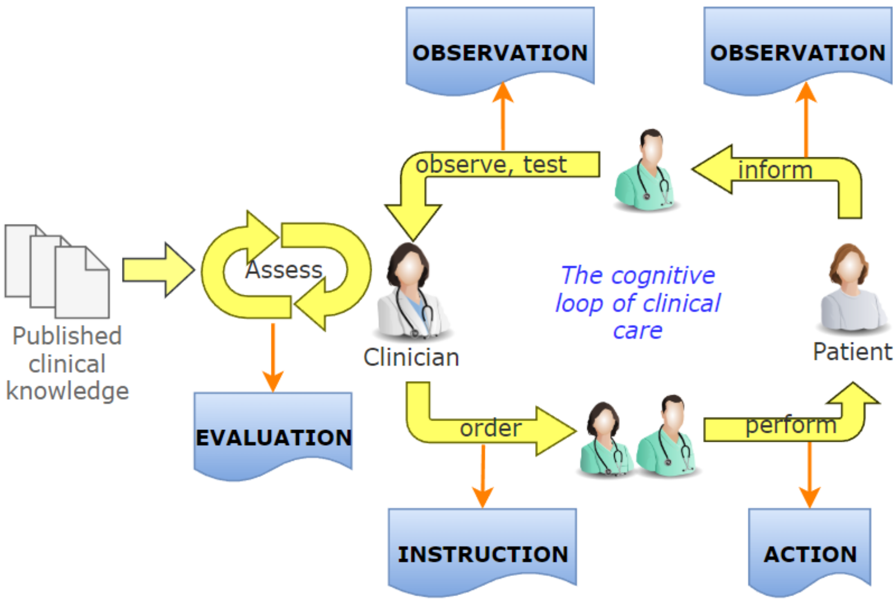

Figure 9: Clinical Investigator Recording Process diagram.

The process starts with the observation of the patient by a healthcare investigator. The patient provides a self-report of their symptoms and concerns followed by the investigator taking notes, subsequently, the investigator conducts a thorough physical examination, including the collection of vital signs and documentation of any abnormal findings that deviate from the expected range of a healthy individual. With the symptoms, and data collected an evaluation of the patient's condition is performed. The investigator, using published clinical knowledge, compares the symptoms exhibited by the patient with possible diseases/conditions and performs a final diagnosis. With the diagnosis done instructions are created for how to treat the patient and provide him with the care and/or medication needed.

As seen in the image above the CIRP has 4 steps to accomplish the goal of creating a consistent and organized patient diagnosis investigation. These steps are represented in the ENTRY archetypes with the following subclasses:

- Observation: This subclass represents an assertion about a particular fact, such as a measurement or a statement of a patient's condition. For example, blood pressure, temperature, or a patient's symptoms could be represented as observations.

- Evaluation: This subclass represents an assessment of a particular fact or situation, such as a diagnosis or an interpretation of test results. For example, a radiologist's interpretation of a CT scan or a physician's diagnosis of a patient's condition could be represented as evaluations.

- Instruction: This subclass represents an order or requests for a particular action to be taken, such as a medication order or a lab test request. For example, a physician's order for a patient to receive a certain medication or a request for a lab test could be represented as instructions.

- Action: This subclass represents the performance of a particular action or procedure, such as administering medication or conducting a surgical procedure. For example, a nurse's administration of a medication or a surgeon's performance of a procedure could be represented as actions.

Subtypes that represent each step of the CIRP are inserted into a higher hierarchy folder, the Care_Entry class. As its name implies this includes archetypes that contain valuable information about patient care. Additionally, the Entry class can have an additional subfolder to store purely administrative or bureaucratic hospitalization information called the Admin_Entry. Administrative information is typically created by non-clinical staff performing administrative tasks to coordinate clinical processes by recording details such as patient admission, appointments, discharge or transfer, and billing or insurance information. Care_Entry on the other hand can only contain information strictly created by clinical professionals since those are the only ones knowledgeable enough to partake in the CIRP.

While administrative information does not directly impact clinical decision-making, it is crucial to the smooth functioning of healthcare systems. Without this information, it would be difficult for healthcare providers to coordinate appointments, transfers, or billing, and patients would not have access to important details about their care. Despite its importance, removing administrative information from the EHR would not compromise clinical integrity, it would simply make it more challenging for caregivers to stay informed about important details related to patients they are responsible foropenEHR Foundation [2022a].

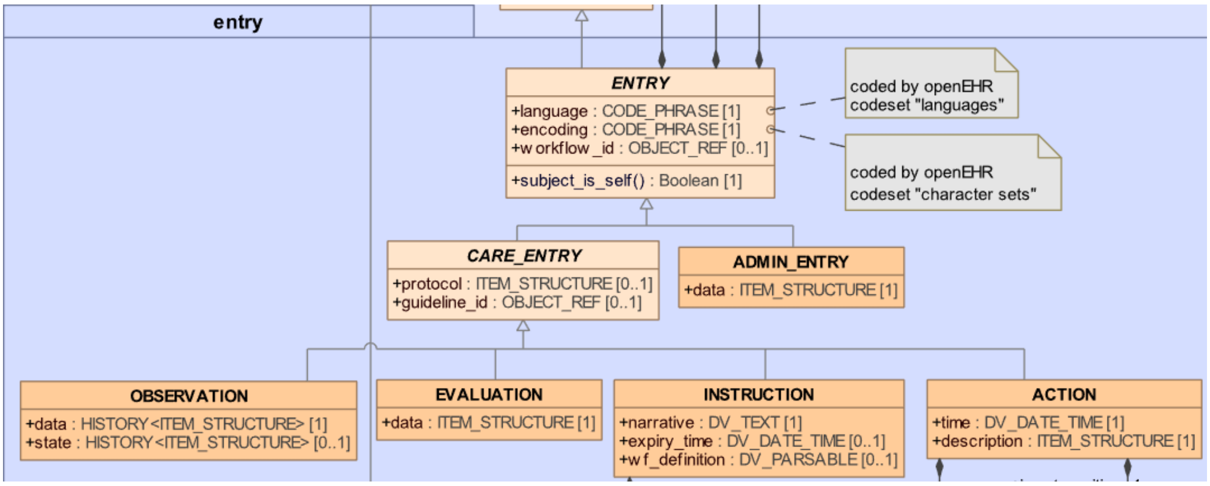

Figure 10: The hierarchy of an Entry class. Entry is divided into a Care_Entry and Admin_Entry classes. Care_Entry contains the archetypes related to the CIRP.

Cluster

Essentially the "lowest" container class of archetypes. They can be found inside Entry, and Composition archetypes as well as other Cluster archetypes. Clusters are particularly useful in cases where recursiveness is important, such as when a data element needs to be repeated or nested within another data element. An example of this use of recursion can be seen in a hypothetical OBSERVATION archetype that contains a Cluster inside, about symptoms seen in a patient. If the patient had heart palpitations the Cluster could contain other ClusterS which describe conditions normally associated with the said symptom (like having a stroke or anxiety).

3.4.3 Datatypes and Datastructures

The following are the datatypes supported by openEHR

- Text, this can include just plain text or coded text
- Date/times
- Encapsulated multimedia data or parsed text files
- Basic boolean types or state variables like temperature or blood pressure levels
- Quantities like weight or height with their corresponding values and units of measurement

Figure 11 shows an example of how these data types can be found in a given archetype. For a prescription archetype, you have the attributes for Drug Name, Dose quantity, and start date and their corresponding datatype

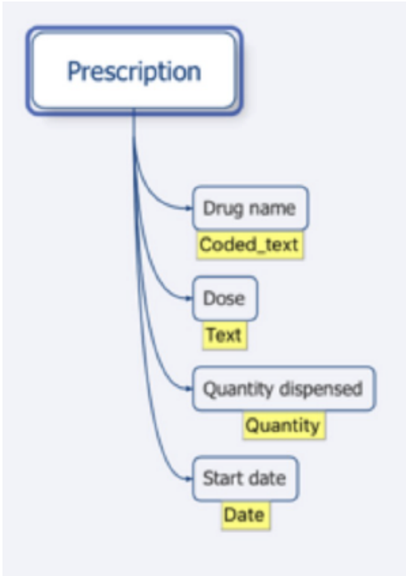

Figure 11: Example of the various datatypes a single archetype can have.

3.4.4 Inheritance

The openEHR archetype framework allows for the creation of archetypes that can be inherited from other archetypes, known as "specialization". This process is used to create new archetypes that are based on existing ones but with additional constraints or modifications.openEHR Foundation [2022b]

This is made possible by employing the "specialize" keyword, enabling the referencing of a preexisting archetype as the parent archetype. The child archetype possesses the capability to introduce specific changes while ensuring compatibility with the parent archetype. These changes may involve the addition of new attributes or the constraining of existing ones. This allows for a hierarchical framework of archetypes to be established, with more specialized archetypes inheriting characteristics from broader ones. Additionally, this mechanism supports multiple inheritance, allowing an archetype to inherit attributes from multiple parent archetypes.

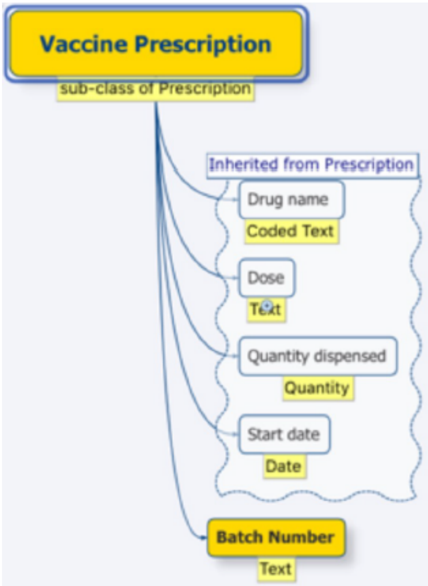

Figure 12: Example of a child class (Vaccine prescription) inheriting properties from a parent class (Prescription). The attributes for drug name, dose, quantity dispensed, and start date are all common in both parent and child archetypes while the batch number attribute is only present in the Vaccine Prescription archetype.

3.4.5 Clinical Data sets vs openEHR solutions

There are several problems associated with the use of traditional clinical datasets that openEHR tries to solve. The lack of consistency in basic data types, presentation formats, and design principles used for example can make it difficult to compare and integrate data from different sources. The time of data capture, interpretation of data, and integrity constraints can also vary, leading to further inconsistencies. Another problem with CDS is the replication of domain knowledge and the lack of multi-language support, which can make it difficult to share and understand data across different systems and organizations. CDS also heavily relies on non-integrated specialist applications, which can make it challenging to share information between different systems and authorities.

OpenEHR was built as a solution that addresses these problems by providing a reference model and the use of the archetype structure. Archetypes are able to support the definition of time series and have all relevant information for the interpretation of measurement, making it easier to understand and use the data and define integrity constraints in a uniform way, which helps to ensure the quality and consistency of the data.

Another important aspect of openEHR is the mechanism of making archetypes freely available at one central place, which helps to avoid 'reinventing the wheel' and ensures that the latest and most accurate archetypes are used. The openEHR also ensures that any translation occurs within one archetype only, which helps to maintain the semantic meaning of the data. Furthermore, archetypes can be shared by multiple HIS and authorities, which allows information to be exchanged between different systems while keeping the semantic meaning allowing for more efficient and effective sharing and integration of health information across different systems and organizations.

3.5 Archetype Definition Language

As established previously in this chapter, without a consistent and standardized way of defining archetypes, different systems, applications, and organizations may have different interpretations of the same concept, which can lead to inconsistencies and interoperability issues.

The openEHR foundation created a rather complex syntax to define and build archetypes formally named the Archetype Definition Language ADL. This language has a characteristic structure that is designed according to the following image:

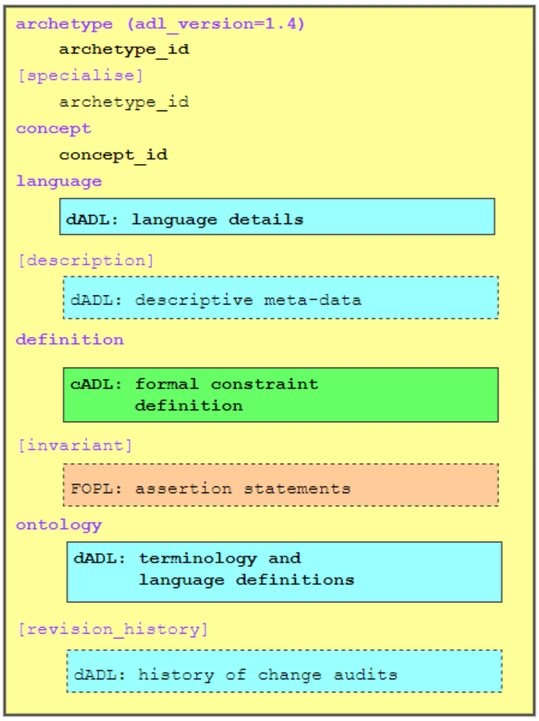

Figure 13: Representation of the structure of an ADL archetype file.

The terms represented in bold (archetype, concept, language, definition, ontology) are the main parts of the document present in every file while the terms in parenthesis are merely optional and are not present in every ADL file.

There are two main syntaxes for representing data in ADL files. Data ADL dADL and Constraint ADL cADL. The dADL structure is used to define most of the content of a common ADL file including the header, concept, language, description, and ontology sections. The following sections exemplify how this syntax is written as well as the special characters used and their meanings.

3.5.1 Data ADL

Figure 14 shows a snippet of a data ADL syntax being used. This section is best characterized by the use of the smaller than (<) and greater than (>) symbols to enclose and delimit object definitions that are associated with a certain attribute name (e.g. the family_name attribute). Attributes can have sub-attributes characteristics (e.g. name attribute) and form more complex tree branch-like paths.

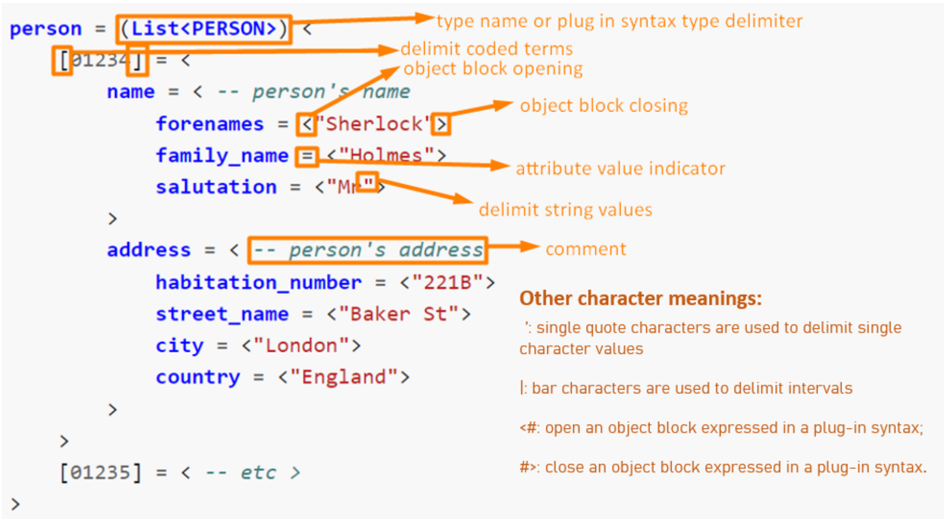

Figure 14: Representation of a snippet of a Data ADL syntax made for a person attribute. Written in orange are the special characters used as well as their meanings.

The dADL syntax is designed to represent data in a way that is both human-readable and machine-processable while making the fewest assumptions about the underlying information model. While the XML schema could also compete for this purpose it however is mainly intended for machine processing. dADL is an abstract syntax for object-oriented data that provides a comprehensive set of leaf data types, adheres to object-oriented semantics, and requires half the space of the equivalent XML while not using the confusing XML notion of 'attributes' and 'elements' to represent object properties, and more than one information model can be compatible with the same dADL-expressed data openEHR Foundation [a].

3.5.2 Constraint ADL

This is a special syntax designed to let the user put restrictions on data defined by object-oriented information models in archetypes by defining data structures that conform to the general object model defined, running computational systems that check data against cADL in an archetype. Syntax wise it's similar in structure to dADL since both are defined by blocks, however, cADL uses brackets, and the "matches" keyword (the math symbol ∈ and the "is_in" keyword can also be used) instead of greater/lesser than and the equal sign.

```
PERSON[at0000] matches {                                -- constraint on a PERSON instance
    name matches {                                      -- constraint on PERSON.name
        TEXT matches {/.+/}                             -- any non-empty string
    }
    addresses cardinality matches {0..*} matches {      -- constraint on
        ADDRESS matches {                               -- PERSON.addresses
            -- etc --
        }
    }
}
```

Figure 15: Representation of a snippet of a constraint ADL syntax. The syntax is mostly similar however it replaces the arrows with curly braces.

In cADL terms that are written in all caps, the PERSON class in the previous example, are the main constraint identifiers while lowercase terms like name or addresses correspond to attributes constrained to the main identifier. cADL also recognizes certain keywords that offer additional constraint instructions for certain data that get parsed. The cardinality keyword written in red is used to define container attributes that set constraints on the number of members that an instance of a container type, like a list or a set, can have. The example above essentially says that the addresses property has a cardinality that matches a number between zero to many.

Using the previous image as an example, it is evident that a "name matches" attribute comes with a constraint that the PERSON type identifier must match the regular expression "/.+/" in the TEXT field. This regular expression matches any character occurring at least once. Regular expressions like this can be used to define constraints that allow the inclusion of other archetypes files, by using the "allow_archetype" keyword. In the following example, we have a SECTION archetype that defines an item's property indicating what OBSERVATION and other SECTION archetypes are allowed. A regex like /.*/ or /.+/ essentially means any archetype is allowed to be included in this context while usually, the "include" keyword will only exhibit one or two regular expressions for files that need to be inside the original archetype.

```
SECTION [at2000] occurrences ∈ {0..1} ∈ {                          -- objective
    items cardinality ∈ {0..*} ∈ {
        allow_archetype OBSERVATION occurrences ∈ {0..1} ∈ {
            include
                short_concept_name ∈ {/.+/}
        }
        allow_archetype SECTION occurrences ∈ {0..*} ∈ {
            include
                archetype_id/value ∈ {/.*/}
            include
                archetype_id/value ∈ {/openEHR-EHR-SECTION\.patient_details\..+/}
        }
    }
}
```

Figure 16: Usage of the allow_archetype term to include certain files.

3.6 Web Applications development

The process of creating software applications that can be accessed over the internet using a web browser requiring the knowledge of various technologies and frameworks, as well as an understanding the unique challenges posed by web-based environments.

Some key concepts one needs to know about web application development include:

- **Client-server architecture**: Web applications typically follow a client-server architecture, where the client (usually a web browser) sends requests to a server, which processes those requests and sends back responses.
- **Frontend development**: The frontend of a web application is the part that is visible to the user and is responsible for handling user input and displaying the application's output.
- **Backend development**: The backend of a web application is the part that is responsible for processing data and business logic.
- **APIs**: Web applications often use APIs (Application Programming Interfaces) to allow different parts of the application to communicate with each other and get specific data.
- **Security**: Web applications can face numerous security risks. Security flaws can occur due to unwanted SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). It is imperative to possess knowledge on crafting secure code, employing encryption techniques, and

incorporating authentication and authorization mechanisms to ensure the development of robust and protected web applications.

- **Performance**: Minimizing load times, reducing server response times, and optimizing data processing are all important aspects of a well performed web application.
- **Testing**: Web applications must be thoroughly tested to ensure they function as intended and are free of bugs. Testing frameworks and techniques such as unit testing, integration testing, and end-to-end testing are usually done to build reliable web applications.

The general architecture of a web application follows a model where you have users interacting with an interface that can trigger requests to the App's scripting on the backend side. The web server is then able to extract any data the user requested initially and send the response where it's displayed in the frontend for the user to see and interact with. An overview of this web application architecture can be seen in Figure 17:

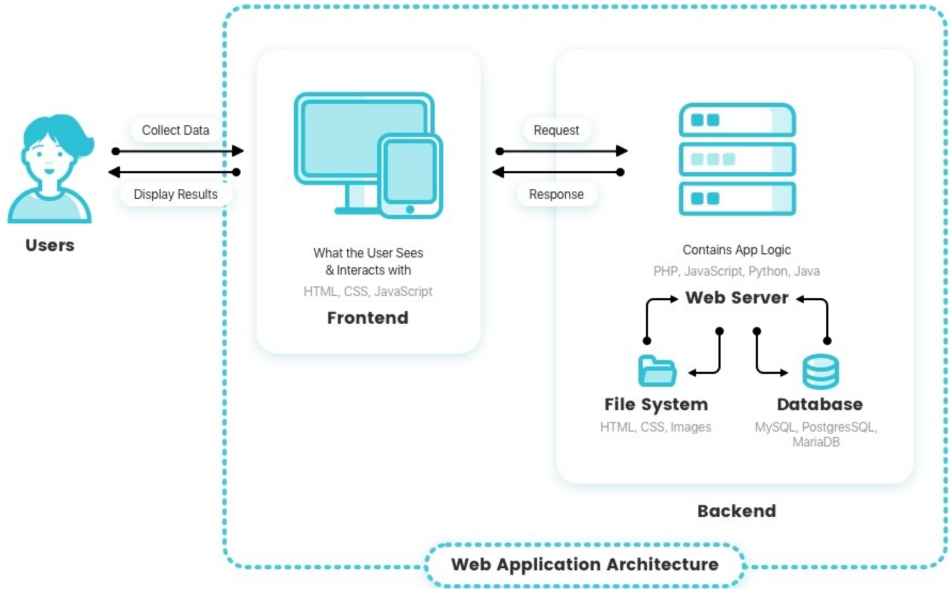

Figure 17: General architecture of a web application.

3.6.1 REST API

A REST API (or Representational State Transfer API) is an application programming interface that conforms to the constraints of REST architecture, an architectural style for distributed hypermedia systems. The primary purpose of an API is to offer a streamlined web services database, enabling applications to efficiently communicate and exchange data across the internet. It is specifically designed to be lightweight, ensuring optimal performance and ease of integration.

REST APIs typically use the HTTP protocol and its request methods (GET, POST, PUT, and DELETE) to send and receive data. The client and server components of a REST API are loosely coupled, and communication is done through a stateless interface, as such the client does not need to maintain any state information about the server, and the server does not need to keep track of the client's state.

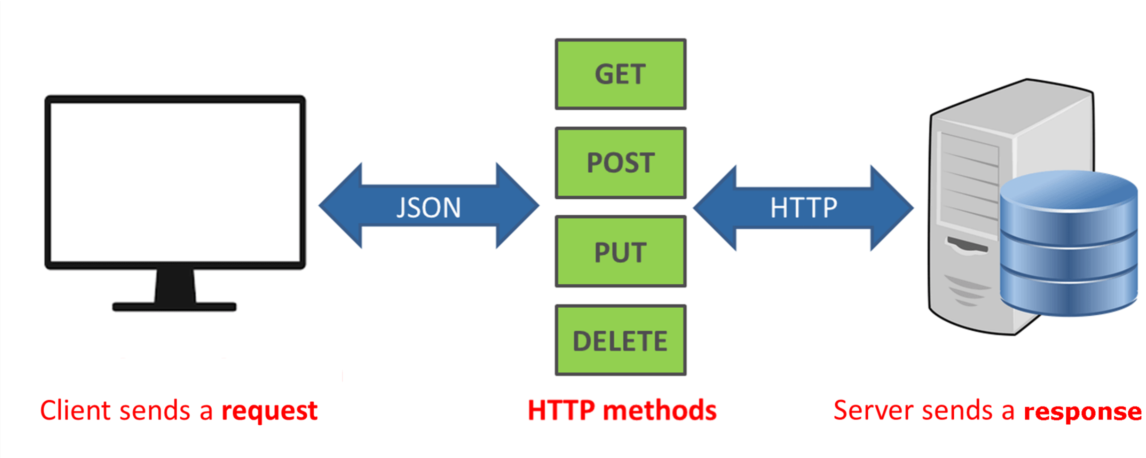

Figure 18: Representation of a REST API.

The primary design principles of REST APIs are based on the Representational State Transfer (REST) architectural style. This style was originally proposed by Roy Fielding in his 2000 doctoral dissertation, and it provides a set of rules for designing web services that are lightweight, maintainable, and scalable Fielding [2020]. The main principles of REST APIs are as follows:

- **Client-server separation**: The client and server should be independent and should not rely on each other. This allows the client and server to be developed and deployed independently.
- **Statelessness**: The server should not maintain any state information about the client. The client should instead send all the necessary data required for each request.
- **Cacheability**: The server should provide information about which responses can be cached and for how long.
- **Uniform interface**: The server should provide a consistent interface for clients, which should be independent of the underlying implementation.
- **Layered system**: The server should support a layered system, which allows for different levels of authentication and authorization.
- **Code-on-demand**: The server should provide the client with the ability to download code and execute it. This can be used to provide additional functionality, such as dynamic HTML.

3.6.2 Front-End

Frontend refers to the part of a software system or application that is visible to and interacts directly with the user. It includes the user interface, which provides a way for users to interact with the application, as well as the frontend logic that processes user input and sends requests to the backend.

In a typical web application, the frontend consists of the user interface that is displayed in the user's browser. This interface is built using primarily HTML code, with CSS offering styling choices and JavaScript handling all the complex logic behing the scenes.

The frontend logic is responsible for handling user input and generating requests to the backend, as well as processing responses from the backend and updating the user interface accordingly. The logic is typically implemented using JavaScript, which allows for dynamic behavior and interactivity in the application. Multiple frameworks and libraries have been developed to ease developers into building functional web applications, for javascript development the more popular choices are React and Angular but more are continuously being produced.

3.6.3 Back-end

While the frontend deals with the user interactions the backend refers to the part of a software system or application that is responsible for processing data and performing operations that are not directly visible to the user, that includes the server-side logic that powers the application, manages the database, and handles requests from the frontend.

In a typical web application, the backend consists of a server, application server, and database. The server receives requests from the client (e.g. a browser), routes them to the appropriate part of the application, and sends responses back to the client. The application server is responsible for handling the business logic and processing data, while the database stores and retrieves data as needed by the application.

A system's backend can be developed using a variety of programming languages and frameworks. Software professionals skilled in API development, server-side programming, and database management will usually be given the job to build the backend of applications.

3.6.4 JavaScript Object Notation

JavaScript Object Notation or JSON is a data interchange format first introduced in 2002 by Douglas Crockford as an alternative to XML, which was becoming increasingly complex and difficult to use Florescu and Fourny [2013]. Similar to the syntax of JavaScript object literals, JSON is built using a collection of key-value pairs where the keys are strings surrounded by quotation marks and values can be any supported data type like numbers, Booleans, arrays or other complex nested JSON objects.

It's generally agreed that using a JSON structure to represent data when transferring between a server and web applications is a better alternative to other text-based formats like XML or, in the context of openEHR, ADL files Simec and Maglicic [2014]. Compared to the XML format, JSON is strictly typed meaning objects can be recognized as any data type supported like string, numbers, or booleans, unlike XML where every value data is recognized as a string type only. JSON objects are also far more lightweight allowing for faster data transfers and consuming less bandwidth between the client side and the server side of an application.


```
{
  "rules": {
    "align": [false,
      "parameters",
      "arguments",
      "statements"],
    "ban": [true,
      ["angular", "forEach"]
    ],
    "class-name": true,
    "comment-format": [false,
      "check-space",
      "check-lowercase"
    ],
```


Figure 19: Example of a JSON structure supporting string, list, and boolean types recognized by the editor with multiple colors.

JSON is used extensively in modern software development, particularly API development. The popularity of the JSON format has become such a stapple in the programming landscape most languages will include built in methods or libraries to work with JSON parsed data. Some notable examples include the JavaScript JSON.parse() and JSON.stringify() methods, the JSON Schema standard, and popular JSON libraries like Jackson (for Java) and Newtonsoft.Json (for .NET) . Its support in various programming languages means it's far easier to maintain the interoperability of openEHR data assuming the data is parsed in a JSON object since almost all modern languages recognize JSON structures and have prebuilt methods used to extract only certain values needed by the user.

Chapter 4

Materials and Methods

This section lists various tools used to create the web application part of this project, namely the main language, libraries and frameworks used. A brief look into what they are, and their functionalities is also described here.

4.1 Javascript

Introduced in 1995 by Netscape Communications Corporation, JavaScript emerged as a programming language aimed at creating interactive elements and dynamic content within web pages. It a language used both for developing the client-side interface as well as server-side scripting. On the client side, it is used in conjunction with HTML and CSS to create more complex interactive elements on web pages, such as forms or animations that responds to certain user inputs. In conjunction with the back-end side it's possible to create dynamic applications that continuously communicate with a server to display data from a database.

Its relevance can be attributed to its ability to offer developers a way to construct high-powered and reliable web applications and websites. JavaScript garners additional appeal due to its widespread support across all major web browsers, ensuring easy accessibility for developers of varying expertise levels Wirfs-Brock and Eich [2020]. To this day, JavaScript is the most used client-side language for any web browser and many major companies and organizations still use it W3Techs, making it a well-supported language with tons of libraries, frameworks, and documentation.

4.2 React

Created by Facebook in 2013, React is one if not the most popular JavaScript library to design user interfaces. It's specially designed to simplify the creation of fast and dynamic user interfaces by using a virtual DOM, which is an in-memory representation of a web page, to improve the performance of web applications by only updating elements that need to be updated at a certain time. React also makes use of a declarative programming style, which means that developers can focus on the code that is necessary to create the user interface, rather than having to worry about the order of operations.

The biggest feature in the React library is in the powerful component-based architecture which makes it easy to break complex user interfaces into smaller components. Each component is a self-contained piece of code that can be reused throughout the application, making it easier to manage and maintain the overral code.

Facebook, Instagram, Airbnb, and Netflix Evgeniy Fetisov [2023] are just some of the few large companies that actively incorporate this library with many smaller companies actively using this technology in their day-to-day life. With its component-based architecture, easy-to-learn syntax, and performance-oriented design, React is a popular choice for developing web applications and websites and it's why it was the main library used to build the user interface of this project.

4.3 Bootstrap

Bootstrap is an open-source front-end framework that is used for developing responsive, websites. It was created by Mark Otto and Jacob Thornton in 2011 at Twitter and is currently maintained by the Bootstrap core team with contributions from the open-source community Spurlock [2013].

It's built on HTML, CSS and JavaScript code and is used to create modern, interactive webpages with high user engagement. It provides a wide range of components and plugins that help developers create webpages quickly and efficiently at the same time offering plenty of customizable options, allowing developers to customize the look and feel of the website to their own liking.

Being one of the most popular frontend frameworks it's extremely well documented, with plenty of tutorials, courses and resources available to help developers get up and running quickly. Any developer with not a lot of experience in UI design can build a functional and aesthetically pleasing website with minimal effort making it a popular choice.

4.4 NextJS

A fullstack web development framework that provides an intuitive way to build web applications while integrating both the server side and front-end code together. It was created by Zeit, a technology company based in San Francisco, California, in 2016, and it has quickly gained traction among developers due to its numerous benefits and features Konshin [2018].

This framework uses server-side rendering to deliver fast web pages that load quickly, even on slow connections. This approach also improves search engine optimization (SEO) by providing search engines with optimized HTML that is easy to index. Additionally, Next.js provides a feature called Incremental Static Regeneration, which enables developers to re-render only the parts of the page that need to be updated, resulting in faster load times and a more responsive user experience, it also provides excellent support for static site generation, which makes it ideal for building content-heavy websites that require fast load times and improved SEO Konshin [2018].

4.5 Github API

GitHub is a web-based platform founded in 2008 by Tom Preston-Werner, Chris Wanstrath, and PJ Hyett. It's widely used for hosting, managing, and collaborating on software development projects as well as general file storage van der Vlist [2013]. It's popularity arrised from it's usage in open-source software development, used by both large businesses and individual developers to store and manage their code repositories. For this project GitHub was chosen to manage the files in the CKM repository due to its ease-of-use API and the compatibility with the Archetype Designer web software.

Part II

Core of the Dissertation

Chapter 5

ADL to JSON conversion

Parsing ADL files to a valid JSON structure was not a simple one step process, due to JSON's strict text formating rules in order for the script to recognise any string given to it and convert to a JSON structure certain changes in the files had to be made, in particular when it came to symbols that had to be replaced or removed. In order to make this easier the program created JSON structures based on information about the header, concept, language, description, definition, and ontology. Only after creating each of these structures the final JSON formatted file can be formed by adding the previous JSONs into one big compiled JSON. This part of the work required the usage of regular expressions to either extract all the information or modify certain parts of the ADL text document. The following table examplifies the regex created to get each section mentioned previously.

Section	**Regular Expression**
Header	/archetype[\w\W]*?concept/
Concept	/concept[\w\W]*?language/
Language	/language[\w\W]*?description/
Description	/description[\w\W]*?definition/
Definition	/definition[\w\W]*?ontology/
Ontology	/ontology[\w\W]*>/

Table 1: Regular expressions used to capture each section in the ADL file.

It's important to note that the files used to test this conversion algorithm are all ADL version 1.4 and the openEHR reference model version used was the 1.1.0. More information about each archetype component used to write this algorithm can be found in the "specifications-ITS-JSON" GitHub repository. Available at: https://github.com/openEHR/specifications-ITS-JSON (Last Accessed: April 13, 2023). All the files used are available freely in the clinical knowledge manager CKM, a tool created by openEHR to view archetype and template files. Table 3 shows an extensive list of all the files that were used for testing purposes.

5.1 Archetype Header

This following section contains information about the adl version, uid code and file name, and the corresponding JSON formated header section with this information. Creating this section was fairly straightforward using regular expressions all that was needed to do here was to add the necessary curly braces, quote symbol and colons while removing the unneeded parentheses.

```
archetype (adl_version=1.4; uid=71ef98e0-22d6-4548-a375-251cb5d776da)
        openEHR-EHR-EVALUATION.alcohol_consumption_summary.v1
```

```
{
  archetype: {
    adl_version: 1.4,
    uid: '71ef98e0-22d6-4548-a375-251cb5d776da',
    file_name: 'openEHR-EHR-EVALUATION.alcohol_consumption_summary.v1'
  }
}
```

Figure 20: Example of a header section. The grey square is the original unmodified header while the black square is the final JSON structure.

5.2 Concept and Specialize

Both these sections only have a single parameter, concept only has a ontology code and it's corresponding definition while an the specialize section will only have the adl file name for the associated archetype that gets it's characteristics inherited.

```
concept
        [at0000]        -- Alcohol consumption summary
```

```
{ concept: { at0000: 'Alcohol consumption summary' } }
```

Figure 21: Example of a concept section and it's corresponding JSON formatted structure.

5.3 Language

The language section contains two main parameters, the original language the archetype was written and any translations the ADL file can have as well as the author of said translation and related information (name, organization, email). According to the Archetype Definition Language 1.4 specifications page,the arrow symbol > and < are used as self enclosed parameter tags ("<" being the opening tag and ">" the closing tag). Knowing this information, these arrow can get replaced by curly braces originating a JSON structure seen in the below image.

```
language
	original_language = <[ISO_639-1::en]>
	translations = <
		["de"] = <
			language = <[ISO_639-1::de]>
			author = <
				["name"] = <"Ramona Wellmann, Natalia Strauch">
				["organisation"] = <"Medizinische Hochschule Hannover">
				["email"] = <"wellmann.ramona@mh-hannover.de, Strauch.Natalia@mh-hannover.de">
			>
		>
	>
```

```
{
  language: { original_language: 'en', translations: { de: [Object] } }
}
```

Figure 22: Example of a language section and it's corresponding JSON formatted structure.

It should be noted that to create this JSON structure the parameters with self closing brackets like the name, organization and email had to have said brackets removed. Only the information was needed, the JSON does not need for these parameters to be in brackets.

5.4 Description

This section encompasses descriptive information about the archetype, including references, contributors, and licensing. It also includes metadata for searching and indexing files in repositories. The final JSON structure created for this work only retains information about the original creator and language details, omitting other information.

```
{
  description: {
    original_author: {
      name: 'Heather Leslie',
      organisation: 'Atomica Informatics',
      email: 'heather.leslie@atomicainformatics.com',
      date: '2013-05-14'
    },
    details: { de: [Object], en: [Object] }
  }
}
```

Figure 23: Example of a JSON formatted description section.

5.5 Ontology

An archetype's ontology section is mostly a term_definitions subsection where the definition for node identifiers are found, usually written in multiple languages. The following parameters can also be found but occur more rarely in the ontology schema:

- terminologies_available
- term_bindings
- constraint_definitions
- constraint_bindings

Each term_definitions node is written with a code starting with the letters "at" followed by 4 digits. The same rule applies to the "terminologies_available" and "term_bindings" subsections, but for the constraint-related section, the prefix "ac" is used instead. Inside each node there's a text parameter with a title followed by a description parameter with a short explanation about said title. Sometimes there's also a comment attribute but it's not mandatory unlike the text and description attributes and even more rarely there can be a provenance attribute if the term definitions came from another different terminology openEHR Foundation [2021]. An example of an ontology section after being parsed to a valid JSON object can seen in figures 24.

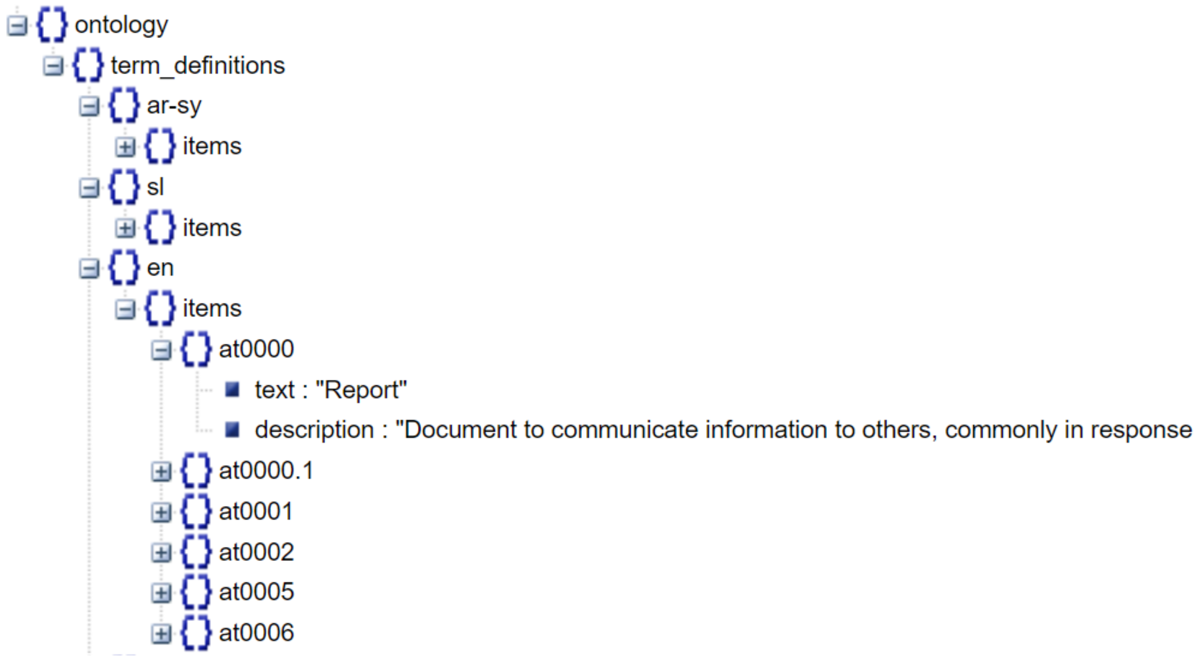

Figure 24: Snippet of an Ontology section. The "ar-sy" , "sl" and "en" codes correspond to languages. A node (at0000) with a text and description attributes inside the English language section for term definitions is also seen here.

5.6 Definition

The definition portion of the archetype definition language outlines the layout and restrictions for a specific kind of clinical document, such as a progress note or discharge summary, for that very reason it uses the cADL syntax, unlike every previous section that followed the dADL syntax general guidelines.

It's in this section that the information in each unique archetype class is contained as such to start off it's important to have the element that identifies the archetype class. The first node is represented by the at0000 code preceded by the archetype class (in all caps), thus a cluster archetype will be identified by the code "CLUSTER[at0000]".

In the JSON structure, this first node will have a rmType, an occurrences, and a node attribute. The rmType stands for reference model type and here the class name of the archetype is identified. The occurrences parameter has a lower occurrences and upper occurrences elements which get its values either by an item matches line that is directed followed up, or if this line doesn't exist it's assumed that the lower and upper occurrences both have a value of 1. Finally, the node attribute has the code, text, and description elements which are directly taken from the ontology section of the file, for that very reason it's important to define and create the JSON section of the ontology before the definition section, even though in the adl schema ontology is the very last section in the file.

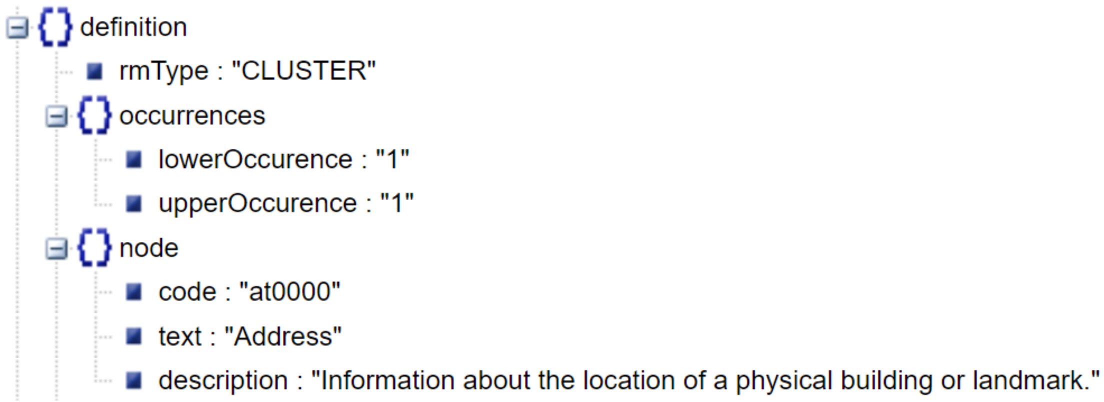

Figure 25: Example of a representation of a CLUSTER root node in a JSON string.

With the initial node identified It's now possible to fill up the rest of the JSON. Using a CLUSTER archetype as an example, Whenever there's an "items matches" property the JSON will have a items list to fill up with ELEMENT objects that the CLUSTER contains. Like the first node, ELEMENT objects will have the same attributes identified however there's an additional value_matches object here. This is a parameter used to define the data type of the data brought by the element. A glossary containing a detailed description of all the DV types can be checked through in Table 2 in the appendix section of this work.

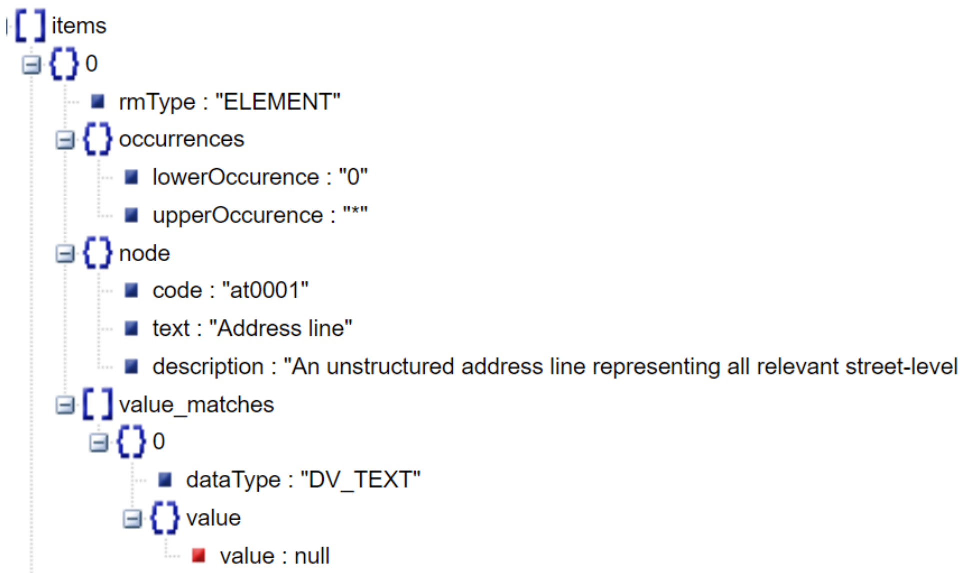

Figure 26: Example of an element leaf node.

The previous example shows an element node with a DV_TEXT containing an object with a single key/item pair of "value" followed by a null keyword. The dataType in this example has an object structure inside that the user is supposed to fill out, however, for the purpose of this work all files have these structures with null values with the only exception being the DV_CODED dataType. If the DV_CODED inside the element node contains the keyword "local" followed by an "at" or "ac" code its data is taken from the ontology section of these files otherwise this section stays blank. An example of this can be seen in Figure 27.

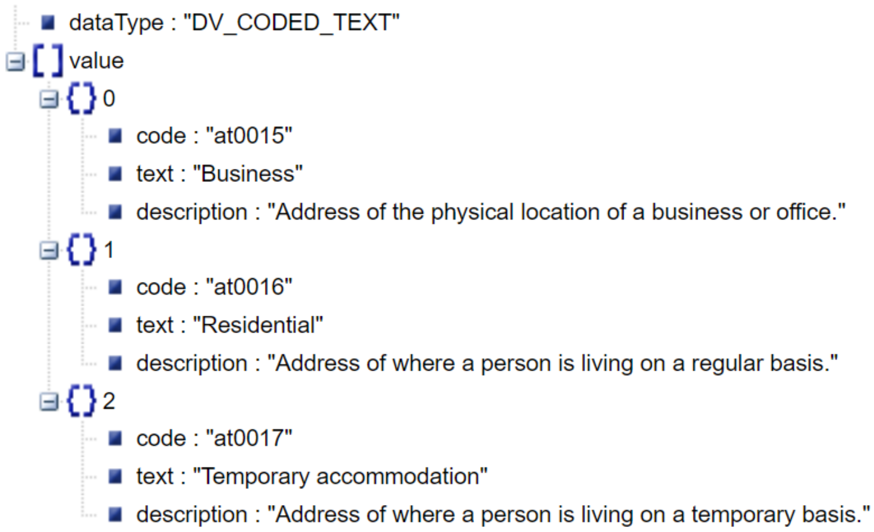

Figure 27: A DV_CODED value matches filled with ontology data.

Every other object created for each DV type can be checked thoroughly in Table 3 in the appendix section of this work.

5.6.1 Include Archetype

To make the inclusion of other archetypes possible whenever an "allow_archetype" property is detected the program will extract the regular expression for the file/files that need to be included and it will recursively call for the main function to make an API call to the repository until it eventually reaches an archetype with no valid "allow_archetype" property. Every archetype called will then be parsed and added to the final JSON. Archetypes that are included inside other archetypes get the added property called the "xsi_type : C_ARCHETYPE_ROOT" to give the indication they are archetypes included inside other archetypes.

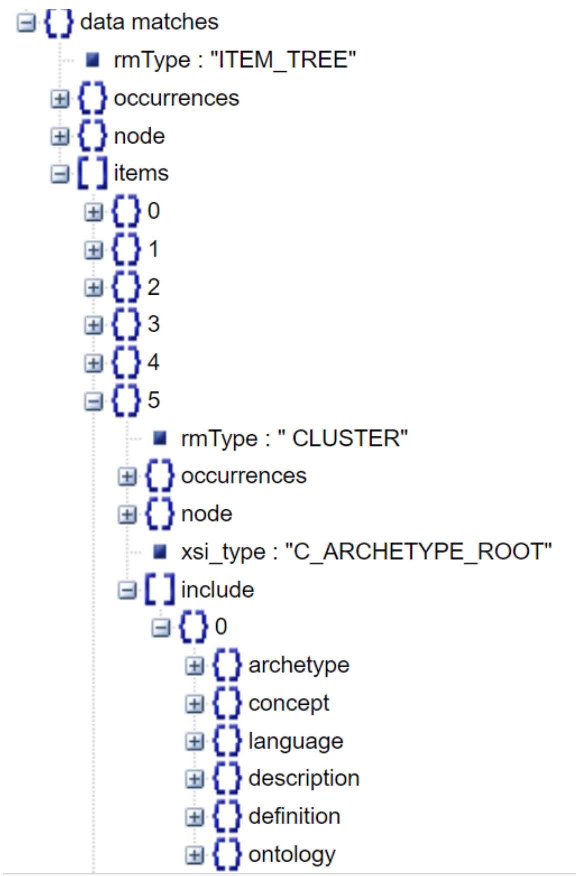

Figure 28: Example showing an ITEM_TREE subclass with another cluster inside represented with the xsi_type property.

As seen in the previous image, the included archetype will retain information about its type, occurrences, and node and the "Include" list has a complex nested object with every section an archetype has.

5.6.2 Property Matches

While CLUSTER classes will only have item object matches, certain archetype classes will include additional properties that need to be added. These properties are easily identified by a lowercase naming convention right after the class name identifier. An archetype can have multiple properties and these can be nested inside other properties as seen in Figure 29.

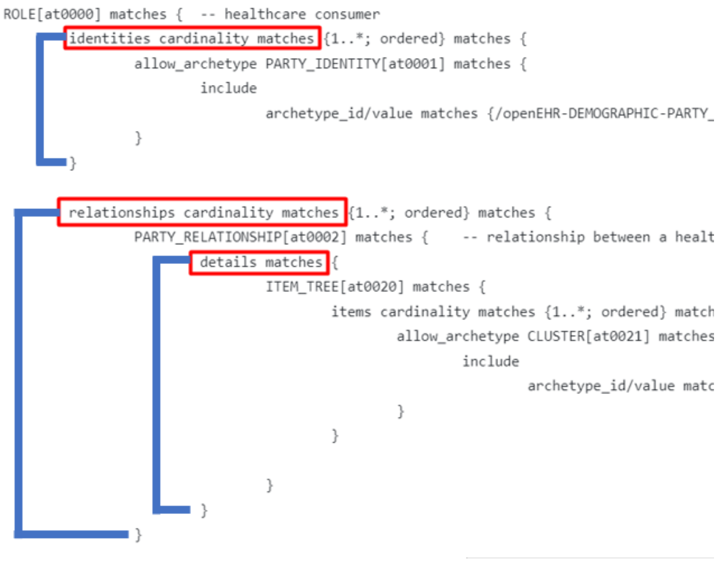

Figure 29: Examples of property matches (red squares) in a role class archetype. The blue brackets identify the information contained in each property.

After creating the first node the following step is to extract all the property matches names and content. That is accomplished by using the following regular expression:

```
[a-z_]+ matches\s+{.+?}(?=\s+[a-z]+ matches|$)
```

The script will run a check to see which property the regex matched with and use the according function to create an object. This is all done with an object constant that stores information about the existing property matches and then a function created to parse properties in the most reliable manner.

```
const objMatchCheck = {
    "name matches" : createNameMatches,
    "category matches" : createNameMatches,
    "details matches" : createDetailsMatches,
    "identities matches" : createDetailsMatches,
    "contacts matches" : createDetailsMatches,
    "relationships matches" : createRelationshipMatches,
    "description matches": createDetailsMatches,
    "credentials matches" : createDetailsMatches,
    "items matches" : createDetailsMatches,
    "context matches" : createContextMatches,
    "ism_transition matches" : createTransitionMatches,
    "protocol matches" : createProtocolMatches,
    "activities matches" : createActivityMatches,
    "data matches" : createDetailsMatches,
    "state matches" : createDetailsMatches

 }
```

Figure 30: Object created with various archetype property names and their corresponding parse functions.

Something to note here is that certain properties while having different names are essentially parsed the same way according to the RM schema. That's the main reason why some matches use repeated functions. An example of a parsed ADL file with properties specific to EVALUATION archetypes can be seen in Figure 31:

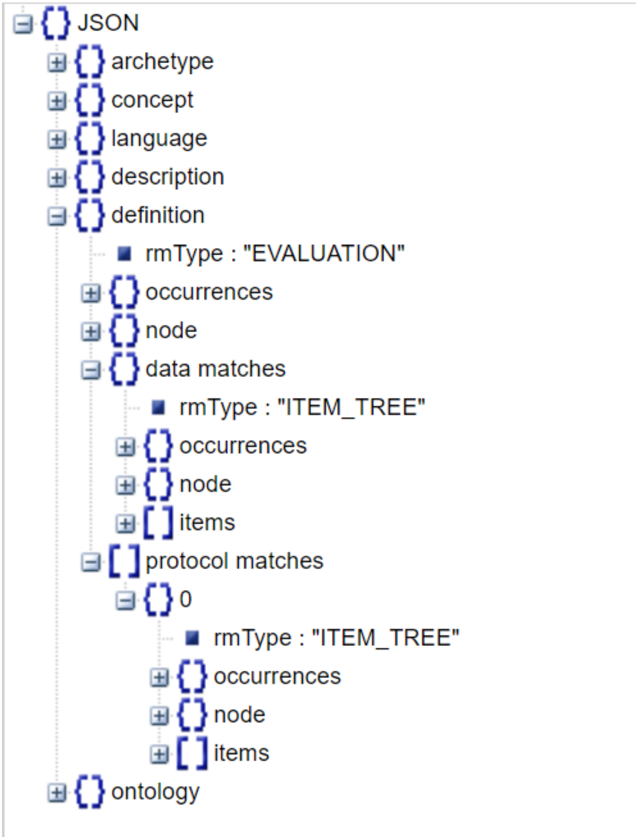

Figure 31: Example of an EVALUATION archetype with a data and protocol matches properties.

Chapter 6

Web Application Development

The main objective of this application was to provide a simple way to manage files taken straight from a given repository and directly convert them to a JSON formatted structure. To keep it relatively simple and easy to use the Web application follows a general layout comprised of a header that when clicked exposes the sidebar dropdown menu with the content body being rendered on the right side. A representation of the application schema can be seen in Figure 29.

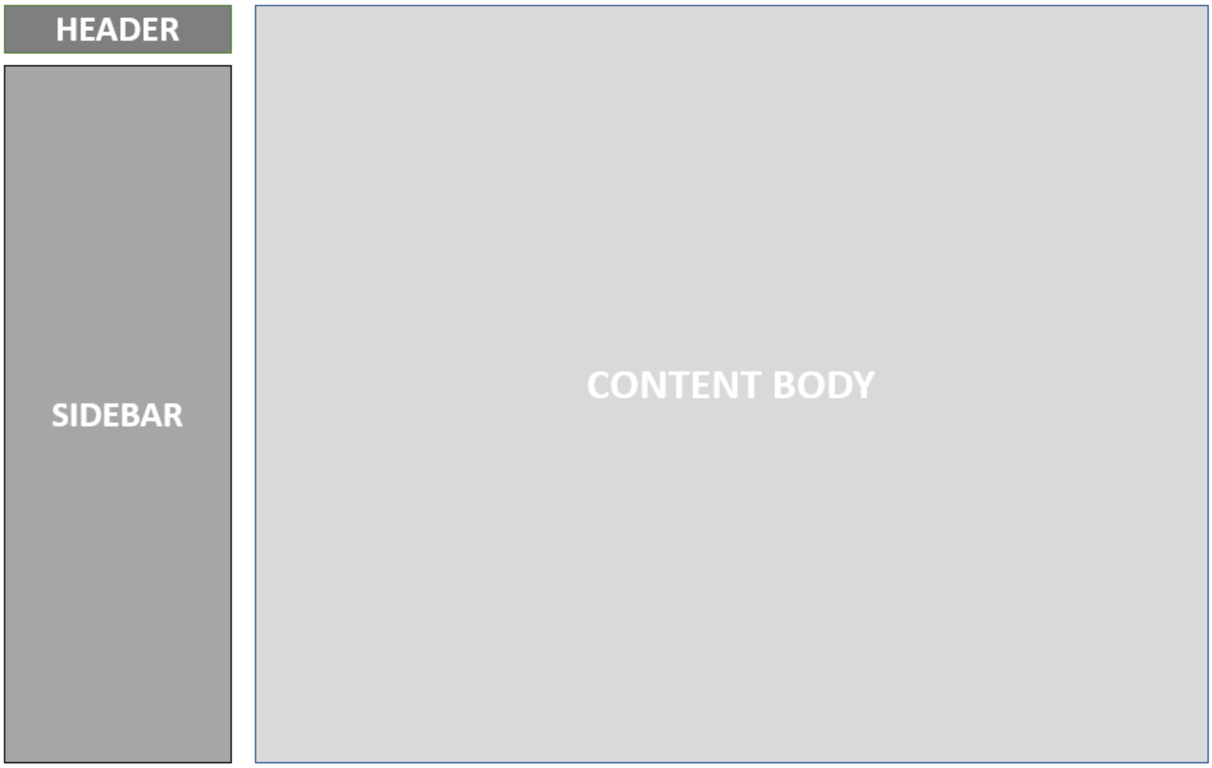

Figure 32: Representation of the application schema.

To carry out the work, the front-end part of the application was designed using React, and the components of the pages, namely the dropdown menu on the side and buttons, were done using the bootstrap framework since it provides easy to use and good looking templates for any HTML element. On the server side of things, nextJS was used to not only make the react application run smoothly but also for rooting and to make API calls to a GitHub repository containing clinical EHR models taken from the Clinical Knowledge Manager (CKM), made freely available by the openEHR Foundation International. The result of this work can be seen in the following image.

Figure 33: The web application main window.

The drop-down menu on the side is essentially the file system for the folders available on the repository. The root directory is the CKM repository button, clicking it shows every child folder that is inside. The archetype class name buttons are the leaf nodes of this file system and clicking them will immediately trigger an API call to the GitHub repository by providing a list with all the files available. Clicking on any file will in consequence trigger yet another API call for a GET request, this time to provide the contents of the file in question. After the file is retrieved it is then parsed. If the file retrieved from the repository is a template the application uses the package xml2json to convert the OPT file into a valid JSON, when the file is an ADL the application uses the program created in the previous section providing a JSON formatted structure seen in figure 31.

Figure 34: The web application window after the file is parsed to a JSON.

This section used the "react-json-view" component, which is available in the following GitHub repository: https://github.com/mac-s-g/react-json-view. This component enables the elegant rendering of JSON data with the added benefit of allowing the user to directly copypaste the entire JSON or just certain sections. JSON structures can be easily examined using the built-in method, which not only provides information about the data type but also indicates the number of items within the object, making it an ideal tool for manual JSON structure checking.

Figure 35: A section of a JSON structure rendered using the react-json-view component.

Chapter 7

Conclusions and future work

7.1 Conclusions

More and more, the support of new technologies for the provision of health care to the patient, in institutions, is emerging. In this regard, the development of hospitalization support systems aims to contribute to improving the quality of patient care. The project described throughout this dissertation had the improvement of organization and archival of clinical data as its main motivation, this being a rather important aspect in the efficient work of an institution that involves health professionals from the most diverse academic backgrounds, from doctors to nurses, health assistants or even IT developers. In pursuit of this, the research question presented below was initially formulated.

"Will this new approach provide a solution for the significant challenges faced by professionals in the medical industry with the increasing complexity of propagation of healthcare record data?"

Combined with the high pace of work evidenced, the standardization of clinical information and the need to make information available to all professionals, this boosted the development of the initially idealized artefact. Multiple goals were initially established in the introductory section of this dissertation. The first two goals included a literature review of articles in the subject area of healthcare data, analysis of past similar work and the choice of research methods to help carry out the work in the most efficient manner. These objectives had a major influence in fully understanding the complexities of the technologies used and the data standards for the openEHR which in consequence helped the planning of a general architecture for the final artefact.

The third objective was the entire artefact creation using the knowledge acquired by all the extensive research accomplished with the state of the art section. As far as the development goes the initial descriptions and procedures necessary carried out, from the prototype to the developed platform. The idealized REST API using ADL files succeeds at creating a standardized approach to transfer high-speed data across any web platform ensuring that other developers can use this API for future application development instead of creating new tools to parse ADL files. The API was tested by building a prototype web application that reads files from a github repository and displays the openEHR data but formatted in a JSON object.

The final goal was a critical analysis of the artefact built. In terms of improvements, the user can view the data associated with a particular EHR in a more user-friendly manner, especially since these structures will include any nested archetypes inside. A big limitation of the original ADL file format was this lack of inclusion of any data related to ”allow_archetype” sections. JSON's powerful and lightweight format proves that it's a vital solution for the interoperability problem present in Healthcare data and the added value of converting these files also helps in making future developers interested in making software using different libraries or programming languages. Regarding the constraints or shortcomings of the built artefact, when it comes to the back end created for this project the GET responses were limited in only using objects from the files that used ADL version 1.4. For the reference model schema, the latest release as of the time of the writing of this dissertation was version 1.1.0 and as such the script only tries to adapt ADL files using this version. Also due to the repository that was used as the main database not having any 2.0 formatted ADL files the decision to adapt to the syntax defined only in the 1.4 version was decided beforehand.

Going back to the main research question proposed in the beginning, It's possible to conclude that this artefact can indeed help provide a solution to the ongoing interoperability problem however there would need to be an active effort to improve the platform and the ADL conversion script.

7.2 Prospect for future work

If the goal of achieving an interoperable database of ADL data endpoints is in order, there needs to be an active effort to update this script to be more adaptable to both future and past versions of the ADL syntax. Therefore, regular updates to the APIs are necessary to ensure compatibility with different ADL versions.

As for the web application, certain aspects could be improved to include more functionality. This prototype is only connected to a single repository, in the future, the user could be given the choice to log in to their desired online storage file system and manage any files directly in the web application by dragging and dropping files or deleting unnecessary files. The user could also have the option to export files to their own device or the ability to select only certain parameters from the JSON. Finally, the creation of a mobile application could also be made using the REACT-NATIVE library or at the very least optimizing the current web application to work better on a mobile phone.

Bibliography

Adetokunbo AA Adenowo and Basirat A Adenowo. Software engineering methodologies: a review of the waterfall model and object-oriented approach. *International Journal of Scientific & Engineering Research*, 4(7):427–434, 2013.

Adobe Communications Team. Waterfall Methodology: A Complete Guide. https://business.adobe.com/blog/basics/waterfall, March 2022. Accessed on 13/04/2023.

Altexsoft. Pacs and vna: What's the difference? https://www.altexsoft.com/blog/pacs-vna/, 2022. Accessed on 15/12/2022.

Ashish Atreja, Steven M Gordon, Daniel A Pollock, Russell N Olmsted, and Patrick J Brennan. Opportunities and challenges in utilizing electronic health records for infection surveillance, prevention, and control. *American journal of infection control*, 36(3):S37–S46, 2008.

Julius Quarshie Azasoo and Kwame Osei Boateng. A retrofit design science methodology for smart metering design in developing countries. In *2015 15th International Conference on Computational Science and Its Applications*, pages 1–7. IEEE, 2015.

Shivani Batra, Shelly Sachdeva, and Subhash Bhalla. Entity attribute value style modeling approach for archetype based data. *Information*, 9(1):2, 2017.

Thomas Beale, Sam Heard, et al. An ontology-based model of clinical information. *Studies in health technology and informatics*, 129(1):760, 2007.

Régis Beuscart, Emmanuel Chazard, Jacques Duchêne, Grégoire Ficheur, Jean-Marie Renard, V Rialle, and N Souf. E-health. *Medical informatics, e-health: fundamentals and applications*, pages 405–427, 2014.

Guthrie S Birkhead, Michael Klompas, and Nirav R Shah. Uses of electronic health records for public health surveillance to advance public health. *Annual review of public health*, 36:345–359, 2015.

Christine A Caligtan and Patricia C Dykes. Electronic health records and personal health records. In *Seminars in oncology nursing*, volume 27, pages 218–228. Elsevier, 2011.

Kitty S Chan, Jinnet B Fowles, and Jonathan P Weiner. Electronic health records and the reliability and validity of quality measures: a review of the literature. *Medical Care Research and Review*, 67(5): 503–527, 2010.

Kristin M Corey, Sehj Kashyap, Elizabeth Lorenzi, Sandhya A Lagoo-Deenadayalan, Katherine Heller, Krista Whalen, Suresh Balu, Mitchell T Heflin, Shelley R McDonald, Madhav Swaminathan, et al. Development and validation of machine learning models to identify high-risk surgical patients using automatically curated electronic health record data (pythia): a retrospective, single-site study. *PLoS medicine*, 15 (11):e1002701, 2018.

José C Delgado. Structural interoperability as a basis for service adaptability. In *Adaptive web services for modular and reusable software development: Tactics and solutions*, pages 33–59. IGI Global, 2013.

Evgeniy Fetisov. Top 30 Companies Using React JS Development. https://jaydevs.com/top-companies-using-react-js/, April 2023. Accessed on 11/04/2023.

Roy T Fielding. Chapter 5: Representational state transfer (rest). 2000. *URL https://www. ics. uci. edu/~fielding/pubs/dissertation/rest_arch_style. htm. Accessed*, pages 06–25, 2020.

Daniela Florescu and Ghislain Fourny. Jsoniq: The history of a query language. *IEEE internet computing*, 17(5):86–90, 2013.

William Goossen and Laura Heermann Langford. Exchanging care records using hl7 v3 care provision messages. *Journal of the American Medical Informatics Association*, 21(e2):e363–e368, 2014.

Luis Guijarro. Semantic interoperability in egovernment initiatives. *Computer Standards & Interfaces*, 31 (1):174–180, 2009.

Sandra Heiler. Semantic interoperability. *ACM Computing Surveys (CSUR)*, 27(2):271–273, 1995.

Masoud Hosseini and Brian E Dixon. Syntactic interoperability and the role of standards. In *Health Information Exchange*, pages 123–136. Elsevier, 2016.

Mohannad A Hussain, Steve G Langer, and Marc Kohli. Learning hl7 fhir using the hapi fhir server and its use in medical imaging with the siim dataset. *Journal of digital imaging*, 31(3):334–340, 2018.

IHTSDO. Snomed ct concept identifier. https://confluence.ihtsdotools.org/display/DOCGLOSS/concept+identifier, 2013. Accessed on 14/12/2022.

Intely. Semantic interoperability: The next big challenge for connected healthcare. https://www.intely.io/blog/semantic-interoperability-the-next-big-challenge-for-connected-healthcare, 2022. Accessed on 14/12/2022.

Md Mohaimenul Islam, Tahmina Nasrin Poly, and Yu-Chuan Jack Li. Recent advancement of clinical information systems: opportunities and challenges. *Yearbook of medical informatics*, 27(01):083–090, 2018.

Jawahar Jagarapu and Rashmin C Savani. A brief history of telemedicine and the evolution of teleneonatology. In *Seminars in Perinatology*, volume 45, page 151416. Elsevier, 2021.

Kirill Konshin. *Next. js Quick Start Guide: Server-side rendering done right*. Packt Publishing Ltd, 2018.

Rita Kukafka, Jessica S Ancker, Connie Chan, John Chelico, Sharib Khan, Selasie Mortoti, Karthik Natarajan, Kempton Presley, and Kayann Stephens. Redesigning electronic health record systems to support public health. *Journal of biomedical informatics*, 40(4):398–409, 2007.

Gilad J Kuperman, Anne Bobb, Thomas H Payne, Anthony J Avery, Tejal K Gandhi, Gerard Burns, David C Classen, and David W Bates. Medication-related clinical decision support in computerized provider order entry systems: a review. *Journal of the American Medical Informatics Association*, 14(1):29–40, 2007.

Elyes Lamine, Wided Guédria, Ariadna Rius Soler, Jordi Ayza Graells, Franck Fontanili, Léonard Janer-García, and Hervé Pingaud. An inventory of interoperability in healthcare ecosystems: Characterization and challenges. *Enterprise Interoperability: INTEROP-PGSO Vision*, 1:167–198, 2017.

Jake Luo, Min Wu, Deepika Gopukumar, and Yiqing Zhao. Big data application in biomedical research and health care: a literature review. *Biomedical informatics insights*, 8:BII–S31559, 2016.

Lingtong Min, Qi Tian, Xudong Lu, Jiye An, and Huilong Duan. An openehr based approach to improve the semantic interoperability of clinical data registry. *BMC medical informatics and decision making*, 18(1):49–56, 2018.

Lorenzo Moja, Koren H Kwag, Theodore Lytras, Lorenzo Bertizzolo, Linn Brandt, Valentina Pecoraro, Giulio Rigon, Alberto Vaona, Francesca Ruggiero, Massimo Mangia, et al. Effectiveness of computerized decision support systems linked to electronic health records: a systematic review and meta-analysis. *American journal of public health*, 104(12):e12–e22, 2014.

Mario Mustra, Kresimir Delac, and Mislav Grgic. Overview of the dicom standard. In *2008 50th International Symposium ELMAR*, volume 1, pages 39–44. IEEE, 2008.

NIH. Snomed ct overview. https://www.nlm.nih.gov/healthit/snomedct/snomed_overview.html, 2016. Accessed on 14/12/2022.

Daniela Oliveira, Ana Coimbra, Filipe Miranda, Nuno Abreu, Pedro Leuschner, José Machado, and António Abelha. New approach to an openehr introduction in a portuguese healthcare facility. In *World Conference on Information Systems and Technologies*, pages 205–211. Springer, 2018.

Daniela Oliveira, Rui Miranda, Nuno Abreu, Pedro Leuschner, António Abelha, Manuel Santos, and José Machado. Management of a pandemic based on an openehr approach. *Procedia Computer Science*, 177:522–527, 2020.

openEHR Foundation. Ehr information model. https://specifications.openehr.org/releases/RM/latest/ehr.html#_class_descriptions_2. Accessed on 08/03/2023.

openEHR Foundation. Archetype Definition Language (ADL) 1.4 Specification. https://specifications.openehr.org/releases/AM/latest/ADL1.4.html, a. Accessed on 08/03/2023.

openEHR Foundation. openEHR Archetype Model, b. URL https://specifications.openehr.org/releases/BASE/Release-1.0.3/architecture_overview.html#_archetype_model_am. Accessed on 19/02/2023.

openEHR Foundation. Archetype Definition Language ontology section. https://specifications.openehr.org/releases/AM/latest/ADL1.4.html#_ontology_section, March 2021. Accessed on 08/03/2023.

openEHR Foundation. openehr information model: Entry class. https://specifications.openehr.org/releases/RM/latest/ehr.html#_the_entry_class, 2022a. Accessed on 29/02/2023.

openEHR Foundation. Adl1.4-openehr specification. https://specifications.openehr.org/releases/AM/latest/ADL1.4.html%13_specialise_section, March 2022b. Accessed on 14/12/2022.

openEHR Foundation. What is openehr. https://www.openehr.org/about/what_is_openehr, 2022c. Accessed on 13/12/2022.

Rene Spronk. The early history of health Level 7. http://www.ringholm.com/docs/the_early_history_of_health_level_7_HL7.html, September 2014. Accessed on 11/04/2023.

Max J Romano and Randall S Stafford. Electronic health records and clinical decision support systems: impact on national ambulatory care quality. *Archives of internal medicine*, 171(10):897–903, 2011.

Susan A Sherer, Chad D Meyerhoefer, and Lizhong Peng. Applying institutional theory to the adoption of electronic health records in the us. *Information & Management*, 53(5):570–580, 2016.

Alen Simec and Magdalena Maglicic. Comparison of json and xml data formats. In *Central European Conference on Information and Intelligent Systems*, page 272. Faculty of Organization and Informatics Varazdin, 2014.

Jake Spurlock. *Bootstrap: responsive web development*. " O'Reilly Media, Inc.", 2013.

Maria Staroselsky, Lynn A Volk, Ruslana Tsurikova, Lisa Pizziferri, Margaret Lippincott, Jonathan Wald, and David W Bates. Improving electronic health record (ehr) accuracy and increasing compliance with health maintenance clinical guidelines through patient access and input. *International journal of medical informatics*, 75(10-11):693–700, 2006.

Fernando N van der Vlist. *A Technicity of Collaboration: Coordinating Distributed Software Development on GitHub*. PhD thesis, University of Amsterdam, 2013.

W3Techs. Usage statistics and market share of javascript for websites. URL https://w3techs.com/technologies/details/cp-javascript. [Accessed 15 Apr. 2023].

Hans Weigand, Paul Johannesson, and Birger Andersson. An artifact ontology for design science research. *Data & Knowledge Engineering*, 133:101878, 2021.

Allen Wirfs-Brock and Brendan Eich. Javascript: the first 20 years. *Proceedings of the ACM on Programming Languages*, 4(HOPL):1–189, 2020.

Cecil L Wittson and Reba Benschoter. Two-way television: helping the medical center reach out. *American Journal of Psychiatry*, 129(5):624–627, 1972.

Kun-Hsing Yu, Andrew L Beam, and Isaac S Kohane. Artificial intelligence in healthcare. *Nature biomedical engineering*, 2(10):719–731, 2018.

Part III
Appendices

Appendix A
Support work

Table 2: DV types and their definitions.

DV type	Data structure
DV_TEXT	It corresponds to a free text that can be of three types: a simple input, a text area or a text editor. This choice is made in the form designer. on the forms, free text is rendered through the draft-js library due to the possibility of text being used in a text editor. If in the JDT the "value" property is on the format of a string instead of structure defined by draft-js this string is processed and transformed to the appropriate format.
DV_CODED_TEXT	Corresponds to a multiple choice field that can be two types: a dropdown or radio buttons. This choice is made in the form designer. the choices possible are defined in the property "itemsList" which corresponds to an array of objects. All objects must have "code" and "text" properties. There is also a property "constraints" that allows defining the choice selected by default.
DV_DURATION	Corresponds to a duration with value and unit. The units are defined in the "units" property. The authorized range of values for each unit is defined in the "units" property as well as the precision, and for Different units may have different ranges and precisions.
DV_DATE_TIME	Corresponds to a date and time.

Continued on the next page

Table 2 – *Continued from previous page*

File Name	Class Type
DV_QUANTITY	Corresponds to a quantity with value and unit. The units are defined in the ”units” property. The authorized range of values for each unit is defined in the ”units” property as well as the precision, and for different units there may be different ranges and precisions.
DV_BOOLEAN	Corresponds to a Boolean that takes one of these two values: Yes or No.
DV_DATE	Corresponds to a date.
DV_TIME	It corresponds to one hour.
DV_COUNT	Corresponds to a count that assumes only whole numbers. At the JDT, the values of items of type DV_COUNT can appear as follows:
DV_IDENTIFIER	Corresponds to an identifier.
DV_MULTIMEDIA	Corresponds to a multimedia (files, images, etc).
DV_ORDINAL	Corresponds to a multiple choice field for scores and scales. The possible choices are defined in the ”itemsList” property that corresponds to an array of objects. All objects must have ”code”, ”value” and ”text”.
DV_INTERVAL<DV_DATE>	Corresponds to a date range.
DV_INTERVAL<DV_COUNT>	Corresponds to a range of counts in number whole.
DV_INTERVAL<DV_DATE_TIME>	Corresponds to a range of quantities. To the units are defined in the ”units” property. The authorized range of values for each unit is defined in the ”units” property as well as the precision, being that for different units there may be different ranges and precisions.
DV_PROPORTION	Corresponds to a range of proportions. So far, the package supports Percent or Unitary type proportions. The type is specified in the ”type” property of items of type DV_PROPORTION. In addition, in The ”numerator” property defines the range of authorized values for the proportion.

Table 3: Data Structures created for each DV type.

DV type	Data structure
DV_TEXT	{”value”: null}
DV_CODED_TEXT	{”value”: null}
DV_DURATION	{”value”: {”value”:null, ”unit”: null}}
DV_DATE_TIME	{ ”value”: { ”date”: null,”time”: null}{
DV_QUANTITY	{ ”value”: { ”value”: null,”unit”: null}}
DV_BOOLEAN	{”value”: null}
DV_URI	{”value”: null}
DV_DATE	{”value”: null}
DV_TIME	{”value”: null}
DV_COUNT	{”value”: null}
DV_IDENTIFIER	{”value”: null}
DV_MULTIMEDIA	{”value”: null}
DV_ORDINAL	{”value”: null}
DV_INTERVAL	{ ”value”: { ”start”: null,”end”: null}}
DV_INTERVAL<DV_COUNT>	{ ”value”: { ”start”: null, ”end”: null}}
DV_INTERVAL<DV_DATE_TIME>	{ ”value”: {”date”: {”start”: null,”end”: null}”time”: {”start”: null,”end”: null}}}
DV_INTERVAL<DV_TIME>	{ ”value”: { ”start”: null,”end”: null } }
DV_INTERVAL<DV_QUANTITY>	{ ”value”: { ”value”: { ”start”: null,”end”: null }, ”unit”: null } }
DV_PROPORTION	{ ”value”: null } }

Appendix B
Listings

Table 4: List of Files in the CKM repository used to work on the JSON parser.

File Name	Class Type
openEHR-EHR-CLUSTER.address.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.adhoc_cluster_heading.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.adverse_reaction_event_summary.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.anatomical_location.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.anatomical_location_circle.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.anatomical_location_precise.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.anatomical_location_relative.v2.adl	CLUSTER
openEHR-EHR-CLUSTER.anatomical_pathology_exam.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.art_container_details.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.biobank_storage.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.bioreagent.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.boston_bowel_preparation_scale.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.case_identification.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.cessation_attempts.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.change.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.citation.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.clinical_evidence.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.conditional_medication_rules.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.consent_details.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.country_visited.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.ctcae.v0.adl	CLUSTER

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-CLUSTER.device.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.device_details.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.dietary_phytochemicals.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.distribution.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.document_entry_metadata.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.dosage.v2.adl	CLUSTER
openEHR-EHR-CLUSTER.dwelling.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.ear_cleaning.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.education_record.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.electronic_communication.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.environmental_conditions.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-abdomen.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-anterior_chamber_eye.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-anus.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-aqueous_humour.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-auscultation-bowel_sounds.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-auscultation-breath_sounds.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-auscultation.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-breast.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-breasts.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-cardiovascular_system.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-chest.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-conjunctiva.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-cornea.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-cranial_nerves.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-ear.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-ears.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-external_auditory_canal.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-eye.v0.adl	CLUSTER

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-CLUSTER.exam-eyelid.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-eyes.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-face.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-finger.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-fingernail.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-foot.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-fundus_eye.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-hand.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-heart.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-inspection-cervix.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-inspection-rectum.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-inspection-vagina.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-inspection.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-iris.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-lens.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-lower_limb.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-lung.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-middle_ear.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-mouth.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-muscle.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-neck.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-nervous_system.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-optic_disc.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-palpation-cervix.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-palpation-prostate.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-palpation-rectum.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-palpation-uterus.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-palpation-vagina.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-palpation.v0.adl	CLUSTER

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-CLUSTER.exam-penis.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-peripheral_nervous_system.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-posterior_chamber_eye.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-pupil.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-respiratory_system.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-sclera.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-scrotum.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-skin.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-testicle.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-throat.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-thyroid.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-toe.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-toenail.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-tongue.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-tooth.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-tympanic_membrane.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-upper_limb.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam-vulva.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_blastocyst.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_blastocyst.v1 toâ¦	CLUSTER
openEHR-EHR-CLUSTER.exam_burn.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_embryo.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_embryo.v1 to Pubâ¦	CLUSTER
openEHR-EHR-CLUSTER.exam_faeces.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_hydration.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_lesion.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_oocyte.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_oocyte.v1 to Pubâ¦	CLUSTER

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-CLUSTER.exam_tendon_reflexes.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_wound.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_zygote.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.exam_zygote.v1 to Pubâ¦	CLUSTER
openEHR-EHR-CLUSTER.exclusion_exam.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.exclusion_symptom_sign.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.family_prevalence.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.fetal_biometry.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.fetus_abdominal.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.fetus_vaginal.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.financial_record.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.free_text.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.gait.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.genetic_variant_presence.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_conversion_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_copy_number_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_deletion_insertion_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_deletion_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_duplication_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_insertion_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_inversion_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_repeated_sequence_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_substitution_variant.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.genomic_variant_result.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.health_event.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.hip_arthroplasty_component.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.housing_record.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.housing_record.v1 to â¦	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-abnormality-adnexal_mass.v0.adl	CLUSTER

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-CLUSTER.imaging_exam-abnormality.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-bladder.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-fallopian_tube.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-fetus.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-gestational_sac.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-liver.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-lymph_node.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-lymph_node_group.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-ovary.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-rectouterine_pouch.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-scrotum.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-testicle.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-uterine_body.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam-uterine_cervix.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_exam.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_finding.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.imaging_series.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.information_resource.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.inspection_body_fluid-sputum.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.inspection_body_fluid-urine.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.inspection_body_fluid.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.inspired_oxygen.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.interpreter_request.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.interpreter_request.vâ¦	CLUSTER
openEHR-EHR-CLUSTER.issue.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.knowledge_base_reference.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.laboratory_test_analyte.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.laboratory_test_panel.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.language.v1.adl	CLUSTER

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-CLUSTER.level_of_exertion.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.lymph_node_metastases.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.macronutrients.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.macroscopy_colorectal_carcinoma.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.macroscopy_lung_carcinoma.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.media_file.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.medication.v2.adl	CLUSTER
openEHR-EHR-CLUSTER.medication_authorisation.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.medication_order_summary.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.medication_supply_amount.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.micronutrients.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.microscopy_breast_carcinoma.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.microscopy_colorectal_carcinoma.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.microscopy_lymphoma.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.microscopy_melanoma.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.microscopy_prostate_carcinoma.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.myringoplasty.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.myringotomy.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.notifiable_condition.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.occupation_record.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.oedema.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.operative_procedure.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.organisation.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.other_significant_conditions.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.outbreak_identification.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.overcrowding_screening.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.person.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.pews_original.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.physiological_monitoring.v0.adl	CLUSTER

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-CLUSTER.problem_qualifier.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.procedure_preparation.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.promis_bank_v10_anxiety.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.promis_bank_v10_depression.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.promis_bank_v10_fatigue.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.promis_bank_v10_sleep_disturbance.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.promis_bank_v11_pain_interference.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.promis_bank_v20_ability_participate.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.promis_bank_v20_physical_function.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.promis_scale_v12_global_health.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.reference_sequence.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.refraction_details.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.religion.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.religion.v1 to Published	CLUSTER
openEHR-EHR-CLUSTER.sade.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.sequencing_assay.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.service_direction.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.service_direction.v1 â¦	CLUSTER
openEHR-EHR-CLUSTER.simple_variant.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.skin_sensation.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.specimen.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.specimen_container.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.specimen_measurements.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.specimen_preparation.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.specimen_transport.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.strategy.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.structured_name.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.symptom_sign.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.therapeutic_direction.v1.adl	CLUSTER

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-CLUSTER.timing_daily.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.timing_nondaily.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.tnm-pathological.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.tnm.v1.adl	CLUSTER
openEHR-EHR-CLUSTER.tnm.v1 to Published	CLUSTER
openEHR-EHR-CLUSTER.tos.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.translocation_variant.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.tumour_colorectal_staging_non_tnm.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.tumour_invasion.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.tumour_resection_margins.v0.adl	CLUSTER
openEHR-EHR-CLUSTER.waveform.v0.adl	CLUSTER
openEHR-EHR-COMPOSITION.advance_care.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.adverse_reaction_list.v1.adl	COMPOSITION
openEHR-EHR-COMPOSITION.care_plan.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.encounter.v1.adl	COMPOSITION
openEHR-EHR-COMPOSITION.event_summary.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.family_history.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.health_summary.v1.adl	COMPOSITION
openEHR-EHR-COMPOSITION.lifestyle_factors.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.medication_list.v1.adl	COMPOSITION
openEHR-EHR-COMPOSITION.notification.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.obstetric_history.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.pregnancy_summary.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.prescription.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.problem_list.v2.adl	COMPOSITION
openEHR-EHR-COMPOSITION.progress_note.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.report-procedure.v1.adl	COMPOSITION
openEHR-EHR-COMPOSITION.report-result.v1.adl	COMPOSITION
openEHR-EHR-COMPOSITION.report.v1.adl	COMPOSITION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-COMPOSITION.request.v1.adl	COMPOSITION
openEHR-EHR-COMPOSITION.review.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.self_monitoring.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.social_summary.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.therapeutic_precautions.v0.adl	COMPOSITION
openEHR-EHR-COMPOSITION.transfer_summary.v1.adl	COMPOSITION
openEHR-EHR-COMPOSITION.vaccination_list.v0.adl	COMPOSITION
openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-ADDRESS.address.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-ADDRESS.electronic_communication-provider.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-ADDRESS.electronic_communication.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CAPABILITY.individual_credentials.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.biometric_identifier_iso.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.birth_data_additional_detail_br.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.high_level_address_other_data_br.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.identifier_other_details.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.individual_credentials_iso.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.individual_provider_credentials_iso.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.person_additional_data_br.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.person_additional_data_iso.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.person_birth_data_iso.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.person_death_data_iso.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.person_identifier-provider.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.person_identifier.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.person_other_birth_data_br.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.person_other_death_data.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.provider_identifier.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-CLUSTER.registration_other_data.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-ITEM_TREE.person_details.v0.adl	DEMOGRAPHIC

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-DEMOGRAPHIC-ORGANISATION.organisation.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-PARTY_IDENTITY.organisation_name.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-PARTY_IDENTITY.person_name-individual_provider.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-PARTY_IDENTITY.person_name.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-PERSON.person-patient.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-PERSON.person.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-ROLE.healthcare_consumer.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-ROLE.healthcare_provider_organisation.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-ROLE.individual_provider.v0.adl	DEMOGRAPHIC
openEHR-DEMOGRAPHIC-ROLE.third_party_payer.v0.adl	DEMOGRAPHIC
openEHR-EHR-ACTION.care_plan.v0.adl	ACTION
openEHR-EHR-ACTION.health_education.v1.adl	ACTION
openEHR-EHR-ACTION.imaging_exam.v0.adl	ACTION
openEHR-EHR-ACTION.informed_consent.v0.adl	ACTION
openEHR-EHR-ACTION.medication.v1.adl	ACTION
openEHR-EHR-ACTION.procedure.v1.adl	ACTION
openEHR-EHR-ACTION.review.v0.adl	ACTION
openEHR-EHR-ACTION.screening.v0.adl	ACTION
openEHR-EHR-ACTION.service.v0.adl	ACTION
openEHR-EHR-ACTION.transfusion.v0.adl	ACTION
openEHR-EHR-ADMIN_ENTRY.arrival.v0.adl	ADMIN_ENTRY
openEHR-EHR-ADMIN_ENTRY.episode_institution.v0.adl	ADMIN_ENTRY
openEHR-EHR-ADMIN_ENTRY.legal_constraint.v0.adl	ADMIN_ENTRY
openEHR-EHR-ADMIN_ENTRY.transfer_of_care.v0.adl	ADMIN_ENTRY
openEHR-EHR-ADMIN_ENTRY.translation_requirements.v1.adl	ADMIN_ENTRY
openEHR-EHR-ADMIN_ENTRY.travel_event.v0.adl	ADMIN_ENTRY
openEHR-EHR-ADMIN_ENTRY.triage.v0.adl	ADMIN_ENTRY
openEHR-EHR-EVALUATION.absence.v2.adl	EVALUATION
openEHR-EHR-EVALUATION.advance_care_directive.v1.adl	EVALUATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-EVALUATION.advance_intervention_decisions.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.adverse_reaction_risk.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.alcohol_consumption_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.art_cycle_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.biospecimen_summary-embryo.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.biospecimen_summary-oocyte.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.biospecimen_summary-semen.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.biospecimen_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.cause_of_death.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.clinical_synopsis.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.communication_capability.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.consumer_note.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.container.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.contraceptive_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.contraindication.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.death_details.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.device_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.differential_diagnoses.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.education_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.estimated_date_delivery.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.ethnicity.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.exclusion_global.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.exclusion_specific.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.exposure.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.family_history.v2.adl	EVALUATION
openEHR-EHR-EVALUATION.financial_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.food_nutrition_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.gambling_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.gender.v1.adl	EVALUATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-EVALUATION.goal.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.health_risk.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.housing_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.immunisation_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.infant_feeding.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.infectious_disease_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.infectious_exposure.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.last_menstrual_period.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.living_arrangement.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.medication_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.menstruation_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.obstetric_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.occupation_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.physical_activity_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.physical_characteristics.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.precaution.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.pregnancy_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.problem_diagnosis.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.reason_for_encounter.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.recommendation.v2.adl	EVALUATION
openEHR-EHR-EVALUATION.sexual_health_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.smokeless_tobacco_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.social_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.source.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.substance_use_summary.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.support_network.v0.adl	EVALUATION
openEHR-EHR-EVALUATION.tobacco_smoking_summary.v1.adl	EVALUATION
openEHR-EHR-EVALUATION.travel_summary.v0.adl	EVALUATION
openEHR-EHR-INSTRUCTION.care_plan_request.v0.adl	INSTRUCTION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-INSTRUCTION.health_education_request.v0.adl	INSTRUCTION
openEHR-EHR-INSTRUCTION.informed_consent_request.v0.adl	INSTRUCTION
openEHR-EHR-INSTRUCTION.medication_order.v3.adl	INSTRUCTION
openEHR-EHR-INSTRUCTION.notification.v0.adl	INSTRUCTION
openEHR-EHR-INSTRUCTION.service_request.v1.adl	INSTRUCTION
openEHR-EHR-INSTRUCTION.supplemental_oxygen_order.v0.adl	INSTRUCTION
openEHR-EHR-INSTRUCTION.transfusion_order.v0.adl	INSTRUCTION
openEHR-EHR-OBSERVATION.abbey_pain_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.abc_score_massive_transfusion.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.abc_stroke_risk_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.abcd2_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.acoustic_reflex_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.acvpu.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.adl.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.adverse_reaction_monitoring.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.affected_body_surface_area-burn.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.affected_body_surface_area.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.age.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.air_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.alcohol_audit.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.alcohol_intake.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.aldrete_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.alsfrs_r.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.alvarado_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.apgar.v2.adl	OBSERVATION
openEHR-EHR-OBSERVATION.asa_status.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.atria_bleeding_risk.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.atria_stroke_risk.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.audiogram_result.v0.adl	OBSERVATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-OBSERVATION.audiology_speech_test_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.auditory_brainstem_response_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.avpu.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.behavioural_observation_audiometry_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.bishop_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.blood_pressure.v2.adl	OBSERVATION
openEHR-EHR-OBSERVATION.body_composition.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.body_mass_index.v2.adl	OBSERVATION
openEHR-EHR-OBSERVATION.body_segment_area.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.body_segment_circumference.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.body_segment_length.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.body_surface_area.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.body_temperature.v2.adl	OBSERVATION
openEHR-EHR-OBSERVATION.body_weight.v2.adl	OBSERVATION
openEHR-EHR-OBSERVATION.braden_scale.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.braden_scale_q.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.bristol_stool_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.bvc.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.cage.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.capillary_refill.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.caprini_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.ccs_angina_status.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.cgas.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.chadsvas_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.cheop_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.chest_circumference.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.child_growth.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.child_pugh_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.clinical_frailty_scale.v1.adl	OBSERVATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-OBSERVATION.clinical_frailty_scale2.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.comfort_behaviour_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.condition_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.conference.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.container.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.cormack_lehane.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.cow_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.crb_65.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.critical_pain_observation_tool.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.crusade_bleeding.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.curb_65.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.das28-CRP.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.das28.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.dash_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.demo.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.device_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.diabetic_wound_wagner.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.downton_fall_risk_index.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.easi_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.ecg_result.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.ecog.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.edinburgh_pnd_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.embryo_assessment.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.epic_cp.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.esas_r.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.estimated_glomerular_filtration_rate.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.exam.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.exclusion_pregnancy.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.exposure_screening.v0.adl	OBSERVATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-OBSERVATION.fact_g-Hep.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fact_g.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.faecal_output.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fagerstrom.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fetal_growth.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fetal_heart-monitoring.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fetal_heart.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fetal_movement.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fitzpatrick_skin_type.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fluid_balance.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fluid_input.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fluid_output-blood.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fluid_output.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.food_item.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.four_at.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.four_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.fundoscopic_examination.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.gad_7_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.gestation.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.glasgow_coma_scale.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.glasgow_coma_scale_paediatric.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.gpaq.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.grace_admission.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.grace_discharge.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.growth_velocity.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.hannallah_pain_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.harris_hip_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.has_bled.v0.adl	OBSERVATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-OBSERVATION.head_circumference.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.hearing_screening_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.heart_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.height.v2.adl	OBSERVATION
openEHR-EHR-OBSERVATION.hip_circumference.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.hirsutism_scales.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.honos.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.hscore.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.humpty_dumpty_falls_risk_assessment_tool.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.iga_eczema_treat.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.imaging_exam_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.infant_feeding.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.intermacs_profile.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.intraocular_pressure.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.intravascular_pressure.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.ipss.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.iss-revised.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.iss.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.issue_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.jugular_venous_pressure.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.kads.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.karnofsky_performance_status_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.laboratory_test_result.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.leg_length_discrepancy.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.malinas_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.management_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.mantoux.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.mayo_score.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.medication_screening.v0.adl	OBSERVATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-OBSERVATION.medication_statement.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.menstrual_diary.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.menstruation.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.modified_aldrete_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.modified_barthel_index.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.modified_rankin_scale.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.msfc_score.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.mskcc_bowel_function_instrument.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.mskcc_motzer.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.mst.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.murray_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.must.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.neonatal_skin_risk_assessment.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.news2.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.news_uk_rcp.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.nine_hole_peg_test.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.nutrition_intake.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.nyha_heart_failure.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.oucher_pain_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.paced_auditory_serial_addition_test.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.padss.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pasi_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pefr_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.penetration_aspiration_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pews.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pf_ratio.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pga_eczema_treat.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.phfrat1.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.phq_9.v0.adl	OBSERVATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-OBSERVATION.physical_activity.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.poem_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pregnancy_status.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pregnancy_test.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.procedure_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.progress_note.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.promis.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pulse.v2.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pulse_deficit.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.pulse_oximetry.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.qsofa_score.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.reach_b.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.refraction.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.respiration.v2.adl	OBSERVATION
openEHR-EHR-OBSERVATION.richmond_agitation_sedation_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.rinne_weber_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.sara_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.scoff_questionnaire.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.scorad_index.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.six_cit.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.skeletal_age.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.soas_r.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.soas_re.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.social_context_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.sofa_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.speech.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.spirometry_result.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.story.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.stroke_scale_neurological_assessment.v0.adl	OBSERVATION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	Class Type
openEHR-EHR-OBSERVATION.substance_use.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.substance_use_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.symptom_sign_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.tanner.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.telecommunication.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.temperature.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.testicular_volume.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.third_party_observation.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.timed_25_foot_walk.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.travel_screening.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.trunk_impairment_scale.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.tympanogram_226hz.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.tympanogram_hf.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.urinalysis.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.uterine_contractions.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.visual_acuity.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.visual_field_measurement.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.waist_circumference.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.waist_height_ratio.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.waist_hip_ratio.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.waterlow_score.v0.adl	OBSERVATION
openEHR-EHR-OBSERVATION.ygtss_revised.v1.adl	OBSERVATION
openEHR-EHR-OBSERVATION.ymrs.v0.adl	OBSERVATION
openEHR-EHR-SECTION.adhoc.v1.adl	SECTION
openEHR-EHR-SECTION.adverse_reaction_list.v0.adl	SECTION
openEHR-EHR-SECTION.conclusion.v0.adl	SECTION
openEHR-EHR-SECTION.diagnostic_reports.v0.adl	SECTION
openEHR-EHR-SECTION.family_history.v0.adl	SECTION
openEHR-EHR-SECTION.immunisation_list.v0.adl	SECTION

Continued on the next page

Table 4 – *Continued from previous page*

File Name	**Class Type**
openEHR-EHR-SECTION.lifestyle_risk_factors.v0.adl	SECTION
openEHR-EHR-SECTION.medication_list.v0.adl	SECTION
openEHR-EHR-SECTION.problem_list.v0.adl	SECTION
openEHR-EHR-SECTION.referral_details.v0.adl	SECTION
openEHR-EHR-SECTION.soap.v0.adl	SECTION
openEHR-EHR-SECTION.vital_signs.v0.adl	SECTION