FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Interoperable Assistive Technologies

Eduardo Miguel Moreira Guedes Osório

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Rui Filipe Lima Maranhão de Abreu (PhD)

Co-supervisor: Liliana da Silva Ferreira (PhD)

15th February 2013

Interoperable Assistive Technologies

Eduardo Miguel Moreira Guedes Osório

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Nuno Honório Rodrigues Flores (PhD)

External Examiner: António Carlos da Silva Abelha (PhD)

Supervisor: Rui Filipe Lima Maranhão de Abreu (PhD)

15th February 2013

Resumo

O número crescente de idosos na nossa sociedade aumenta o esforço imposto aos recursos dos sistemas de saúde tradicionais. Soluções na área de Ambient Assisted Living (AAL) procuram reduzir o impacto desta tendência bem como permitir aos idosos continuar a viver o máximo tempo possível no seu ambiente caseiro com uma boa qualidade de vida e independência. No entanto, dada a natureza diversa dos produtos e serviços existentes nesta área, é importante assegurar a interoperabilidade entre estes de modo a proporcionar soluções flexíveis e adequadas às necessidades particulares de cada indivíduo.

O objectivo desta dissertação é apresentar uma possível solução para este problema, utilizando Registos Electrónicos de Saúde e normas de informação de saúde. Para este fim, foi implementado um repositório para armazenar os registos em formato Extensible Markup Language (XML) seguindo a norma openEHR, bem como uma aplicação na linguagem de programação Java para criar, gerir e pesquisar estes registos no repositório e um motor de interface (Mirth Connect) foi utilizado para fornecer a capacidade de transformação de mensagens implementado em JavaScript entre diferentes normas, particularmente Health Level Seven versão 2.x (HL7 v2.x) e openEHR.

Abstract

The increasing number of elderly citizens in our society puts an increasing strain on traditional healthcare systems' resources. Ambient Assisted Living (AAL) solutions seek to reduce the impact of this tendency as well as provide a way for the elderly to continue to live in their home environment as long as possible with a good quality of life and independence. However, due to the diverse nature of existing products and services, ensuring interoperability between them is important in order to provide flexible and adequate solutions for an individual's particular needs.

The goal of this dissertation is to present a possible solution for this problem using Electronic Health Records (EHRs) and health information standards. For this purpose a repository was implemented to store records in Extensible Markup Language (XML) format following the openEHR standard, as well as an application in the Java programming language to create, manage and query these records in the repository and an interface engine (Mirth Connect) was used to provide message transformation capabilities implemented in JavaScript between different standards, particularly Health Level Seven version 2.x (HL7 v2.x) and openEHR.

Acknowledgements

I would like to acknowledge and thank everyone who helped me through this dissertation, as well as the rest of my academic journey. Firstly, to FEUP and Fraunhofer Portugal AICOS as a whole for affording me the opportunity and conditions to accomplish this work and gain a new working experience. In particular, I would like to acknowledge my supervisor Rui Maranhão, my co-supervisor Liliana Ferreira and Filipe Sousa for their guidance, support and patience throughout this process.

Of course, none of this would ever be possible without my family and friends so a very special thank you must go out to my parents for their love and sacrifice that allowed me to reach this stage of my life and to the rest of my family and friends for their counsel and support even those who I haven't been able to spend much time with.

Eduardo Miguel Moreira Guedes Osório

Contents

List of Figures

List of Tables

Abbreviations

AAL	Ambient Assisted Living
AAL4ALL	Ambient Assisted Living For All
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
DDK	Driver Development Kit
EHR	Electronic Health Record
HL7	Health Level Seven
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IM	Information Model
REST	REpresentational State Transfer
RM	Reference Model
SDK	Software Development Kit
URL	Uniform Resource Locator
XML	EXtensible Markup Language
XSLT	EXtensible Stylesheet Language Transformation

Chapter 1

Introduction

1.1 Motivation

The number of elderly people is steadily increasing in all developed countries and this fact will put an ever increasing strain on traditional methods of providing social support, health and care services as the total population of countries is expected to decrease, yet the percentage of elderly people increases [1]. Due to the increasing costs and demands in resources necessary to address this problem, coupled with changing lifestyle preferences, the possibility of continuing to live and receive care in the accustomed home environment is becoming more and more attractive.

These findings have motivated research in the field of Ambient Assisted Living (AAL), whose primary aim is to enable an independent life for the elderly in their own homes and provide support for people with special needs.

It is in this context that the Ambient Assisted Living for All (AAL4ALL) [2] project emerged as a joint effort from several academic, industry and research institutions. Fraunhofer Portugal Research Center for Assistive Information and Communication Solutions (FhP-AICOS) is one of the involved research institutions and it is where work on this dissertation was developed. AAL4ALL aims at creating a standard platform upon which to build AAL solutions ensuring interoperability between all its components and external healthcare systems.

This dissertation arises, within the scope of AAL4ALL, to present a possible solution to the interoperability problem within AAL systems by using Electronic Health Records and health transaction standards.

1.2 Objectives

The primary goal of this dissertation will be to propose and develop a prototype system that will enable interoperability between applications such as eHealthCom within an AAL system.

At the start of this work it will be necessary to acquire knowledge in the fields of eHealth and Ambient Assisted Living, learning their state of the art and research activities, the diferences between them and how they intersect. More specifically, it will be important to review some of the available health storage and transaction standards.

Working towards a solution, the problem will have to be more clearly defined and a proposal for the architecture to use will have to be presented. Its development will hinge on a choice of available technologies based on which a prototype following the proposed architecture must be produced.

1.3 Structure of the Dissertation

Besides the introduction, this dissertation contains a further five chapters. In chapter 2, a state of the art review is presented, introducing some concepts in the field of eHealth as well as describing important standards and terminologies that can be used. It also presents various products and services in the AAL field to provide an overview of the existing kinds of information and applications that might be integrated into an AAL solution.

A detailed description of the problem at hand is presented in chapter 3, as well as the intended solution's architecture and Chapter 4 provides a detailed description of the development process, from choosing the used technologies to presenting components and functionalities.

An evaluation of the defined objectives and main difficulties found throughout the course of this work as well as a performance evaluation of the resulting prototype is presented in chapter 5, while finally the conclusions drawn from this dissertation and prospects for future work are presented in chapter 6.

Chapter 2

State of the Art

2.1 Introduction

In order to satisfy the demands of an increasing elderly segment of the population that wishes to continue living independently within their home environment, it is necessary to provide solutions that enable not only monitoring of health conditions and communication with care providers, especially for chronic diseases typical of this age group, but also the prevention of social isolation and various other factors of overall wellness.

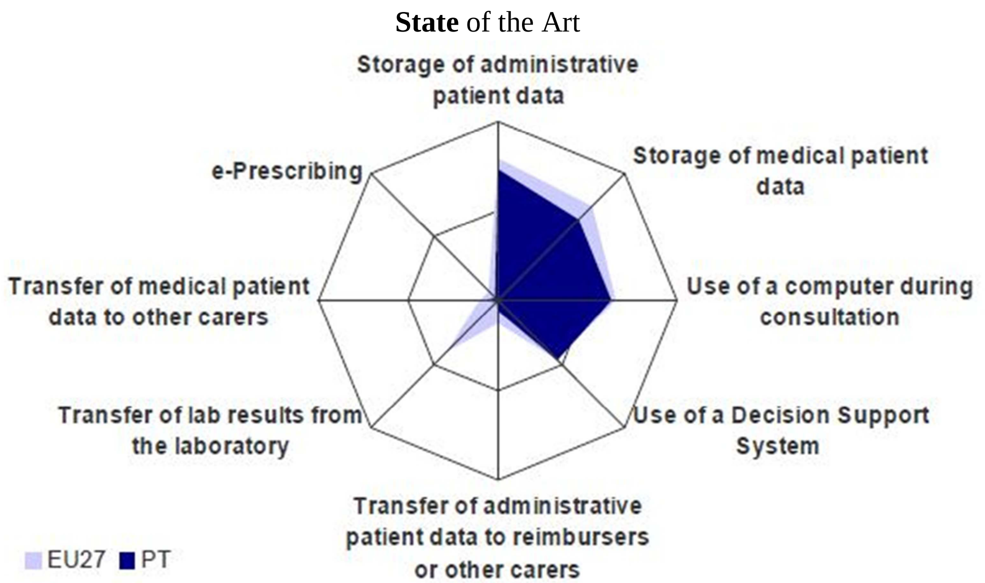
This chapter presents some important definitions and standards in the fields of eHealth and Ambient Assisted Living, Portugal's position within the context of the European Union and various products, services and research projects currently available or in development in these fields.

2.2 eHealth

The World Health Organization (WHO) currently defines eHealth as "the use of information and communication technologies (ICT) for health. Examples include treating patients, conducting research, educating the health workforce, tracking diseases and monitoring public health" [3]. Major advancements in ICT have been made in recent years to provide tools and methods for the development of new ways of efficiently providing healthcare services.

A study [4], performed by the European Comission to classify the eHealth implementation level in European Union (EU) member states, shows that general practitioners in Portugal are considered average eHealth performers in the EU27, receiving a score of 2.1 on a scale from 0 (not used) to 5 (used by professionals across the country).

As Figure 1 [4] shows, some of the indicators used in the study, such as use of a computer during consultation (3.2) and use of a Decision Support System (2.3), scored on par with the



Indicators	Compound indicators of eHealth use (cf. indicator annex for more information). Index scores ranging from 0 (not used at all) to 5 (used by all GPs in the country).
Source	empirica, Pilot on eHealth Indicators, 2007.

Figure 1 – eHealth use in Portugal in 2007

EU27 average, yet none surpassed it. The transfer of patient data and e-Prescribing were serious points of concern, achieving very low scores at the time.

Since then, however, there has been some progress made in the eHealth area and measures taken by the Portuguese government, such as a mandatory e-Prescription system (PEM, from the Portuguese *Prescrição Electrónica de Medicamentos*) in place since the 1st of August of 2011, have helped to improve Portugal's standing.

A more recent study [5], focused on eHealth benchmarking for acute hospitals, shows that some indicators have clearly surpassed the EU average while most of the others remain below but close to that average, accompanying the EU's natural progression in this field, as can be seen in Figure 2 [5].

Rules and regulations regarding the aforementioned Portuguese e-Prescribing system are only mandatory for software products that result in medical prescriptions to be reimbursed by the Portuguese National Health System (SNS, from the Portuguese *Serviço Nacional de Saúde*) and aim to help control medication invoicing and improve health care service efficiency and effectiveness.

However, this is not the only project launched by the Portuguese government in the field of eHealth. A National Network of Users (RNU, from the Portuguese *Rede Nacional de Utentes*), a medical appointments online booking service (eAgenda) and an integrated surgery registration management system (SIGIC, from the Portuguese *Sistema Integrado de Gestão de Inscritos para Cirurgia*) were created and recently integrated for access through the Portuguese Citizen Portal (Portal do Utente).

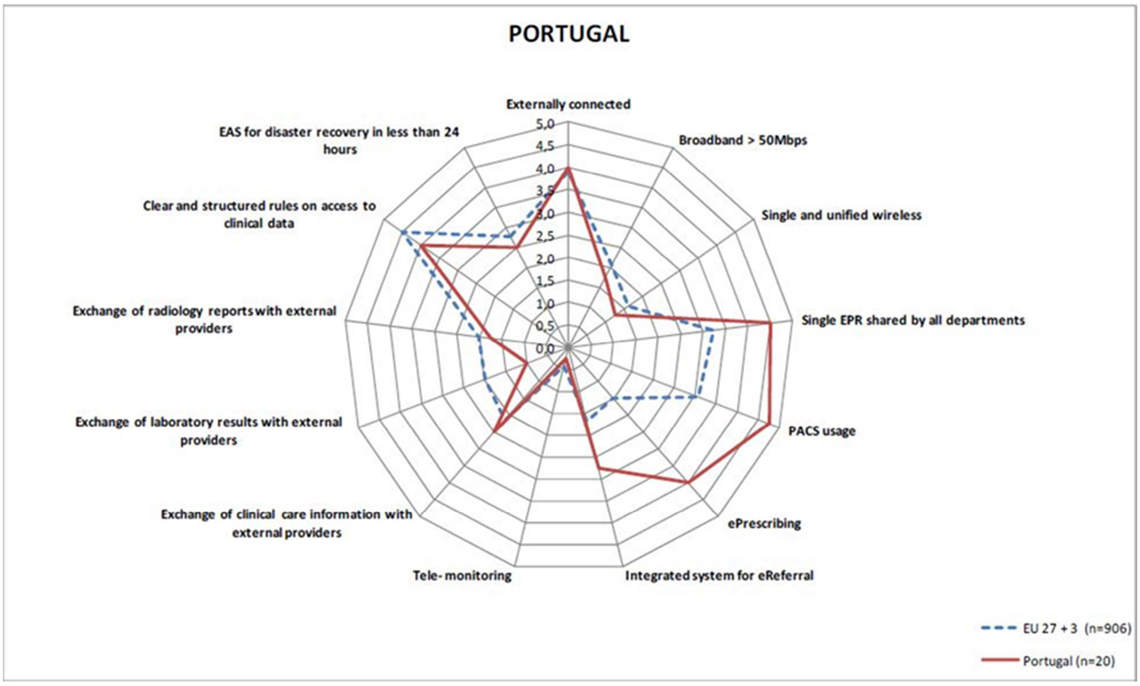


Figure 2 – eHealth use in Portugal in 2011

2.2.1 Electronic Health Records

An Electronic Health Record (EHR) is defined by HIMSS Analytics as "A subset of each care delivery organization's EMR, presently assumed to be summaries like ASTM's Continuity of Care Record (ASTM CCR) or HL7's Continuity of Care Document (HL7 CCD), is owned by the patient and has patient input and access that spans episodes of care across multiple CDOs within a community, region, or state (or in some countries, the entire country)" [6].

An EHR should possess certain characteristics. It should be patient-centred, long-term (possibly birth to death), include all types of carers and provider institutions, and not only previous events, but also decisions, plans, goals, orders and evaluations.

2.2.1.1 OpenEHR

OpenEHR is a set of open specifications for an EHR architecture, not a software application. It aims to enable semantic interoperability of health information within and between EHR systems in a non-proprietary format.

The necessary clinical knowledge concepts are captured outside the software in archetypes, the types of which support the recording of common clinical activities, such as observations, evaluations, instructions and actions. Their creation is almost purely a task for clinicians, so they may define for themselves the breadth, depth and complexity needed in the health record for their practice. These archetypes can be simple or complex and "contain a maximum data set about each clinical concept, including attendant data such as: protocol, or method of measurement; related events; and context that is required for the clinical data to be interpreted accurately" [7]. Data built according to these archetypes are stored in 'composition' structures within an EHR, which have their own archetypes and are similar to a document resulting from a performed clinical event, such as a consultation record or discharge summary.

According to [7] "Aggregations of archetypes are combined in openEHR 'templates' in order to capture the data-set corresponding to a particular clinical task, such as an ICU discharge summary or antenatal visit record. When clinicians look at templates, the information contained within them inherently makes sense and doesn't require significant training for interested clinicians to be able to create templates for their own purposes – be it domain, organization or purpose specific. Templates can be used to build generic forms to represent the approximate layout of the EHR in a practical sense, and these can be used by vendors to contribute to their interface development".

The openEHR specification can be implemented in various ways [7]:

- Scalable EHRs
    - Personal Health Records
    - Small/medium/large organizations to regional or state clinical record systems
    - National eHealth programs
- Message-based, web-service, middleware applications
- Integrating existing clinical systems, including virtual federation of data for research or public health purposes

There are also a number of ways in which openEHR differentiates itself from other EHR models:

- It is an open source initiative, which means the openEHR specifications are available for free under an open license;
- It maintains a clear separation between the technical model which forms the basis of the software and the clinical knowledge model;
- It allows active and direct contribution of healthcare professionals in the development of clinical knowledge models;
- It is terminology agnostic, meaning it allows any or all terminologies to be used through archetypes or templates, which in turn provide context to minimize the need for post-coordination and complexity;
- Combining terminologies with archetypes enables powerful possibilities for semantic querying of repository data;
- Archetypes may begin in any language and be translated to multiple other languages for usage in different countries;
- It consists of only generic data types, structures and a small number of generic patterns, which results in a small, sustainable and stable information model. This allows a clinical data repository to be totally independent of software applications and technology changes;

- It is comparatively easy to implement because it requires little infrastructure, the required software is small, due to the compact and stable object-oriented reference model on which it is based;
- It is in continuing development and enhancement, based on international collaborator feedback;
- Archetypes can be created, agreed upon and used as the basis for consistent data exchange between various systems, providers and even countries;
- Its development is based on a collaborative model between a broad international community of clinicians and software engineers, rather than a standards-based path. This does not mean standards are rejected, they are just not a part of the development process, which has allowed it to be relatively rapid and pragmatic while maintaining formal and rigorous processes through the oversight of both Architectural and Clinical Review Boards, composed of world experts in their fields.

OpenEHR is currently being used in various countries in both research and commercial activities. “Research on openEHR is being conducted in Sweden, Australia, United Kingdom, USA, Sri Lanka and Spain. Commercial development is occurring in Australia, United Kingdom’s NHS Connecting for Health, Netherlands, Belgium, Sweden, Turkey and the USA” [7].

2.2.2 Plataforma de Dados de Saúde – Portuguese Electronic Health Record

The Portuguese Electronic Health Record is based on a web platform, currently in development by the Comission for Clinical Informatics (CIC, from the Portuguese *Comissão para a Informatização Clínica*), created by a dispatch from the Health Secretary of State (SES, from the Portuguese *Secretário de Estado da Saúde*) near the end of 2011, and the Ministry of Health’s Shared Services (SPMS, from the Portuguese *Serviços Partilhados do Ministério da Saúde*), that provides a central clinical information storage and sharing system in keeping with the requirements of the National Comission of Data Protection (CNPD, from the Portuguese *Comissão Nacional de Proteção de Dados*). It allows access to registered users’ information by health professionals throughout the SNS. Each access to this information is restricted and registered in an access history.

This platform targets the entire part of Continental Portugal (it does not include the islands of Madeira and Azores), it aggregates cross-institutional clinical data from hospitals and from primary care units and will comply with the epSOS core services of Patient Summary based on the Portuguese Patient Summary (RCU2, from the Portuguese *Resumo Clínico Único do Utente*), in the context of epSOS II.

The *Plataforma de Dados de Saúde* (PDS) [8] is the national electronic health record data sharing facility, utilizes webservice technology to link both old and new applications and through it provides different Portals for access to information by different stakeholders.

There will be a total of 4 Portals available:

- The Citizen Portal (Portal do Utente)

- This portal was launched on the 31st of May, 2012. It allows the insertion of data such as emergency contacts, health data, habits, medications, allergies, diseases, authorizations/audits or health clinic/family health unit contacts. It also enables access to services such as the RNU, the SIGIC, eAgenda to book medical appointments or request renewal of prescriptions for chronic patients and SIM-Cidadão to leave suggestions, complaints, complements and aknowledgements to the SNS.

- The Health professional Portal (Portal do Profissional)
- The Institutional Portal (Portal Institucional)
- The International Portal (Portal Internacional) – within the scope of epSOS pilot project participation

2.2.3 European Patients Smart Open Services

The European Patients Smart Open Services (epSOS) [9] is a project that aims to design, build and evaluate a service infrastructure to demonstrate cross-border interoperability between electronic health record systems in Europe. It started on the 1st of July, 2008 and will extend until the 31st of December, 2013 and encompasses 23 European countries, of which 20 are European Union member states.

The primary goals in the intended cross-border eHealth services are the improvement of healthcare quality and safety for citizens travelling between European countries. Its focus is in the development of a practical eHealth framework and infrastructure that allows secure access to patient health information among different European healthcare systems and can contribute to the reduction of medical errors, provide quick access to documentation and thus, potentially life-saving information, as well as reduce the repetition of diagnostic procedures.

The concepts developed within the epSOS project's framework are subjected to an extensive practical testing phase spanning a period of one year and its cross-border eHealth services are tested in the different areas.

The Patient Summary, which is a standardized set of basic medical data such as general information about the patient (name, date of birth, gender), a summary of the most important clinical patient data (allergies, current medical condition, implants or major surgical procedures), a list of the current medication and information about the Patient Summary itself such as time of generation and updates, is tested in a first phase as well as cross-border use of electronic prescription services.

In a second phase, the integration of emergency services and of the European Health Insurance Card (EHIC) and patient access to data is tested.

2.2.4 Personal Health Records

A Personal Health Record (PHR) can be defined as an electronic application that can store and manage health information of an individual introduced by himself or another duly authorized user, such as a physician or family member. It also allows the sharing of this information in a secure and confidential environment.

This type of record belongs to the user whose information it stores and may contain information considered relevant to an individual's health and well-being, yet not directly related to any specific consultation or examination.

Examples of PHR systems include Google Health and Microsoft HealthVault.

2.2.4.1 Google Health

Google Health [10] was the system previously used by the eHealthCom platform to store a PHR but has since been retired on 1 January 2012 by Google for not having the broad impact expected in the community. It allowed the storage of patient information, such as:

- Wellness
- Problems
- Medications
- Allergies
- Test Results
- Procedures
- Immunizations
- Insurance

The Google Health API was based on a subset of the ASTM CCR standard, supported .NET, Java, Python and PHP programming languages and allowed the integration of third-party systems.

2.2.4.2 Microsoft HealthVault

Microsoft HealthVault [11] is a PHR that lets the user store, gather, and share his health information online. The information stored therein is structured according to a health information transaction standard (ASTM CCR and HL7 CCD) and so, can be easily transferred and allows interaction with various healthcare entities, such as hospitals, physicians, and retail pharmacies.

Microsoft provides a SDK for solution providers to build applications (in .NET, Java, Windows Phone, iOS and Android languages) that communicate with the HealthVault platform servers to determine data access rights and to manage all types of data in the system. It also provides a DDK to build device drivers for HealthVault compatible devices, such as blood pressure monitors, glucometers and more.

With the exception of special relationships, however, Microsoft HealthVault accounts are not available outside the United States and it supports the storage of various types of information such as Conditions, Files, Health History, Measurements and Medications but does not support Observation Patterns, Agenda, Alerts or Reminders.

2.2.5 Electronic Medical Records

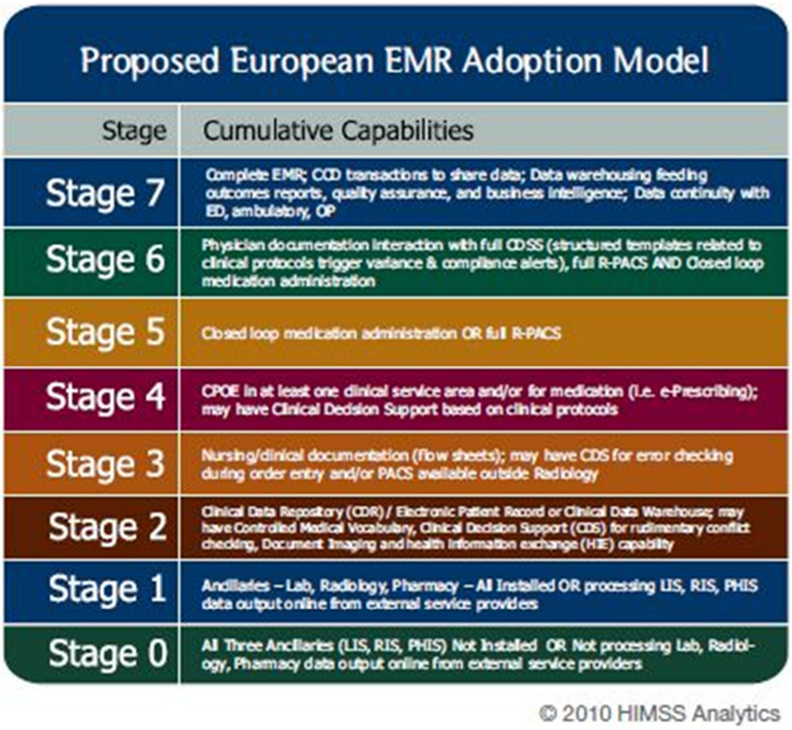
An Electronic Medical Record (EMR) can be defined as “An application environment composed of the clinical data repository, clinical decision support, controlled medical vocabulary, order entry, computerized provider order entry, pharmacy, and clinical documentation applications. This environment supports the patient’s electronic medical record across inpatient and outpatient environments, and is used by healthcare practitioners to document, monitor, and manage healthcare delivery within a care delivery organization (CDO). The data in the EMR is the legal record of what happened to the patient during their encounter at the CDO and is owned by the CDO” [6].

This record is the sole responsibility of the care delivery organization and cannot be accessed or modified by the patient.

In order to understand the level of EMR capabilities in healthcare providers in Europe, the Healthcare Information and Management Systems Society (HIMSS) Analytics Europe created the European EMR Adoption Model based on the EMR Adoption Model already established across the U.S. and Canada. They also developed a methodology and algorithms for automatic scoring of care delivery organizations and comparisons of their progress towards EHR integration. This model is composed of eight stages, described in Figure 3 [12].

The following is a list of EMR projects under implementation or use in Portugal:

- *Processo Clínico Electrónico Único (Sistema Regional de Saúde – Região Autónoma da Madeira)*
- *Rede Telemática da Saúde – Aveiro Digital*
- *UPIP – Urgência Pediátrica Integrada do Porto*
- *Processo Clínico Electrónico – Hospital Geral de Santo António*
- *HSL.ICU (Informação Clínica do Utente)*
- *Unidade de Hematologia do Centro Hospitalar de Coimbra*



Proposed European EMR Adoption Model	
Stage	Cumulative Capabilities
Stage 7	Complete EMR; CCD transactions to share data; Data warehousing feeding outcomes reports, quality assurance, and business intelligence; Data continuity with ED, ambulatory, OP
Stage 6	Physician documentation interaction with full CDSS (structured templates related to clinical protocols trigger variance & compliance alerts), full R-PACS AND Closed loop medication administration
Stage 5	Closed loop medication administration OR full R-PACS
Stage 4	CPOE in at least one clinical service area and/or for medication (i.e. e-Prescribing); may have Clinical Decision Support based on clinical protocols
Stage 3	Nursing/clinical documentation (flow sheets); may have CDS for error checking during order entry and/or PACS available outside Radiology
Stage 2	Clinical Data Repository (CDR) / Electronic Patient Record or Clinical Data Warehouse; may have Controlled Medical Vocabulary, Clinical Decision Support (CDS) for rudimentary conflict checking, Document Imaging and health information exchange (HIE) capability
Stage 1	Ancillaries – Lab, Radiology, Pharmacy – All Installed OR processing LIS, RIS, PHIS data output online from external service providers
Stage 0	All Three Ancillaries (LIS, RIS, PHIS) Not Installed OR Not processing Lab, Radiology, Pharmacy data output online from external service providers

© 2010 HIMSS Analytics

Figure 3 – European EMR Adoption Model

2.3 Health Terminologies and Ontologies

Terminologies are “best described as language-oriented artifacts that relate the various senses or meanings of linguistic entities with each other. Terminologies are generally built to serve well-defined purposes like document retrieval, resource annotation, the recording of mortality and morbidity statistics, or health services billing” [13] and ontologies are “advertized to precisely describe domains in detail and to employ these descriptions in many types of applications, ranging from natural language processing to logic reasoning and decision support systems” [13].

2.3.1 International Classification of Diseases

Created by the World Health Organization (WHO), the International Classification of Diseases is “the standard diagnostic tool for epidemiology, health management and clinical purposes” [14]. Its main purpose is the classification of diseases and other health problems recorded on various types of health and vital records and, in so doing, enables diagnostic information to be stored and retrieved for various purposes, such as clinical, epidemiological and quality evaluation and comparison, as well as the compilation of mortality and morbidity statistics and resource allocation decision-making by countries.

The current version "ICD-10 was endorsed by the Forty-third World Health Assembly in May 1990 and came into use in WHO Member States as from 1994. The 11th revision of the classification has already started and will continue until 2015" [14].

2.3.2 SNOMED CT

Formed in a joint development between the College of American Pathologists (CAP) and the NHS in England by the convergence of SNOMED RT and the United Kingdom's Clinical Terms Version 3 in 1999, the Systemized Nomenclature of Medicine - Clinical Terms (SNOMED CT) is a comprehensive, multilingual clinical terminology. It is currently owned, maintained and distributed by the International Health Terminology Standards Development Organization (IHTSDO) as a result of intellectual property rights transfer from the CAP to the SNOMED SDO in 2007 in the formal creation of the IHTSDO.

SNOMED CT can be defined as "A work of clinical terminology for coding, retrieving and analyzing data about health and health care" [15] and as being "Comprised of codes, terms and relationships, for use in precisely recording and representing clinical information across the scope of health care" [15]. It provides core general terminology for electronic health records and as such, can be used as an integral part of their production.

2.3.3 Unified Medical Language System

Produced by the U.S. National Library of Medicine (NLM), the Unified Medical Language System is defined as "a set of files and software that brings together many health and biomedical vocabularies and standards to enable interoperability between computer systems" [16]. Its main usage is "linking health information, medical terms, drug names, and billing codes across different computer systems" [16], making the development of health information applications easier.

The most important tool of the UMLS is the Metathesaurus, which is a set of terms and codes from many vocabularies, including CPT [17], ICD-10-CM, LOINC [18], MeSH [19], RxNorm [20], and SNOMED CT.

2.4 Health Transaction Standards

A standard, as defined by the International Organization for Standardization (ISO), is a "document, established by consensus and approved by a recognized body, that provides, for common and repeated use, rules, guidelines, or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context" [21]. Given this definition, standards are extremely important for interoperability between health information systems, because they can define data models for the structurization of health information, communication channels to be used and what kind of information can be sent in order to ensure successful communication between those systems.

2.4.1 ASTM CCR

The Continuity of Care Record (CCR) [22] is a patient health summary standard used for health information transaction between different systems and developed by the standards development organization (SDO) ASTM International with contributions by members of the American Academy of Family Physicians (AAFP), the Massachusetts Medical Society (MMS), the American Academy of Pediatrics (AAP) and other healthcare instiutions and vendors.

It is defined by the ASTM itself as “a core data set of the most relevant administrative, demographic, and clinical information facts about a patient’s healthcare, covering one or more healthcare encounters” [22] whose primary use is “to provide a snapshot in time containing the pertinent clinical, demographic, and administrative data for a specific patient” [22].

The CCR standard defines an XML structure schema consisting of a header, a footer and a body composed of 17 optional sections in order to ensure interchangeability of electronic CCRs between health information systems.

The defined XML structure schema for this standard is as follows:

- Header (ID, Language, Version, Date/Time, Patient ID, From, To and Purpose)
- Body (optional sections, which include Payer, Advance Directives, Support, Functional Status, Problems, Family History, Social History, Alerts, Medications, Immunizations, Vital Signs, Results, Procedures, Encounters, Plan of Care and Healthcare Providers)
- Footer (Actors, References, Comments and Signatures)

2.4.2 HL7 v2.x

Created in 1989, the HL7 v2.x [23] standard is the most widely used health transaction standard in the world, including 95% of healthcare organizations in the USA.

It was designed to provide a data exchange framework between disparate health information systems and achieves this by allowing the negotiation of a portion of needs on an interface-by-interface basis. Currently, this standard defines a series of messages to exchange administrative, logistical, financial and clinical processes information, and can contain the following data:

- Patient Demographics
- Patient Charges and Accounting
- Patient Insurance and Guarantor
- Clinical Observations
- Encounters including Registration, Admission, Discharge and Transfer

- Orders for Clinical Service (Tests, Procedures, Pharmacy, Dietary and Supplies)
- Observation Reporting including Test Results
- The synchronization of Master Files between systems
- Medical Records Document Management
- Scheduling of Patient Appointments and Resources
- Patient Referrals – Specially messages for primary care referral
- Patient Care and problem-oriented records

HL7 v2.x messages use a delimiter-based textual syntax, commonly known as encoding rules seven (ER7), allow for optional fields and additional portions of messages in order to support local variations in data interchanges. These messages are sent in response to trigger events and their names are derived from the message type and a trigger event. Their two main components are:

- Message syntax
    - "Message syntax describes the overall structure of messages and how the different parts are recognized. Each message is composed of segments in specified sequence, each of which contains fields also in a specified sequence; these fields have specified data types" [24].
- Data types
    - "Data types are the building blocks of the fields and may be simple, with a single value, or complex, with multiple components. These components themselves have data types, which can be simple or complex, leading to subcomponents" [24].

All versions of this standard are generally compatible with earlier versions because it allows applications to ignore unexpected elements in messages.

2.4.3 HL7 CDA

First released in 2000 and also known as HL7 v3 [25], this standard was strongly influenced by government and medical information users. It was created to address some issues found in the HL7 v2.x standard, such as:

- Implied, instead of consistent, data model
- No formal methodologies to model data
- None or ill-defined applications and message roles
- Too much flexibility, translating into lack of a full solution

In light of these issues, the HL7 v3 uses a well-defined methodology based on a reference information data model (HL7 RIM), which "specifies the grammar of V3 messages and, specially, the basic building blocks of the language (nouns, verbs, etc), their permitted relationships, and Data Types. The RIM is not a model of health care, although it is healthcare-specific, nor is it a model of any message, although it is used in messages" [24].

Both CDA v1 and v2 are composed by a header containing meta-data for computer processing and a body containing human-readable text or images. However, the CDA v2 body can be unstructured, like the CDA v1 (and thus allowing compatibility with it), or one or more structured sections, each composed by a single narrative block containing XML markup that can be rendered in human-readable format.

This standard is, however, not compatible with the widely adopted HL7 v2.x standard which makes its implementation costly and usually limited to applications with no legacy communication requirements, no historical use of HL7 v2.x or in locations with high government enforcement of its usage.

2.4.4 HL7 CCD

This standard was created as a joint effort between ASTM International and HL7 in order to address the incompatibility between their previously defined standards: ASTM CCR and HL7 CDA.

The CCD standard utilizes templates, which are a set of constraints, to define the usage of CDA elements that comprise the document while the clinical data itself is defined by the CCR standard. “Each template may have further supporting templates, as required” [26].

The following is a list of CCD templates (excludes supporting templates):

- Header
- Purpose
- Problems
- Procedures
- Family History
- Social History
- Payers
- Advance Directives
- Alerts
- Medications
- Immunizations
- Medical Equipment
- Vital Signs
- Functional Stats
- Results
- Encounters
- Plan of care

The CCD is composed by a header, which includes the document unique identifier, source, intended recipients, purpose and metadata, and a body, which is in turn composed by the main sections of the CCR standard.

The joining of the CDA standard's flexibility with the CCR standard's inflexible data structure allows the documents to be read by a physician and understood by health information systems at the same time. The CCD standard also allows for broad compatibility and easy adoption by health information systems that already use the CDA and the CCR standard.

The CCD is defined as the recommended standard for exchange of health summary information by the HIMSS and the Healthcare Information Technology Standards Panel (HITSP).

2.5 Ambient Assisted Living

Although the term "Ambient Assisted Living" does not currently have a single unifying definition, it is mainly used in research activities and generally understood to refer to a field focused on providing solutions, based on Information and Communication Technologies (ICT), to enable independent living of elderly people and provision of health and care services to them in their own home environment. According to [1] "An AAL product can be everything, starting from hardware components and ending in complex system solutions that integrate devices as well as services".

The following is a brief description of some products and services developed in the field of AAL or that may be incorporated into an AAL system in Portugal.

2.5.1 AAL Products

2.5.1.1 VirtualECare and iGenda

VirtualECare [27] is a research project developed by the University of Minho's Informatics Department which aims to connect a series of different environments focused on the provision of personalized healthcare. Domestic, external, hospital or day care and family or care provider environments are all considered and linked through group decision support tools in order to provide a high level environment in which all parties are connected with the goal of improving patients' safety and quality of life.

To achieve this, the patient should be monitored constantly, at home or outside, in order to quickly detect threats to the patient's safety and, if necessary, locate him/her and take immediate action. In addition, this project focuses on learning mechanisms capable of determining user preferences and habits and replicating them autonomously, freeing the user from monotonous and possibly difficult tasks due to physical limitations, all with the goal of improving patient quality of life.

The same institution also develops the iGenda [28] project which focuses on memory limitations that arise with ageing. The concept of Memory Assistants is used to approach this issue with the goal of developing environments that can help users remember events and occurrences in a proactive and reactive fashion. It also aims to encourage the development of relaxation and socialization activities through interaction with other users, in order to avoid loneliness situations. The current implementation essentially boils down to the development of calendar managers capable of registering important events, such as consultations, and filling vacant time slots with socialization activities envolving available family members or nearby users.

2.5.1.2 PRK-Treatment – Exercise System for Parkinson Continuous Treatment and Rehabilitation

PRK-TREATMENT – Exercise System for Parkinson continuous treatment and rehabilitation [29] is a project currently under development by a consortium comprising INOVA+, Associação Portuguesa de Doentes de Parkinson (APDP), Artica Telemedicina and Asociación Parkinson Madrid under the Eurostars program. It focuses on the improvement of Parkinson treatment and patient care by providing a tool for therapy and rehabilitation exercises and tests monitored by healthcare professionals.

This project has been running since 2010 and is expected to deliver tangible results at the beginning of 2013. It has a few distinguishable objectives which include the creation of an Exercises Platform addressing different categories of treatment by providing customizable exercises specifically designed for Parkinson patients to monitor their evolution and the inclusion of social networking tools to promote social integration and a common network of Parkinson treatment.

2.5.1.3 AAL@HOME and AAL@INSTITUTIONAL

Developed by the company PLUX, the aal@home [30] system consists in a wearable system of biosignal acquisition integrating one or two independent wireless sensor nodes to provide continuous monitoring and portability. A mobile phone is used as a mobile gateway, with data buffering and re-transmission capabilities for when it is out of network coverage, between the sensor nodes and a remote monitoring station and as a self-monitoring tool by providing local on-screen visualization of the monitored parameters. The remote monitoring station is provided with a database with information of each monitored user, stores their collected signals sent by the mobile gateways and provides a web interface for the caregivers to access the information.

The aal@home system is also capable of detecting anomalous situations and sending notifications and alarms to the monitoring station, ensuring the possibility of a fast and effective assistance to the patient whenever required.

The aal@institutional system, developed by the same company is similar to aal@home but directed for use in hospitals or other healthcare facilities. It is designed for localization, fall detection, heart rate monitoring and device removal detection and is based on a wireless sensor network (WSN) architecture composed by a coordinator computer, routers and end-devices.

A coordinator computer is installed on each floor of the building and communicates with the routers through radio frequencies. The routers are themselves strategically positioned throughout the various floors and ensure the communication between the coordinator computer and the end-devices, which in turn are wearable devices for patients and caregivers. Patient end-devices can be necklaces or chest bands, providing heart rate monitoring, fall detection, device removal detection, localization and an emergency button, or wristwatches, providing only device removal detection, localization and an emergency button. Caregiver end-devices, on the other hand, can be placed in the chest and provide information on current anomalous situations.

The coordinator computer functions in the same manner as the aal@home monitoring station, storing and providing access to monitoring information and generating alarms for anomalous situations.

2.5.1.4 ExaNoNeedle and ExaAllinOne

The ExaNoNeedle [31] is a drug delivery system created by Exa4Life [32] which is based on iontophoresis. This means the device is used to administer medication through the patient’s skin via the use of a low electrical current, increasing its permeability and allowing a fast introduction of a charged substance to the skin tissue. This method of drug delivery therefore requires no needle, is painless and non-invasive.

The ExaNoNeedle system in particular offers a few advantages:

- Provides the electrical current frequencies most used by healthcare specialists
- User-friendly interface with a choice of language
- Ability to treat two patients or two locations of the same patient simultaneously
- Provides pre-defined rehabilitation programs developed by healthcare specialists
- Non-invasive and localized drug delivery, avoiding effects on unaffected organs

This device is also small and portable for use in AAL which will reduce the resources required for healthcare services.

ExaAllinOne is a system developed by the same company to monitor human vital signs at home, reducing healthcare costs and making it suitable for AAL environments. It allows the monitorization of the following vital signs [31]: heart rate, body temperature, blood pressure, oxygen’s concentration in blood, respiration rate and body mass index.

Figure 4 [31] shows a simple diagram demonstrating the system's functionality.

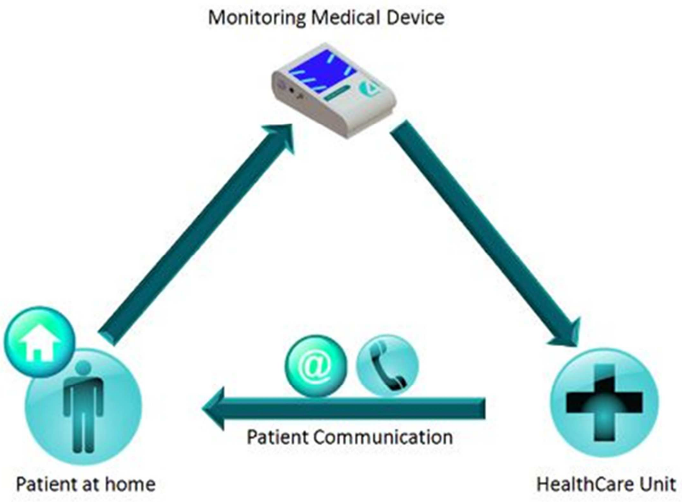


Figure 4 – ExaAllinOne communication diagram

2.5.1.5 ALERT® Patient Data Management System

The Alert® Patient Data Management System (Alert® PDMS) is a software system that enables the collection of healthcare data from monitoring devices and its storage in the patient's EMR. Its main features are [33]:

- Visualization and storage of values collected by clinical devices and external systems in a patient EHR
- Ability to chronologically present and compare the collected data with data from other clinical variables
- Functionalities for human data validation
- Direct order entry through the PDMS system and reutilization of the data by severity scoring systems
- Ability to set parameters by clinical environment, user profile, personal preferences and patient's clinical condition
- Ability to collect data and present monitorization, hemodynamic, neurological and ventilation parameters

Although this system was designed for clinical and healthcare facility environments, it could serve as a basis for an AAL system with similar storage capabilities for monitorization.

2.5.1.6 NetCare – "*Telemetria médica para cuidados de saúde em contínuo: aplicação de redes e sensores wireless aos serviços da saúde*"

The NetCare – "*Telemetria médica para cuidados de saúde em contínuo: aplicação de redes e sensores wireless aos serviços da saúde*" project was carried out by a Consortium formed by INOVA+, FEUP, HSS and SPECULUM under the IDEIA – *Apoio à Investigação e*

Desenvolvimento Empresarial Aplicada program run by AdI. The goal of this project was to develop and integrate the required components for a constant patient monitoring system throughout various locations (hospital, site of an accident, medical emergency vehicle and the patient's home).

The developed system is based on two components [34]:

- Go-wireless kit, composed by a set of Personal Sensor Units (PSUs) that are able to communicate with ad-hoc local networks, a base unit to collect and provide information, a mobile device to visualize patients' vital signs and a set of range expanders to increase local network coverage
- NetCare system server to enable monitorization of the patient and recording of his information, communication between doctors, paramedics, healthcare providers and patients, visualization of the patient's history and access to a medical decision support system

The results obtained from this project demonstrate the feasibility of the concept and the potential for exploitation of associated products. The proposed objectives were exceeded, although there are still issues to address before the go-wireless kit reaches the necessary maturity level for a commercial product.

2.5.1.7 Look4MySounds

The Look4MySounds [35] system can be seen as the implementation of a digital stethoscope that is designed to monitor respiratory sounds and detect abnormal ones in a home environment over extended periods of time.

The patient's respiratory sounds are collected through a stethoscope with the help of a family member according to a pre-defined order of auscultation. The sounds are stored in a memory card folder and a brief medical report with all the sound classifications is made when the recording is finished. This data can then be transferred to the patient's personal computer (PC) or remotely sent to a doctor or healthcare unit.

2.5.1.8 Low Power Wireless Acquisition Module for Wearable Health Monitoring Systems

The prototype device presented in [36] is designed to increase power efficiency in Wearable Health Monitoring Systems (WHMS) by providing a solution capable of monitoring local temperature, physical activity and electrocardiogram (ECG) with minimum power requirements.

To achieve this, flexible dry electrodes, mounted directly on the bottom side of the device's small flexible printed circuit board (PCB) and with comparable performance to conventional electrodes, are used and the device wirelessly transmits the collected data to a base station connected to a PC to be processed and displayed.

The use of dry electrodes also provides a higher degree of comfort to the users because it eliminates the need for skin preparation with electrolyte gel and possible skin irritations arising from it on long monitoring periods.

2.5.1.9 MagTag – A Wearable Device for Localization Applications

MagTag [37] is a very small device developed for localization of patients in a clinical environment. For wearability and usability purposes, this device was restricted to a size similar to a wristwatch, to be easily cleanable and removal resistant and to clinical environment compliant materials. It also allows wireless battery recharging when placed on a base station.

In terms of data transmission, the MagTag device communicates with the existing Wireless Local Area Network (WLAN) infrastructure in order to provide patient tracking capabilities to control centers.

Despite being developed specifically for clinical use, the device can be adapted for use in a home environment and its included movement sensors can be used for more features than just localization, such as activity monitoring. In this way it could play an important role in the care of elderly and chronic patients in their own domestic environment.

2.5.1.10 Enhanced Complete Ambient Assisted Living Experiment

Developed under the European AAL Joint Programme [38], the Enhanced Complete Ambient Assisted Living Experiment (eCAALYX) was a three year project whose main goal was the creation of a solution to improve the quality of life of the elderly with diagnosed chronic diseases. To this end, it was meant to promote user autonomy as well as reduce hospitalization costs for this segment of the population.

Fraunhofer Portugal AICOS was one of the participants in this project and one of its responsabilities was the development of eCAALYX's Home Monitoring System (Figure 5 [39]) which consisted of a home gateway to interpret a set of observation patterns defined by caregivers and a set-top box to visualize the health parameters determined by a health kit prescribed by a healthcare professional.

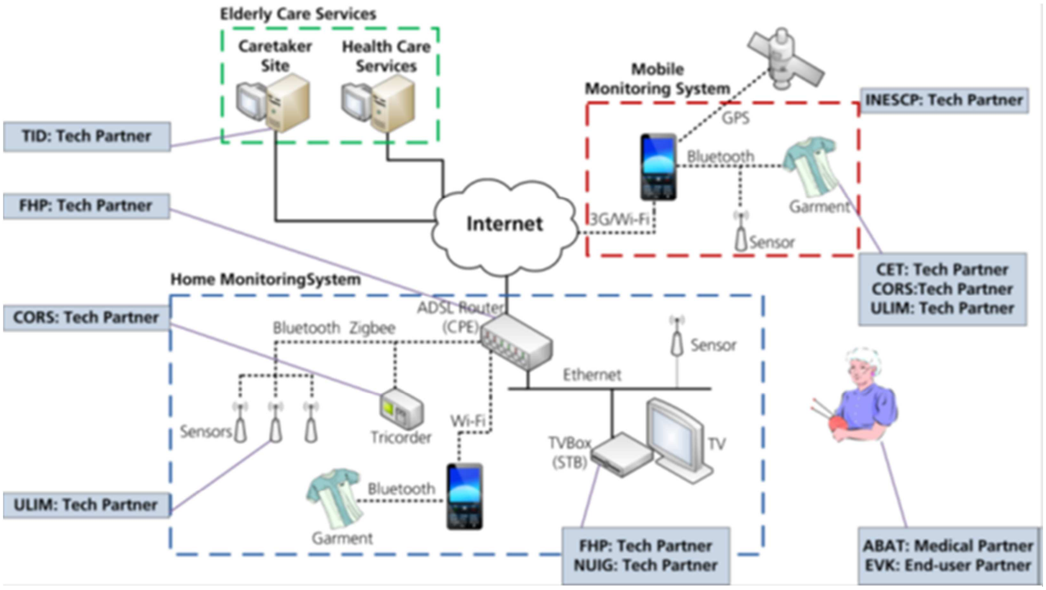


Figure 5 – eCAALYX architecture

2.5.1.11 Mover – Activity Monitor and Fall Detector for Android Mobile Devices

Developed by Fraunhofer Portugal AICOS, Mover [40] is an Android application for user activity monitoring and fall detection. Its main purpose is to help combat increasingly sedentary lifestyles to avoid obesity and heart diseases, especially among the elderly, and to detect the risk of falls. To achieve this, Mover constantly monitors the mobile phone's accelerometer data to determine the user's average activity level and classify it in 6 defined states (sleeper, sitter, lagger, walker, mover and hyper) by comparison with the other community members.

The Mover application runs constantly in the mobile phone's background, unless it is deactivated, and in executed tests allowed more than 20 hours of activity without recharging or approximately 10 hours with fall detection enabled.

2.5.1.12 eHealthCom

The eHealthCom [41] project emerged with the aim of developing a simple prototype of a Caretaker Server to ensure interoperability with different health information systems. It tried to achieve this by using health transaction standards, such as ASTM CCR, to exchange health information between a Home System, the Caretaker Server and the Google Health PHR.

The Home System is constituted by a Set-Top Box and a Home Gateway which retrieves and stores health and personal information from and to the Caretaker Server through Web Services. The system's architecture can be described by Figure 6 [41].

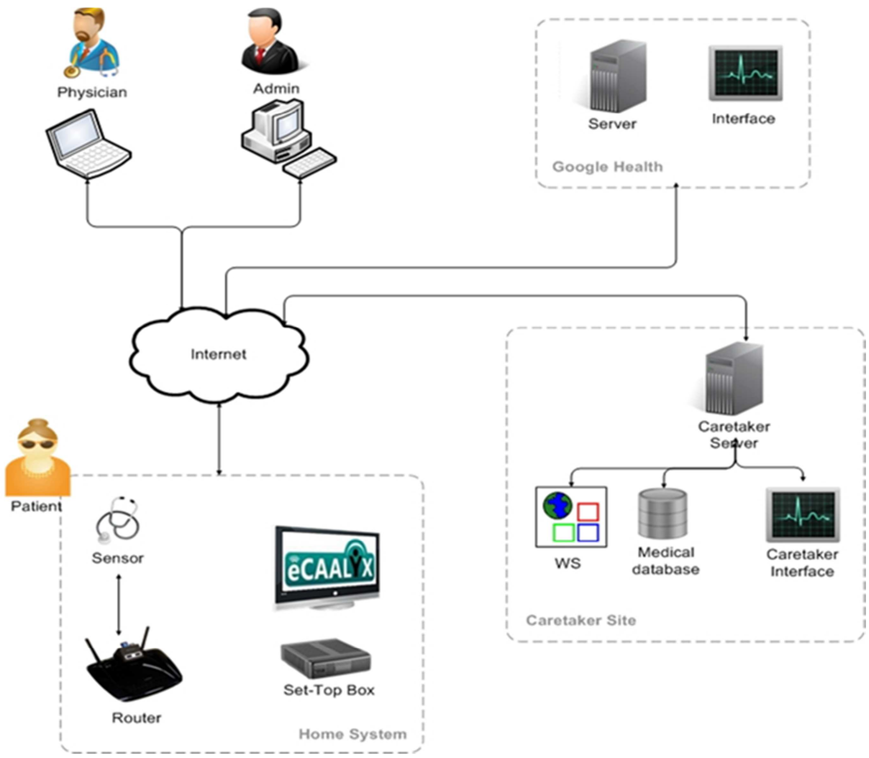


Figure 6 – eHealthCom architecture

However, Google Health has since been retired for not having the expected broad impact on the community and so, Fraunhofer Portugal AICOS is currently looking into understanding the difficulties and challenges to the adoption of PHRs and finding a viable alternative for the system.

The resulting prototype of the project was developed using the Grails [42] framework, which is a high-productivity web framework based on the Groovy [43] language that embraces the coding by convention paradigm, but is designed specifically for the Java [44] platform. This framework was chosen to simplify and speedup the developing process, since the goal was to create a simple prototype.

The Web Services required for the communication between the Home System and the Caretaker Server were implemented using the SOAP [45] method, wherein the Caretaker Server is the service provider and the Home System is the requester. The messages are constructed using the XML structure schema defined by the ASTM CCR and HL7 CCD standards.

The Web Interface that allows users to interact with the Caretaker Server supports authentication to avoid unauthorized access to user information. It implements three user profiles with different functionalities:

- Administrator
    - Creation, deletion and update of:
        - Patient profiles
        - Physician profiles
        - Pathologies
        - Sensors
        - Vital signs
        - Vital sign fields
    - Assignment of a physician to a patient
    - Search for patient or physician profiles

- Physician
    - Visualization and analysis of patient alarms
    - Search for:
        - Patient alarms
        - Patients
        - A patient’s prescribed medication
        - A patient’s assigned questionnaires
        - Questionnaire results
    - Creation, deletion and update of:
        - Appointments
        - Medications

        - Treatments
        - Conclusions
        - Questionnaires
    - Listing of:
        - Appointments
        - Assigned patients
        - A patient’s prescribed medication
        - A patient’s assigned questionnaires
        - Questionnaire results
        - Conclusions
        - Created questionnaires
        - Reminders
    - Selection of a specific patient
    - Visualization of patient profiles and measurements
    - Definition of search parameters for patient measurements
    - Assignment and unassignment of:
        - Pathologies
        - Conclusions
        - Questionnaires

- Patient
    - Listing of:
        - Appointments
        - Alarms
        - Reminders
        - Prescribed medication
    - Search for alarms
    - Visualization of measurements
    - Definition of search parameters for measurements

The eHealthCom Caretaker Server database was built on the Enterprise Architect [46] software and was developed using the MySQL [47] database and the InnoDB [48] storage engine due to support transactions and foreign key constraints.

2.5.2 AAL Services

2.5.2.1 Smart and Interactive Textiles

Textile products are currently being increasingly considered for uses other than just aesthetics and protection. They can, for instance, be used to hidrate and relax a person's skin or serve as support for devices to monitor or control vital signs, as well as communicate and transmit data.

Smart textiles can be defined as fabrics with measuring capabilities that react to certain events or have electronics-related functionalities. Interactive textiles, on the other hand, require a wearable device to be embedded in them and operated through a control panel or command buttons.

These products must provide at least one type of functionality (sensors, actuators, data processors, energy or communication) and are mostly used in healthcare where their applications in health monitoring and care, prosthetics, surgical implants and materials and hygiene are particularly important, especially for the growing segment of elderly population.

2.5.2.2 Living Usability Lab

The Living Usability Lab (LUL) [49] is a collaborative Research and Development (R&D) project between the Portuguese industry and academia with the goal of developing technologies and services to support senior citizens' health, activity and productivity with special attention given to their specific usability needs. It adopts the principles of universal design and natural user interfaces, such as speech and gesture, making use of the benefits of next generation networks and distributed computing.

In this context, the Microsoft Language Development Center (MLDC) has launched a few projects:

- Your Speech – A speech collection platform with the goal of advancing speech recognition technology for human-computer interaction and speech synthesis for people with special needs. It consisted of two applications, one of which consisted of a simple quiz game where the users answered with their own voice, while the second application consisted of a voice synthesizer which allowed users to generate their own synthesized voice.

- Personal Life Assistant (PLA) [50] – A distributed system prototype that enables access to services such as social networks, email, schedule and audio/video conferencing through natural user interfaces like speech and touch, as well as more traditional means of mouse and keyboard, through a PC or smartphone. **Error! Reference source not found.** [50] and **Error! Reference source not found.** [50] respectively show PLA's architecture and interfaces.

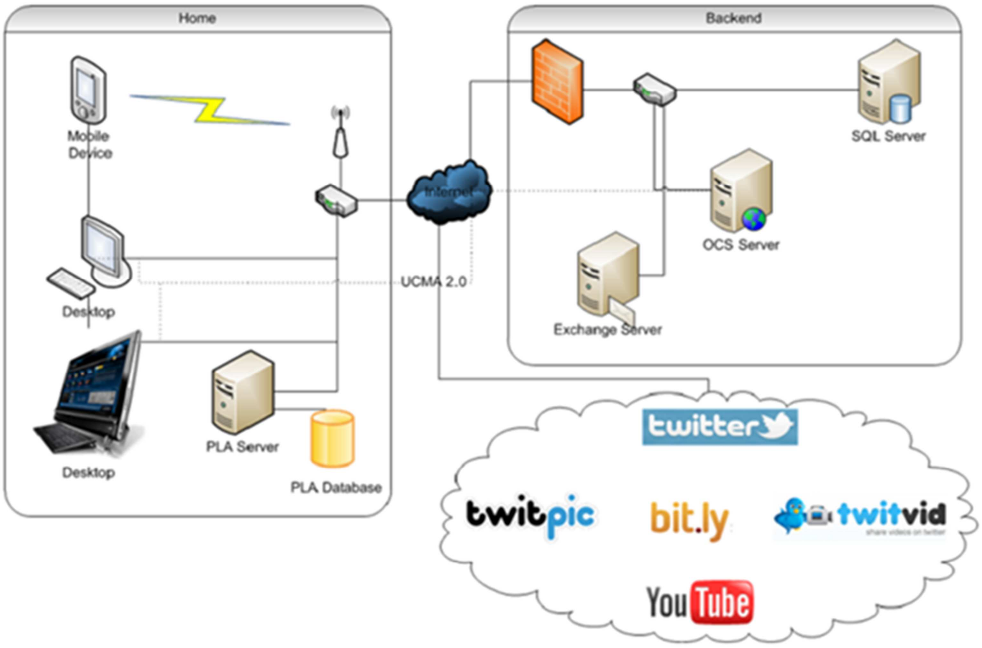


Figure 7 - PLA prototype architecture

The MLDC also launched a national campaign in 2010 to collect senior citizens' voices, dubbed "Doar a Voz", with the same aim as the Your Speech project but focused on the elderly segment of the population.

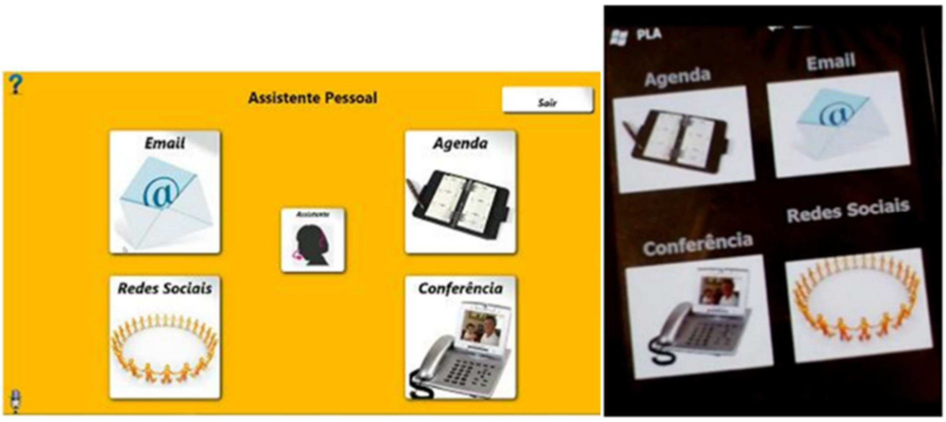


Figure 8 - PLA desktop and mobile interfaces

2.5.2.3 Activo PC Sénior

Activo PC Sénior [51] is an initiative launched by Microsoft, Caixa Geral de Depósitos, Rutis and Inforlândia to promote information literacy among the elderly by providing them with a loptop specifically designed with their usability requirements in mind. This laptop, shown in Figure 9 [51], possesses keyboard lighting, larger keys with more spacing between them, an ergonomically designed mouse and an embedded Third Generation (3G) broadband modem.

Figure 9 – Activo PC Sénior

2.5.2.4 Iniciativa Segurança Idade Maior

Through a collaboration between Optimus and the Braga Civil Government, the Iniciativa Segurança Idade Maior (ISIM) [52] project has the main goals of fighting loneliness and isolation among the elderly and provide them with an easy means of communication with family, friends and basic healthcare and security services.

To this end, Optimus provides pre-configured mobile phones with the numbers of family members, friends and healthcare and security institutions on speed dial and a sticker on the back with this information.

Optimus is the exclusive mobile operator on this project and provides these devices for people who meet some age and income requirements, leaving their delivery up to the Braga Civil Government.

2.5.2.5 Terceira Idade Online Project

The Terceira Idade Online Project (Projecto TIO) [53], created in 1999, is a Portal dedicated to the Portuguese elderly population developed by Byweb, who also developed the Net@vó project which was distinguished by the Portuguese Ministry of Education and participated in the VIVER project developed at the European level and regarded as a good practice by experts of the European Commission.

Professionals involved in these Byweb projects went on to form the Associação VIDA (Associação Valorização Intergeracional e Desenvolvimento Activo) in 2003.

2.5.2.6 Tele-assistance Services

There are a number of tele-assistance services currently operating in Portugal which are meant to provide support in emergency situations or social and medical counseling. These services are usually available 24 hours a day, every day, are mostly aimed at the elderly and people with disabilities and are provided by an assortment of private companies and other organizations.

Serviço Teleassistência PT

Portugal Telecom (PT) provides one such service [54] with their own specific equipment which includes a specially designed home phone and a pendant, both with an emergency button. It is meant to help provide emergency medical assistance and counseling and establishes immediate contact with their call center when the emergency button is pressed, where a licensed professional will take the call, diagnose the problem and activate the appropriate means of assistance, which can range from contacting family members or neighbours to provide assistance to sending a healthcare professional or the *Instituto Nacional de Emergência Médica* (INEM) with an ambulance to the user's home.

Portuguese Red Cross Tele-assistance Service

The Portuguese Red Cross (*Cruz Vermelha Portuguesa* – CVP) [55] provides a tele-assistance service both to a user's home, through a speaker terminal and pendant with an emergency button, or anywhere, through the use of a device similar to a mobile phone with GSM network communication andu ser localization capability through GPS.

Similarly to other such services, pressing the emergency button on any of the devices establishes contact with the call center where the operator identifies the user and activates the adequate means of support or rescue and maintains contact until the emergency is solved.

Montepio Residências Tele-assistance Service

The Montepio bank offers a tele-assistance service [56] to its clients who remain in their homes for health or other reasons and aims to provide security and comfort in a simple and efficient manner.

Through the push of a button on a necklace or bracelet, access is granted to three main services:

- *Emergência 24 Horas* – Provides immediate support in emergency situations, allowing the dispatch of ambulances, firefighters and police, medical counseling over the phone and indication of available hospitals, clinics and pharmacies.
- *Voz Amiga* – Provides regular contact with clients and their family members, a reminder service for medications and various appointments, miscellaneous information and loneliness support.
- *Assistência 24 Horas ao Lar* – Provides the dispatch of qualified professionals such as plumbers, electricians, painters and movers whenever required.

EnfermeirosPT Tele-assistance Service

Redenced-EnfermeirosPT [57] is a group that provides healthcare and home support in the field of nursing and their tele-assistance service installs equipment in patients' homes, monitors them and activates rescue measures whenever necessary, sends healthcare or other specific professionals, provides tracking systems for Alzheimer or other mental patients, automatic calling systems and a 24 hour support call center.

Helpphone Tele-assistance Service

The Helpphone [58] company operates a tele-assistance center based on a call center with a fast and secure communication system. With the push of a button on a device worn as a wristwatch or necklace, contact with the call center is immediately established through an intercom connected to a phone. This intercom has a unique identifier that, together with the information provided by the user at the time of registration, allows the call center operator to have immediate knowledge of information necessary for emergency situations even if the user is unable to speak.

Helpphone also provides users a card with contact information and the same service with a mobile phone.

2.5.2.7 Programa Aconchego and Projecto Lado a Lado

The *Aconchego* [59] and *Lado a Lado* [60] projects are both meant to promote mutual support between college students and the elderly, *Aconchego* in Oporto and *Lado a Lado* in Coimbra.

The initiatives consist in providing lodging for college students in the elderly's homes, while the students in this way would provide support to the elderly they are living with and help combat their loneliness and social isolation.

2.5.2.8 Primus Care

Created in 2006, Primus Care [61] provides healthcare and home support services in the Lisbon Metropolitan Area with a focus on the elderly population. For this purpose, it provides a range of services and solutions for home environments as well as day care centers, such as:

- Nursing
- Physical and speech therapy
- Psychology
- Occupational therapy
- Training
- Technical orientation and assistance
- Advisory
- Alert system
- Integrated operational management

The integration and flexibility of these services intends to increase the elderly's quality of life and their independence as well as their families'.

2.5.2.9 Fundação Vodafone Portugal

A not-for-profit, self-funded organization, the *Fundação Vodafone Portugal* [62] has as its main goals the promotion, support and execution of projects to develop the Information Society, fight info-exclusion and spread mobile telecommunication technologies, as well as the implementation of social and philanthropic initiatives that contribute to the integration of all citizens.

In the field of healthcare, the *Fundação Vodafone Portugal* and the Pulmonology Service of the Pulido Valente Hospital jointly developed the TELEMOLD [63] (*Telemonitorização de Oxigenoterapia de Longa Duração*) system. It aims to improve the quality of life of patients with breathing deficiencies by remotely monitoring blood oxygen levels and their physical activity in order to optimize oxygen prescription.

The TELEMOLD system tries to bring benefits on three major fronts:

- Patients and their families
  - Continuous monitoring
  - Prescription adjustments at any time
  - Avoid frequent hospitalizations
  - Increase in quality of life
- Doctors
  - More rigorous diagnostic data acquisition
  - Follow up of a greater number of patients
  - Anomalous situation recognition through an alert system
- Society
  - Relief of pressure in hospital emergency rooms
  - Adequate financial means for treatments

2.5.2.10 Futuro Feliz em Família

The *Futuro Feliz em Família* [64] provides services for domestic support with the goal of improving quality of life mainly to the elderly and their families, as well as supporting people with varying degrees of disability or in need of healthcare in their homes. They act according to a previously elaborated Individual Development Plan that aims to keep their clients within their day to day environment, close to family members, neighbours and friends, whatever their degree of dependency.

Futuro Feliz em Família provides its services from 2 to 24 hours a day, every day of the year, which include:

- Social services
  - Company and entertainment
  - Habitational and personal hygiene
  - Preparation and accompaniment of meals
  - Control and administration of medication
  - Support in domestic chores
  - Accompaniment outside (medical appointments, walks)
  - Biopsychosocial support
  - Support for mentally handicapped people (APPACDM/CERCI protocol)
- Medical services
  - Specific care for recovering stroke, Alzheimer's, Parkinson and Diabetes patients
  - Post-discharge and continued care
  - Nursing, clinical analysis and cardiological exams
  - Physical, occupational and respiratory therapy
  - Podology
  - Internal medicine, geriatric and home emergency consultations
  - Acupuncture, Shiatsu and therapeutic massages
- Engineering and construction services
  - Accessibility and suppression of architectural barriers
  - Plumbing and unclogging
  - Gas, electricity, phone and data
  - Windows, fixtures and blinds
  - Doors and locks

  - Floating floors
  - Painting, infiltrations and waterproofing
  - Carpentry and furniture
  - Kitchen and bathrooms
  - Heating, ventilation and air conditioning
  - Household appliances
  - Gardening
  - Periodic gas and electricity inspections
- Technical help and hospital material (commercialization)
  - Hygiene
  - Wheelchairs
  - Walking aids
  - Hospital material
  - Anti-pressure ulcer material
  - Elevation and transfer
  - Accessibility
  - Rehabilitation
  - Immobilization and compression
  - Podology

Futuro Feliz em Família currently provides these services in the Lisbon Metropolitan Area.

2.5.2.11 Comfort Keepers

Comfort Keepers [65] is a multinational company dedicated to providing domestic support, especially to the elderly and incapacitated or recovering adults. In Portugal it currently operates in the Lisbon and Oporto Metropolitan Areas, Zona Centro, Trás-os-Montes, Algarve and Azores.

The company provides services 24 hours a day, every day of the year and designs personalized care plans according to a family's and each individual's needs. These services are divided in five major areas:

- Personal care
  - Bathing
  - Mobility
  - Transport and body position
  - Incontinence
  - Intimate Hygiene
  - Special diet and meal preparation
  - Shopping
- Family care
  - Company and conversation
  - Personalized preparation of meals
  - Light domestic chores
  - Recreational activities
  - Shopping
  - Company in occasional appointments
  - Phone calls and follow up contacts
  - Oral hygiene
  - Clothing and personal image aid
- Tele-assistance service
  - Assistance and immediate support in emergency situations
  - Orientation and medical emergency
  - Indication of available hospitals, clinics and pharmacies

  - Alert services (medications, medical appointments, waking up)
  - Home assistance
- Hospital and medical assistance
  - Pharmaceutical management
  - Psychological support
  - Physical therapy
- Special care
  - Accompaniment programs for chronic, degenerative and oncologic diseases such as Alzheimer's and Parkinson's
  - Permanent care at home 24 hours a day, 7 days a week

2.6 Conclusions

Considering the wide array of information and different applications, such as the previously presented ones, with which a truly interoperable AAL system would have to communicate, an EHR flexible enough to store various kinds of data in a standardized way and share it with different systems is required. For this purpose, the openEHR standard seems to be the ideal choice at this point. It is in continuous development and improvement as part of an international community effort, is capable of storing different kinds of data and provides a reference implementation written in Java as a starting point for software applications looking to implement this standard. The separation of the technical and domain models is also a welcome design feature as it should provide flexibility in regards to evolving information structures and concepts, requiring only small or no changes to existing applications to deal with these evolutions on the data they store and transmit. One caveat that should be made, however, is that openEHR's reference model is complex due to the flexibility it strives to achieve and will require some time and effort to fully understand and implement successfully.

In order to tackle the interoperability issue, communication of the stored information is of vital importance and while it is unfeasible to expect every single available application or system to use the same standard and implementation for this purpose, at this time the HL7 v2.x standard seems to be the most widely used and therefore the most likely to provide greater coverage of the existing scenarios. As a result of this widespread use there are some available tools to support the understanding and implementation of this standard, such as Mirth Connect, which will be discussed later in this document.

Chapter 3

Interoperability and proposed system

This chapter presents a brief description of the problem at hand as well as its proposed solution's architecture and information flow between the desired components.

3.1 Description of the problem

Interoperability is defined in the IEEE Glossary as "the ability of two or more systems or components to exchange information and to use the information that has been exchanged" [66] and can be divided into syntactic and semantic interoperability.

Syntactic interoperability is required for further interoperability and is achieved when systems can simply exchange data between them. A well defined data format, such as XML, is required for this purpose. The openEHR foundation has provided schemas that define and validate the structure of data to be transmitted according to the openEHR standard in XML format. Both XML and HL7 v2.x are well known and widely used formats in healthcare applications which makes them good choices for data transmission.

Semantic interoperability is achieved when applications not only exchange data but also understand its meaning, enabling automatic interpretation. This could be achieved through the use of openEHR archetypes that provide a means to correctly interpret the data within the structure, provided that the communicating applications use the same set of archetypes which should be agreed upon and defined beforehand.

A possible solution for the interoperability problem using the previously chosen standards, which will be explored in this dissertation, can be divided into three parts. Firstly, the intended system must provide a way to store health and other related data in a standardized way in order to allow different applications to then correctly access a user's relevant information from one source. OpenEHR should be the standard used in storage as any application that can interpret the used archetypes will be able to understand the meaning of the data and so, a repository of records following the openEHR standard should be a component of the system. It must also provide basic management capabilities in order to allow the insertion, removal and alteration of said user's information in storage through a client application of the repository. Finally, the solution must provide a way to receive and transmit data, possibly according to different standards depending on the application that is sending or requesting that data. This could be implemented as a message broker capable of transforming messages between different standards.

3.2 System architecture

Taking into account the three parts of the proposed solution mentioned earlier, it is necessary to understand how information would flow between them and how they can be implemented into a system. The following is a brief description of the intended system's architecture and information flow between its components.

3.2.1 Repository and client application

As Figure 10 shows, the client application (EHRClient) receives an openEHR composition in XML format to process. The client application first validates the composition received against the openEHR XML schemas to ensure it has the correct structure; it then adds the composition to an existing record in the repository belonging to the composition's subject or creates a new EHR for the subject if it does not yet exist and adds it to the repository. The client application can, in this light, be seen as middleware between the repository and the source of the information, as its main functions are to validate the structure of the information to be shared and to insert it correctly into the repository.

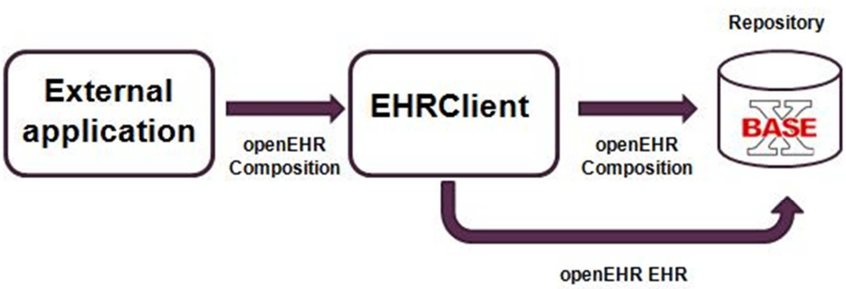


Figure 10 – Client application and repository information flow

3.2.2 Message broker

Figure 11 shows the expected flow of messages through the message broker to be used. It is expected to receive messages following the HL7 v2.x standard and transform them into XML format messages following the openEHR standard. The broker is expected to receive messages in the two possible formats of the HL7 v2.x standard, which are the original HL7 v2.x format of messages made up of segments separated by the ASCII Carriage Return character and composed by data fields separated by another ASCII displayable character, usually the vertical bar (|), and the encoding specification of HL7 v2.x messages in XML format, dubbed v2.xml. These formats will be presented in more detail later in this document. It then transforms them into the openEHR XML format to send to the client application. The broker should also be able to do the inverse transformation when receiving an openEHR composition from the client application to send to another application.

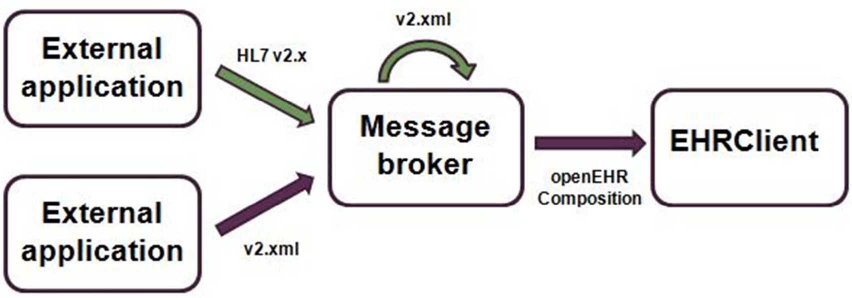


Figure 11 – Message broker information flow

3.2.3 System overview

The complete system, shown in Figure 12, should then be able to receive HL7 v2.x messages in either of the available encoding specifications, transform them into openEHR compositions in XML format through the message broker and send them to the repository or, alternatively, receive a message from an external application already as an openEHR composition and insert it into the repository. The client application always validates the received compositions against the openEHR XML schema before storing them in the repository in order to ensure they are in the correct format.

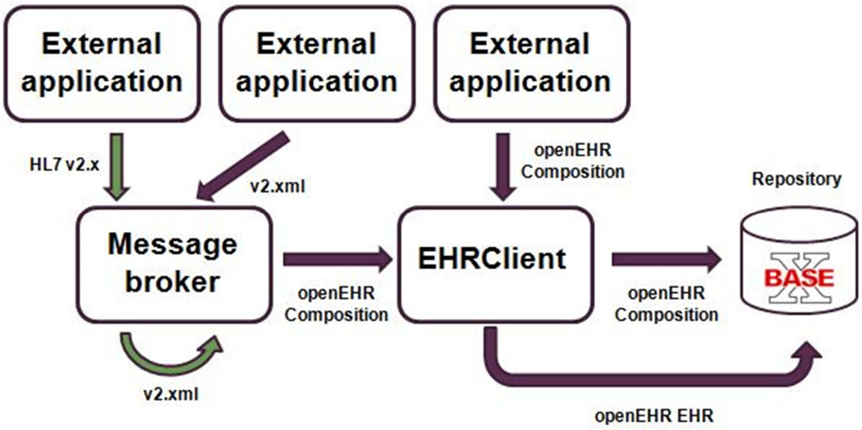


Figure 12 – Complete system overview

Chapter 4

Prototype development

This chapter details the development phase of this dissertation, from the analysis and choice of technologies used to the detailed description of the components implemented for the target prototype.

4.1 Technologies and tools

The following is a description of the more relevant parts of the chosen technologies for this project. It details the general openEHR architecture as defined by the openEHR foundation and how it is used in the developed system, it also presents the chosen storage technology as well as the message broker and a brief explanation of the choices made against other available options.

4.1.1 OpenEHR

Before attempting any implementation and choosing technologies for that purpose it is necessary to study and understand the openEHR architecture and identify the most important components needed for the proposed system.

Figure 13 [67] shows the components of a minimal EHR system based on the openEHR standard, which are an EHR repository, an archetype repository, terminology (if available) and demographic/identity information, which may be in the form of an openEHR demographic repository or an already existing Patient Master Index (PMI) or other directory.

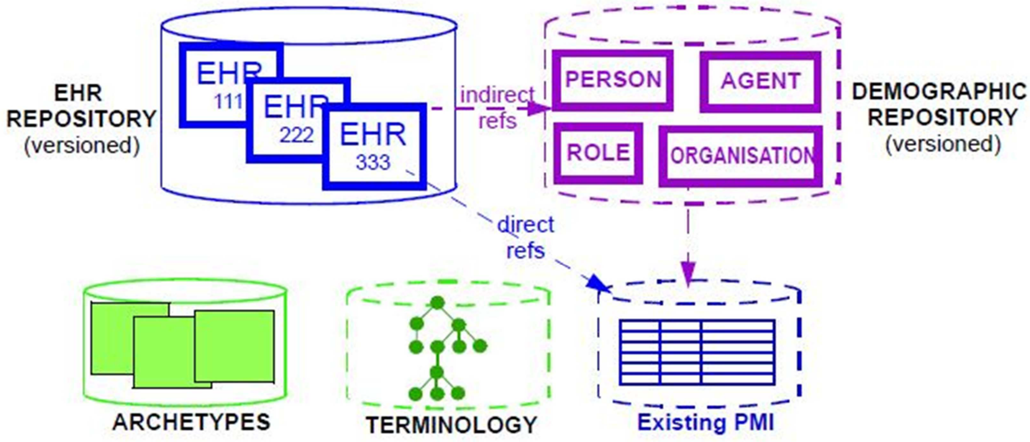


Figure 13 – Minimal openEHR EHR system

It is important to note that the openEHR standard was designed to provide a basis for a complete and flexible EHR system and as such, given the limited time and resources available for this dissertation, only a subset of the available specifications will be used since the goal is not to implement a fully functional EHR system but to assess the possibility that EHRs constructed according to this standard can be part of a solution to the interoperability problem. For this purpose, the component which will be focused on is the EHR repository.

The openEHR architecture is composed of three major packages, each of them composed in turn by more specialized packages defining detailed models. The three major packages are RM (Reference Model), AM (Archetype Model) and SM (Service Model) and the relationships between them are shown in Figure 14 [67], in which dependencies are shown to exist only from higher to lower packages. Each of these packages defines a local context for definition of classes.

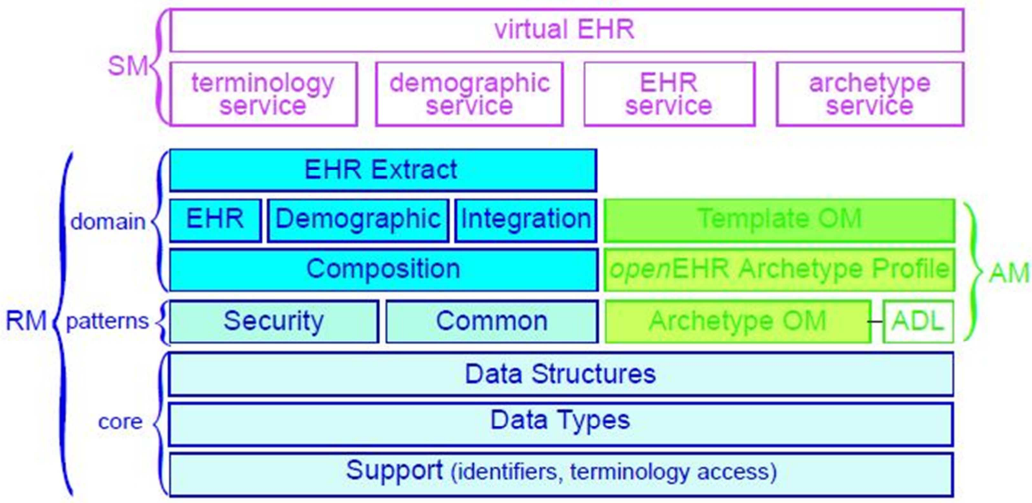


Figure 14 – openEHR package structure

The packages shown in the *core* group are generic, used in all the openEHR models, in all the outer packages and provide identification, access to knowledge resources, data types and structures, versioning semantics and support for archetyping. The packages in the *domain* group define the semantics of enterprise level health information types, including the EHR and demographics.

The following subsections provide a brief overview of the most important RM packages for this project, with a more detailed description of the EHR Information Model since this is where the main concepts of an EHR's structure are defined.

4.1.1.1 Support Information Model

This package is comprised of the *Definitions*, *Identification*, *Terminology* and *Measurement* packages and describes the most basic concepts required by all other packages. It is in these packages where the semantics that allow all other models to use identifiers and have access to knowledge services, such as terminology, are defined. A special package, named *assumed_types* is also included in the support package. It is a guide for integrating openEHR models into the type systems of implementation technologies by describing what basic types are assumed by openEHR in external type systems.

4.1.1.2 Data Types Information Model

This package comprises a set of clearly defined data types that provide a number of general and clinically specific types required for all kinds of health information. The categories of data types defined are:

- *Text* – plain text, coded text, paragraphs
- *Quantities* – any ordered type including ordinal values, measured quantities with values and units, and so on
- *Date/times* – date, time, date-time types and partial date/time types
- *Encapsulated data* – multimedia, parsable content
- *Basic types* – boolean, state variable

4.1.1.3 Data Structures Information Model

The following is a list of the generic structures defined in this package that are used by archetypes to define the particular structure needed to express content.

- *Single* – single items, used to contain any single value
- *List* – linear lists of named items
- *Table* – tabular data, including unlimited and limited length tables with named and ordered columns, and potentially named rows
- *Tree* – tree-shaped data, which may be conceptually a list of lists or other deep structure
- *History* – time-series structures, where each time-point can be an entire data structure of any complexity, described by one of the above structure types. Point and interval samples are supported.

4.1.1.4 Common Information Model

This package defines recurring concepts for higher level packages, such as the LOCATABLE and ARCHETYPED classes, which provide the link between information and archetype models, as well as the ATTESTATION and PARTICIPATION classes, which are generic domain concepts that appear in various reference models. It also provides the *change_control* package that defines a formal model of change management and versioning which applies to any service that needs to be able to supply previous states of its information, in particular the demographic and EHR services.

4.1.1.5 EHR Information Model

The EHR Information Model (IM) defines the containment and context semantics of the major coarse-grained components of the EHR. The packages within the EHR IM are shown in Figure 15 [68].

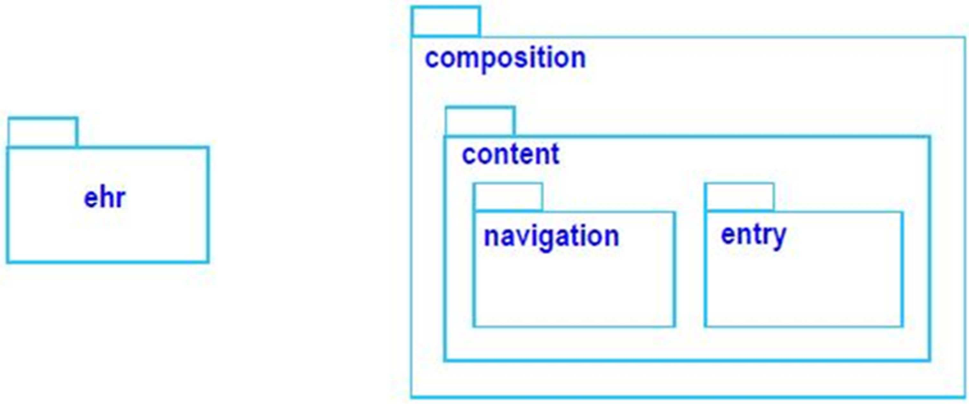


Figure 15 – openEHR EHR package structure

The **ehr** package contains the EHR's top level structure itself and is composed by an EHR_ACCESS object, an EHR_STATUS object and versioned data containers called VERSIONED_COMPOSITIONs, which may be indexed by a hierarchical directory of FOLDERs, as well as a collection of CONTRIBUTIONs documenting the changes made to the EHR over time.

The **composition** package, described by the COMPOSITION class, is the EHR's top level data container and represents a record's unit of committal.

The **content** package contains the **navigation** and **entry** packages which describe the structure and semantics of Composition contents through their classes.

The **navigation** package is composed by the SECTION class, under which other SECTIONs or ENTRYs may appear, providing a navigational structure to the record.

The **entry** package contains the classes that form the generic structures meant to record health and care data, which are ADMIN_ENTRY, OBSERVATION (records all observed phenomena, measured in any way, and responses in interview), EVALUATION (records assessments, diagnoses and plans), INSTRUCTION (records actionable statements such as medication orders, monitoring, reviews) and ACTION (records instances of performing Instructions).

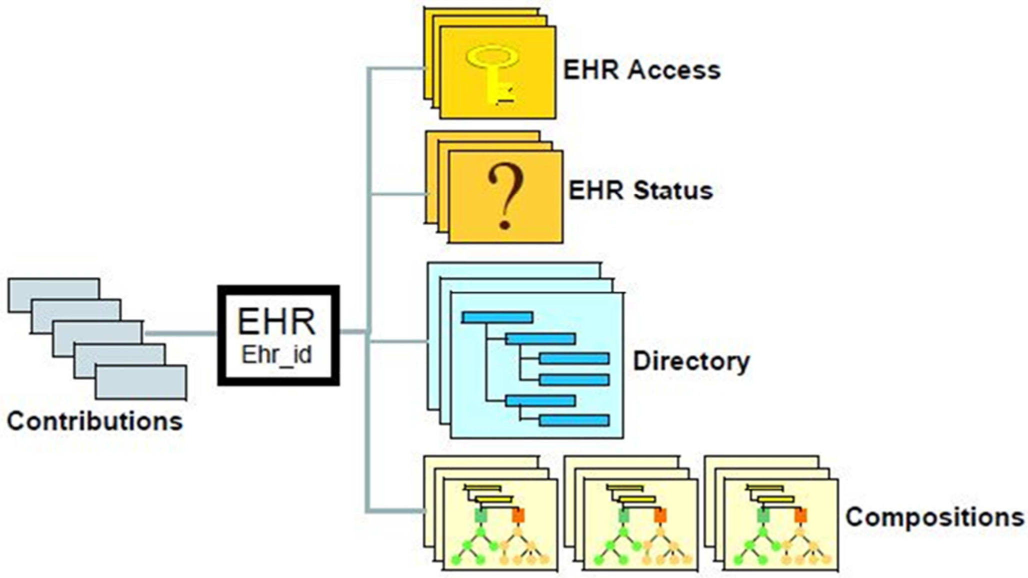


Figure 16 – High level structure of the openEHR EHR

As shown in Figure 16 [68], the openEHR standard defines six components in the high-level structure of an EHR:

- *EHR* – central root object itself uniquely identified by an identifier field, it also records the identifier of the system in which it was created and its time of creation. Acts as an access point for the component parts of the EHR
- *EHR_access* (versioned) – object containing access control settings for the record, such as default privacy policy, lists of identified accessors and exceptions to the default policies. All changes to this object are versioned so that the visible view of the EHR at any previous point in time is reconstructable.
- *EHR_status* (versioned) – object containing status and control information, may also contain the record's subject identifier and other EHR-wide metadata, such as runtime environment settings, software application names and version ids, identification and versions of data resources and tools, in an archetyped 'other_details' part of the object.
- *Directory* (versioned) – optional hierarchical structure of Folders used for logical organization of Compositions
- *Compositions* (versioned) – containers of all clinical and administrative content of the record, can be divided into two types: Persistent and Event compositions
- *Contributions* – change-set records for every change made to the record, each references a set of one or more versions of any versioned items in the record that were commited or attested together by a user to an EHR

Compositions

The most important component to focus on within the EHR for the purposes of this dissertation is the compositions, since this is where all the relevant data will be stored and transmitted, serving as the committal unit of the EHR.

The Composition concept in openEHR originated from the Transaction concept of the Good European Health Record (GEHR) project [69], which in turn is based on the concept of a unit of information corresponding to the interaction of a healthcare agent with the EHR. However, it has been expanded and more formally defined in openEHR. Firstly, the idea of a unit of committal has been formalized by the openEHR model of change control and secondly, a Composition may now capture categories of clinical data with long-lived significance, such as problem and medication lists, in addition to data from clinical events such as a patient contact. So, there are two types of Compositions to reflect the two general categories of information found in an EHR: Event and Persistent Compositions.

Event Compositions record information gathered during healthcare system events with or for the patient, even when the latter is not present or a participant (surgery or pathology testing, for instance). In addition to this data, this type of Composition must also record the event's context information, which includes the responsible party, place, time and cause; as such, a specific class is associated with event Compositions in the formal model representing clinical context.

Persistent Compositions are used to record information with more longevity, such as problem lists, current medications, family history and care plan, while Event Compositions record things that were true or happened but at a specific point in time. "Persistent Compositions can be thought of as *proxies* for the state or situation of the patient – together they provide a picture of the patient at a point in time" [68].

Compositions in the openEHR EHR are versioned through the VERSIONED_OBJECT<T> type from the *change_control* package of the common information model which is explicitly bound to the COMPOSITION class, via the VERSIONED_COMPOSITION class in the *composition* package which in turn inherits from the type VERSIONED<COMPOSITION>. Any Composition commited to an EHR is a single version to be contained in a VERSIONED_COMPOSITION, which means it is either added to an already existing VERSIONED_COMPOSITION as its latest version or a new VERSIONED_COMPOSITION is created altogether.

The openEHR foundation provides XML schemas to define and validate the structure of compositions transmitted, thus indicating an ideal format to exchange information to and from an EHR following this standard.

4.1.2 Storage technology

One of the components previously identified for implementation is a repository for the records. The first step towards deciding the technology to use for storage is to study the available options. Since a previous thesis from the University of Aveiro had already been published in 2011 [70] focusing on the very subject of an openEHR repository, it was used as a basis for this choice.

The aforementioned thesis presents a brief comparison between relational, object oriented, XML and plain text databases and concludes that there is no best database. In order to select the most appropriate database model, the nature of the data to store and the functionalities to support must be analysed. However, it does point out that "XML databases are indicated when the data model is very flexible and must be easily modified" [70] which is in line with the requirements for openEHR record storage.

Ultimately, the type of storage chosen was a native XML database due to it being a relatively new technology at the time, therefore making it an interesting case of study for performance comparisons with other approaches, but also because XML is optimal for document centric applications, which makes it a good fit for an EHR repository that is patient centric and may have an XML file per patient containing all of his data. It is also a known standard, facilitating communication with other systems. More specifically, BaseX was chosen as the native XML database system to implement the repository.

BaseX is defined as a light-weight, high-performance and scalable XML database engine and XPath/XQuery 3.0 processor which includes full support for the World Wide Web Consortium (W3C) Update and Full Text extensions [71]. It is also a completely open source and platform independent software written in Java. The update functionality is a particularly welcome feature since it allows access to a specific position of a file to alter its value instead of the traditional way which would require reading and rewriting the entire file. It also offers a client-server architecture and a Representational State Transfer (REST) API for systems with distributed database access. REST facilitates a fast and simple access to databases through HTTP, using the GET, PUT, DELETE and POST HTTP methods to interact with the database.

BaseX also provides a number of indexes used to rewrite expressions and speed up query evaluation that are thus important performance-wise. They are divided into two categories [71]:

- **Structural Indexes** – These will always be present and cannot be dropped by the user
    - **Name Index** – Contains all element and attribute names of a database and the fixed-size index ids are stored in the main database table. New names are automatically added upon database updates. The index may also contain statistical information, such as the distinct (categorical) or minimum and maximum values of its elements and attributes. This information, however, is discarded after database updates and requires a call of the optimization command to be recreated. This index is applied to pre-evaluate location steps that will never yield results, for example.
    - **Path Index** (or **Path Summary**) – Stores all distinct paths of the documents in the database and contains the same statistical information as the name index which is discarded and recreated in the same manner. It is applied to rewrite descendant steps to multiple child steps which can be evaluated faster, as fewer nodes have to be accessed.

- **Resource Index** – Contains references to the *pre* values of all XML document nodes and speeds up the access to specific documents in a database. It is automatically updated when updates to the database are performed.

- **Value Indexes** – These can be created and dropped by the user. The text and attribute indexes will be created by default
  - **Text Index** – Speeds up string-based equality tests on text nodes and also supports range queries based on string comparisons. This index can be kept up-to-date by activating the UPDINDEX option. The current index structures do not support queries for numbers and dates.
  - **Attribute Index** – Similar in both benefits and update methods to the text index but applied to attribute values.
  - **Full-Text Index** – Speeds up queries using the *contains text* expression. Provides two index structures internally: the default sorts all keys alphabetically by their character length and is particularly fast if fuzzy searches are performed; the second is a compressed tree structure, which needs slightly more memory, but is specialized in wildcard searches. Both of these structures will be merged in a future BaseX version.

BaseX then seems a logical choice, not only because it aligns itself with the requirements for the repository component of the proposed solution, but also because of the possibility to expand upon the work of the thesis from the University of Aveiro and compare performance when this storage technology works in conjunction with the rest of the solution system.

4.1.3 Message broker

Another component required for the proposed solution is a message broker capable of transforming and routing messages of different standards, especially openEHR and HL7 v2.x. For this purpose the HL7 application programming interface (HAPI) and Mirth Connect were considered for implementation.

HAPI [72] is an open source, object-oriented HL7 v2.x parser for Java which can be used to add HL7 capabilities to applications. It was briefly considered as a possible way to expand the client application to be implemented with functionalities to parse HL7 v2.x messages and, in conjunction with openEHR’s Java Reference Implementation, be used to transform those messages into openEHR compositions to be stored in the repository as well as retrieve requested compositions from the repository and transform them into HL7 v2.x messages to be sent to other applications. This approach would require the solution to concentrate the message broker and storage client into a single application which would possibly be more robust, yet less flexible. It would also only be able to process HL7 v2.x messages and openEHR compositions, limiting possible future expansions.

Mirth Connect [73] is an open source healthcare integration engine based on standards designed to facilitate the routing, filtering and transformation of messages between health information systems over a variety of protocols. It is sponsored and primarily developed by the Mirth Corporation and currently supports the following standards and protocols:

- Message standards
  - HL7 v2.x

  - HL7 v3
  - XML
  - X12
  - Electronic Data Interchange (EDI)
  - Digital Imaging and Communications in Medicine (DICOM)
  - National Council for Prescription Drug Programs (NCPDP)
  - Delimited Text
- Transfer Protocols
  - Minimal Lower Layer Protocol (MLLP)
  - TCP/IP
  - HTTP
  - Files
  - Database
  - S/FTP
  - Email
  - Java Message Service (JMS)
  - Web Services
  - PDF/RTF Documents
  - Custom Java and JavaScript

This means that Mirth Connect not only has the necessary standards for this project, but can also be expanded upon in the future to enable the system to deal with further standards and to use various protocols to communicate and transfer the necessary information, so it provides flexibility for communication with various other applications.

The interface engine is composed of four key components [74]:

- **Mirth Connect Server** – Contains the back-end for the management interface and the integration engine component, which performs message filtering, transformation and transmission.
- **Mirth Connect Server Administrator** – Graphical user interface that connects to the Mirth Connect Server and allows the configuration of interfaces, the monitoring of interface activity and browsing of the message store.
- **Mirth Connect Server Manager** – Graphical user interface that manages the Mirth Connect service, displays log files and contains other configuration settings for the Mirth Connect Server.
- **Mirth Connect Command Line Interface (CLI)** – Command line tool that allows connections to the Mirth Connect Server to deploy/import/export channels and perform other administrative tasks.

The graphical user interfaces are a particularly welcome feature as they provide a user-friendly environment in which to implement the features required such as the message transforms.

4.2 BaseX client application

The first component of the system to be implemented was an application that serves as a client for the BaseX repository which is capable of receiving information as openEHR compositions in XML format, validate them against the openEHR XML schemas and send them to the repository as either new compositions, new versions of existing compositions or as the first composition in a newly created record.

In order to achieve these implementation goals it was necessary to use both the openEHR Java Reference Implementation and the BaseX API.

4.2.1 OpenEHR Java Reference Implementation

The Java Reference Implementation was first created as an implementation of all the stable specifications of the openEHR Reference Information Model by a team from Sweden and then donated to the openEHR foundation. It was adopted in March 2005 and released under open source licenses [75]. It provides building blocks for EHR systems and encourages EHR implementation projects around the world.

This implementation is available online [76], uses the Java 5.0 platform and several well-established open source libraries, such as log4j and commons-lang from the Apache Software Foundation [77] and its goal is to keep the Java look and feel and to remain as faithful as possible to the openEHR specification at the same time. This means that to ensure interoperability with similar systems implemented in other programming languages, the complete semantics of the design must be correctly implemented in the Java language, which involves mapping between native Java types and openEHR assumed data types. The implementation is released under three different open source licenses, allowing users to choose the one that best suits their purposes between the Mozilla Public License (MPL), the Gnu Public License (GPL) and the Less Gnu Public License (LGPL). It implements all the packages from the openEHR Reference Model, illustrated in Figure 17 [67].

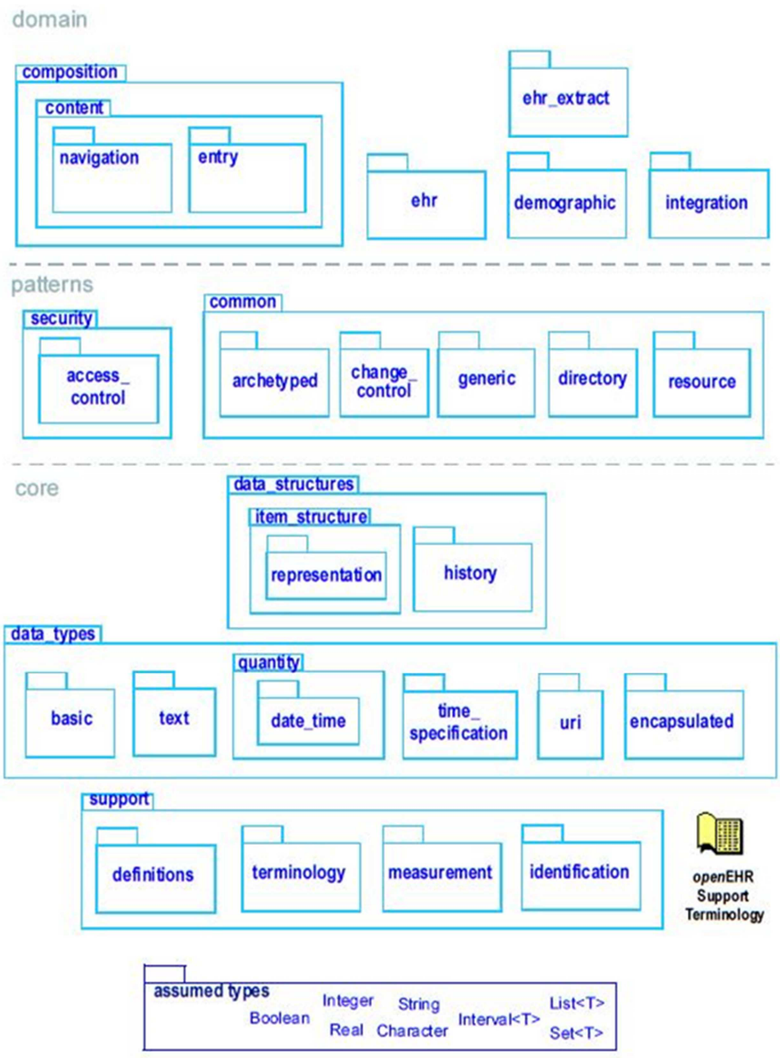


Figure 17 – openEHR Reference Model package diagram

This reference implementation was used to develop an application that can read compositions in XML format, validate them against the openEHR XML schema and manipulate the data received in runtime to create new EHRs, versioned compositions and contributions to add to a repository. For this purpose most of the packages within the RM package were utilized, particularly the *composition* and *ehr* packages for the main coarse-grained components of the EHR and the *support*, *datatypes* and *datastructure* packages for the more fine-grained components necessary. The *build* package provided by the Java Reference Implementation was also vital as it provided the RMObjectBuilder class that was responsible for the construction of most openEHR runtime objects through its *construct* method.

This *construct* method requires the name of the intended RM class to build and the necessary values for the object's elements. Therefore, it was necessary to implement a parser that could retrieve the required values from an XML file as well as functions to support the building of an EHR object or a versioned composition and write it to XML format.

These functionalities were distributed among a few classes created in a package called *main* and are described in a little more detail in the following subsections.

4.2.1.1 XMLParser

For the purpose of validating the XML compositions received, the javax.xml.validation API [78] was used through the Schema, SchemaFactory and Validator classes. A Schema instance is created through a SchemaFactory instance and the location of the XML Schema Definition (XSD) file to use. Then this Schema instance is used to create a Validator instance which will compare the XML file's structure with the created Schema.

In order to retrieve the data values required from the XML file, the JDOM API [79] is used to navigate the file's elements in runtime and place the values in a Map structure for the RMObjectBuilder to use to construct a Composition object and all of its component objects.

These functionalities are implemented in the XMLParser class.

4.2.1.2 DBBuilder

It is also necessary to construct VersionedComposition objects to be inserted in the repository, as well as Contribution objects that reflect the change in the record that the insertion of the composition represents and the EHR object itself when the goal is to create a new record in the repository.

These functionalities are implemented in the DBBuilder class which uses the XMLParser class when it is necessary to create a new EHR object with a received composition file, also uses the RMObjectBuilder to construct the versioned objects required in a record and the XMLWriter class to write the objects created into XML files ready to be added to the repository.

This is mainly a support class to prepare all the objects required when inserting new records or compositions into the repository in the correct format.

4.2.1.3 XMLWriter

This class implements the writing process from runtime EHR, Composition and Contribution objects to XML format. For this purpose, the classes from the javax.xml.bind package [80] were used as well as calls to DBBuilder functions when it is required to construct VersionedCompositions.

The writing process is achieved by creating a JAXBContext instance, using the classes that may become root elements in XML format, which in turn will be used to create a Marshaller instance. This Marshaller instance can then write the required objects in XML format with one simple step. However, in order for this process to work, the right XML annotations had to be inserted at the classes to be marshalled to indicate which were root elements, how some attributes were to be written and which attributes should be ignored in the writing process.

4.2.2 BaseX API

BaseX provides means to create standard client applications using sockets and sessions or calls to RESTful web services to communicate with the server where the repository is located. The RESTful web services design was chosen for this project as in this design the server may be created and run in one simple step and required functionalities may be implemented through simple HTTP GET, PUT, DELETE and POST calls sent to the server.

The BaseX REST API was introduced in version 7.0 [81], replacing the previous JAX-RX API mentioned in the thesis from the University of Aveiro [70]. Also, in this project the openEHR repository stores each record as a separate resource within the database instead of storing all records under one XML node. This decision was made to reflect the patient centric nature of EHRs, as each record should contain all the information related to a single subject and to facilitate the insertion of new records into the repository which in this way require a simple PUT request to the database instead of a more elaborate POST request containing an XQuery update. These changes may lead to some differences in performance between the two repositories which will be looked upon towards the end of this dissertation.

The BaseX HTTP Server was used with the default settings, which starts an instance of the Jetty Web Server [82], listening to port 8984, and the BaseX Server, listening to port 1984. This server provides the REST services by default at http://localhost:8984/rest.

In order for the client application to use the provided REST services, a few classes had to be created to prepare and send the service calls as well as receive the response from the server. These all receive the necessary parameters for the calls in their respective constructors and have the *execute* method to trigger their execution.

The created classes were based on the examples provided by BaseX [83] and are briefly described in the following subsections, as well as additional classes created and used in this project for testing purposes.

4.2.2.1 BaseXGet

This class implements an HTTP GET request which is used to retrieve information directly from the repository's structure or can also be used to execute some BaseX commands. It is constructed by receiving as arguments the URL where the REST services are available, such as the aforementioned default, a parameter to indicate it is a query or a command call and the path of the information to retrieve or the name of the command to execute.

4.2.2.2 BaseXPut

This class implements an HTTP PUT request which is used to insert files into the repository. It receives the URL for the REST services and the full path to the file to insert as its constructor arguments.

4.2.2.3 BaseXDelete

This class implements an HTTP DELETE request which is only used to remove an entire record from the repository. It receives the URL for the REST services and the path to the record to remove.

4.2.2.4 BaseXPostAdd

HTTP POST requests in BaseX are expected to include an XML structure in the body following the provided POST Schema [84]. For this reason the implementation of POST requests was split into two classes and this one is used for the most common usage of POST requests to the repository, which is to add elements to a record, such as new compositions or contributions. This class receives the URL for the REST services, the content to add to a record in XML format and the path in which to insert this content.

4.2.2.5 BaseXPostQuery

This class implements a more general use of the HTTP POST request in BaseX. It is used to execute full text queries to the repository, to update single field values and to remove compositions from a record. It receives the URL for the REST services and the content to send within the XML structure in the body of the request, which must be prepared beforehand according to the desired functionality to use.

4.2.2.6 EHRClient

This is the core class of the application which provides a simple textual user interface for testing purposes. It requires the server's address, the port where REST services are provided and a name to use for the database. All the previously created classes to communicate with the BaseX HTTP server which houses the openEHR repository are used here and it provides the following functionalities:

- **Create a new database** – Creates a new repository with no records or importing records in XML format available in a specified folder. Uses the BaseXPut class.
- **Create a new record** – Creates a new record using an available composition in an XML file. Uses the DBBuilder and BaseXPut classes.
- **Retrieve general information about the database** – Retrieves information such as the number of documents in the repository, its size and the status of the indexes. Uses the BaseXGet class.
- **Execute simple queries** – Retrieves data directly from a path within the repository specified by the user. Uses the BaseXGet class.
- **Insert a new record** – Adds a new record available in an XML file to the repository. Uses the BaseXPut class.
- **Delete a record** – Removes an entire record from the repository. Uses the BaseXDelete class.
- **Add a composition to a record** – Inserts a composition available in an XML file to a record in the repository. Uses the XMLParser, XMLWriter and BaseXPostAdd classes.

- **Retrieve all or one composition of a record** – Queries the repository for one or all of a record's compositions and returns the result. Uses the BaseXGet class.
- **Remove a composition from a record** – Removes the composition specified by the user from a record and adds a contribution to the record that reflects this change. Uses the BaseXPostAdd and BaseXPostQuery classes. The content argument of the BaseXPostQuery constructor is prepared with the appropriate elements to delete a node in the record of the repository.
- **Retrieve all or one contribution of a record** – Identical to the previous functionality to retrieve compositions, but used for contributions.
- **Update a single field's value** – Changes a field's value directly in the database and is mainly intended to allow for small corrections. Uses the BaseXPostQuery class.
- **Search for text value** – Applies a full text search to search for a specified value in all fields of the repository and returns the ID of the records in which it was found and the parents of the elements where it was found. Uses the BaseXPostQuery class whose content argument is prepared with the appropriate input for a full text search.

4.2.2.7 Main

This is an auxiliary class used to start the execution of both the BaseX HTTP server and an instance of the EHRClient.

4.3 Mirth Connect

Mirth Connect was used in this dissertation to implement the message broker component of the proposed solution and, for this purpose, its graphical user interfaces were used to facilitate the implementation. It was therefore necessary to install Mirth Connect as a standard program beforehand.

An installer for Mirth Connect is provided as an executable file and pre-packaged distributions for individual operating systems such as Windows, Unix/Linux and Mac OS X are also available. The Windows installer, however, has the option to install and start a service which runs in the background and also to install and run the Mirth Connect Server Manager which serves to start and stop the service, view the system logs, change the backend database settings and launch the Mirth Connect Administrator. A Mirth Connect Shell may also be installed to connect to a running Mirth Connect Server through a command line interface.

Prototype development

The Mirth Connect Administrator was used to both develop and manage the channels and transforms created to handle the messages meant to be sent and received by the system.

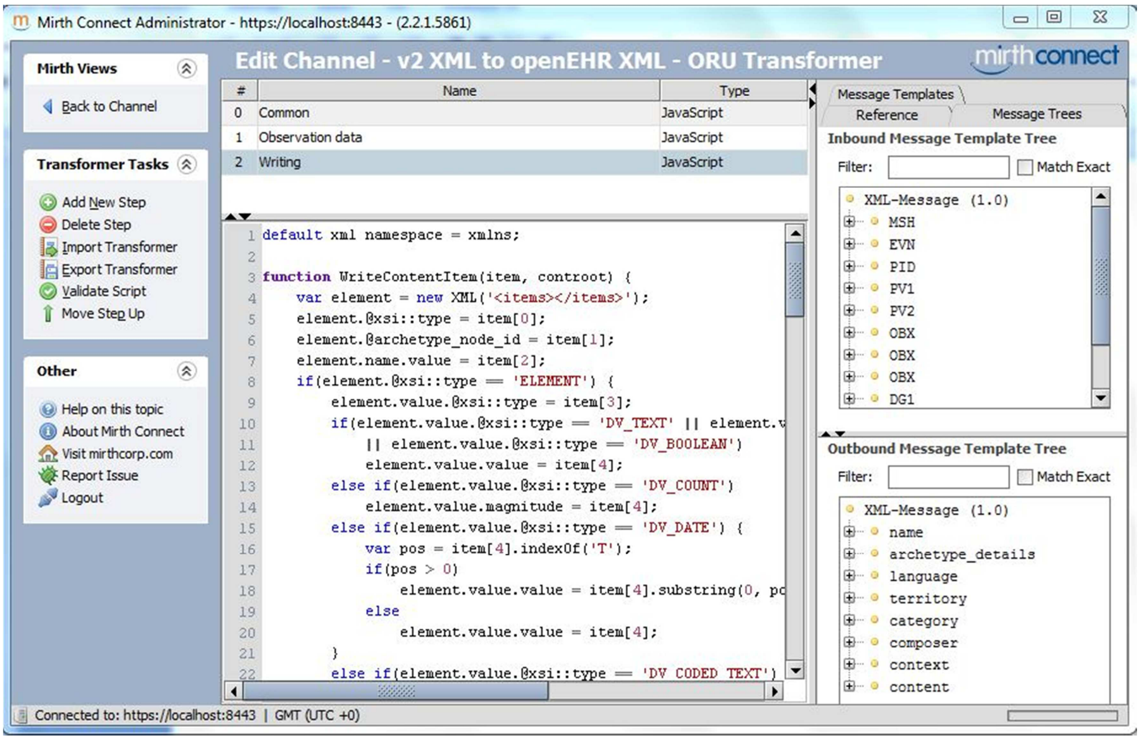


Figure 18 – Mirth Connect transform definition example

Figure 18 shows an example of the interface for defining transforms for the created channels. There are always panels on the left with contextual commands to change views and perform tasks. On the top center the list of transformer steps to follow is shown and their type can be modified. JavaScript was the type used, but Mirth Connect also provides a Message Builder type to build HL7 message segments, a Mapper type to place values from a message into variables which can be used to build other messages, an External Script type to use external JavaScript files and an XSLT Step type. Below is the panel in which to write the code for JavaScript steps or define other types of transformation steps through a provided interface of combo and text boxes. To the right on the Message Templates tab it is possible to define templates for the messages expected to arrive and for the ones to be sent either through a file or manually. These templates are parsed into trees shown in the Message Trees tab which can be used to drag and drop fields to map into the JavaScript code or text boxes in the bottom center to help facilitate the mapping process. The Reference tab provides some predefined functions and the list of available variables for mapping.

4.3.1 Used HL7 v2.x message types

First of all, it is necessary to understand the types and structure of messages that the system will need to handle. Figure 19 shows an HL7 v2.x message in its original format and Figure 20 shows an HL7 v2.x message in v2.xml format. The message structure is mostly the same in both formats, with XML tags replacing the different separators of the original format in the v2.xml format. Component and subcomponent values are wrapped in numbered XML tags indicating their position which in turn are wrapped in another XML tag which represents the message segment, all within the HL7Message tag that is the root element of the message.

```
MSH|^~\&|ADT1|MCM|LABADT|MCM|198808181126|SECURITY|ADT^A01|MSG00001-|P|2.4
EVN|A01|198808181123
PID|||PATID1234^5^M11||JONES^WILLIAM^A^III||19610615|M-||C
PV1|1|I|2000^2012^01||||004777^LEBAUER^SIDNEY^J.|||SUR||-||ADM|A0
AL1|1||^PENICILLIN||PRODUCES HIVES~RASH~LOSS OF APPETITE
DG1|001|I9|1550|MAL NEO LIVER, PRIMARY|19880501103005|F
PR1|2234|M11|111^CODE151|COMMON PROCEDURES|198809081123
```

Figure 19 – HL7 v2.x message example

```
<HL7Message>
    <MSH>
        <MSH.1>|</MSH.1>
        <MSH.2>^~\&amp;</MSH.2>
        <MSH.3>
            <MSH.3.1>SOURCE</MSH.3.1>
        </MSH.3>
        <MSH.4>
            <MSH.4.1>383018129</MSH.4.1>
        </MSH.4>
        <MSH.5>
            <MSH.5.1>PRIORITY HEALTH</MSH.5.1>
        </MSH.5>
        <MSH.6>
            <MSH.6.1>382715520</MSH.6.1>
        </MSH.6>
        <MSH.7>
            <MSH.7.1>2007100914484648</MSH.7.1>
        </MSH.7>
```

Figure 20 – HL7 message in v2.xml format

The HL7 v2.x standard defines many possible message types for different purposes (106 in version 2.6 [85], for instance), all of which consist of a mix of mandatory and optional segments that carry the relevant transmitted data within their components. Due to time constraints, the number of message types, as well as the composing segments, used throughout this work is limited and represents some simplified and common information that might be commonly exchanged between health and care systems. A brief description of each message type and corresponding segments used is included in the following subsections.

4.3.1.1 ADT – Admit Discharge Transfer

As its name suggests, this message type is used to record admissions of patients to a healthcare facility, for instance, as well as their discharge or transfer to other facilities. There are some different types of structure this message type may adopt to reflect the event it records. For the purposes of this work, only the A01 structure was considered and is mainly used to record patient admissions and registrations, so it is a more administratively focused message that may nevertheless include some initial observations with clinical data.

It was chosen for usage in this prototype mainly to represent registrations of new individuals into the system that may already introduce relevant clinical data from observations made in an initial consultation or by the subject himself beforehand.

Within this message structure the segments in Table 1 were used for the transformation process.

Identifier	Name	Optional (Y/N)	Repeating (Y/N)
MSH	Message Header	N	N
EVN	Event Type	N	N
PID	Patient Identification	N	N
PV1	Patient Visit	Y	N
OBX	Observation/Result	Y	Y

Table 1 – ADT message segments

The MSH, PID, PV1 and OBX segments are also present and serve the same function in all the other used message types. The MSH segment transmits the intent, source, destination, as well as some specifics of the syntax of the message. The EVN segment communicates information about the message's trigger event. The PID segment is used to transmit patient identifying and demographic data, such as name, date of birth, and so forth. The PV1 segment is used to transmit account or visit-specific data, such as a patient's admission type, current and prior location, attending, consulting and referring doctor, among others. The OBX segment transmits a single observation or observation fragment and its primary purpose is to carry information about observations in a report message, serving as its smallest indivisible unit. It may, however, also be found in other messages that need to include clinical information.

4.3.1.2 ORU – Unsolicited Observation Message

This message type is used to transmit observations and their results and was designed particularly with laboratory automation processes in mind. It is, however, suitable to transmit most other types of observations as well, such as those made in a routine consultation by a physician for instance. It was chosen to represent transmission of observations such as blood pressure or heart rate measurements and more complex laboratory test results, for example. The segments of this message structure used are presented in Table 2.

Identifier	Name	Optional (Y/N)	Repeating (Y/N)
MSH	Message Header	N	N
PID	Patient Identification	N	N
PV1	Patient Visit	Y	N
OBR	Observations Report ID	N	Y
OBX	Observation/Result	Y	Y

Table 2 – ORU message segments

In this structure, OBR segments serve as report headers. They may include relevant ordering information and contain many of the attributes that usually apply to all of the included atomic observations that follow them. OBX segments, then, are transmitted as a single component of an observation report identified by an OBR segment in this message type. For example, if an OBR segment pertains to a blood pressure report, it may be followed by two OBX segments separately containing the systolic and diastolic values recorded in the observation.

4.3.1.3 PPR – Patient Problem Message

As its name implies, this message type is used to transmit patient problems between applications. These problems include possible symptoms detected by the patient or a caregiver as well as known diagnosis information and it is for this reason that this message type was chosen.

Table 3 presents the segments of this message structure used in the transformation process.

Identifier	Name	Optional (Y/N)	Repeating (Y/N)
MSH	Message Header	N	N
PID	Patient Identification	N	N
PV1	Patient Visit	Y	N
PRB	Detail Problem	N	Y
OBX	Observation/Result	Y	Y

Table 3 – PPR message segments

For this and the following message types, OBX segments from received messages are read and their data is written into the same openEHR composition. However, when transforming from compositions to these message types, observation data is not written. It is instead inserted into a separate ORU message as is explained in the subsection for the corresponding Mirth Connect channel (openEHR XML to HL7 v2.x) further along in this document.

The PRB segments contain the relevant information about an individual's problem, such as the time of its identification by a caregiver, its date of onset, its classification, prognosis and so on.

4.3.1.4 PGL – Patient Goal Message

This message type, as its name suggests, is used to transmit goals for the patient. These goals may be desired states of health and wellbeing to achieve within a time frame defined by a caregiver or the patient himself and for this reason this message type was chosen.

The segments from this message structure used in the transformation process are presented in Table 4.

Identifier	**Name**	**Optional (Y/N)**	**Repeating (Y/N)**
MSH	Message Header	N	N
PID	Patient Identification	N	N
PV1	Patient Visit	Y	N
GOL	Detail Goal	N	Y
OBX	Observation/Result	Y	Y

Table 4 – PGL message segments

OBX segments for this message type are handled in the same fashion as for the previously described message type.

GOL segments are similar in function to the PRB segments used in the previously presented message type, but the data it transmits pertains to an individual's goals with information such as priority, time of establishment, review information, and so on.

4.3.1.5 OMP – Pharmacy/Treatment Order Message

This message type is used to record treatment or medication orders made for a patient by a care provider. It may include information on medications of various nature as well as clinical or other care treatments to be administered to a specific patient. It was chosen for this prototype mainly for the information on medications it can transmit, with other kinds of treatment possible to explore in the future. This decision was made as the corresponding openEHR archetype for medications is simple and well defined whereas other treatment types encompass several archetypes, one for each specific treatment. This message structure's utilized segments are presented in Table 5.

Identifier	Name	Optional (Y/N)	Repeating (Y/N)
MSH	Message Header	N	N
PID	Patient Identification	N	N
PV1	Patient Visit	Y	N
ORC	Order Common	N	Y
RXO	Pharmacy/Treatment Order	N	Y
RXR	Pharmacy/Treatment Route	Y	Y
RXC	Pharmacy/Treatment Component	Y	Y
OBX	Observation/Result	Y	Y

Table 5 – OMP message segments

For this message type, ORC segments are used to record data fields that are common to all kinds of orders possible within the system, while RXO segments are used for data specific to medication or treatment orders. RXR and RXC segments are associated with the preceding RXO segment, meaning their transmitted data pertains only to that specific order. RXR segments contain data about the site and method of administration and RXC segments contain data about the components of the medication or treatment ordered. The OBX segments are handled in the same manner as the previous message types.

4.3.1.6 RAS – Pharmacy/Treatment Administration Message

As its name implies, this message type is used to transmit information regarding the administration of ordered medications or treatments. This type of information may be recorded by a caregiver or the patient himself depending on the kind of treatment being administered. It was designed to record all the relevant information on the time and procedure of one or several administration instances of a particular order and was chosen for this prototype mainly to serve as follow-up to the previous message type and so, the focus is also on medications. Table 6 presents the segments of this message structure used.

Identifier	Name	Optional (Y/N)	Repeating (Y/N)
MSH	Message Header	N	N
PID	Patient Identification	N	N
PV1	Patient Visit	Y	N
ORC	Order Common	N	Y
RXO	Pharmacy/Treatment Order	N	Y
RXR	Pharmacy/Treatment Route	Y	Y
RXC	Pharmacy/Treatment Component	Y	Y
RXA	Pharmacy/Treatment Administration	N	Y
OBX	Observation/Result	Y	Y

Table 6 – RAS message segments

This message type's ORC, RXO, RXR and RXC segments are handled in the same manner as for the previously described message type with the addition of one or multiple RXA segments associated with each ORC segment as well. These RXA segments each contain data about a single instance of administration for the associated order. The OBX segments are handled in the same fashion as the previous message types.

4.3.2 Channels and transforms

In order to implement the transformations of messages required for the proposed system, a total of four Mirth Connect channels were created to both apply the transforms implemented in JavaScript and route the resulting messages to the desired destination. The transforms for the messages are located in the defined destinations of each channel and map the relevant field values of one standard to the appropriate fields of the other standard.

All of the openEHR archetypes used in the channels described here were obtained from the openEHR Clinical Knowledge Manager (CKM) [86].

4.3.2.1 V2 XML router

This is the simplest of the created channels and serves only to route any messages received in the v2.xml format to the correct channel (v2 XML to openEHR XML) which will transform them into openEHR compositions in XML format. This channel was created as a way to avoid repeating transform coding since transforms from v2.xml and the original HL7 v2.x format would be identical, due to their previously presented similar structures, yet they are different file types.

4.3.2.2 HL7 v2 to v2 XML

This channel was created to transform messages received in the original HL7 v2.x format into the v2.xml format using a predefined transform provided by the Mirth Connect program itself. It serves as an intermediate step in the transformation of HL7 v2.x messages into openEHR compositions and routes the results to the channel that will transform v2.xml messages (v2 XML to openEHR XML).

4.3.2.3 V2 XML to openEHR XML

This is the channel that transforms messages in the HL7 v2.x standard to openEHR compositions in XML format. It expects messages in the v2.xml format and applies different transforms according to the message type received. It receives messages directly from the two previous channels and according to their type routes them to one of its implemented destinations through filters defined in each of them to accept only that particular message type.

There are five different destinations for this channel, each with the name of the type of message expected.

ADT

This destination receives HL7 ADT messages and contains four transformer steps:

- **Common** – Contains a few variables and functions used in all transformer steps and maps the fields common to all types of messages
- **Admission data** – Prepares the necessary field values to place in the openEHR composition by retrieving them, mostly from the PV1 message segment, and placing them within an array variable to be used later for writing the composition as well as some other necessary values according to the openEHR-EHR-ADMIN_ENTRY.admission.v1 archetype
- **Observation data** – Prepares the necessary field values in the same fashion described above but for optional OBX segments that may be present in the message and uses the following archetypes:
    - openEHR-EHR-OBSERVATION.body_temperature.v1
    - openEHR-EHR-OBSERVATION.blood_pressure.v1
    - openEHR-EHR-OBSERVATION.body_weight.v1
    - openEHR-EHR-OBSERVATION.heart_rate.v1
- **Writing** – Writes the openEHR composition using the openEHR-EHR-COMPOSITION.encounter.v1 archetype and accessing the previously constructed variables with the required field values

ORU

This destination receives HL7 ORU messages and contains the **Common**, **Observation data** and **Writing** transformer steps described above with slight differences in the **Writing** step since there is no admission information in this message type.

PPR/PGL

This destination receives both PPR and PGL messages and, in addition to the **Common**, **Observation data** and **Writing** steps, contains two more transformer steps:

- **Problem data** – Prepares the necessary field values in the same fashion as the **Admission data** and **Observation data** steps but for PRB segments that may be present in the message and uses the following archetypes:
  - openEHR-EHR-EVALUATION.problem.v1
  - openEHR-EHR-EVALUATION.problem-diagnosis.v1
- **Goal data** – Prepares the necessary field values in the same fashion as the previously described data steps but for GOL segments that may be present in the message and uses the openEHR-EHR-EVALUATION.goal.v1 archetype

OMP

This destination receives OMP messages and contains the **Common**, **Observation data** and **Writing** transformer steps as well as the following one:

- **Order data** – Prepares the necessary field values in the same fashion as the previously described data steps but for ORC segments that may appear in the message as well as its associated RXO, RXC and RXR segments and uses the following archetypes:
  - openEHR-EHR-INSTRUCTION.medication.v1
  - openEHR-EHR-ITEM_TREE.medication.v1
  - openEHR-EHR-ITEM_TREE.medication-formulation.v1

RAS

This destination receives RAS messages and contains the same transformer steps as the OMP destination. However, the **Order data** step is different as it also requires field values from RXA segments of the message and additionally uses the openEHR-EHR-ACTION.medication.v1 archetype.

4.3.2.4 OpenEHR XML to HL7 v2

This channel transforms received openEHR compositions into HL7 v2.x messages and works in the opposite way of the previous channel, yet mapping directly to the original HL7 v2.x format. However, since some of the openEHR compositions may contain very different kinds of data that would normally be transmitted separately in HL7 v2.x messages, this channel may produce more than a single message, separating the data into its appropriate message type. For this purpose, filters are defined for each destination so that only messages with the appropriate data types are routed to it.

There are six different destinations for this channel, each with the name of the type of information that is transformed.

Admissions

This destination receives compositions with content elements that follow the openEHR-EHR-ADMIN_ENTRY.admission.v1 archetype and transforms them into HL7 v2.x ADT messages. It contains only one transformer step:

- **Admission** – Maps the necessary field values from the composition’s elements and creates an ADT message with MSH, EVN, PID and PV1 segments

Observation Reports

This destination receives compositions with content elements of the OBSERVATION type and transforms them into HL7 v2.x ORU messages. It contains two transformer steps:

- **Header & Patient** – Maps the necessary field values from the composition's elements and creates the MSH, EVN, PID and PV1 segments of the ORU message
- **Observations** – Processes the composition's OBSERVATION type content elements and creates their corresponding OBR segments as well as the associated OBX segments of the ORU message, mapping the necessary field values

Problems

This destination receives compositions with content elements that follow the openEHR-EHR-EVALUATION.problem.v1 or openEHR-EHR-EVALUATION.problem-diagnosis.v1 archetypes and transforms them into HL7 v2.x PPR messages. It also contains the **Header & Patient** transformer step and the following:

- **Problems** – Processes the composition's content items following the aforementioned archetypes and maps the necessary field values to create their corresponding PRB segments in the resulting HL7 v2.x message

Goals

This destination receives compositions with content elements that follow the openEHR-EHR-EVALUATION.goal.v1 archetype and transforms them into HL7 v2.x PGL messages. It contains the **Header & Patient** transformer step as well as the following:

- **Goal** – Processes the composition's content items following the aforementioned archetype and maps the necessary field values to create their corresponding GOL segments in the resulting message

Treatment Orders

This destination receives compositions with content elements that follow the openEHR-EHR-INSTRUCTION.medication.v1 or openEHR-EHR-INSTRUCTION.medication-formulation.v1 archetypes and transforms them into HL7 v2.x OMP messages. It contains the **Header & Patient** transformer step as well as the following:

- **Medication Order** – Processes the composition's content items following the aforementioned archetypes and maps the necessary field values to create their corresponding ORC segments with associated RXO, RXR and RXC segments when required

Treatment Administrations

This destination receives compositions with content elements that follow the openEHR-EHR-ACTION.medication.v1 archetype and transforms them into HL7 v2.x RAS messages. It contains the **Header & Patient** transformer step as well as the following:

- **Treatment Administration** – Processes the composition's content items following the aforementioned archetype and maps the necessary field values to create their corresponding ORC segments with associated RXO, RXR and RXA segments when required

Chapter 5

Evaluation

In order for an AAL system to use the kind of solution developed in this dissertation to face the interoperability problem, it is important to measure its performance in some way. Such a system would have to store and manage potentially thousands of records and so, for this to be a viable solution it must be able to deal with such a volume of information in a timely manner.

Taking into account the fact that the University of Aveiro has already published a thesis with performance results for a repository using the same storage technology [70], it would be a good starting point to make a comparison between theirs and this prototype's repository performance results to account for the changes in the BaseX API itself and in the storage of records.

For testing purposes, the repository was populated with some openEHR records created by the developed client application using some found Swedish composition examples [87] and HL7 message examples found on various sites and forums [88]–[91] were modified and used to test the transformation capabilities.

This evaluation process was separated into two aspects. Firstly, some of the repository functionalities were tested in order to compare the results with the aforementioned University of Aveiro thesis to verify if the differences in implementation also yield differences in performance. Secondly, a brief assessment was made regarding the feasibility of this solution based on the obtained results and taking into account the benefits in interoperability brought by the message transformation component of the solution. The repository performance tests used the same number of records as in the University of Aveiro thesis, namely 1,000, 10,000 and 30,000 records, not only for direct comparison but also to project the evolution of performance figures with an increasing number of records.

5.1 Repository evaluation

This part of the evaluation serves mainly as a comparison between the repository implemented in the course of this dissertation and the repository presented in the University of Aveiro thesis mentioned earlier in order to verify if the differences between the implementations produce any significant changes in terms of performance. As such, functionalities are assessed through the Graphical User Interface (GUI) application provided by BaseX which measures the time spent on each of them. It is important to note that in the implementation of this prototype the repository's indexes are always kept up-to-date by the application through calls to the *optimize* command whenever there is a change made in the repository and so all of the tests were run using the respective indexes.

5.1.1 Add new EHR

Figure 21 shows the results obtained for adding a new record to the repository in the three different storage states mentioned earlier through the BaseX GUI application.

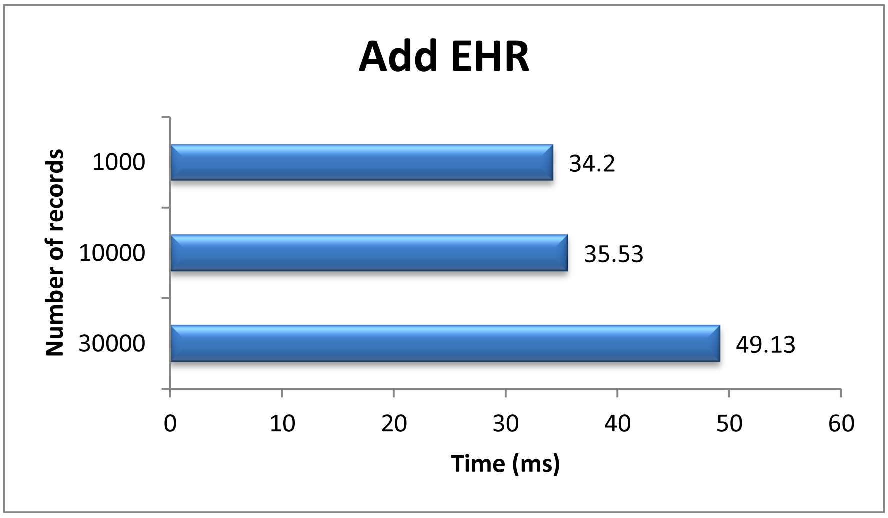


Figure 21 – Insert EHR results

As can be seen, the repository itself performs very well in this implementation when adding a new record. The time required to execute this operation does not significantly increase along with the number of records already stored.

However, testing with the developed client application yielded very different results. While inserting a new record into a repository with just 1,000 records was actually marginally faster (27.77 ms), the time required for the same operation to be executed in larger repositories increased dramatically (189.9 ms for 10,000 records and 562 ms for 30,000 records).

5.1.2 Search for a record

The graph in Figure 22 shows the time spent querying for a record in different positions of the repository. The input used for these queries is of the following type:

- /ehr[ehrID/value/text()='8cAC723e-bB3D-ecCD-dEEB-EBdAb1eF7EC5']

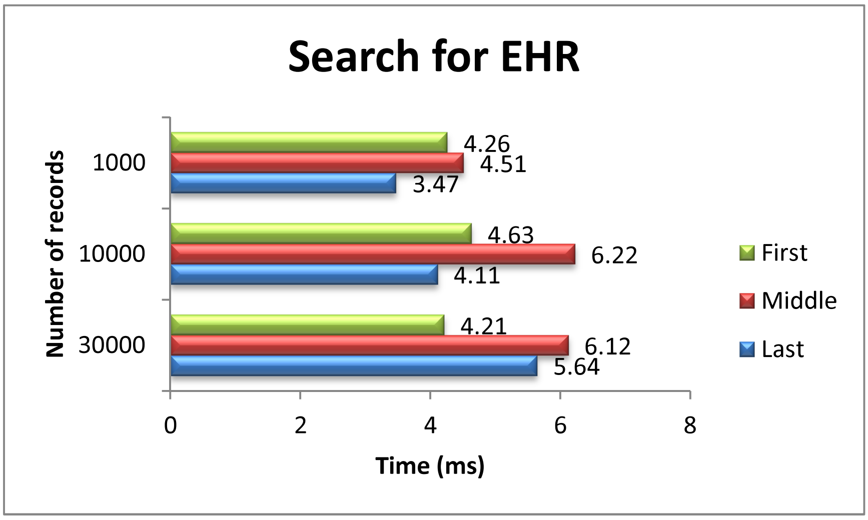
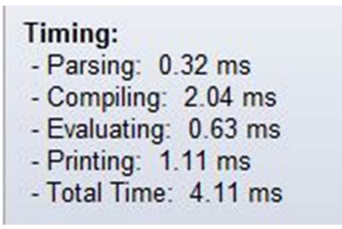


Figure 22 – Search record results

The difference between searching for a particular record at the very top, in the middle and at the very bottom of any of the databases is negligible, as is the difference between searching in a small or large database. It should be noted that, as Figure 23 shows, most of this time is spent on compiling the query as the evaluation time itself is brief. Printing may also take a significant



amount of time when the result of the query is long, as an entire record usually is.

Once again, using the developed client application proved much more time consuming, taking as long as 962 ms for the repository with 30,000 records. It should be noted, however, that with the records being separate resources in the repository it is possible to limit the querying context to a specific record by adding its path to the URL of the REST request. This is useful

Figure 23 – Query timing information

when querying data for a single subject as the search is only evaluated for that specific record and the time spent decreases. For example, to retrieve the entire record when querying a specific file, all that is needed is to return the file's root element and so, time spent is reduced to nearly half (486.91 ms).

5.1.3 Search for an attribute

For this test case three different queries were used to search for attributes at different levels within the records:

1. /ehr[ehrStatus/ownerID/id/@xsi:type='HIER_OBJECT_ID']
2. /ehr[ehrStatus/versions/commitAudit/committer/@xsi:type='PARTY_IDENTIFIED']
3. /ehr[ehrStatus/versions/data/subject/externalRef/id/@xsi:type='GENERIC_ID']

This kind of query is useful for population-wide searches, such as retrieving all blood pressure measurements recorded in the repository for instance, and may be important for research and statistical purposes.

As Figure 24 shows, this kind of query has a poorer performance than others and requires significantly more time with an increase in records. The level at which the attribute is found within the records, however, is not relevant as the times recorded for the different queries used were similar.

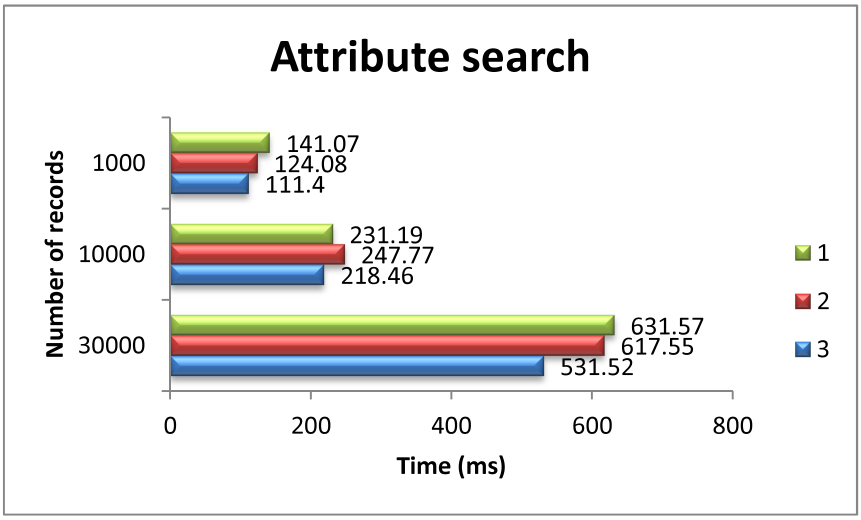


Figure 24 – Attribute search results

These results are not surprising since this repository separates the records as mentioned earlier and the University of Aveiro's thesis had already shown that the response time varies with the number of records already inserted.

5.1.4 Add composition

The recorded times of insertion of a composition to existing records in the repository are shown in Figure 25.

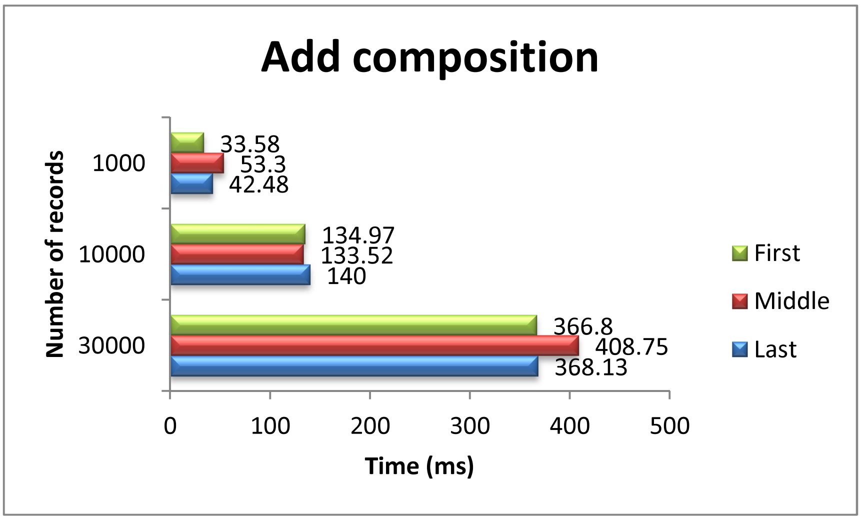


Figure 25 – Insert composition results

Once again, the very top, middle and very bottom positions of the records were tested and there was no significant difference between insertions in these positions.

This repository with the records as separate resources proved to be faster than with all the records under a single XML node, although the same proportional increase in time was recorded with the increasing number of records.

Using the developed client application the execution times were significantly increased again with insertion of a composition taking up to 4 seconds in a repository filled with 30,000 records.

5.1.5 Database size

Not only functionalities are important in a system of this nature. It is also important to be mindful of possible impacts on storage capacity to determine the resources required for a safe implementation.

It was verified that this repository implementation does not sufficiently reduce the storage space required for a large number of records, given that filled with 30,000 EHRs it occupied 302 MB of storage while the corresponding XML files outside the database occupied 456 MB.

5.2 Assessment

Given the results presented above, the repository implementation in the proposed solution is advantageous only for low volume systems where the benefits of storing each record separately outweigh the drawbacks. Adding new records and compositions in these situations is particularly fast, however, as more and more records are stored more impact it will have on the remaining functionalities. Querying within a specific record, on the other hand, still produces acceptable performance if the context is properly set, but population-wide queries are much more time consuming.

The main problem in managing the repository, however, seems to be the overhead of the REST requests as the client application developed presented considerably longer execution times for the same operations than the local BaseX GUI application. In addition to this, while keeping the indexes up-to-date is important to reduce time spent executing the required operations, executing the *optimize* command to update the index structures after an update can itself take a significant amount of time (approximately 30 seconds for the 30,000 record repository).

Overall, this particular implementation of the repository would be suitable for situations with few records to store and where it would be much more important to retrieve information for one single subject at a time than from the population. This is limiting for an AAL system where statistics and research studies are an important factor to consider and so, aggregating all the records under a single node seems to be more suitable for a high volume system.

Regarding the message transformation capabilities, Mirth Connect is capable of running alongside the client application and the tested HL7 v2.x messages were successfully transformed into openEHR compositions and vice-versa. The transformation process was quite fast despite the verbose nature of the openEHR compositions and extensive transforms written in JavaScript and any big performance hit should stem from whatever available protocol might be chosen to communicate between Mirth Connect and external applications that send messages to insert into the repository, since this component was only tested locally for the time being.

Chapter 6

Conclusions

The main goal of this dissertation was to propose a solution to the interoperability problem for use in AAL systems. For this purpose, it was necessary to first study some concepts of eHealth, particularly Electronic Health Records and transaction standards, as well as review different AAL products and services to understand the different application purposes and kind of information required in this field in order to choose which standards could best serve as a basis for the solution. The openEHR standard was chosen for its continuous evolution as part of an international community effort as well as its separation between the technical and domain models that grants flexibility to handle evolving information concepts. The HL7 v2.x standard was also chosen as it is the most commonly used transaction standard and therefore most likely to cover more real world scenarios. The study of the openEHR standard in particular proved challenging for someone new to the health information area as it is a very complete and complex standard.

It was also necessary to propose a simple architecture and define the information flow between its components in order to choose the technologies on which a prototype for the solution could be built. The proposed architecture consists of a repository for openEHR records, a client application to manage these records and a message broker that could transform messages between the two standards' formats. The technologies ultimately chosen were the BaseX XML database in conjunction with the openEHR Java Reference Implementation for the repository and client application as well as the Mirth Connect interface engine for the message broker.

The development of this work resulted in a prototype following the proposed architecture that is capable of dealing with a few common scenarios of information that an AAL system would use. Another difficulty encountered was finding available test data. Although some examples were eventually found, most of them had to be slightly modified, mostly to fill empty data fields or add new ones that would be required for testing with fake information.

When evaluating the proposed system, it was verified that keeping the records as separate resources within the repository did not yield the expected results, as performance of some of the basic functionalities was slower than the performance recorded in a repository with the records aggregated in a single node. However, the required information was inserted and managed correctly using the developed prototype and the implemented message transformation component could prove valuable in conjunction with the right repository structure to provide interoperability as it is expandable to many transaction standards and communication protocols, allowing the system to receive messages and store data from many different applications.

The essential objectives for this dissertation were therefore achieved, although integrating the prototype with an external application, such as eHealthCom, would have been beneficial. Time constraints, however, prevented the execution of this step.

6.1 Future work

Besides the integration with one or more external systems, there are still a number of ways in which the work done in this dissertation can be expanded upon.

At the moment only the HL7 v2.x standard is used for transformation of messages to be exchanged between this prototype and other systems, however, the Mirth Connect interface engine is designed to be able to deal with a number of different standards and formats, so the inclusion of other standards such as version 3 of HL7 for instance, with the correct mapping will allow the use of this type of system with a more varied array of applications. Even within the already used standards, only a subset of them were actually implemented, so it would also be beneficial to expand the use of more openEHR archetypes and more HL7 message types to provide a more complete implementation of the standards.

Mirth Connect was used in this work to receive messages and store their resulting transformations locally in the file system so that these files could then be used by the client application to insert new compositions into records within the repository. This was done for testing purposes so that the user may verify the result of the transformation and confirm the format and content of the information inserted into the repository. However, Mirth Connect's provided communication protocols could be used to automatically transmit its transformed messages to an application through web services, for instance, or the client application could automatically check a set system folder for new files to insert automatically. It would be interesting to also investigate performance figures for these situations.

Also, the focus of this work was on the clinical information stored and exchanged in an AAL system and other considerations such as security, access control and demographic information are assumed to be handled by external applications at this time but it is possible to incorporate these aspects of such a system into this kind of solution. Another important aspect is the fact that AAL systems require more than just clinical information. Other information concerning a subject's well being and required assistance are also part of an AAL system and should be handled and stored as well. This was not done in this dissertation because clinical information is currently much better defined and accepted and so, ideal for a starting point until the other kinds of information are agreed upon and integrated into the system.

References

[1] K. Gassner e M. Conrad, «ICT enabled independent living for elderly», A status-quo analysis on products and the research landscape in the field of Ambient Assisted Living (AAL) in EU-27. Berlin: Institut for Innovation and Technology (iit), 2010.

[2] «AAL4ALL | Ambient Assisted Living For All». [Online]. Available: http://www.aal4all.org/?lang=en. [Accessed: 22-Jul-2012].

[3] «WHO | eHealth», WHO. [Online]. Available: http://www.who.int/topics/ehealth/en/. [Accessed: 22-Jul-2012].

[4] Empirica, «“Benchmarking ICT use among General Practitioners in Europe” Final Report», European Commision, Information Society and Media Directorate General, Abr 2008.

[5] Deloitte & Ipsos Belgium, «eHealth Benchmarking III», European Commision, Information Society and Media Directorate General, Abr 2011.

[6] D. Garets e M. Davis, «Electronic medical records vs. electronic health records: yes, there is a difference», HIMSS Analytics, pp 2–14, 2006.

[7] «openEHR - The World’s Record - openEHR :: future proof and flexible EHR specifications». [Online]. Available: http://www.openehr.org/301-OE.html. [Accessed: 22-Jul-2012].

[8] «Portal da Saúde - Plataforma de Dados de Saúde». [Online]. Available: http://www.min-saude.pt/portal/conteudos/a+saude+em+portugal/noticias/plataforma+dados+saude.htm. [Accessed: 22-Jul-2012].

[9] «epSOS: Home». [Online]. Available: http://www.epsos.eu/. [Accessed: 22-Jul-2012].

[10] «Google Health». [Online]. Available: http://www.google.com/intl/en_us/health/about/index.html. [Accessed: 22-Jul-2012].

[11] «Microsoft HealthVault», Microsoft HealthVault. [Online]. Available: http://www.HealthVault.com. [Accessed: 22-Jul-2012].

[12] «HIMMS Analytics Europe - EMR Adoption Model for Europe». [Online]. Available: http://www.himssanalytics.eu/docs/HA_EMRAM_Overview_EU_07272010.pdf. [Accessed: 20-Jan-2013].

[13] F. Freitas, S. Schulz, e E. Moraes, «Survey of current terminologies and ontologies in biology and medicine», RECIIS, vol 3, n 1, Mar 2009.

[14] «WHO | International Classification of Diseases (ICD)», WHO. [Online]. Available: http://www.who.int/classifications/icd/en/. [Accessed: 22-Jul-2012].

References

[15] R. Rudowski, «SNOMED – a tool for achieving semantic interoperability in e-health systems». [Online]. Available: http://powershow.com/view/1140eb-MDg4Y/SNOMED_powerpoint_ppt_presentation. [Accessed: 06-Fev-2013].

[16] «UMLS Quick Start Guide». [Online]. Available: http://www.nlm.nih.gov/research/umls/quickstart.html. [Accessed: 22-Jul-2012].

[17] «CPT - Current Procedural Terminology». [Online]. Available: http://www.ama-assn.org/ama/pub/physician-resources/solutions-managing-your-practice/coding-billing-insurance/cpt.page. [Accessed: 22-Jul-2012].

[18] «Logical Observation Identifiers Names and Codes (LOINC®) —». [Online]. Available: http://loinc.org/. [Accessed: 22-Jul-2012].

[19] «Fact SheetMedical Subject Headings (MeSH®)». [Online]. Available: http://www.nlm.nih.gov/pubs/factsheets/mesh.html. [Accessed: 22-Jul-2012].

[20] «RxNorm». [Online]. Available: http://www.nlm.nih.gov/research/umls/rxnorm/index.html. [Accessed: 22-Jul-2012].

[21] D. ISO e P. CEI, «ISO/IEC Directives, Part 2». Abr-2011.

[22] E31 Committee, «Specification for Continuity of Care Record (CCR)», ASTM International, 2006.

[23] «HL7 Standards Product Brief - HL7 Version 2 Product Suite». [Online]. Available: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=185. [Accessed: 22-Jul-2012].

[24] T. Benson, Principles of Health Interoperability HL7 and SNOMED. 2012.

[25] «HL7 Standards Product Brief - CDA® Release 2». [Online]. Available: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=7. [Accessed: 22-Jul-2012].

[26] Corepoint Health, «The Continuity of Care Document - Changing the Landscape of Healthcare Information Exchange». 2011.

[27] R. A. F. Costa, P. Novais, L. Lima, D. Samico, J. Oliveira, J. Machado, e J. Neves, «VirtualECare : intelligent assisted living», 2009. [Online]. Available: http://repositorium.sdum.uminho.pt/handle/1822/18997. [Accessed: 22-Jul-2012].

[28] Â. Costa, «IGenda: An Intelligent Agenda Organizer», 2009.

[29] «Parkinson Treatment». [Online]. Available: http://www.parkinsontreatment.eu/. [Accessed: 22-Jul-2012].

[30] «aal@ home: a New Home Care Wireless Biosignal Monitoring Tool for Ambient Assisted Living». [Online]. Available: http://unl-pt.academia.edu/HGamboa/Papers/1096143/aal_at_home_a_New_Home_Care_Wireless_Biosignal_Monitoring_Tool_for_Ambient_Assisted_Living. [Accessed: 22-Jul-2012].

References

[31] «Solutions». [Online]. Available: http://exa4life.com/index.php?option=com_content&view=article&id=47&Itemid=30&lang=en. [Accessed: 22-Jul-2012].

[32] «Home». [Online]. Available: http://exa4life.com/. [Accessed: 22-Jul-2012].

[33] «ALERT® PATIENT DATA MANAGEMENT SYSTEM (ALERT® PDMS) | ALERT® ONLINE - EN». [Online]. Available: http://www.alert-online.com/pdms. [Accessed: 22-Jul-2012].

[34] «NetCare - Fase 1 - Caracterização de Cenários de Utilização». 15-Fev-2008.

[35] T. Sapata, «Look4MySounds - A Remote Platform for Auscultation», 2010.

[36] C. P. Figueiredo, K. Becher, K. P. Hoffmann, e P. M. Mendes, «Low power wireless acquisition module for wearable health monitoring systems», in Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE, 2010, pp 704–707.

[37] S. Silva, P. M. Mendes, e L. Domingues, «Magtag-A wearable wrist device for localization applications», presented at the 4th International Conference on Pervasive Computing Technologies for Healthcare 2010, 2010, pp 1–2.

[38] «AMBIENT ASSISTED LIVING JOINT PROGRAMME | ICT for ageing well.» [Online]. Available: http://www.aal-europe.eu/. [Accessed: 22-Jul-2012].

[39] «eCAALYX Project Presentation». [Online]. Available: http://www.fraunhofer.pt/content/dam/portugal/en/documents/eCAALYX%20Project%20Presentation.pdf. [Accessed: 06-Fev-2013].

[40] M. Silva, P. M. Teixeira, F. Abrantes, e F. Sousa, «Design and Evaluation of a Fall Detection Algorithm on Mobile Phone Platform», in Ambient Media and Systems, vol 70, S. Gabrielli, D. Elias, K. Kahol, O. Akan, P. Bellavista, J. Cao, F. Dressler, D. Ferrari, M. Gerla, H. Kobayashi, S. Palazzo, S. Sahni, X. (Sherman) Shen, M. Stan, J. Xiaohua, A. Zomaya, e G. Coulson, Eds Springer Berlin Heidelberg, 2011, pp 28–35.

[41] L. Ferreira e P. Ambrosio, «Towards an interoperable health-assistive environment: The eHealthCom platform», in Biomedical and Health Informatics (BHI), 2012 IEEE-EMBS International Conference on, 2012, pp 930 –932.

[42] «Grails - The search is over.» [Online]. Available: http://grails.org/. [Accessed: 22-Jul-2012].

[43] «Groovy - Home». [Online]. Available: http://groovy.codehaus.org/. [Accessed: 22-Jul-2012].

[44] «java.com: Java + You». [Online]. Available: http://www.java.com/pt_BR/. [Accessed: 22-Jul-2012].

[45] «SOAP Introduction». [Online]. Available: http://www.w3schools.com/soap/soap_intro.asp. [Accessed: 22-Jul-2012].

References

[46] «UML tools for software development and modelling - Enterprise Architect UML modeling tool». [Online]. Available: http://www.sparxsystems.com.au/. [Accessed: 22-Jul-2012].

[47] «MySQL :: The world's most popular open source database». [Online]. Available: http://www.mysql.com/. [Accessed: 22-Jul-2012].

[48] «MySQL :: MySQL 5.0 Reference Manual :: 14.2 The InnoDB Storage Engine». [Online]. Available: http://dev.mysql.com/doc/refman/5.0/en/innodb-storage-engine.html. [Accessed: 22-Jul-2012].

[49] «MLDC | LUL». [Online]. Available: http://www.microsoft.com/pt-pt/mldc/lul/pt/pt/default.aspx. [Accessed: 22-Jul-2012].

[50] C. G. Pires, F. M. Pinto, E. M. Rodrigues, e M. S. Dias, «On the benefits of speech and touch interaction with communication services for mobility impaired users», AAL, pp 60–73, Jul 2012.

[51] «Activo PC Sénior». [Online]. Available: http://www.activopcsenior.com/index.htm. [Accessed: 22-Jul-2012].

[52] «Iniciativa de Segurança Idade Maior (ISIM)». [Online]. Available: http://www.rcc.gov.pt/Directorio/Temas/ServicosCidadao/Paginas/Iniciativa-de-Seguran%C3%A7a-Idade-Maior-(ISIM).aspx. [Accessed: 22-Jul-2012].

[53] «Projecto TIO». [Online]. Available: http://projectotio.net/. [Accessed: 22-Jul-2012].

[54] «Serviço Teleassistência». [Online]. Available: http://casa.telecom.pt/PTResidencial2/Tabs/MyPTPublico/Servicos/Teleassistencia/Teleassistencia.htm. [Accessed: 22-Jul-2012].

[55] «Cruz Vermelha Portuguesa - Cruz Vermelha Portuguesa». [Online]. Available: http://www.cruzvermelha.pt/. [Accessed: 22-Jul-2012].

[56] «Tele-Assistência». [Online]. Available: http://www.residenciasmontepio.pt/cart%C3%B5es-de-sa%C3%BAdeservi%C3%A7os/tele-assist%C3%AAncia.aspx. [Accessed: 22-Jul-2012].

[57] «Apoio Domiciliário - Médicos e Enfermeiros ao Domicílio». [Online]. Available: http://www.enfermeirospt.com/. [Accessed: 22-Jul-2012].

[58] «HelpPhone :: Entrada». [Online]. Available: http://www.helpphone.pt/index.jsp. [Accessed: 22-Jul-2012].

[59] «Federação Académica do Porto». [Online]. Available: http://www.fap.pt/pt/index.php?Pagb=fapsocial/aconchego&ch_site=fap. [Accessed: 22-Jul-2012].

[60] «Projecto Lado a Lado». [Online]. Available: http://www.cajp2cbr.org/index.php?option=com_content&view=section&id=34&Itemid=213. [Accessed: 22-Jul-2012].

References

[61] «Primus Care». [Online]. Available: http://primuscare.com/. [Accessed: 22-Jul-2012].

[62] «Fundação Vodafone Portugal». [Online]. Available: http://www.vodafone.pt/main/A+Vodafone/PT/Fundacao/. [Accessed: 22-Jul-2012].

[63] «Vodafone - Telemold». [Online]. Available: http://www.vodafone.pt/main/A+Vodafone/PT/Fundacao/ProjectosIniciativas/Saude/TELEMOLD. [Accessed: 22-Jul-2012].

[64] «:: Futuro Feliz em Família - Apoio Domiciliário ::» [Online]. Available: http://www.fff.pt/. [Accessed: 22-Jul-2012].

[65] «Comfort Keepers». [Online]. Available: http://www.comfortkeepers.pt/. [Accessed: 22-Jul-2012].

[66] «IEEE Standard Computer Dictionary. A Compilation of IEEE Standard Computer Glossaries», IEEE Std 610, 1991.

[67] T. Beale e S. Heard, Eds, «openEHR Architecture - openEHR Architecture Overview». 13-Nov-2008.

[68] T. Beale, S. Heard, D. Kalra, e D. Lloyd, Eds, «The openEHR Reference Model - EHR Information Model». 16-Ago-2008.

[69] D. Kalra, «The Good European Health Record», Comput Methods Programs Biomed, vol 45, n 1–2, pp 83–89, Out 1994.

[70] L. M. Velte, «Electronic health record repository based on the openEHR standard», 2011. [Online]. Available: http://ria.ua.pt/handle/10773/7479. [Accessed: 20-Jan-2013].

[71] «BaseX Documentation - Version 7.5». [Online]. Available: http://files.basex.org/releases/7.5/BaseX75.pdf. [Accessed: 06-Fev-2013].

[72] «HAPI - The Open Source HL7 API for Java». [Online]. Available: http://hl7api.sourceforge.net/. [Accessed: 20-Jan-2013].

[73] «Mirth Connect, The Leading Open Source HL7 Interface Engine - Mirth Corporation». [Online]. Available: http://www.mirthcorp.com/products/mirth-connect. [Accessed: 20-Jan-2013].

[74] «Mirth Connect FAQ - Mirth Connect - Confluence». [Online]. Available: http://www.mirthcorp.com/community/wiki/display/mirth/Mirth+Connect+FAQ. [Accessed: 20-Jan-2013].

[75] R. Chen e G. Klein, «The openEHR Java reference implementation project», Studies in health technology and informatics, vol 129, n 1, p 58, 2007.

[76] «Standard Java libraries for Java implementations of openEHR», GitHub. [Online]. Available: https://github.com/openEHR/java-libs. [Accessed: 20-Jan-2013].

References

[77] «Welcome to The Apache Software Foundation!» [Online]. Available: http://www.apache.org/. [Accessed: 20-Jan-2013].

[78] «javax.xml.validation (Java 2 Platform SE 5.0)». [Online]. Available: http://docs.oracle.com/javase/1.5.0/docs/api/javax/xml/validation/package-summary.html. [Accessed: 20-Jan-2013].

[79] «JDOM». [Online]. Available: http://www.jdom.org/. [Accessed: 20-Jan-2013].

[80] «javax.xml.bind (Java EE 5 SDK)». [Online]. Available: http://docs.oracle.com/javaee/5/api/javax/xml/bind/package-summary.html. [Accessed: 20-Jan-2013].

[81] «REST - BaseX Documentation». [Online]. Available: http://docs.basex.org/wiki/REST. [Accessed: 20-Jan-2013].

[82] «jetty - Jetty WebServer». [Online]. Available: http://jetty.codehaus.org/jetty/. [Accessed: 20-Jan-2013].

[83] «basex-examples», GitHub. [Online]. Available: https://github.com/BaseXdb/basex-examples. [Accessed: 20-Jan-2013].

[84] «REST: POST Schema - BaseX Documentation». [Online]. Available: http://docs.basex.org/wiki/REST:_POST_Schema. [Accessed: 20-Jan-2013].

[85] «HL7 v2.6 Data Definition Tables». [Online]. Available: http://www.hl7.org/special/committees/vocab/v26_appendix_a.pdf. [Accessed: 06-Fev-2013].

[86] «Clinical Knowledge Manager». [Online]. Available: http://openehr.org/knowledge/. [Accessed: 20-Jan-2013].

[87] «Old Nabble - openehr-technical - Sample OpenEHR records». [Online]. Available: http://old.nabble.com/Sample-OpenEHR-records-td31034728.html#a31057174. [Accessed: 20-Jan-2013].

[88] «EKG lab example messages», 20-Jan-2013. [Online]. Available: http://wiki.interfaceware.com/544.html. [Accessed: 20-Jan-2013].

[89] «Sample HL7 2.x and 3.x Messages». [Online]. Available: http://www.xmlconverters.com/tutorials/sample-hl7-v2-and-v3-messages.html. [Accessed: 20-Jan-2013].

[90] «HL7 sample messages | Priority Health | Leading Michigan insurance company». [Online]. Available: http://www.priorityhealth.com/provider/manual/office-mgmt/data-exchange/hl7/hl7-samples. [Accessed: 20-Jan-2013].

[91] «Sample HL7 Immunization Messages». [Online]. Available: http://www.dt7.com/cdc/sampmsgs.html. [Accessed: 20-Jan-2013].