



**David Manuel
Carvalho De Abreu
Fontes**

**Sistema Web para Optimização do Workflow em
Serviço Radiologia**

**Web system for Workflow Optimization in
Radiology Service**



**David Manuel
Carvalho De Abreu
Fontes**

**Sistema Web para Optimização do Workflow em
Serviço Radiologia**

**Web system for Workflow Optimization in
Radiology Service**

*“Supreme excellence consists in breaking the enemy’s resistance
without fighting. Sun Tzu”*



**David Manuel
Carvalho De Abreu
Fontes**

**Sistema Web para Optimização do Workflow em
Serviço Radiologia**

**Web system for Workflow Optimization in
Radiology Service**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Carlos Manuel Azevedo Costa, Professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Dedico este trabalho aos meus pais, pelo enorme esforço que fizeram para que eu conseguisse concluir os meus estudos, ao meu irmão Henrique e irmã Joana pelos conselhos e apoio sempre disponível. Gostaria também de deixar um enorme obrigado a todos os meus amigos, especialmente ao Adnre Alves e ao Miguel Valério pelos conselhos, apoio e brincadeiras durante esta fase da minha vida. Seguindo o mesmo pensamento, também gostaria de deixar um enorme obrigado ao Daniel Oliveira, ao Luís Menezes e ao João Ribeiro (Meleon) pela companhia proporcionada nas árduas noites de estudo e trabalho.

o júri / the jury

presidente / president

Prof. Doutor Augusto Marques Ferreira da Silva

Prof. Auxiliar do Dep. Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

vogais / examiners committee

Prof. Doutor Joel P. Arrais

Prof. Auxiliar Convidado Dep. Informática da Universidade de Coimbra

Prof. Doutor Carlos Manuel Azevedo Costa

Prof. Auxiliar do Dep. Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

**agradecimentos /
acknowledgements**

Agradeço toda a ajuda e incentivo disponibilizado pelos meus orientadores, Luís A. Bastião Silva da BMD Software e Doutor Prof. Carlos Manuel Azevedo Costa. Também gostaria de deixar um agradecimento ao grupo UA.PT Bioinformatics por me ter acolhido e contribuído de alguma forma para o meu crescimento intelectual, nas inúmeras partilhas de experiências que se realizaram ao longo do ano lectivo.

Palavras Chave

radiologia, ris, imagem, medica, relatório, PACS, DICOM, DICOMSR, anotações, voz, modelos.

Resumo

A ampla adoção de imagens médicas em formato digital nos diversos tipos de instituições de saúde, levantou novos problemas ao nível da gestão de dados e processos. A normalização destes cenários tem sido alvo de atenção nas últimas décadas, esforço que resultou no desenvolvimento e dinamização de normas como DICOM e HL7. Atualmente coexistem dois tipos de sistemas de informação num laboratório de imagem médica que devem funcionar de forma integrada, os RIS que são responsáveis pela gestão das tarefas administrativas e os PACS que fazem a gestão das imagens e informação associada. Esta dissertação teve como objetivo desenhar e implementar uma solução RIS baseada em ferramentas de utilização livre ou código aberto. Assim, começamos por estudar detalhadamente o estado da arte, incluindo soluções do domínio público e proprietárias, destacando os pontos fortes e fraquezas de cada uma. Para além da análise das tecnologias utilizadas no desenvolvimento de cada solução, este estudo teve contributos determinantes na análise de requisitos efetuada. Nomeadamente, permitiu-nos identificar funcionalidades inovadoras e com elevado valor para os utilizadores. O resultado é um sistema de informação capaz de gerir todas as operações de um departamento de radiologia, incluindo gestão administrativa de utentes, agendamento de exames, realização de relatórios clínicos, entre outras. Em termos de características inovadoras destaca-se o módulo de relatório que permite carregar novos modelos de relatórios com o sistema em produção e a sua exportação para o formato *standard* DICOM-SR, permitindo desta forma a sua integração com as imagens no repositório PACS. Em termos tecnológicos, desenvolveu-se uma aplicação web multiplataforma que segue uma arquitetura modular orientada a serviços e que oferece uma abstração relativamente à camada de persistência de dados.

Keywords

radiology, ris, image, medical, report, PACS, DICOM, DICOMSR, annotations, voice, templates.

Abstract

The widespread adoption of digital medical images in various types of health institutions, has raised new problems regarding data and processes management. The standardisation of these scenarios has been subject of attention in the last decades, resulting in the development and promotion of standards such as DICOM and HL7. Currently, there are two kinds of information systems in medical imaging laboratories, that must operate in a collaborative manner, RIS which is responsible for managing the administrative tasks and PACS that manage images and associated information. This dissertation aimed to design and implement an RIS solution based on tools with no use restriction or open source. We begin by studying in detail the state of the art, including the open source and proprietary solutions, highlighting the strengths and weaknesses of each one. In addition to analysing the technologies used in the development of each solution, this study provided decisive contributions, regarding the project requirements. In particular, it allowed us to identify innovative features with high value to users. The achieved solution is an information system capable of managing all operations in a radiology department, including administrative management of patients, exam scheduling, conducting clinical reports, among others. Regarding innovative features, the reporting module stands out, since it allows to upload new report templates into the system and export these clinical reports in the DICOM-SR standard, thus allowing their integration with the images in a PACS repository. Regarding the technologies aspect, it was developed a multi-platform web application that follows a modular service-oriented architecture and also provides an abstraction in regard to the data persistence layer.

CONTENTS

CONTENTS	i
LIST OF FIGURES	iii
LIST OF TABLES	v
ACRONYMS	vii
1 INTRODUCTION	1
1.1 Overview	1
1.2 Objectives	2
1.3 Thesis Outline	2
2 BACKGROUND	3
2.1 Digital Medical Imaging	3
2.2 Interoperability	4
2.3 PACS	4
2.4 RIS	6
2.5 DICOM	7
2.5.1 DICOM Objects and Organisation	9
2.5.2 DICOM Structure	10
2.5.3 DICOM Communication	11
2.5.3.1 Storage Service	12
2.5.3.2 Query/Retrieve Service	12
2.5.3.3 WADO	13
2.5.4 DICOM-SR	14
3 RELATED WORK	17
3.1 Open Source Solutions	17
3.1.1 GNU Health	17
3.1.2 OpenMRS	19
3.1.3 OpenEMR	22
3.1.4 ClearCanvas	25
3.2 Proprietary Solutions	28
3.2.1 Carestream Vue RIS	28
3.2.2 Fujifilm Synapse RIS	29
3.2.3 Agfa Impax RIS	30

3.3	Final Considerations	30
4	SYSTEM REQUIREMENTS	33
4.1	System Wide Requirements	33
4.1.1	Functional Requirements	33
4.1.2	Non-functional Requirements	36
5	SYSTEM ARCHITECTURE AND IMPLEMENTATION	39
5.1	Smart RIS Architecture	39
5.2	Server Application	41
5.3	Client Application	43
5.4	Data Persistence	44
5.4.1	Integration with Play Framework	45
5.4.2	Database Model	48
5.5	Communications	50
5.6	Reporting	52
5.7	PACS Archive	54
5.8	Security	54
5.9	Global Settings and Initialisation	55
6	RESULTS	57
6.1	Common Operations	57
6.2	Secretary	59
6.3	System Administrator	63
6.4	Technician	67
6.5	Physician	74
6.6	Other aspects	75
7	CONCLUSION	77
7.1	Conclusion	77
7.2	Future work	78
	REFERENCES	79

LIST OF FIGURES

2.1	Integration of a HIS, RIS and PACS	5
2.2	Major PACS Components Compliant with the DICOM standard	8
2.3	DICOM Information Model Hierarchy	9
2.4	IOD Example of a DICOM Object[13]	10
2.5	DICOM Data Elements with TLV Format	11
2.6	DICOM Storage Service	12
2.7	DICOM Query Service	13
2.8	DICOM Retrieve Service	13
2.9	DICOM WADO Service	14
2.10	DICOM SR Document Structure	15
2.11	DICOM SR Content Tree	15
3.1	Gnu Health Interface	19
3.2	OpenMRS Menu	21
3.3	OpenMRS Patient Search	21
3.4	OpenMRS Patient Personal Information	22
3.5	OpenEMR Menu	24
3.6	OpenEMR Patient Search	25
3.7	OpenEMR Patient Personal Information	25
3.8	Study Visualisation on ClearCanvas Viewer	26
3.9	Search Menu in ClearCanvas	27
3.10	Viewer Customisation on ClearCanvas	28
3.11	Open Source Comparison	31
4.1	Use Case Scenario	34
5.1	Smart RIS System Architecture	40
5.2	MVC Diagram	42
5.3	App Folder Tree	42
5.4	Database Model	50
5.5	Query Performed on Dicoogle	54
6.1	Login Interface	58
6.2	User Profile Operation	58
6.3	Personal Information Overview	59
6.4	Patient Form Overview	60
6.5	Patient Record Search	61

6.6	Daily View	62
6.7	Monthly View	62
6.8	New Event Form Overview	63
6.9	New Staff Record Form Overview	64
6.10	New Modality Form Overview	65
6.11	Modality Record Search	65
6.12	New Facility Form Overview	66
6.13	New Room Form	66
6.14	Settings Menu	67
6.15	Menu for Selecting a Subset of Templates	68
6.16	Technician's Worklist	69
6.17	Patient's Previous Events	70
6.18	Exam Data Displayed with Dicoogle	71
6.19	Clinical Report	72
6.20	Pending Reports Notification	72
6.21	Events with a Pending Report	73
6.22	Clinic Report Presented to the Technician	74
6.23	Referring Table	74
6.24	Clinical Report Reviewing	75
6.25	IOS tablet accessing Smart RIS	76
6.26	Android smartphone accessing Smart RIS	76

LIST OF TABLES

2.1	DICOM Services and Related DICOM Commands	11
5.1	Activator Commands	41
5.2	JPA Annotations	45
5.3	HTTP Methods Overview	51
5.4	HTTP Status Code Overview	51

ACRONYMS

PACS	Picture archiving and communication system	IHE	Integrating the Healthcare Enterprise
HIS	Hospital Information System	HL7	Health Level Seven
RIS	Radiology Information System	OpenMRS	Open Medical Record System
EMR	Electronic Medical Record	API	Application Programming Interface
EHR	Electronic health record	FHIR	Fast Healthcare Interoperability Resources
RBAC	Role-based access control	REST	Representational State Transfer
DICOM	Digital Imaging and Communications in Medicine	OpenEMR	Open Electronic Medical Records
DICOM SR	DICOM Structured Reporting	ONC	Office of the National Coordinator
CT	Computer Tomography	ACL	Access Controls Listing
US	Ultrasounds	CDR	Clinical Decision Rules
PET	Positron Emission Tomography	CDS	Clinical Decision Support
XA	Angiography	DES	Data Encryption Standard
MRI	Magnetic Resonance Imaging	SMS	Short Message System
DIM	DICOM Information Model	SOA	Service-Oriented Architecture
UID	Unique identifier	ORM	Object-Relational Mapping
IOD	Information Object Definition	JPA	Java Persistence API
IOD-ER	IOD Entity-Relationship	JDO	Java Data Objects
TLV	Tag Length Value	API	Application Programming Interface
IE's	Information Entities	RAM	Random-Access Memory
AE	Application Entity	JDK	Java Development Kit
AEs	Application Entities	MVC	Model, View, Controller
OSI	Open Systems Interconnection	UML	Unified Modeling Language
SCU	Service Class User	REST	Representational State Transfer
SCP	Service Class Provider	RPC	Remote Procedure Call
WADO	Web Access to DICOM persistent Objects	SOAP	Simple Object Access Protocol
		HTTP	Hypertext Transfer Protocol
		URI	Uniform Resource Identifier
		URL	Uniform Resource Locator

HTML	HyperText Markup Language	MRRT	Management of Radiology Report Templates
HTTPS	Hyper Text Transfer Protocol Secure		
PDF	Portable Document Format	FS	Fully supported
RSNA	Radiology Society of North America	PS	Partially supported
IT	Information Technology		
OOP	Object-Oriented Programming	NS	Not supported

CHAPTER 1

INTRODUCTION

This chapter provides an introduction regarding this document. It also presents an overview about medical imaging scenarios, along with some of the objectives of this dissertation, with a brief description of each chapter.

1.1 OVERVIEW

The fast evolution of Information Technology (IT) raised an growth in the medical imaging area. As a result, several medical facilities have leveraged IT infrastructures, into including medical imaging in their workflows, promoting better healthcare services. Following these trends, in clinical radiology there is also a growing need to manage efficiently all the resources, as well as to delegate administrative responsibilities to specific members of the medical staff. These steps should be performed in a fast way, in order to allow the clinicians to provide a quality healthcare service.

The broad and wide adoption of medical imaging in healthcare institutions, led to the development of standards like Digital Imaging and Communications in Medicine (DICOM) and Health Level Seven (HL7), facilitating the storage and communication processes. Nevertheless, the scheduling of exams or appointments, processes of assigning tasks to medical staff, reporting and other administrative tasks, still require integration efforts and optimisation of workflows.

Radiology Information System (RIS) is the system responsible for managing medical imaging laboratories, processes, equipment and data. It is responsible for implementing all the necessary logic, regarding operations in a radiology environment, such as scheduling exams, save and review clinical reports, management of patient records, and so on. Usually, a RIS works in a integrated way with a Picture archiving and communication system (PACS), allowing data like clinical reports or medical images, to be managed by this separate system, which specifies standards on how to store and distribute such data. The requirements proposed in this dissertation, were gathered from a thorough analysis performed on the open source and proprietary existing RIS, highlighting the strengths and weaknesses of each system. Also in the same study, it is presented a list of features, that could optimise a clinic workflow, some of which based on new standard updates.

1.2 OBJECTIVES

This dissertation proposes a new web-oriented RIS, labelled Smart RIS, capable of managing all operations encompassed in a radiology clinic's workflow, as well as supporting the integration of distinct PACS systems, to achieve an organised DICOM environment.

Clinical reports are crucial in a healthcare institution. For this reason, this document also proposes a way to manage normalised report templates, allowing the integration of new templates on a system in production. By providing structured reports, compliant with the DICOM Structured Reporting (DICOM SR) standard, new ways of searching and storing reports can be implemented.

With tools for managing a radiology clinic, such as workflow management and role-based operations, the proposed solution can maximise the resources usage, as well as acclaim for responsibility segregation, making it suitable for standalone institutions. For this reason, a set of administrative operations is also presented in Smart RIS.

1.3 THESIS OUTLINE

The presented document is divided into several chapters, in order to provide an organised walk-through, of the addressed matters. A brief description of each chapter, will now be given:

- **Chapter 2:** Introduces the necessary background information, in order to understand and explain the current situation in medical imaging laboratories. This is achieved by approaching key subjects like Interoperability, PACS, RIS, DICOM.
- **Chapter 3:** Presents the currently existing solutions, that make up the state of the art. Additionally, it is presented a few examples of RIS systems, along with the discussion of key features and disadvantages of each one. A comparison table is also included in this chapter, in order to briefly resume the analysed open source solutions.
- **Chapter 4:** Exposes the requirements for the development of this project. Consequently, allowing for a better understanding of the implementation decisions.
- **Chapter 5:** Provides a brief explanation regarding implementation aspects utilised in Smart RIS. As it stands, it offers a concise knowledge regarding third-party components employed, technologies that were used, database model applied in this solution, and so on.
- **Chapter 6:** Shows the obtained results of the proposed system. Focusing on key features, along with a brief explanation and illustrated examples.
- **Chapter 7:** Presents final considerations along with future work, that can improve the developed solution.

CHAPTER 2

BACKGROUND

This chapter introduces some background knowledge, in order to facilitate the state of the art discussion, about radiology information systems that will be presented in the next chapter. To do so, each area of interest is divided into a smaller section, providing a scope division regarding each subject.

2.1 DIGITAL MEDICAL IMAGING

It is well known that technology is in rapid evolution. This also applies to the healthcare sector, where technological innovations can greatly improve diagnosis, treatment and follow up of patient health. As a result, there is a continuous search for technological improvement in different areas, regarding the healthcare sector.

In the last couple of decades, there was an evolution in all aspects regarding medical imaging. For instance, a shift from film-base media to digital media, as the mean of storage and distribution. This change led to significant improvement in the physician's workflows, and consequently an improvement in the healthcare provided [1].

As it stands, medical imaging encompasses all the techniques and processes, used to create a visual representation of the interior of the human body. As a result, a physician can gather as much information as possible, in order to make reliable decisions regarding the patient's diagnosis.

Currently, medical imaging is divided into different modalities, like Computer Tomography (CT), Positron Emission Tomography (PET), Ultrasounds (US), Magnetic Resonance Imaging (MRI), Angiography (XA), and so on. Since there is an extensive set of modalities, a health care facility will not support all modalities required to correctly evaluate a patient. As a consequence, dispersion of medical data can occur, since a patient may need to visit different institutions, in order to perform all the exam modalities needed. For this reason, is crucial for various institutions, to safely exchange patient's medical data , as well as some clinical data relevant to the study. This data share can be a challenge, since it relies on interoperability between institutions, as well as real-time access to the required data, in order to avoid latency and delays associated with accessing file remotely [2].

Currently, the adoption of equipment capable of retrieving digital medical images has been increasing, since the price has decreased significantly. Also contributing to this wide adoption, is the fact that these devices can cause a significant improvement in the healthcare provided, since it reduces physicians workflow delays.

2.2 INTEROPERABILITY

Nowadays almost every information system requires some kind of communication. This also applies to health care systems, which rely on communication between different machines, using distinct protocols, on several propagation means, as well as some other kinds of variations. In order to achieve the communication, it is necessary to have some kind of interoperability between the systems that will interact on the communication. As it stands, interoperability can be seen as a system's property, that represents the full understanding of the interfaces used to communicate with other systems without any restrictions [3].

In the radiology area, there are widely spread guidelines that aim for a uniformization to achieve interoperability. In order to resolve issues regarding information sharing, that can impact the quality of the care provided in medical imaging, the Integrating the Healthcare Enterprise (IHE) Radiology was created. Since then, IHE has developed and documented standard-based solutions to these problems, promoting the coordinated use of established standards such as DICOM and HL7, to address specific clinical needs in support of optimal patient care [4]. As a result, different profiles of guidelines have been tested, improved and documented, resulting in a reliable effort. With the implementation of interoperable systems, there is an ease of sharing information in a safe way, meaning that different types of equipment may communicate with each other, even if such devices are not in the same facility or institution. Consequently, these standards lead to an improvement in the health service provided, as well as a decrease in the vendor lock in anti-pattern.

2.3 PACS

PACS embraces a set of technologies, to create a system that allows medical images to be acquired, distributed and visualised. This system is aimed for radiology infrastructures, in order to capture medical images electronically, rather than in film-based media, but was later extended to other services in the health care infrastructure. Since it can facilitate the communication process between an image technician and a physician, the use of this system can speed up diagnoses time. Consequently, it improves the efficiency of the healthcare delivery process [5]. PACS can also solve some of the problems that are associated with film-based media [6], for instance:

- **Scalability** - a film-based study is only available in one place at a time, which can lead to a delay in the patient care since it is not immediately available to the physician.
- **Fault tolerance** - film-based studies are not compliance with disaster/catastrophe plans, considering they are usually locally stored. As a result, it is very difficult to have redundancy or backup data, of these files.
- **Security** - it is essential to keep patient data secured, considering hospital have large amounts of personal data. As it stands, a film-based study is not compliant with user access tracking, neither role-based access, making it impossible to know which users have visualised such study.
- **Data migration** - with the current pace of technology, there is a constant search for better systems, therefore, it is crucial to achieve cross-system data compatibility, which can be very hard to achieve with film-base media.

Usually a PACS system is integrated with a Hospital Information System (HIS) and RIS. Figure 2.1 illustrates the workflow and integration of the referred system, with a health care infrastructure.

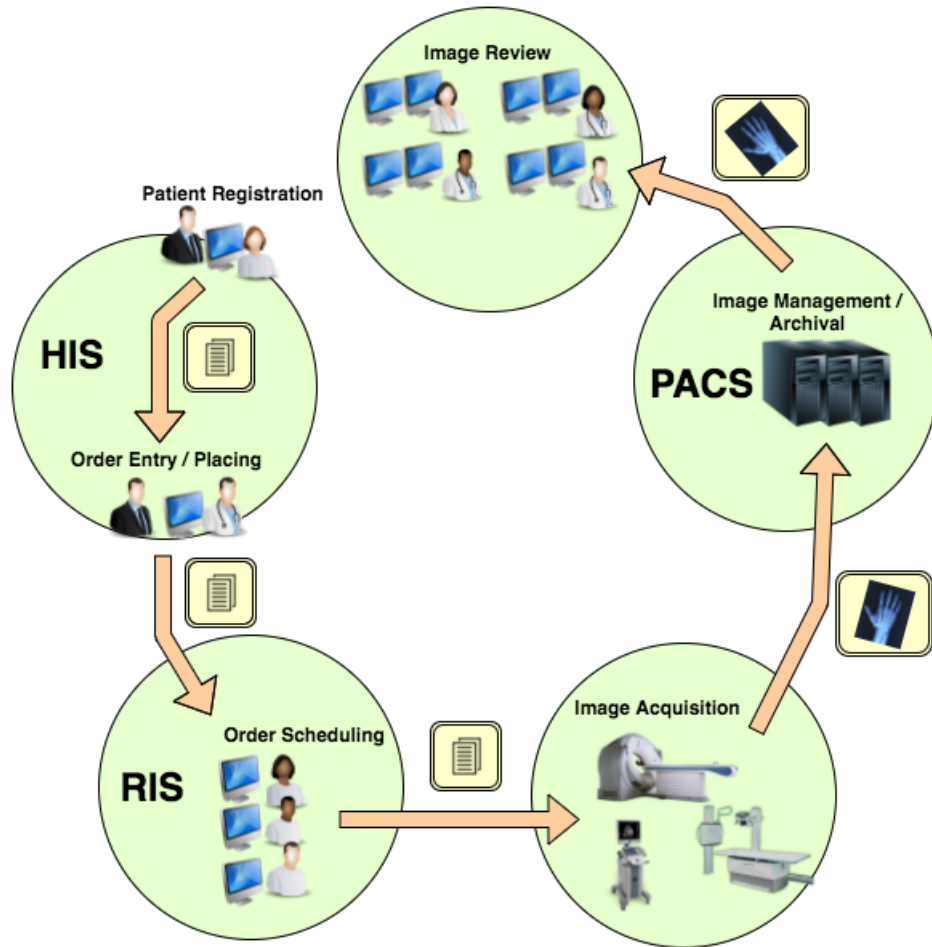


Figure 2.1: Integration of a HIS, RIS and PACS

As shown in Figure 2.1, a patient is registered within the HIS by an administrative employee, which then starts all the process regarding the patient's care. This process is then accessed by the patient's physician that evaluated the study and refers him to perform a certain exam if necessary. The patient process now changes scope from HIS to RIS, and it is accessed by an image technician which can deal with the managing operations, associated with an image based exam, scheduling, form filling, and so on. After this step, the exam is performed, which represents the first paradigm of a PACS system, **image acquisition**, consisting in the process of capturing an image and its codification in a digital format. The main focus of this paradigm is to obtain an a visual representation of the patient body, in order to help physicians deliver the best health care possible. The output of the performed exam will later be forward to the PACS system, allowing medical images and associated data to be **stored** and **distributed** to workstations. This feature can be extremely useful since it allows images to be accessed outside the acquisition department, resulting in a reduced diagnose time, as well as an easier exchange of case studies. A cost decreased, can also be a result of the stated feature since it eliminates the manual process associated to image printing and communication and can avoid loss of important medical data. The final step in the workflow consists in **image visualisation**. This

element can be seen as the connecting entity between a PACS and physician since it allows to retrieve, visualise, manipulate, save and evaluate studies about a patient condition.

The ecosystem created with the integration of a PACS with a RIS and HIS can be a complex and expensive, but the advantages that derive from doing so are not to be discarded [7].

2.4 RIS

As explained in subsection 2.3, a RIS usually is used in conjunction with a PACS system, allowing for a better modular solution and interoperability between external systems. Currently, it can be considered the core system of imaging departments, since its main purpose is to support the operational workflow, as well as a business model like billing information, within a radiology department.

It should provide some basic features [8], such as:

- **Patient management** - for every health care facility, it is crucial to have a Electronic health record (EHR). This consists of an official health record, shared among multiple facilities, that contain personal information about the patient and his clinical history. A RIS should allow managing these records by adding, editing or even merging different patient records when needed.
- **Order scheduling** - in order to keep all the workflow in proper functioning and maximise all of the facility resources, it is important to have a good planning. Therefore, it is crucial to have a good scheduling system, that avoids resources overlap. A RIS may or may not be included on a hospital campus hence, it should also allow inpatients and outpatients.
- **Image visualisation** - for a correct diagnosis, is essential to have access to reliable medical images. For this reason, a RIS should provide a way to visualise and manipulate these images, using PACS resources. The ease of access should also be taken into consideration since it can greatly improve the turn around time, between image acquisition and diagnoses time.
- **Reporting** - in order to register findings or points of interest in a certain study, is essential to have a reporting feature. This feature should follow normalised templates and also allow exporting these reports in a structured format like DICOM SR or simply in PDF format. A voice-to-text feature or voice recordings should also be provided with a RIS, as it can ease the reporting job for the physician.
- **Security** - as every other health care information system, a RIS handles great amounts of sensitive information, so it is essential to have robust security mechanisms. A good approach is to implement a Role-based access control (RBAC) system, where the access to a resource is regulated by the role a system user may have. An robust authentication mechanism should also be provided by the RIS, so that only a user may access the platform.
- **Integration** - the software solution should provide a way to be integrated with other systems, as well as allowing to change different components, such as the PACS system, in order to better suit the institution needs at the moment.
- **Billing and administrative management** - since it can be the only system for a current health care facility, it should have means to provide billing and also allow the configuration of different aspects regarding the business model adopted by the facility.

As it stands, the features available in a RIS can have a direct impact on the quality of the healthcare a physician can provide therefore, a correct implementation of the features mentioned above will result in an optimal and robust RIS. This is critical to an efficient workflow of radiology practises, which can be translated into a smoother experience for the patient, as well as for the physician, resulting into an improving regarding the quality of the service provided.

2.5 DICOM

At the beginning of the digital medical imaging era, there was no standard way to handle, store, print, and transmit information in health care provider systems. With the growing massification of the use of digital medical images, this lead to manufacturers develop proprietary solutions and equipment, resulting in a vendor lock-in pattern.

To counter this effect, DICOM was born with the main purpose of creating a universal standard in digital medical imaging [9]. As a result, different proprietary solutions or equipment could work together since this standard aims for interoperability and compatibility between health care information systems. DICOM is not just an image or file format [9], instead, it can be seen as a built-in set of tools that allow data transfer, storage and display protocol, providing all the necessary building blocks for the implementation of different functionalities in a health care information system.

Since the DICOM main purpose is to eliminate compatibility issues between different imaging devices [10], it is easily understood that its adoption and recommendation helped the dissemination of PACS, as it provides guidelines that allow store, transmit and handle of digital medical images.

Figure 2.2 depicts the integration of a PACS system with the standard DICOM [11].

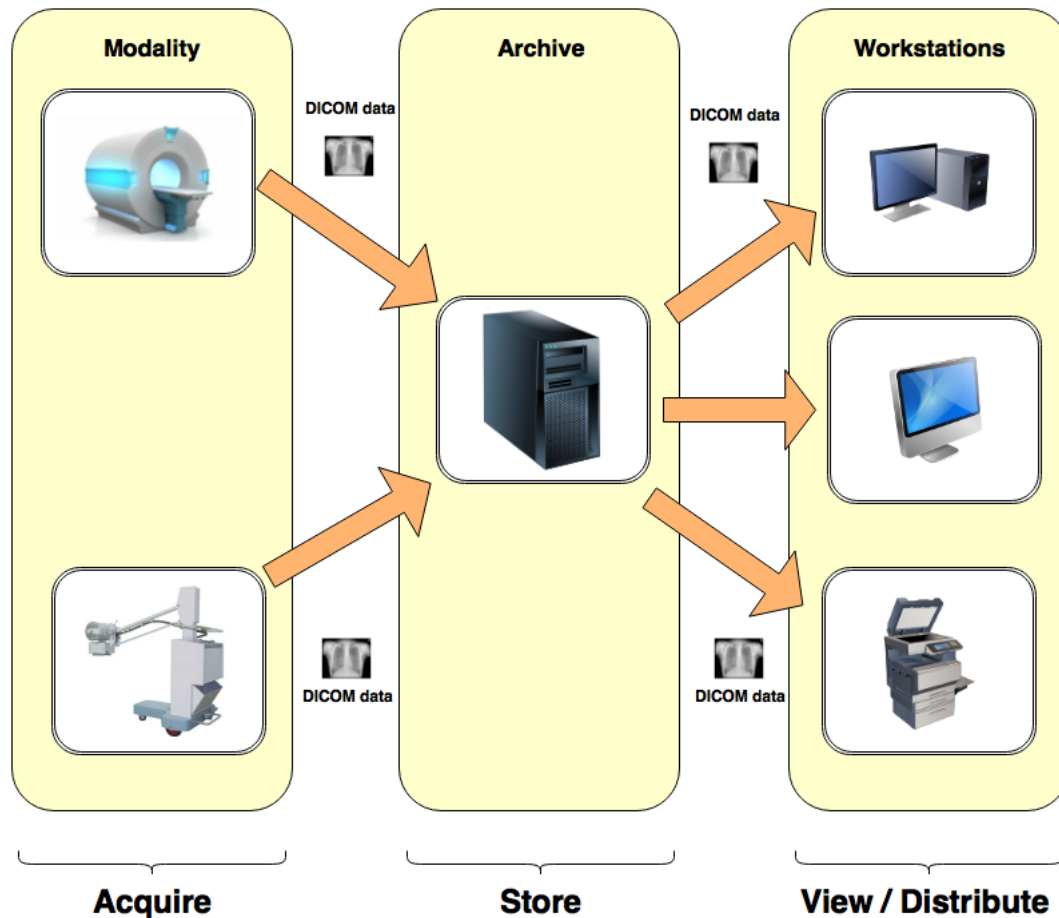


Figure 2.2: Major PACS Components Compliant with the DICOM standard

As it shown in the Figure 2.2, a PACS system comprises tree main components:

- **Modalities** - encompasses all digital image acquisition devices, that will output DICOM compliant data.
- **Image archives** - that represents the backbone on which the medical data will be stored.
- **Workstations** - represents the place where the imaging technician or patient's physician will visualise and manipulate the medical data.

Figure 2.2 illustrates that a medical image is acquired by the modality equipment, from a wide range of devices. These images are outputted as compatible with the format and organisation established by DICOM. Afterwards, the DICOM data is then transferred to an image archive, using DICOM network services, on which will later be stored. As a result, the medical image and its associated data can be accessed from every workstation that has privileges to do so, therefore facilitating the physician's workflow. By using this standard, each entity on a PACS system can be pictured as a block, that can be changed without affecting the entire information ecosystem.

Currently, it is almost impossible to picture interoperability between medical imaging information systems, without using the DICOM protocol.

2.5.1 DICOM OBJECTS AND ORGANISATION

To allow data from the real world to be represented in the DICOM standard, the DICOM Information Model (DIM) was created, representing how objects should be mapped, organised and how should their relationships be specified [12]. To achieve this, DICOM organised studies in a hierarchy similar to the one presented in Figure 2.3.

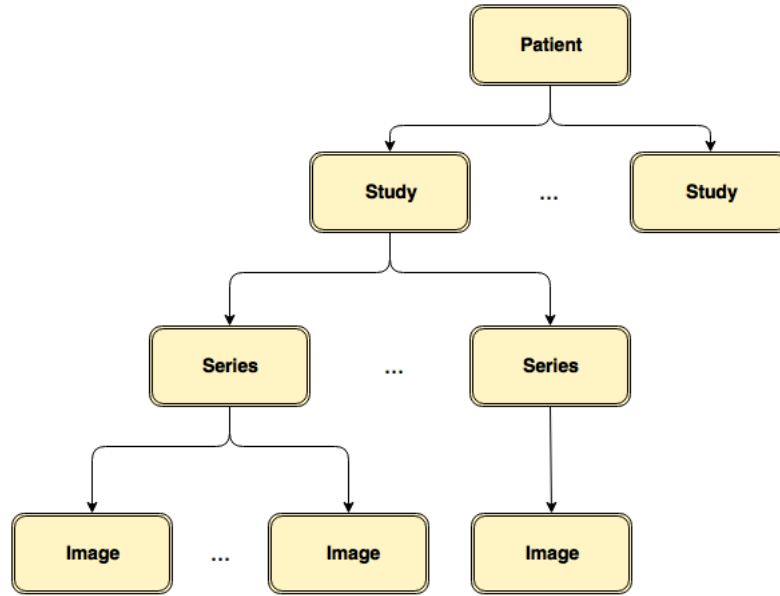


Figure 2.3: DICOM Information Model Hierarchy

The diagram presented on Figure 2.3 can be seen as a real world scenario, where a patient may participate in one or more studies, which may, in turn, implicate one or more series. This series can be seen as different modalities, that may use different protocols and output a varied number of medical data, which is also represented in this diagram, but in a lower level of the associated series.

In order to identify each level of the hierarchy, its used a Unique identifier (UID), composed of two parts:

- **Organisation root** - used to identify the organisation.
- **Suffix** - used to identify the instantiated object.

When used together, with the following format **organisationRoot.suffix**, it provides a way to unequivocally identify a certain instance of an DICOM object in an organisation.

All data related to patients, studies, medical images, reports and such, is represented as DICOM objects. These representations are defined in DICOM dictionaries, providing a universal format, accessible to other DICOM entities. In order to provide an object-oriented abstraction, a model called Information Object Definition (IOD) is used, which describes information regarding the real object in the DICOM format. To better represent such data, it is also used the IOD Entity-Relationship (IOD-ER) model, which allows specifying relationships between Information Entities (IE's).

Presented on Figure 2.4, is an example of a composite image IOD:

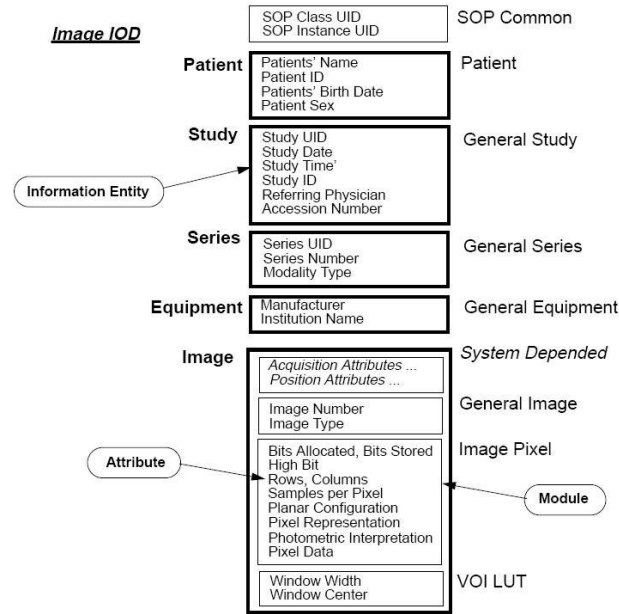


Figure 2.4: IOD Example of a DICOM Object[13]

As one can observe, the IOD presented on Figure 2.4 contains multiple information entities such as a patient, a study, a series, and others. For this reason, it is considered a composite DICOM object, otherwise, it would be considered a normalised DICOM object.

2.5.2 DICOM STRUCTURE

Although the DICOM standard supports several types of data, DICOM attributes are composed of three mandatory fields, plus one optional parameter [14]. This format is called Tag Length Value (TLV):

- **Tag** - it is composed of four bytes, where the first two bytes represent the Group ID and the following two bytes represent the Element ID. Each Tag is unique, meaning that there is never duplicated tags, eliminating the possibility of ambiguity.
- **Value representation** - usually composed of 2 bytes, but it can also be composed of 4 bytes, where the last two bytes are 0x0000. It is used to specify how the attributes are encoded, e.g. PN - Person Name, UI - Unique Identifier, among others. This field is optional in the DICOM structure.
- **Length** - this field uses a byte scale and it represents the length of the value field since it may vary for each Tag.
- **Value** - this field holds the actual data, that is the element, of the tag associated with it.

The following image represents the TLV format within a data set:

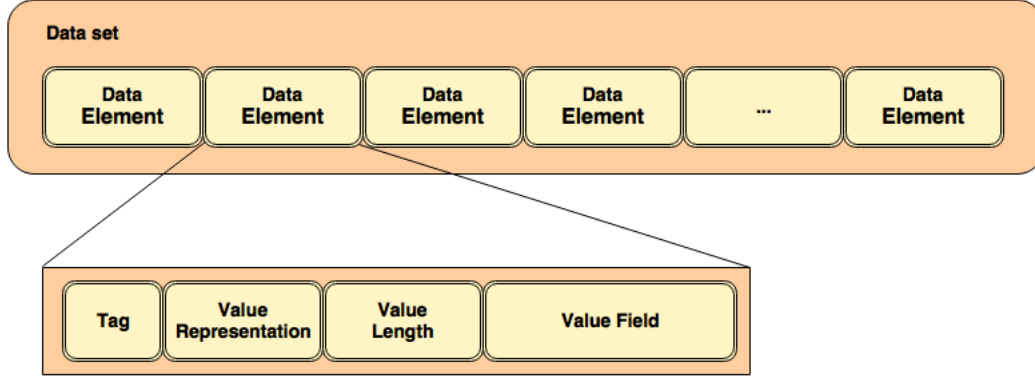


Figure 2.5: DICOM Data Elements with TLV Format

2.5.3 DICOM COMMUNICATION

Since DICOM aims for interoperability with medical equipment, providing a communication protocol that supports a set of services (Table 2.1), it relies on the Open Systems Interconnection (OSI) model, which is a conceptual model that standardises the communication functions of a computing system, with no regard to what is their underlying internal structure and technology. To achieve intercommunication, DICOM works over the TCP/IP protocol, allowing to assure a reliable communication between the two nodes. A node can be seen as an application or service, that exists in a certain network and it is identified by an IP address, a TCP port and a Application Entity (AE).

The services provided by DICOM Application Entities (AEs) follow a client-server architecture, where the client is called Service Class User (SCU) and the server is called Service Class Provider (SCP). Depending on the type of the device, a DICOM equipment can have different roles on this paradigm [15]. As an example, x-ray machine can be responsible for generating a medical image and store it on a PACS Archive. Correlating with the architecture previously described, this means that the X-ray machine can be considered a SCU since its the service requesting entity, and the PACS archive is the SCP as it provides a mean to store the image.

The first phase of communication between two DICOM devices, it is called association, and its purpose is to negotiate transfer parameters, such as encoding, services supported, the duration of the association, and so on. The next phase can only take place after a successful association, and it relies on an agreement by both the SCU and SCP on the negotiation of the parameters. On this phase the service goal is performed, resorting to service commands that are changed between SCU and SCP. The most relevant commands are presented on 2.1.

Service	Description	DICOM command
Storage	Request store to a DICOM device	C-STORE
Query	Encompasses search of data from a DICOM device	C-FIND
Retrieve	Encompasses fetch of data from a DICOM device	C-MOVE/C-GET
Verification	Check availability of a DICOM device	C-ECHO

Table 2.1: DICOM Services and Related DICOM Commands

On the following subsections, it will be described the main services used on the DICOM.

2.5.3.1 STORAGE SERVICE

The main objective of this service is to provide a way to transfer and store data, usually medical images, in remote places. In this case, the equipment that sends a medical image is considered to be the SCU and the service that provides the storage, a PACS archive for example, is considered to be the SCP, resulting in an implementation of the SCU/SCP architecture explained previously. The interaction between the two DICOM nodes can be seen on the Figure 2.6 and consists on: the SCU sends a C-STORE-RQ, containing all the information about the DICOM objects that are aimed to be stored, to the SCP. Upon receiving the SCP sends a C-STORE-RSP acknowledging the data reception.

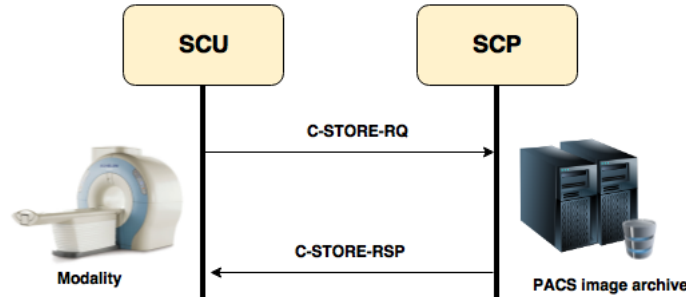


Figure 2.6: DICOM Storage Service

2.5.3.2 QUERY/RETRIEVE SERVICE

The main objective of the Query/Retrieve is to query a certain archive and eventually retrieve the queried content. This service can be broken down in two DICOM commands C-FIND and C-MOVE. The query phase allows the workstations in a PACS system to search for a certain data, such as a study or a patient, specifying some DICOM attributes as query parameters, which may be a patient name, modality, sex, age, among others. The retrieve phase allows the query results, i.e. the studies, to be transferred to the SCU.

The interaction between the two DICOM nodes can be seen on Figure 2.7 and Figure 2.8, consisting on: the SCU sends a C-FIND-RQ to the SCP, containing the query "Patient Name = David Fontes". After processing this message and its respective query, the SCP will send a C-FIND-RSP for each object found, plus one that symbolises the end of results. Once the query phase is finish, the SCU sends a C-MOVE-RQ to the SCP, containing a list of the object's UIDs that correspond to the query result. To each object requested, the SCP will respond with a C-STORE-RQ, on which the SCU will respond with a C-STORE-RSP.

The retrieval of the Query response objects is done via the C-MOVE command. A C-MOVE-RQ command with a list of the desired object's UIDs is sent by the SCU, to which the SCP responds with a C-STORE-RQ for every object requested. Like in the Storage Service Class, every C-STORE-RQ is replied with a C-STORE-RSP by the SCU. To end the retrieval phase and the interaction process, the SCP sends a C-MOVE-RSP, consequently terminating the transfer.

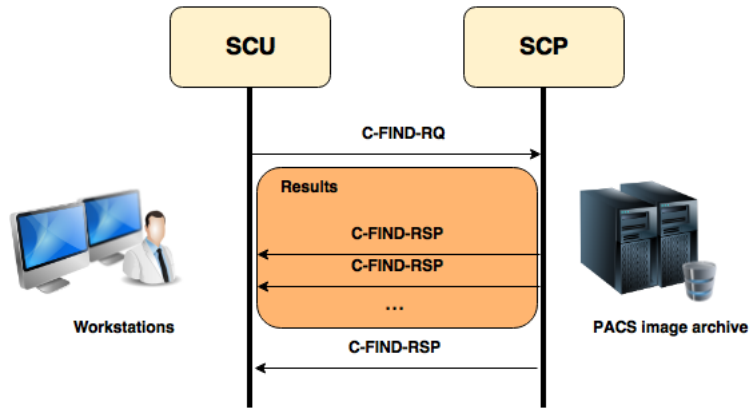


Figure 2.7: DICOM Query Service

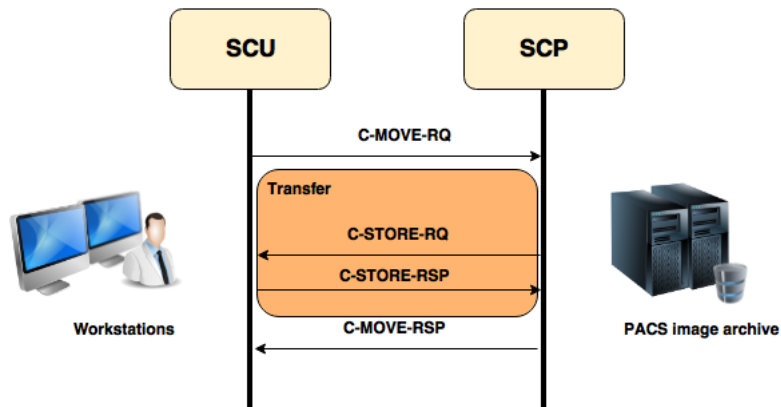


Figure 2.8: DICOM Retrieve Service

2.5.3.3 WADO

Web Access to DICOM persistent Objects (WADO) can be seen as a web-based extension of DICOM. It relies on the HTTP protocol, using the methods GET and POST from the HTTP standard, providing a simple and easy way to integrate web applications with simple features of the DICOM standard. WADO aims to solve some problems related to security restrictions, inherited from the TCP that DICOM relies on. For instance, the port range used in the TCP can be blocked on the enterprise firewall, thus making it difficult to access DICOM data from outside or even inside the health care provider facility. The ports that HTTP uses are most likely not to be blocked, since nowadays it is widely used on the web. Although some of this problems can be overcome, WADO does not provide all the standard DICOM services. The interaction between the two DICOM nodes can be seen in the Figure 2.9

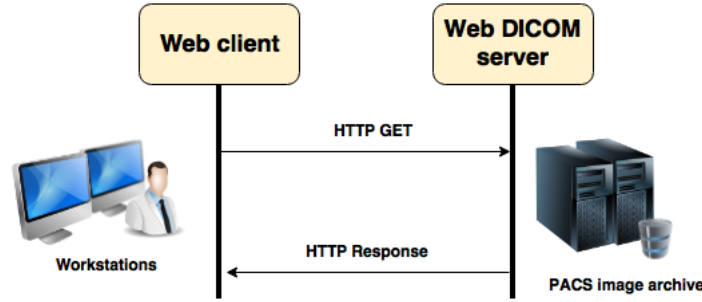


Figure 2.9: DICOM WADO Service

Currently, the latest DICOM versions provide a set of RESTful services [16], that aim to shift traditional communications to web communications making use of the previously presented WADO. These services consists mainly of:

- **WADO-RS** - added to the DICOM standard in 2011, this service provides a way for clients to access multiple SOP instances in one HTTP request [17]. Currently, it uses the REST architecture style, hence the RS in the name, meaning RESTfull.
- **STOW-RS** - it relies on an RESTfull service to store instances with the C-STORE command into the underlying DICOM server [18]. It allows for single or multiple DICOM instances to be transferred, using XML/DICOM and JSON as media types.
- **QIDO-RS** - provides a service for performing queries based on ID's regarding the DICOM Objects [19]. Such queries are then translated into DICOM C-find requests, relying on DICOM+XML and JSON as media types.

2.5.4 DICOM-SR

DICOM SR is a document architecture designed for encoding and exchanging clinical information and findings, in a structured way regarding the context of Radiology imaging. Leveraging an existing DICOM environment, it reduces the ambiguity associated with natural language report format, enhancing the value of clinical information. This standard can be applied in various clinical context, for instance, store several types of classification, as well as specifying tissue injuries on a Mammography screening. As a result, it simplifies workflow issues and allows for more accurate information, regarding the clinical diagnosis.

All documents compliant with this standard, rely on the data elements and network services from the DICOM standard, using the same information model presented in Figure 2.3, as well as the hierarchical object tree (Figure 2.11) [20] containing all associated items. In addition to this, templates can also define a set of constraints for specific tree nodes, provided an input validation regarding the produced clinical report.

Figure 2.10 provides a brief overview of an DICOM SR document structure, which consists of a file header and body [21]. The header section contains information regarding various aspects, such as patient, study and so on. The **Root Container** holds the attributes for the root content, for instance, the code for the report title. Lastly, the **Content Tree** holds all data regarding the clinical report, in a structured manner.

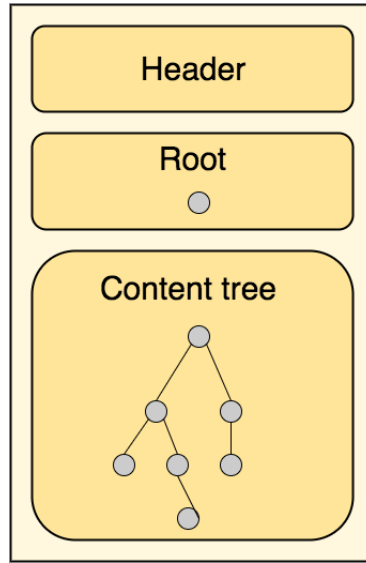


Figure 2.10: DICOM SR Document Structure

Illustrated on Figure 2.11 is an example of the tree structure created to hold information, regarding a finding in a mammography. Depicted in this figure are parent-child relationships, presented to the reader as arrows. From there relationships range from various values, such as **Contains**, **Has properties**, **Inferred from**, among others. Also presented in Figure 2.11 are circular entities that represent content items. From there entities can hold information in various formats, for instance, text, numeric, code, image, and so on. Content items can also work as containers, that hold more nodes of the hierarchically structured tree.

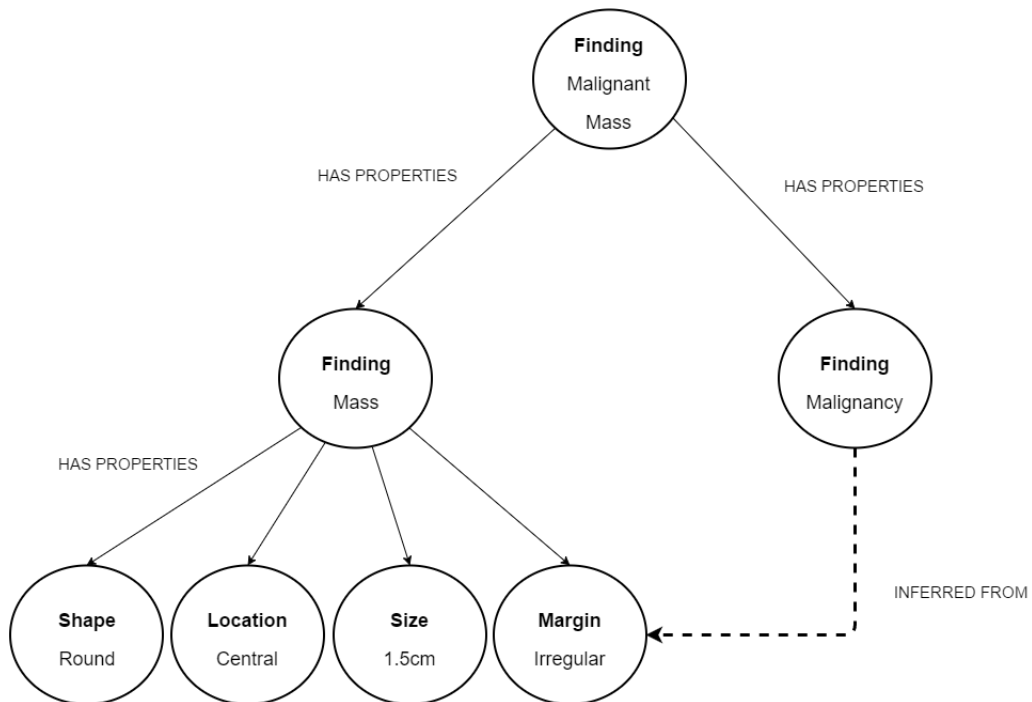


Figure 2.11: DICOM SR Content Tree

RELATED WORK

This chapter presents the current RIS solutions. The stated software systems are divided into two categories, open source and proprietary solutions, a brief overview of each one will be presented further on. In addition to key features provided by each solution, the performed analysis had a great focus on criteria composed of a couple of principles, such as supporting record management, event scheduling management, infrastructures management and the ability to export clinical reports compliant with the DICOM SR standard. All open source solutions have been installed in a new virtual machine, in order to mimic a clean deploy of the system in the field of work. As a result, all information presented in this chapter, regarding the open source solutions, has been gathered from a deep analysis performed in that new installations.

3.1 OPEN SOURCE SOLUTIONS

It is a development model that promotes universal access via a free license to a product's design or blueprint, and universal redistribution of that design, including subsequent improvements to it by anyone. It usually refers to a computer program, in which the source is available to the general public for use and/or modification from its original design. This term gained hold with the rise of the Internet and the attendant need for massive retooling of the computing source code. Currently, this model also enables a self-enhancing diversity of production models, communication and interactive communities [22].

3.1.1 GNU HEALTH

GNU Health [23] is a free open source project for health institutions. This project aims for a modular pattern, in order to cover different use case scenarios, ranging from small medical clinics to large national public health infrastructures, providing Electronic Medical Record (EMR) and HIS features. Since it is an open source project, it relies on a wide community that provides a vast knowledge covering diverse areas. Currently, GNU Health relies on a modular approach, where each module implements a certain feature, resulting in an improved system scalability, since it does not have to

load unnecessary features. A consequence of the use of this pattern is dependencies between different system modules. To solve this, GNU Health inherits classes and models from the core module, making them available to other modules. As a result, all dependencies and inter-module communication are dealt internally.

To help physicians, GNU Health also provides demographic data from the population inserted on the health institution action radius. To achieve this, GNU Health provides a census feature, included in the core module, that allows gathering data like domestic and economic conditions, disease control, family-related information, and many others. By doing so, GNU Health promotes the integration of the healthcare provider with the community, in order to leverage physicians on critical situations like in epidemic control, domestic disputes and others.

Responsibility segregation is also a concern included in the GNU Health, meaning that for instance, demographic data can be gathered by social workers, patient data can be managed by health professionals, healthcare facility management can be assigned to administrative personnel and accountants, and so on.

GNU Health provides an inventory and billing systems, meaning that the competent system role can perform inventory lists, resulting in a better resource equitability between departments. By having a billing system, this platform allows for a full implementation of the business model. In conjunction, this two features leverage the physician to focus only on providing the best healthcare possible, resulting in a smoother experience for physicians and patients.

Interoperability between different systems is a crucial factor nowadays. GNU Health includes a Fast Healthcare Interoperability Resources (FHIR) REST server, a standard for exchanging electronic health information developed by HL7. This standard defines an interface, in this case, a Representational State Transfer (REST) Application Programming Interface (API), for interoperability between medical software. Currently, this feature only allows reading data from the GNU Health's database, blocking untrusted users from overriding crucial information held in the system.

GNU Health does not support medical imaging out of the box. In order to allow physicians to provide a correct diagnosis using medical images, it is necessary to install an additional module, labelled "health_imaging". The installation process is rather easy since it is fully explained in the documentation section of the GNU Health website [24]. After the installation of the aforementioned module, GNU Health allows for medical data to be saved in a local database or in a remote PACS archive, compliant with the DICOM standard and ensuring interoperability with existing devices.

Regarding system records, GNU Health has full support for the management of patient and clinical staff records, in other words, it provides a way to add, modify and delete such records. Currently, patient records hold various kinds of data, such as clinical history (clinical/surgical procedures, patient prescriptions, genetic conditions, etc), biographical data, demographic data, small reports (following an already predefined format in LibreOffice document), and others.

The scheduling feature included in the GNU Health is rudimentary, since it does not take into consideration available resources when an appointment is scheduled. In other words, it is possible to schedule two appointments in the same room at the same time, with different patients or even schedule two appointments to the same physicians at the same time, in different rooms. This can lead to bad resource management, which in turn may result in a bad coordination between equipment, physicians and appointments, slowing down the workflow in a radiology laboratory. GNU Health also allows to add or delete modalities on the system's database, but does not support the association of an equipment to a modality, neither the specification of procedure's estimated time.

Focusing on the report feature, GNU Health allows writing a clinical report on a patient record

when an appointment occurs. These reports can be dynamically exported with ODT extension, PDF or printed directly. Currently, all clinical reports produced with GNU Health, are not compliant with the DICOM SR standard, make then difficult to export to other enterprise-grade systems, or even provide a reliable and secure storage system.

Figure 3.1 illustrates an example of the user interface provided by GNU Health, more specifically when a user intends to obtain personal information regarding a patient record. On the left side of the interface, there is a search bar that allows a user to search for a specific module or functionality. Under this search bar, there is a list, containing all the installed modules in the system. When an entry on this list is selected, a small menu expands revealing available operations to the user. The menu displayed in Figure 3.1, is the result of selecting the option **Patients** from the side menu, followed by selecting a specific patient record. As a result, a detailed view containing the patient's personal information is displayed to the user. Depending on the system user currently consulting the record, it is also possible to modify existing information, as well as introduce new data on such record.

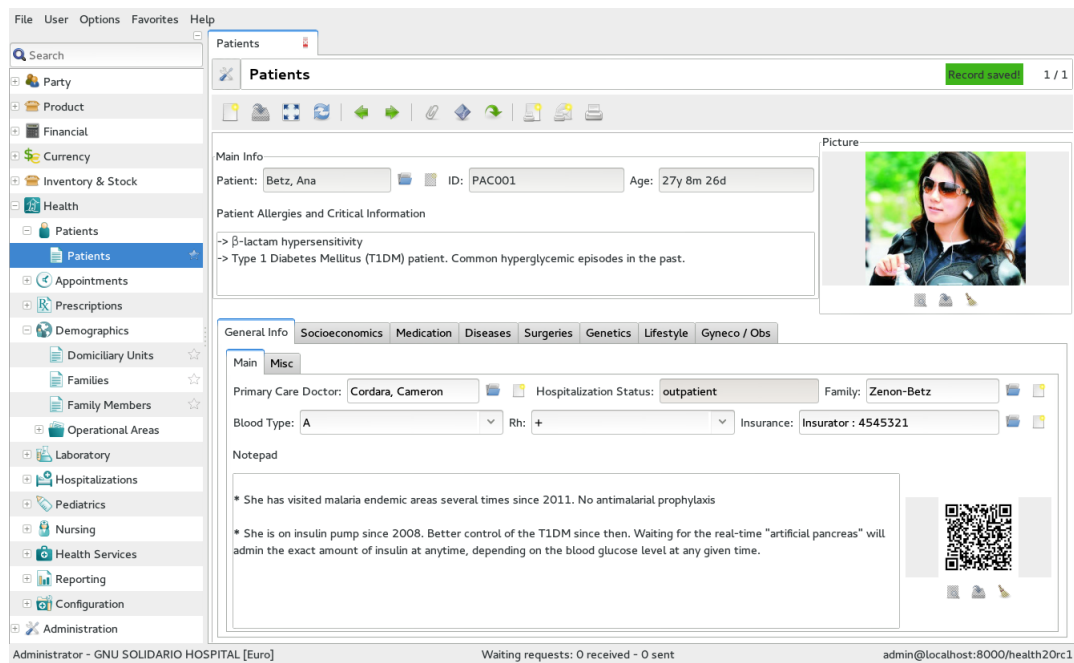


Figure 3.1: Gnu Health Interface

3.1.2 OPENMRS

Open Medical Record System (OpenMRS) [25] is an open source platform that aims to implement a medical record system, having the developing countries as the main target. This project was created by a collaborative effort of Regenstrief Institute and Partners In Health, with the main objective of improving the healthcare service of underprivileged people around the world. Currently, the deployed system is required to have a reliable operation even when there is a constraint of resources, such as processing power, energy delivery, network communication, or any other kind.

As many other solutions, OpenMRS follows a modular design pattern, where modules can be easily added or removed from the system. This is achieved by warping a complicated data model into a API, allowing web-based applications to access the referred model. As a result, OpenMRS can

hide the complexity inherent to the data model and ensures that modules and web-applications are compliant with the business model implemented. Currently, to add a new module to the system, the administrator must navigate through the repository and select the pretended module. These modules may have full access to the system core, and consequently, can modify the local system behaviour, meaning that each facility may have its own solution, instead of following one solution that attempts to fit all use case scenarios.

By default, OpenMRS does not have support for radiology operations. To achieve this, it is necessary to install the Radiology Module, enabling OpenMRS to store and visualise medical images and associated meta-data in a PACS system compliant with the DICOM standard. This module also enables a user to associate a certain patient record with a medical image and its respective report. As a result, a physician can access the full medical history regarding a patient record. Along with the previously stated features, Radiology Module also enables a user to perform a inventory list of the radiology department.

With the full system deployed, OpenMRS follows a client-server architecture, meaning that a server may be running in a remote and secure location, while workstations access it, if a reliable connection is available. Since the client-side logic is supported by a web-application, this system can be considered multi-platform, where there is no vendor lock-in regarding the OS.

One of the strengths of the OpenMRS is the support found in the documentation's website, this being the main point of connection with a full supportive community. An administrator has access to a step-by-step guide on how to install the system, add modules and update the system. To help new users acquiring information about the scalability of the system and what hardware can be used, the documentation also provide reports of real use case scenarios regarding full deployed systems. Along with this, an extensive approach on how to setup the system architecture, how the local network should be configured and how to deal with power backup, is also presented in the documentation's website.

Nowadays, scalability may be an issue for this kind of systems. To overcome this problem, OpenMRS provides a way to monitor system performance, using frameworks like Javamelody [26] and Nagios [27]. By doing so, the system administrator can direct efforts, such as upgrades and bottleneck solutions, to where it is really needed.

Regarding system security, OpenMRS relies on a robust user authentication, so that only an authorised user can access the system, reducing the risk of unwanted intrusions. A user should not have access to all data in a system's database, so OpenMRS strengthens the security aspect with RBAC, where a user can only access data if he has clearance to do so.

Developing countries may have old refurbished medical equipment, so it is crucial to have means to integrate them on the deployed OpenMRS system. For this reason, a HL7 engine for data import is available with the solution, as well as a data export feature, that exports data into a spreadsheet format.

OpenMRS is multi-language and can also provide full support for UTF-8 [28]. Currently, different clinical facilities may use different standards or specifications. To overcome this problem, OpenMRS provides medical dictionary management, where a system administrator can manage a full set of standard, such as ICD-10 [29], SNOMED CT [30] among others.

Regarding the record management, OpenMRS provides a full management of patient and clinical staff records. This means that it supports the creation of new records, as well as the modification of personal data in existing records. Currently, it does not allow for a patient record to be erased from the system. Instead, it provides a way to merge two records if presumably they reference the same person. Each patient record can hold biological and vital data, helping to keep track of patient's past

events and actions.

OpenMRS allows the creation of new appointment types, where it can be specified the modality associated with it, a brief description of the procedure and the estimated time duration. A user can also manage the appointment schedule of a certain physician, where a set of calendar views, such as weekly and monthly, are provided to better visualise the worklists. Currently, the scheduling mechanism provides a robust validation of the input fields, preventing overlap appointments.

Regarding clinical reports, this solution allows for paper-based forms to be transposed into a digital format. To do so, an IT technician will have to rewrite those forms in the HTML format, so the system can interpret then. The user interface provided with the report writing operation can be confusing, consequently slowing down the physician’s workflow. Currently, clinical reports generated in the OpenMRS are not compliance with the DICOM SR standard.

After a user performs the login operation with his credentials, it is presented with the menu screen and depending on the permissions he may have, more option can appear. This menu was designed for a fast and intuitive workflow, where a user can quickly select the pretended option. An example of the menu that OpenMRS provides to the user, is depicted in Figure 3.2.

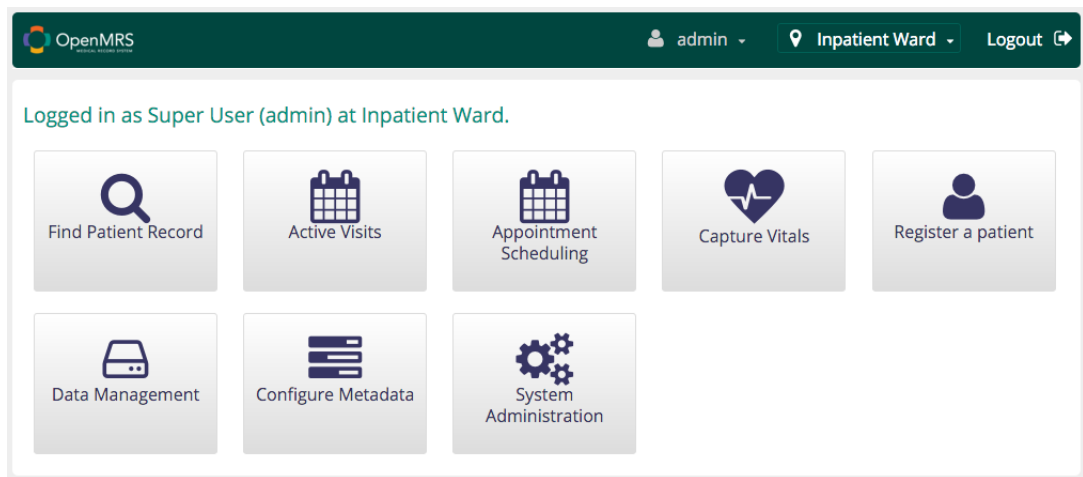


Figure 3.2: OpenMRS Menu

Figure 3.3 illustrates an example of the interface provided to a user, after the selection of the operation **Find a patient record**. From there, a user can easily search for a specific patient record or scroll through the patient list and select the pretended record.

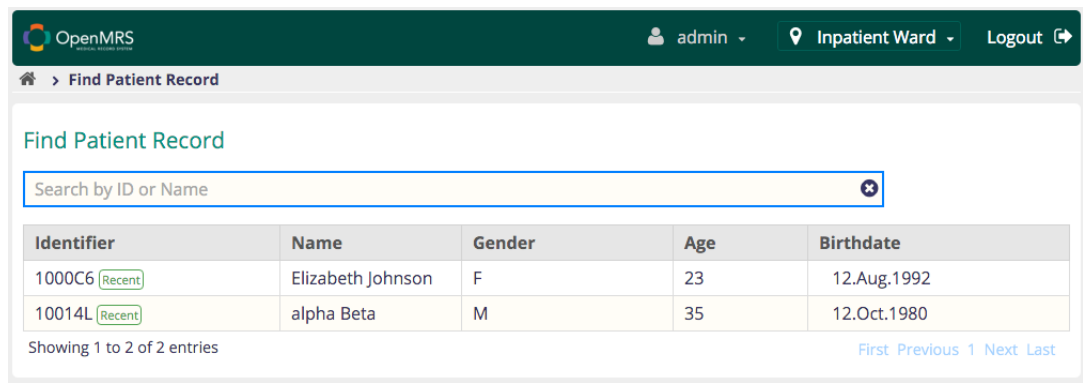


Figure 3.3: OpenMRS Patient Search

After selecting a patient record from the list presented in Figure 3.3, an interface similar to the one depicted in Figure 3.4 is displayed to the current system user. There, all the patient's personal data, such as vital reads from previous appointments, recent appointments, recent diagnosis, is provided to the user. A quick action menu is also provided in the detailed view, allowing a user to easily select a new action, related to this patient record, reducing the navigation time between menus. This quick action menu consists in the **General Actions** blue area, presented in the right side of Figure 3.4.

The screenshot shows the OpenMRS interface for a patient named Elizabeth Johnson. The header includes the OpenMRS logo, user 'admin', location 'Inpatient Ward', and a 'Logout' button. The patient's name 'Elizabeth Johnson' is displayed with 'Given' and 'Family Name' labels. Below the name, it says 'Female 23 year(s) (12.Aug.1992)' with 'Edit' and 'Show Contact Info' links. The 'Patient ID' is '1000C6'. The main content area is divided into four sections: 'DIAGNOSES', 'RECENT VISITS', 'ALLERGIES', and 'VITALS'. The 'DIAGNOSES' section lists: Rash, "Ankle instability Left", "Cavus foot", MALARIA, Superficial injury of hip or thigh, Acute Pharyngitis, Schizophrenia, Severe malaria, Physical trauma, and Umbilical hemorrhage after birth. The 'RECENT VISITS' section shows a table with dates and visit types: Today (Outpatient), Today (Inpatient), Today (Outpatient), Today (Outpatient), and 28.May.2015 - Today (Outpatient). The 'ALLERGIES' section shows 'Unknown'. The 'VITALS' section shows 'Last Vitals: Today 01:37 PM' with fields for Height (cm), Weight (kg), BMI, Temperature (°C), Pulse (/min), and Respiratory (/min). On the right side, there is a 'General Actions' menu with options: Start Visit, Add Past Visit, Merge Visits, Chart Search, the one, Schedule Appointment, and Request Appointment.

Figure 3.4: OpenMRS Patient Personal Information

3.1.3 OPENEMR

Open Electronic Medical Records (OpenEMR) [31] is a Office of the National Coordinator (ONC) certified system, for electronic health records and medical practice management. This project is supported and maintained by a strong community of volunteers and professionals.

Considering that this system may be used anywhere in the world, it provides full support for UTF-8[28] encoding and a full set of pre-installed languages, resulting in an enhanced usability experience. Currently, many healthcare facilities have physicians with various kinds of nationalities. As a result, OpenEMR allows the system's language to be configured on the client side, meaning that the system does not have to be exclusively confined to one language. The large number of supported languages, is due to the wide community involved in the translating process that uses a collaborative Google document, allowing any language to be integrated into the system.

By providing a web-application for the client side, OpenEMR can be considered a cross-platform system, since it can be accessed with computers with various operating systems, such as Microsoft Windows, Linux, Mac OS, and others.

Regarding the security aspect, OpenEMR uses an authentication system, where a user validates his account credentials on a log-in screen. To secure sensitive data from not trusted system users, OpenEMR relies on a Access Controls Listing (ACL) system. Such system is harder to manage when compared with a RBAC system, since a change in a certain user profile has to be updated in all permissions files throughout the system. Confidential data regarding a patient can also be encrypted and decrypted with Triple Data Encryption Standard (DES) in conjunction with a pass phrase, providing a reliable and secure way to store sensitive data within the system's database.

Available on the project's website is an extensive documentation, containing a large variety of guidelines, allowing a administrator to correctly setup an instance of this system.

Since OpenEMR's main objective is to provide a robust and free of charge healthcare information system, many of the adopters of this project will probably be a small clinic that gathers a set of services under one enterprise roof. This solution has also implemented a billing system, allowing to scroll and searching past bills on the system and generate new receipts and bills.

Another feature available on OpenEMR is Clinical Decision Rules (CDR), which consists on the incorporation of sets of rules related to clinical information about the patient. This feature is also known worldwide as Clinical Decision Support (CDS) and aims to provide physicians and other clinical staff with the necessary knowledge to enhance the healthcare provided. It correlates health data obtained from the patient with health knowledge, in order to help physicians deliver the best healthcare possible.

With the massification of Internet access, a large number of households have now an Internet connection of some kind. Leveraging this fact, OpenEMR provides a web portal where a patient can check his/her personal information and notifications, browse old clinical reports, prescriptions, appointments schedules, and so on. Currently, an institution employee is required to manually provide access to a certain patient and handout the user credentials to the patient.

OpenEMR also provides prescriptions management, allowing to search on-line for a drug name or information, scroll through patient prescriptions and medications, print, email or create new prescriptions. Prescriptions may have different formats and headers, so this solution also allows to customise the prescription layout and insert state license number on the prescription header. A small pharmacy can be found in almost every healthcare institution, so a dispensary management is also provided, allowing full support to in-house pharmacies.

Regarding the record management, OpenEMR provides full management of patient and clinical staff records, including the addition of new records, as well as the ability to erase or modify existing ones, from the system database. Each patient record holds appointments history and its clinical report, biological and vital data, patient notes, dated reminders, lab results, immunisations and many others parameters, in order to keep track of patients medical history.

As it stands, modality management is provided by OpenEMR, meaning that an administrator can add or erase modalities to the system. Currently, this feature does not allow the specification of the average duration time, neither the association of resources used with the referred modality.

Leaning on the scheduling feature, the system allows a user to create appointments and to modify existing ones, including the specification of a duration on a certain appointment. Currently, this system notifies the employee when an event overlap occurs, but does not avoid it, meaning that a certain physician can have two appointments at the same time, with different patients. By providing a set of calendar views, such as weekly and monthly, an employee can choose the best view to suit his needs. The scheduling feature can be configured to manage multiple facilities or departments, as well as to notify a patient, by email or Short Message System (SMS), on an upcoming appointment. It can also

restrict the number of a certain type of appointments per day, if so is necessary.

Focusing on the report feature, this system provides a reporting feature that is included in the appointment notes. Such reports can later be dynamically exported with DOCX extension or PDF. It supports the integration of new report templates, but it is necessary for a developer to make alterations directly on the project source code. As it stands, OpenEMR does not implement either follows the DICOM SR standard.

Figure 3.5 illustrates the menu presented to the user, after a successful log-in. On the left side with an orange hue, a full set of operations is available, where the user can select an available operation. On the right side is the result of selecting a menu operation, in this case, is a daily view of the scheduled appointments. The interface follows the same design pattern as the previously presented solutions, where a menu containing all operations is listed on the left side of the screen and the main information is presented to the user on the right side. This interface can be overwhelming to the user as it tries to pack a lot of information at once.

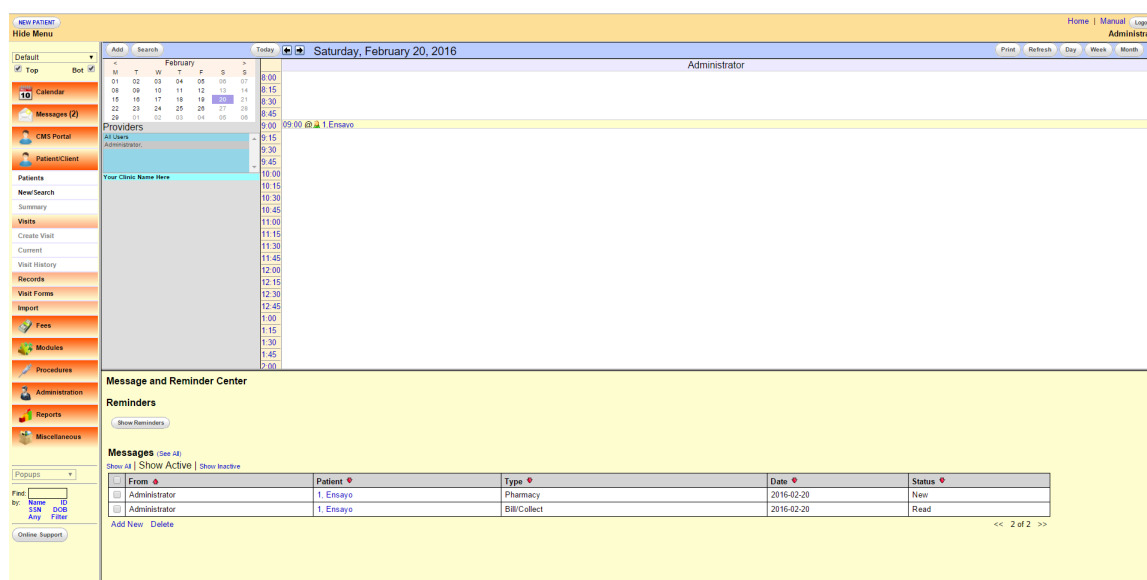


Figure 3.5: OpenEMR Menu

Depicted in Figure 3.6 is the output given by the system, after selecting the operation **Patients**, which consists of a list containing all the registered patients. From there, a user can scroll through the patient list or perform a search by record fields, in order to find a record.

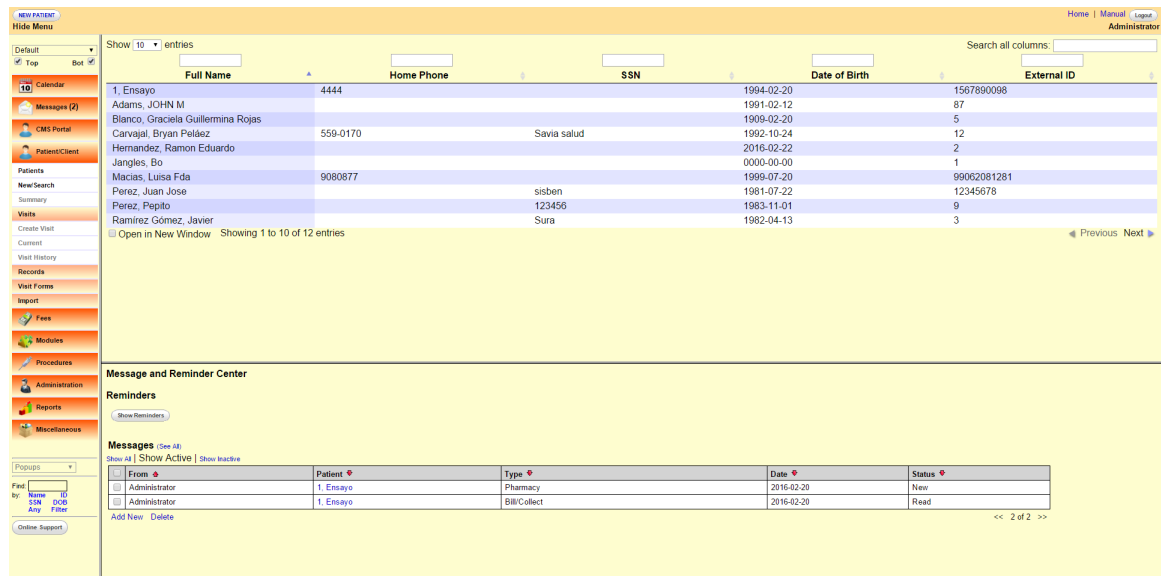


Figure 3.6: OpenEMR Patient Search

Figure 3.7 illustrates the information displayed to the user, after the selection of a certain patient record, making every information contained in the patient record, visible for the user.

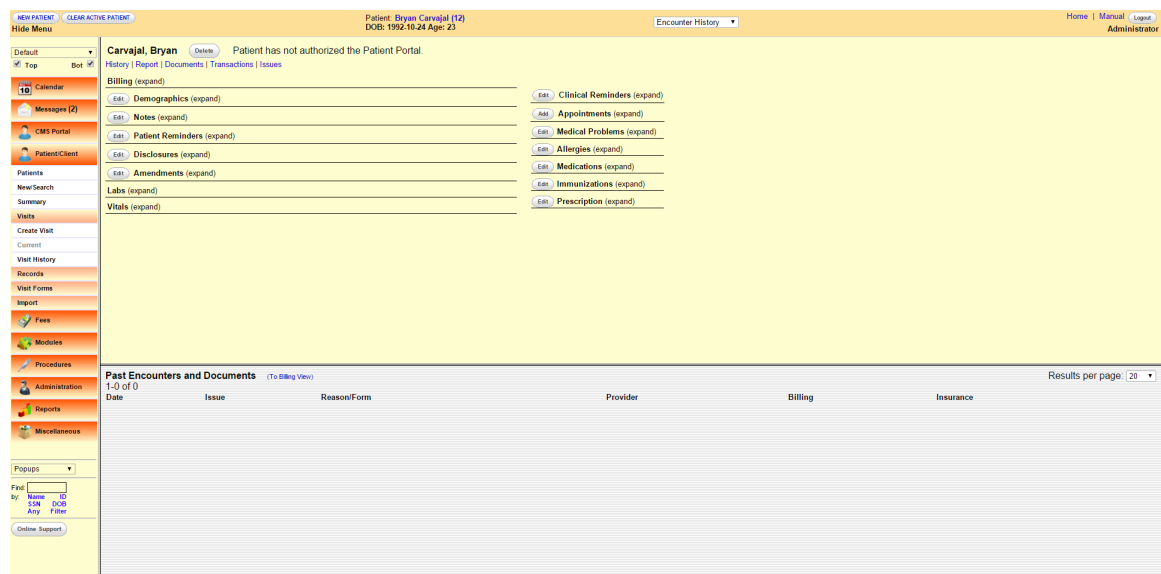


Figure 3.7: OpenEMR Patient Personal Information

3.1.4 CLEARCANVAS

Of all the evaluated open source solutions, ClearCanvas [32] is the only one strictly developed with the main objective of implementing a RIS workflow.

The source code can be found on the project's Github page [33], along with some instructions on how to deploy the system. The documentation available on the mentioned page is poorly achieved, as it only provides a high-level guideline about certain topics, thus not guiding the user to a full

understanding of what lies underneath the project source code. The user's guide section can be a great example of the poorly designed documentation, since it may be the first section a new user reaches for and it is completely empty since 2013.

This project is not cross-platform, as it relies on frameworks and tools that are only available for Microsoft operating systems. Since the tools used are outdated, even on the Microsoft OS's, there is some kind of restrictions on which version to use since these tools cannot be installed on more recent Microsoft operative systems. Moreover, this project may not be suitable to use in developing countries, since it is necessary to purchase licenses for the operating systems and for some of the Microsoft tools required to build the project for deployment, thus limiting the budget for the equipment required to construct a RIS infrastructure.

ClearCanvas allows for medical images to be stored in a PACS system, taking advantage of all the features this system offers. To visualise and manipulate medical data, that may be stored in a PACS, ClearCanvas requires a proprietary DICOM viewer, making it very hard to couple this solution with other software, due to its proprietary lock-in.

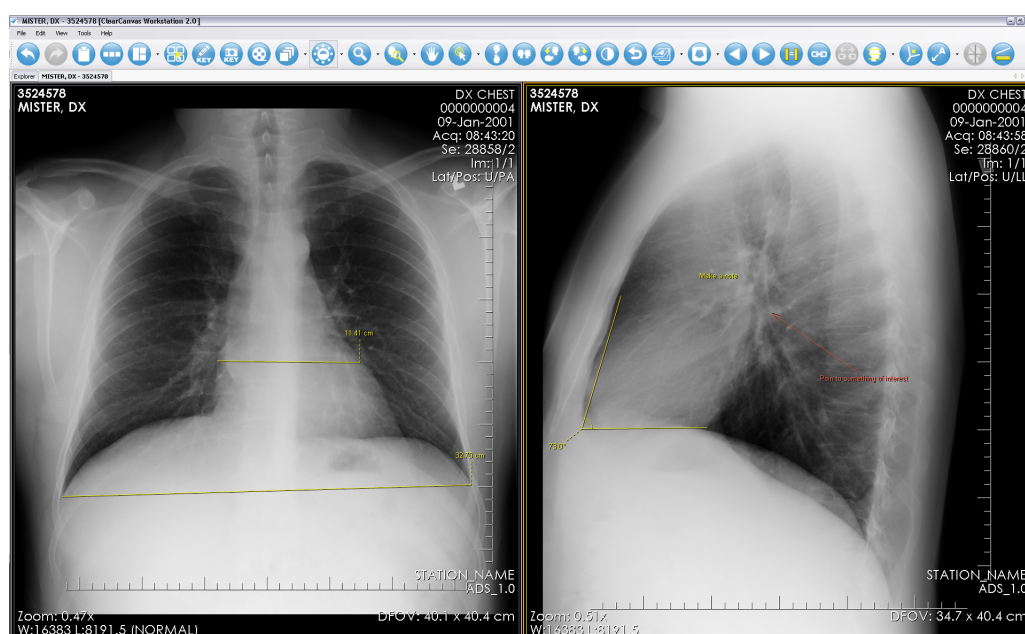


Figure 3.8: Study Visualisation on ClearCanvas Viewer

ClearCanvas allows the integration with plug-ins, in order to implement features, that may not exist in the current version. Such plug-in modules are developed using the .NET framework and then are placed inside a folder in the projects directory called `/plugins`. Afterwards, the system finds all .NET assemblies and loads them dynamically, so they can perform the designated tasks.

To eliminate or reduce unwanted intrusions, ClearCanvas relies on an authentication system, where a user is prompted to introduce his credentials in order to gain access to the system.

Leaning on the record management, ClearCanvas provides a way to create, edit and delete patient records that contain the patient's personal information. The same applies to the clinical staff records. It also supports records search, either by patient or by staff, providing multiple search parameters and the ability to search for specific studies.

Currently, it is possible to create new types of procedures or modify existing ones in the system's database, specifying the average duration time and modality. Along with this, modalities management

is also provided by the system, but a modality record does not have in consideration the resources that are used to perform the task associated with it.

Regarding the scheduling feature, ClearCanvas enables a user to create, edit and delete appointments on the system's database. Usually, not all patients have the same urgency on the scheduled appointment. For that reason, ClearCanvas provides a way to specify the urgency of an appointment, improving the resources usage on the healthcare institution. Currently, when a user is prompt to create a new appointment, the system allows him to add notes or even entire documents, as important information regarding the scheduled event. As it stands, ClearCanvas does not provide a way to customise the view of the appointments list, which can result in a mitigation of the institution workflow.

Currently, ClearCanvas only supports clinical reports as a simple text entry, meaning that all reports produced in this system, are not compliant with the DICOM SR standard, neither follow a structured organisation.

Figure 3.9 illustrates the search interface presented to the user. ClearCanvas has an advanced search engine, that allows a user to perform system queries on various parameters. The interface design is different from the one used on other solutions, aiming for a simpler layout, increasing the usability of the system.

Depicted in Figure 3.9, on the left lower side of the interface, a **Servers** menu is available, allowing for a quick and easy access to the DICOM servers that are available on the network or locally.

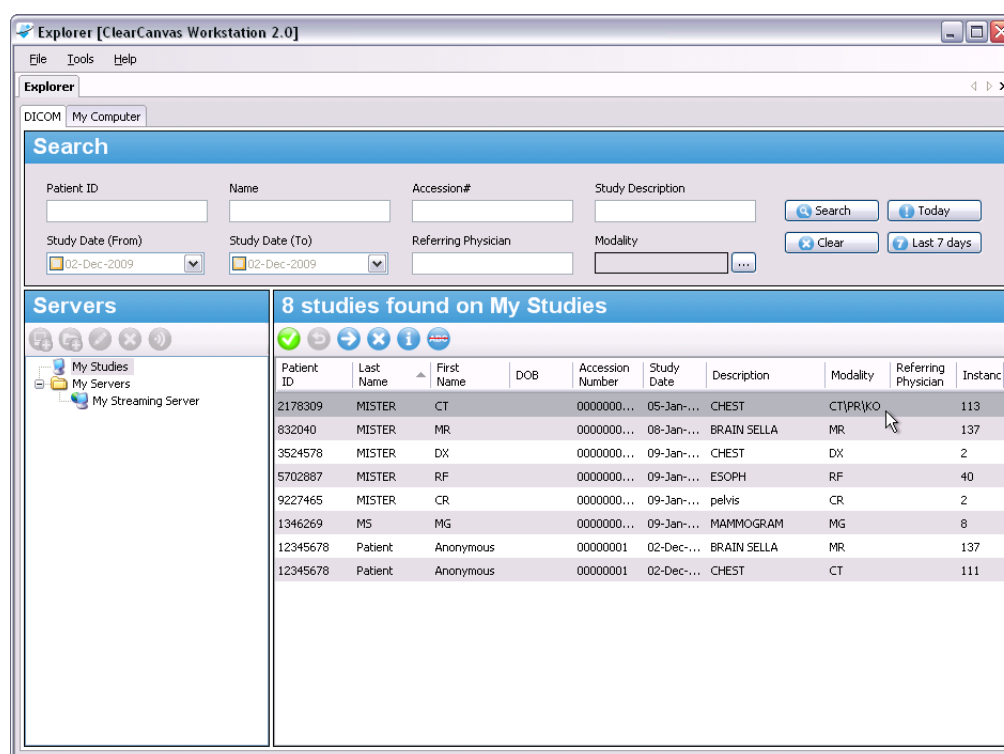


Figure 3.9: Search Menu in ClearCanvas

Since each physician may have his own ideal setup to maximise his workflow, a way to customise the image viewer is a crucial feature on a RIS. ClearCanvas provides a full set of tools to customise the viewer and to manipulate the medical images, which can be seen in the Figure 3.10, allowing the system to adapt to the physician and not the other way around.

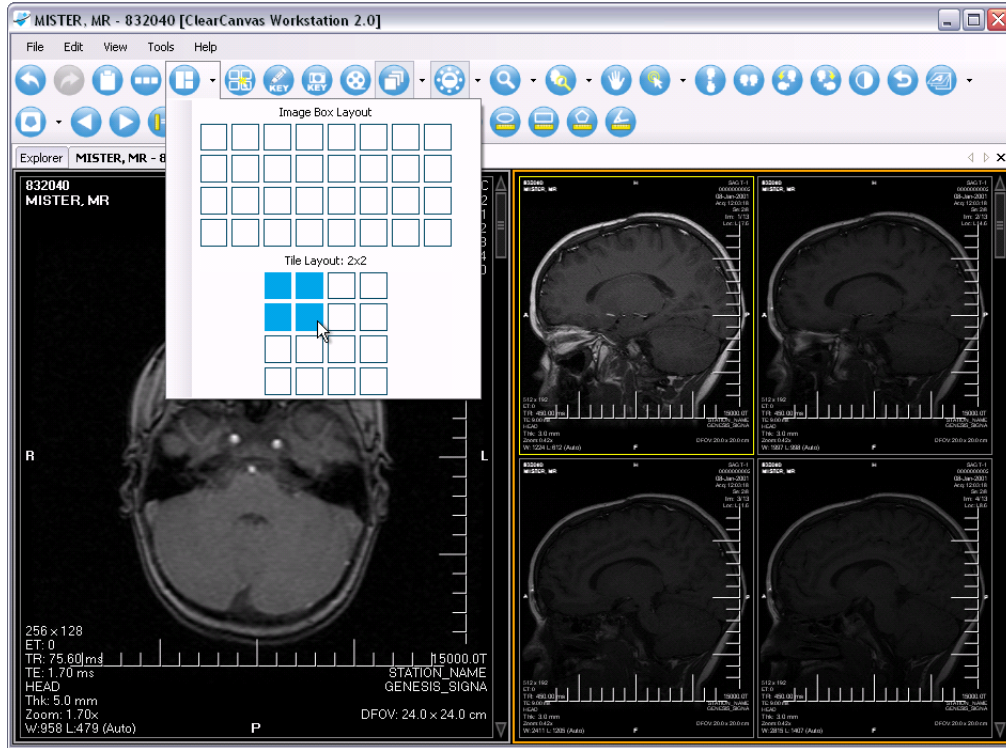


Figure 3.10: Viewer Customisation on ClearCanvas

3.2 PROPRIETARY SOLUTIONS

A proprietary software solution can be considered every software design that does not meet the requirements to be an open-source solution. Meaning that, any kind of restriction on the use, modification and distribution, can label a software solution as a proprietary solution since it does not allow or encourages a user to see and manipulate the source code, which in many cases is not even available to the community.

The solutions presented in this section have a different marketing target than those presented in the previous sections. They are quite expensive and requires a valid license to be used or even tested, making them ineligible for academic research. For this reason, the performed analysis was based on data gathered from public on-line forums, marketing brochures and software enterprise websites. As a result, the level of detail will not be the same, as the one presented in the previous section.

3.2.1 CARESTREAM VUE RIS

This solution was developed by Carestream Health, a company that specialises in healthcare solutions.

Carestream Vue RIS [34] allows the management of patient records, meaning that it has support for adding new records to the system, edit old ones and delete unnecessary patient records from the system's database. The same management principles also apply for clinical staff records.

Regarding the scheduling feature, it provides a way to schedule exams or appointments for patients, where a modality can be associated with that event. Along with the event scheduling, an employee can specify notes on the appointment that may be important for the well being of the patient, for instance, a patient with a pacemaker can not perform a MRI exam. Yet on this matter, modality management is also provided by this system, where a user can add or remove new modalities from the system, specifying the average duration of each one. Currently, urgent events are highlighted with colour flags, in order to emphasise the urgency in the physician work lists.

Carestream Vue RIS also provides a way for the physician to draw on top of medical images, this allows a better understanding of the diagnosis between physicians since, it allows to emphasise certain areas on studies where physicians may need to get a second opinion regarding the diagnosis. This feature does not overwrite the original file since the new image is created and saved, in a separate file with the highlighted areas.

About the report feature, this solution allows for the reports to be exported in PDF extension and are saved internally respecting the DICOM SR standard. To facilitate the physician writing job a **voice-to-text** feature is also available, meaning that the reporting time can be significantly reduced since a physician can dictate the report, as it continues performing the patient diagnosis. Yet on this matter, a voice command feature is also available. This feature works by predefining a paragraph or a small text, that is always repeated in a certain condition or diagnosis and pair it with a voice command, resulting in an easy to use completion tool. As an example of this feature, suppose that a certain paragraph is always written on the report when an aneurysm is presented on a study, if the physician has configured that paragraph to a voice command, he can simply say "auto text aneurysm" and as a result, that same paragraph will be transcript to the study report.

3.2.2 FUJIFILM SYNAPSE RIS

Fujifilm Synapse RIS [35] aims for interoperability between operating systems, offering a web-based solution for radiology management, with an integrated and secure messaging service. This feature may help improve the ease of communication between physicians, or even redefine the way employees communicate on a healthcare enterprise. It also provides a tool for performing requests such as equipment, rooms and meetings online.

With a full-featured record management, Synapse RIS provides a way to create, edit and delete patient and clinical staff records from the system's database.

Regarding the scheduling feature, this solution allows the management of appointments, which comprises create new appointments, edit or delete existing appointments from the system. Upon setting up a new appointment, Synapse RIS also provides a way to specify the modality and the urgency, associated with the event. The event scheduling can be done automatically by the system, where the worklists are distributed accordingly with a working quota and availability of each physician, or manually by a facility employee. To achieve this, it is used a inventory list of all equipments in the institution, in conjunction with a list of all supported modalities. As a result, Synapse RIS supports the management of modalities and equipments that may coexists in the institution.

Providing a way for physicians to draw on top of medical images is also a feature comprised by the Synapse RIS, which in turn may improve emphasis on a certain area of interest on a study. It is crucial to not overwrite the original file, so this solution saves the image with the highlighted areas in a separate file, that can be exported and shared if the patient and physician so desire.

On the report feature, with Synapse RIS reports can be distributed with various secure methodologies, improving the communication between physicians, if a second opinion regarding a study is needed. It has full support for the DICOM SR standard and allows for reports to be exported with PDF extension. A voice recognition tool, as well as an electronic signature tool, are included with this solution, allowing to reduce the turn around time for report writing, in addition to ensuring authorship of the produced study report.

3.2.3 AGFA IMPAX RIS

Impax RIS [36] is a solution created by the AGFA Healthcare that aims for improving productivity and workflow, by providing a way to manage radiology operations. This system can be used in standalone mode or in conjunction with other solutions since it provides HL7 interfaces. It also can be paired with other proprietary modules to implement more features, such as scheduling and business statistics, in order to provide a more complete solution.

This solution has full support for patient records management, which compromise add new records to the system, as well as edit or erase existing records, from the system's database. These patient records hold personal information, such as medical history, vital signs gathered from the patient, alerts about patient health and some other important information relevant to the physician. Since Impax RIS may be integrated with a HIS infrastructure, it also provides a way to import existing demographic records from external platforms, allowing a seamless integration between the two systems. Upon performing an exam, this solution also allows the physician to have access to the patients record information, which may contain crucial information about the patient's healthcare condition. Yet in this matter, Impax RIS allows the association, on a patient record, of dosage consumables, such as contrast and ultrasound measures, that may be used to perform an exam. This information is later saved to the patient's history record. Currently, when performing an exam, a physician can navigate on a chronological view of the patient history with all associated data, such as reports from previous exams, appointments and so on.

The scheduling management is not fully supported with this solution, since it relies on an external system, AGFA iPlan software, to implement it. Currently, it provides a way to manage worklists of different physicians, as well as a visualisation tool for the scheduled worklists.

Regarding the report feature, Impax RIS provides a way for adding new reporting templates on the system, so that physicians can adapt their report according to the modality being performed at the moment. It also has full support for the DICOM SR standard and study reports can be printed or sent on by email or fax. These clinical reports can be exported to other systems, using the available HL7 interfaces.

3.3 FINAL CONSIDERATIONS

This section presents the reader with final considerations regarding the performed study. Table 3.11 illustrate a brief overview of core features, supported by the open source solutions. This table consists in a 3 level rating scale, comprising Fully supported (FS), Partially supported (PS) and Not supported (NS). Currently, this table only depicts open source solutions, since these are amenable

to extension or modification. Aspects regarding the proprietary solutions are also exhibited in this section, as a mean to fully analyse features currently available in existing RIS.

Features	GNU Health	OpenMRS	OpenEMR	ClearCanvas
Patient and staff record management	Fully supported	Fully supported	Fully supported	Fully supported
Modality and resources management	Partially supported	Partially supported	Partially supported	Partially supported
Event scheduling	Partially supported	Fully supported	Partially supported	Partially supported
Export clinical reports (PDF, DICOM-SR)	PDF	PDF	PDF, DOCX	PDF
Report templates	Not supported	Partially supported	Partially supported	Not supported
Speech recognition	Not supported	Not supported	Not supported	Not supported
Voice annotations	Not supported	Not supported	Not supported	Not supported
Web application	Not supported	Fully supported	Fully supported	Not supported
Data access API	FHIR	REST	Not supported	Not supported
User account authentication	Partially supported	Fully supported	Fully supported	Fully supported

Fully supported
 Not supported
 Partially supported

Figure 3.11: Open Source Comparison

As it is depicted in Figure 3.11, open source solutions lack some of the crucial features crucial in an optimal RIS, such as voice reports, DICOM SR standard, support the addition of new report templates, as well as the ability to manage facility resources to improve workflow.

From the presented analysis, it can be concluded that the open source solutions tend to provide an all-in-one approach, regarding the facility where healthcare system will be implemented. As a result, these solutions do not offer a capable and resourceful RIS solution. Currently, most of these solutions are developed with the primary objective of serving as a HIS and as a second thought, implementing a RIS operation management if possible. Consequently, some of the presented solutions only implement a medical image viewer, not providing the environment required for a radiology clinic.

DICOM SR main focus is to provide interoperability between different system regarding clinical reports, allowing both the medical image and the associated report, to be saved in the same PACS repository. It also provides a new way to extend and extract the clinical data, associated with a study case, which is not achievable with free text reports or proprietary structure [37]. Currently, a lot of medical imaging departments, do not harvest the benefits of this standard [38], as a consequence of the complexity of such implementation, as well as the difficulty in exporting structured data to other information systems, that are not compliant with the standard.

Recently, healthcare information systems have been converging into a web application design, resulting in the possibility of centralising the system backbone on a secure place, with all the suitable measures. This phenomenon also provides operating system interoperability, allowing the system administrators to better manage the available resources as needed.

Voice-to-text is a feature that can be found in almost every proprietary solution but, has not yet been implemented in open source solutions, due to the complexity associated, as well as the hit rate

associated with this feature. This feature can be of great value for both the physician and patient, since it can speed up the reporting tasks, resulting in a reduced diagnose turnaround time.

Regarding the resource management, some proprietary solution approaches this subject by providing a way to create an inventory for a specific department. Afterwards, modalities can be associated with certain equipment or examination rooms, allowing for an optimal scheduling, as well as optimal workload distribution.

Since a healthcare facility can support more than one modality, is crucial for a healthcare system to support multiple report templates. Currently, the services provided by a healthcare facility may change over the time, or evolve to new techniques. For this reason, it is crucial for a RIS solution to provide a way to integrate and update various universal templates for reporting.

As it stands, there is no open source solution that comprises all the crucial features for a flawless management of operation in a RIS. Regarding the proprietary solutions, although the operations management can be achieved, some of this solution may result in computer-intensive tasks on the client side, as well as implementing a vendor lock-in anti-pattern.

SYSTEM REQUIREMENTS

This chapter aims to present, in a precise and organised manner, the system requirements gathered from the comprehensive study presented in section 3. As previously stated, Smart RIS aims to solve some of the limitations presented in existing RIS solutions, such as the lack of voice reports, the ability of export or print reports in a structured manner, provide a small and scalable application for small healthcare facilities, modality configuration and management, a good and reliable scheduling feature and so on. The correct implementation of these features allows physicians to have a smoother user experience, since the administrative burden can be shifted to another facility staff member, which may not be the case with small healthcare facilities that use other solutions.

4.1 SYSTEM WIDE REQUIREMENTS

Nowadays the health service provision relies heavily on medical imaging and diagnostic imaging tools, in order to deliver the best healthcare possible. The current solutions for radiology clinics may be lacking a few sets of features, as are the open source solutions, or may be expensive and heavy reliant on the vendor lock-in pattern, as proprietary solutions do.

Smart RIS aims to counteract this phenomenon, by offering a web-based application that can fit the set of operations encompassed in a RIS facility workflow. Since small clinics may also have the need for medical imaging, the presented solution was developed as a standalone application, providing full facility management and scheduling of operations, as well as other administrative features.

The proposed solution was developed from scratch since modifications in current open source solutions, would have led to the implementation of features not usable in a RIS operation workflow. Consequently, not resulting in a lightweight and scalable solution.

4.1.1 FUNCTIONAL REQUIREMENTS

Currently, this project is intended to be implemented in a radiology clinic, where it may exist a diverse set of employees with different qualifications. As a result, one of the first requirements is to support a list of fundamental roles in the radiology workflow, as well as to provide a secure and reliable

way to authenticate a user, within the system. As a result of this requirement, Smart RIS provides four different roles for staff members, namely System Administrator, Technician, Physician and Secretary.

The Secretary role encompasses the patient record management, as well as the appointments or exam scheduling.

The Technician role has the ability to browse his own worklist for a certain day, search for patient records, fulfil a report either by text or by voice, as well as the ability to visualise medical images associated with a certain exam.

The Physician role encompasses the technician role, in conjunction with the ability to sign a report, referred to him, by a technician.

At last, the System Administrator role includes all the above-stated operations, plus the ability to manage facilities and its respective rooms, staff member records and modality records. However, it can not access the exam viewing feature neither can produce a clinical report, since it could be a major liability for a healthcare institution.

Figure 4.1 provides a support for what was previously stated, allowing the reader to better visualise the existing system roles, as well as the set of operations encompassed in each role, within the system.

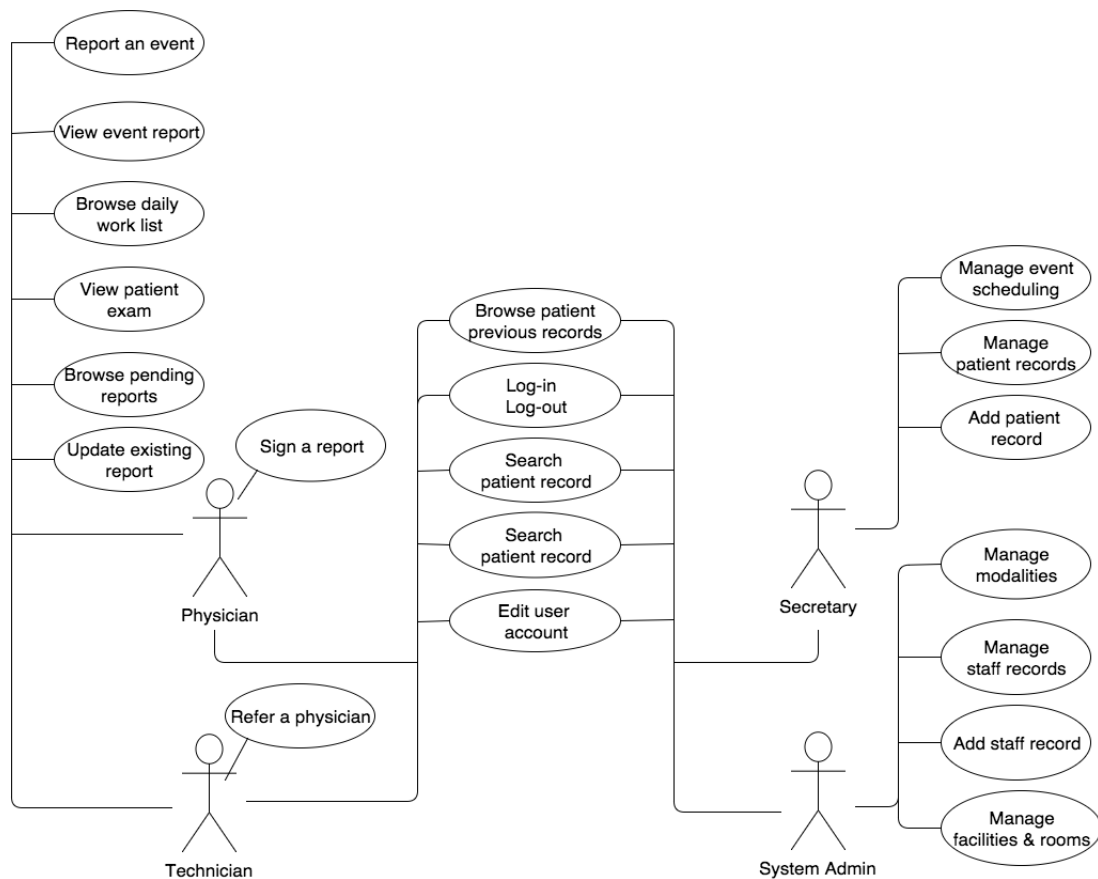


Figure 4.1: Use Case Scenario

The following list, provides a brief description of each use case scenario, depicted in Figure 4.1.

- Log-in / Log-out - A user should be prompt to enter his credentials, in order to gain access to Smart RIS. If the user chooses to log out, all session elements should be deleted from the browser.

- Search patient records - User performs a search throughout patient records and browses a list of results. If a record is selected, patient detailed information will be displayed.
- Browse patient previous events - After selecting a patient record, a list of all previous records will be presented to the user. If one particular event is select, detailed information regarding the event will be displayed.
- Edit user account settings - A staff member edits his personal information, regarding his account on Smart RIS.
- Report an event - A user selects a pending report to fulfil. All the associated information such as patient data, event data and such, shall be made available for the user to consult.
- Update a Report - A user selects a pending report to fulfil. If there is already a first version of it, the ability to modify shall be provided.
- View event report - If available, a user should be able to visualise a clinical report associated with a specific event.
- View patient exam - If available, a user should be able to visualise an exam that has been performed on the associated event.
- Browse daily work list - If a certain day is selected on the calendar, all events regarding the selected day and related to the current user shall be displayed.
- Browse pending reports - A list of pending events to report should be displayed to the user.
- Sign a Report - A report should be loaded from the system database, allowing a user to sign it, if all the contained information is correct.
- Refer a physician - A list of available physicians should be displayed to the user, allowing him to choose which ones to include in the refer list.
- Manage event scheduling - A calendar with multiple views shall be made available to the user. If a day and an hour are selected, a form to add a new event will be displayed. If an already existing event is selected, the ability to update information should be provided to the user.
- Add a patient record - A form to input all the information needed to create a record should be displayed to the user.
- Manage patient records - A list containing patient records will be displayed to the user. If a patient record is selected, the ability to update or delete information should be made available to the user.
- Manage modalities - A list containing all modalities in the system will be presented to the user. If a certain modality is selected, detailed information will be presented to the user in conjunction with the ability to update such information, or delete it.
- Add a staff record - A form to input all the information needed to create a new staff record will be displayed to the user.
- Manage staff records - A list containing staff member records will be displayed to the user. If a certain record is selected, the ability to update or delete information should be available to the user.

- Manage facilities and rooms - Depending on the pretended operation, a list containing records regarding facilities or rooms, will be presented to the user. If a certain record is selected, detailed information should be displayed and the possibility to update or delete such record should be available to the user.

Providing a way to schedule all operations in a radiology facility, is also a requirement of this solution since it can improve workflow and maximise resource usage. Smart RIS should also provide the ability to browse through a calendar having two distinct sets of results. When a user with the Secretary role examines the calendar, all events of the current facility should be displayed. On the other hand, when a user with the role Physician or Technician examines the calendar, only events that the current user intervened, either by being a physician of an appointment or a technician of an exam, are shown. As a result, these two system roles should have the ability to check their worklist for a certain day.

Be able to report an event is one of the most important requirements of a RIS. Nowadays with the evolution of information technologies, it is necessary to support not only text, as an input for a clinical report, but also voice report, eventually with a voice-to-text feature. The implementation of such features enables a physician, as well as a technician, to better employ the time they may have available for reporting. By joining a written report with voice report, Smart RIS can provide a way to better complement the report data. Moreover, the developed solution should also provide a way to export a clinical report in a structured way, following the DICOM SR standard, resulting in an improvement regarding the interoperability between different systems.

By enabling physicians and technicians to access and manipulate medical images, a RIS allows for a better healthcare provision, since it encourages physicians to use all available data resources, related to a certain patient. As a consequence of the previous sentence, providing a mean to visualise, manipulate and store medical images, is another important requirement that needs to be fulfilled, to archive an excellent solution.

These requirements can be seen as a result of the deep analysis performed on open source and proprietary solutions, presented and discussed in chapter 3.

4.1.2 NON-FUNCTIONAL REQUIREMENTS

Besides the previously stated requirements, Smart RIS also has a set of non-functional requirements, which describe how the system should behave.

Regarding the performance aspect, the solution should provide a latency free user experience. This can be achieved, by implementing a back-end system that supports asynchronous accesses, as well as a database engine that can also handle multiple queries being performed simultaneously.

Considering now the security aspect, the system should have a reliable and robust authentication mechanism. A user can only access the system data or perform a certain system operation if, a valid log-in has been performed. In order to securely store log-in credentials on the database, the system should first use a hashing algorithm on the user password, resulting in a lower chance of user accounts being compromised.

Simultaneously, reliability is also a requirement for this solution. This means that it should provide a reliable and trustworthy way to access the system, as well as a predictable operation output when handling information being held on the database. For this reason, the system should use database transactions to ensure integrity on the database.

Since the target audience of this solution may not be tech savvy, the web application implemented by the system, should be intuitive for the user and provide a user-friendly interface, allowing an easy navigation through the system, until reaching the pretended operation.

The correct implementation of the previously stated non-functional requirements, will result in a solution that provides a trustworthy and responsive system, with no latency experience to the end user.

CHAPTER 5

SYSTEM ARCHITECTURE AND IMPLEMENTATION

In this chapter, the system architecture will be presented, as well as a brief overview of the technologies that have been used in the development of Smart RIS. Some important aspects regarding the implementation processed, such as database model and authentication mechanism, will be presented and discussed in this chapter.

5.1 SMART RIS ARCHITECTURE

Smart RIS follows the service-oriented best practises, where a collection of smaller controllers are used together to implement the intended business model. Service-orientation presents an ideal vision of an environment where resources are clearly partitioned and consistently represented.

Responsibility segregation was a key factor during the development of this solution, this means that a certain controller is bound to fulfil every operation regarding its responsibilities.

Smart RIS is provided and deployed as a single service, where each controller can be seen as a service alike entity, that deals with a set of responsibilities. This approach allows for an easy and fast deployment, resulting in an effortless way to use or test the developed solution. Consequently, it is only required to run one play project instance, originating a lower usage of resources. However, the services can be decoupled and run in distributed server units.

As it stands, Smart RIS implements a client-server architecture paradigm. In this case, the client is a web browser that access the provided solution. The client-sided logic is responsible for providing the user interface, as well as receiving and validating inputs that a user may provide. The server side is responsible for implementing the business model, handle requests from the clients, access the database, among others.

To better illustrate the developed system architecture, Figure 5.1 is now presented:

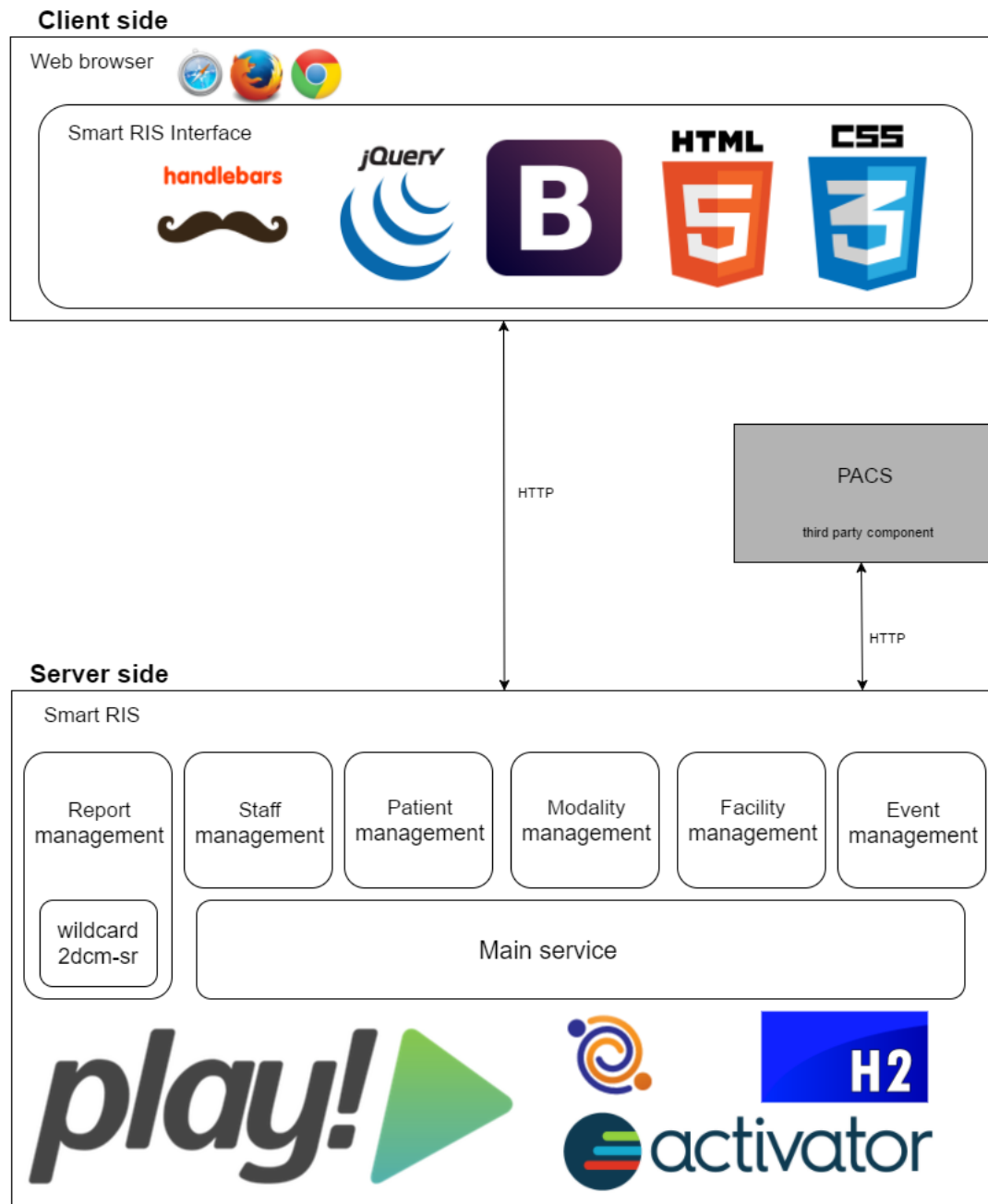


Figure 5.1: Smart RIS System Architecture

To promote responsibility segregation, a modular approach was implemented on the server side, as it is depicted in Figure 5.1. Each small rectangle illustrates a set of capabilities, provided by a module. As it stands, a module is represented by an application controller, that interacts with a Model and respond to certain actions a user may want to perform. (see Figure 5.2) Currently, the report management controller relies on a preexisting component, in order to allow converting a certain clinical report compliant with DICOM SR standard. Smart RIS is also endowed with a controller labelled **Main service**, implementing all the broad scope logic, such as authentication, role base operations, report notifications and so on.

Regarding the client side, it consists of a set of Javascript methods that work in conjunction with the controllers, in order to implement all the necessary logic. Following the same approach of the server

side, each set of methods that implement a bundle of related features is located in a particular *.js* file, achieving responsibility segregation. As a result, a more organised structure is provided, regarding code presentation. Furthermore on Figure 5.1, a third-party component is also coupled with the client logic. This component implements the ability to visualise and store medical images on Smart RIS.

Communications between system entities are achieved using the Hypertext Transfer Protocol (HTTP) protocol. This subject will be discussed further on. (see section 5.5)

As it stands, Smart RIS can be converted to follow a Service-Oriented Architecture (SOA) pattern. To achieve this, for each service required to run in standalone mode, it is necessary to create a new Play project. Afterwards, it is necessary to add the respective controller and route paths associated with the new project. The last step consists on configuring the database fields, in the new project's configuration file, as shown in Listing 7. If all steps were correctly executed, the client-side application should be able to access the segregated services, since all the Uniform Resource Identifier (URI)s are mapped on the route file.

5.2 SERVER APPLICATION

Taking into consideration that Smart RIS relies heavily on a web application, it is necessary to use a framework that is capable of providing a set of requirements, such as being able of responding to asynchronous requests, lightweight, stateless, scalable, reliable among others. Currently, Play fulfils all the previously stated factors [39], in addition to of being an open source solution widely used by the web community. It provides a set environments to develop web applications, either using Java or Scala programming languages, in order to appeal to a larger community of programmers. As a result, Smart RIS was developed over this framework. Play does not need to be installed in the Operative System. As long as the latest Java Development Kit (JDK) version is installed on the computational system, it is possible to download the latest activator *.zip* file, from the Play website, and extract the content to a folder with execution permission enabled. After completing the extraction, a script is available to the developer allowing him to create a new project, execute an existing project and browse project templates. Table 5.1 displays some of the operations that can be performed with the previously described script.

Command	Description
<code>./activator run</code>	Execute an existing project in the current folder
<code>./activator new</code>	Create a new project, specifying the programming language
<code>./activator clean</code>	Clear all dependencies on a existing project
<code>./activator ui</code>	Displays a list of existing templates with search abilities

Table 5.1: Activator Commands

Currently, Play Framework follows the Model, View, Controller (MVC) architectural pattern (Figure 5.2), that splits the application into three interconnected entities [40], which are:

- **Model** - express all the data contained in a Java object, in other words, represents the information a web application may have.
- **View** - represents the visualisation a certain Model may have. The associated data is afterwards rendered to an interface and displayed to the user.
- **Controller** - represents actions a user may perform. Controllers act on both models and views, in other words, they control the data flow into a model object and updates the view accordingly.

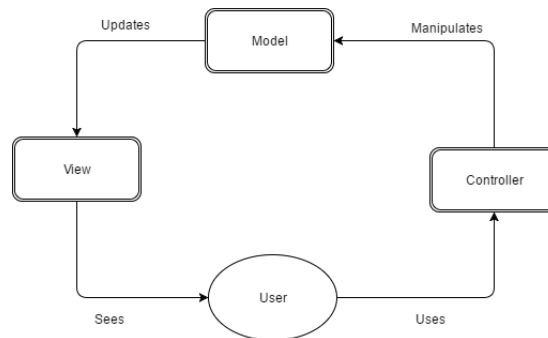


Figure 5.2: MVC Diagram

This pattern is clearly visible in the project structure, where Play specifies three sub-folders that correspond to each one of the three entities this pattern refers to. Figure 5.3 depicts all controllers used in Smart RIS are placed inside the controllers folder. The full set of models consumed by this solution is included in the models folder. Lastly, all views used in this solution are included in the views folder. Smart RIS relies on the single-page application pattern, as a result there is only one view specified on this folder, which is dynamically updated with pre-defined templates, compiled with handlebars.

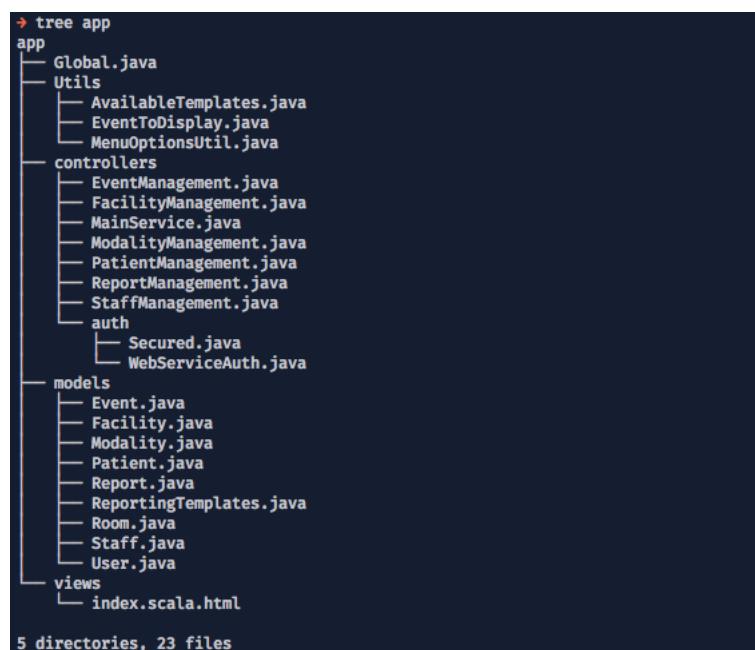


Figure 5.3: App Folder Tree

5.3 CLIENT APPLICATION

As in every web application, there is a significant effort to develop a usable and intuitive user interface, since it represents the bridge between system and the user. For this reason, the web technologies used in the development of the client application will be described in this section.

Starting with JQuery [41], this library offers an easy to use environment, providing a simpler syntax and allowing a developer to achieve the same feature in significantly fewer lines of code, when compared to plain Javascript. JQuery also provides a set of pre-build functions that are not available in some Javascript libraries. Along with this, there is a strong open source community supporting this project, providing well-written documentation and a large number of tutorials. By also featuring Ajax support, JQuery can implement asynchronous requests to the backend server, allowing actions to be performed on web pages without requiring the entire page reloading. This small footprint library also provides selectors that can manipulate style, react to events, implement animations and so on.

In order to achieve code consistency and organisation, template engines should be used, allowing for a separation between markup and application logic. To accomplish this, it was used a semantic templating language called Handlebars [42], with a syntax reminiscent of HyperText Markup Language (HTML). Along with this, Handlebars provides a feature called expressions, allowing for high-level logic like comparisons, on the compiled template. This can be beneficial since it appeals to the reuse of templates, as one template can fit various scenarios. Handlebars also supports pre-compiled templates, allowing a faster start up time. Another key feature of this framework is that it achieves a significant performance increase when compared with other similar tools like, for instance, Mustache.

Since Smart RIS provides a web application, it is necessary to support a way for the client-side logic request a certain template. Presented in Listing 1 an example of a method for retrieving a template that is mapped to a public assets folder. Such folder is configured as public on the route configuration file, meaning that all included files are accessible to the client logic. After a client-side application successfully retrieves a certain template, a callback is triggered in order to compile the fetched template. An example of a template fetching, data loading and template display can be seen in the code block given on Listing 2.

```
function getTemplate( name ) {  
    return $.get( 'assets/templates/'+name+'.hbs' ).then(  
        function( src ) {  
            return Handlebars.compile( src );  
        });  
}
```

Listing 1: Method to load templates on

```
getTemplate( "template_searchEvent" ).done( function( template ) {  
    $( '#page-wrapper' ).empty();  
    $( '#page-wrapper' ).html( template({ title: "My appointments" }));  
});
```

Listing 2: Example of loading an template to a HTML div

A problem associated with web-applications is how to manage a large set of client-side dependencies.

To solve this problem, Smart RIS uses Webjars [43], consisting of libraries that are compiled into JAR files and provided to the client by the Play Framework. In order to add a Webjar to a Play project, it is necessary to edit the `build.sbt` file and add the respective *SBT* import line. The available *SBT* dependencies can be consulted in the Webjars website, where a search interface is provided to developers. Listing 3 illustrates an example of three dependencies that are used in the Smart RIS.

```
libraryDependencies += Seq(
  "org.webjars" % "jquery" % "2.2.1",
  "org.webjars" % "bootstrap" % "3.3.4",
  "org.webjars" % "handlebars" % "4.0.2"
)
```

Listing 3: Webjar dependencies

In addition to Webjars, there is also a need to include a set of scripts that implements the client-side logic. These scripts are separated by the areas of interest, allowing for responsibility segregation. As a result, there is a script responsible for managing the client-side logic regarding the patient records, another one to handle facility and rooms management, modalities, and so on. Since these scripts are placed inside the public assets folder and this folder is marked as a public asset, a client application such as a browser can easily retrieve them.

Alerts are a necessary feature of every information system. For this reason, Smart RIS relies on the **SweetAlert** [44] library to provide highly customizable alerts. These alerts can provide useful feedback to the user, as well as gather input data from such user.

Currently, Smart RIS needs to provide a set of calendar views in order to fulfil its requirements, regarding the scheduling feature. To achieved this, it was used an open source project labelled **Bootstrap Calendar** [45]. This Javascript library was modified in order to achieve the pretender aesthetic of the interface, as well as to implement asynchronous load of system events.

To implement voice recording on the client-side logic of the application, it was used another Javascript external library named **VoiceJS**. This library was heavily modified in order to support audio data encoding/decoding in base64, multiple thread access, audio playback and file download to the user's computer.

As it stands, **Bootstrap Calendar** and **VoiceJS** libraries were heavily modified in order to fulfil the necessary functionalities. For this reason, these libraries are not provided as Webjars, but rather in public assets form.

5.4 DATA PERSISTENCE

Play framework allows the integration of multiple tools such as Hibernate, Ebean and basic Java Persistence API (JPA). Currently, Hibernate is widely used in the Java community, but Ebean is generally the chosen tool to integrate with Play Framework, meaning that proper documentation is available on-line, for developers to browse. Considering that Ebean fulfils all the requirements, has lower complexity than the current competitors, it was integrated into Smart RIS.

Java Ebean allows entities to be developed following the Object-Relational Mapping (ORM)

pattern. As many other solutions, such as Hibernate, Ebean [46] implements part of the JPA standard. JPA can be seen as a specification of Java application programming interface, which describes the management of data information in applications, using Java platform [47]. Since Ebean relies on a session less architecture and does not require a JPA Entity Manager or a Java Data Objects (JDO) Persistence Manager, it provides a much simpler API when compared with JPA. Currently, Ebean expects the developer to annotate his models with JPA [48] annotations, allowing to configure the behaviour of his entities. Annotations are a mean provided by JPA that permits to configure meta-data in an entity class, that will later be compiled into a Java class. These annotations allow overriding the default values placed by the JPA environment and have the peculiarity of always start with a *@*. Table 5.2 presents a few annotations available, as well as the associated functionality.

Annotation	Description
@Entity	Specifies that the class is an entity
@Id	Specifies the primary key of an entity
@GeneratedValue	Provides the specification of generation strategies for the values of primary keys
@Column(unique = true, nullable = false)	Is used to specify the mapped column for a persistent property or field
@OneToOne	Defines a single-valued association to another entity class

Table 5.2: JPA Annotations

5.4.1 INTEGRATION WITH PLAY FRAMEWORK

Taking into consideration that Ebean is prearranged as default with play framework, it is necessary to enable this module by editing some configuration files on the project. The following description explains the configurations adopted by Smart RIS.

- Starting with the *plugins.sbt* file, it is necessary to add the Play Ebean plug-in to the project. This is achieved by adding the following line on the mentioned file:

```
addSbtPlugin("com.typesafe.sbt" % "sbt-play-ebean" % "3.0.0")
```

Listing 4: New sbt plug-in

- Afterwards, it is necessary to enable the recently added plug-in. To do so is necessary to edit the *build.sbt* file, resulting in the following:

```
lazy val myProject = (project in file(".")).enablePlugins(PlayJava,
  PlayEbean)
```

Listing 5: Enable Ebean plug-in

- In order to configure the package where the model classes will be placed, it is necessary to edit the *conf/application.conf* file and add the following line:

```
ebean.default = ["models.*"]
```

Listing 6: Models package

- The last step is to configure the database driver, path and credentials if necessary. To do so, it is essential to add the following lines on the *conf/application.conf* file:

```
db.default.driver=org.h2.Driver
db.default.url="jdbc:h2:~/RIS/_DB"
db.default.username=root
db.default.password=""
```

Listing 7: Database configurations

In the configuration presented in Listing 7, it is illustrated that H2 database engine is being used. However, it can be easily replaced by a MySQL engine, through simply changing the driver to `com.mysql.jdbc.Driver`. The physical location of the database files is defined by the Uniform Resource Locator (URL) parameter, being the home directory in this case. Regarding the credentials, the presented values correspond to the *root* user and an empty string for a password, in this case the default credentials provided by Ebean. If a developer for some reason, wishes to use a memory only database, the URL field should be changed to *jdbc:h2:mem:play*. However, it is important to note that if the project execution stops for some reason, the database will be lost, in consequence of being located in the computational system's Random-Access Memory (RAM).

To better illustrate what has been described, an example of a model class is presented, in Listing 8. Falling back on what was previously stated, the presented class is an entity *User*, as we can see from the annotation **@Entity**. The field *id* represents the primary key and is automatically generated by unitary increments, as it can be understood from the annotations **@id** and **@GeneratedValue**. Regarding the annotations on the column *role* and on the column *password*, it is enforced that any record which is being inserted into the database has a value different of null on those two columns. On the column *email*, the previous conditions are also applied with the addition of each inserted value being unique within the database. As a good practice in Object-Oriented Programming (OOP) development, all fields are set to private, instead of public. The access to these fields is mediated by a collection of methods called getters, to get a field from a class, and setters to set a field on a class. This practice allows modifications on the class without changing the interface, meaning that is not

necessary to change the code at a higher level on the project. The implementation of this practice can be seen in the Listing 8. Also presented in this listing, is a static method that represents a query to be performed on a set of User objects. This method takes as argument a String and returns a User object containing an email equal to the argument string. Since the email is a unique column, this method will not return more than one object at a time. If the required object is not found, the method will return a null object.

```
@Entity
public class User extends Model {

    @Id
    @GeneratedValue
    private Long id;

    @Column(nullable = false)
    @Constraints.Required
    private String role;

    @Column(unique = true, nullable = false)
    @Constraints.Required
    private String email;

    @Constraints.Required
    @Column(nullable = false)
    @JsonIgnore
    private String password;

    public String getPassword()
    public void setPassword(String password)
    public Long getId()
    public void setId(Long id)
    public String getEmail()
    public void setEmail(String email)
    public String getRole()
    public void setRole(String role)

    public static Finder<Long,User> find = new Finder<>(User.class);

    public static User findUserByEmail(final String email) {
        return
            find.where()
                .eq("email", email).findUnique();
    }
}
```

Listing 8: Ebean entity example

Currently, Ebean uses database transactions by default. However such transaction is created and committed or rolledback, after every single query, update, create or delete. In order to perform more than one action per transaction, it is necessary to use a similar approach to the one presented on the

Listing 9.

```
/*
    Check if a certain email account already exists on the system database
*/
@Transactional
@Security.Authenticated(Secured.class)
public Result checkEmailOneness()
{
    JsonNode json = request().body().asJson();
    if (json == null) {
        return badRequest("Expecting Json data");
    }

    Staff s1 = Staff.findStaffByEmail(json.findPath("email").asText());
    s1.done = true;
    s1.save();

    if (s1==null)
        return ok("true");
    else
        return ok("false");
}
```

Listing 9: Transaction usage example

5.4.2 DATABASE MODEL

In order to better illustrate the database model, Figure 5.4 offers a brief description of the entities used on Smart RIS. This image consists of a Unified Modeling Language (UML) diagram, providing an overview of each entity, as well as attributes and relationships between entities. In this type of diagram, an entity is represented by a rectangle, where each attribute is included inside of such rectangle. A line that connects two different rectangles describes a relationship between those entities. The small characters that usually are placed near a relationship line and an entity, is the relationship multiplicity, allowing for a UML diagram to specify cardinality.

As it is depicted on Figure 5.4, there are a few entities presented on the Smart RIS solution. A brief introduction of each one will now be given, in order to give the reader a better understanding of the purpose of such entity.

- **User** - this entity is responsible for holding all data a staff member may have, regarding the system account, in other words, it is responsible for holding information like password, email and the role a certain user has in a facility associated in Smart RIS.
- **Staff** - this entity is also responsible for holding information regarding the staff members. The main difference between this class and the **User** class is that this entity is responsible for holding general information regarding a certain staff record such as name, address, phone number and so on.

- **Facility** - to allow multiple facilities working under one healthcare enterprise, this entity was created. This class allows the storage of common information, such as phone number, address and name, regarding a radiology facility or department where multiple rooms may coexist.
- **Modality** - since a radiology infrastructure may support a diverse set of modalities, there is a need for a way to associate different modalities to a room or to an event. As a result, this entity was created, allowing an association between the supported modalities on a room or on an event schedule.
- **Room** - as it stands, a facility may have a diverse number of exam rooms and may support a varied set of modalities. As a result, this entity allows to store information regarding a room, which can later be associated with a facility object. A modality is also paired with a room since it can greatly improve the scheduling feature and provides a way to better management the available resources.
- **Patient** - as in every other healthcare software, there is a need to save a patient record on the system. To solve this problem, **Patient** class was developed to keep useful information regarding a patient, such as a name, address, phone number, warnings on conditions a patient may have, among others.
- **Event** - the main purpose of this entity is to store all information regarding an event. For this reason, this class can be seen as an association of a lot of other entities that may interact at the event, along with some fields that help to better define an event.
- **Report** - as a result of an event, such as an exam or an appointment, a clinical report may be produced. Consequently, there was a need to associate the generated report with the related event and some other information such as the actual report text, voice annotations, reviews and so on. For the previously mentioned reasons, this entity was added to the solution.
- **ReportingTemplate** - currently, Smart RIS allows an easy integration of multiple reporting templates, by simply draggin them to a folder, and consequently, performing some kind of indexation. Smart RIS performs the referred indexation when it starts. As a result, there was a need to save this information, leading to the use of this entity. The **ReportingTemplate** class provides a mapping between a resource path and the name of the template, meaning that when a user, on the client side, requests a certain template, the server will perform a query, extract the path for such template and load the file, in order to serve it to the user.

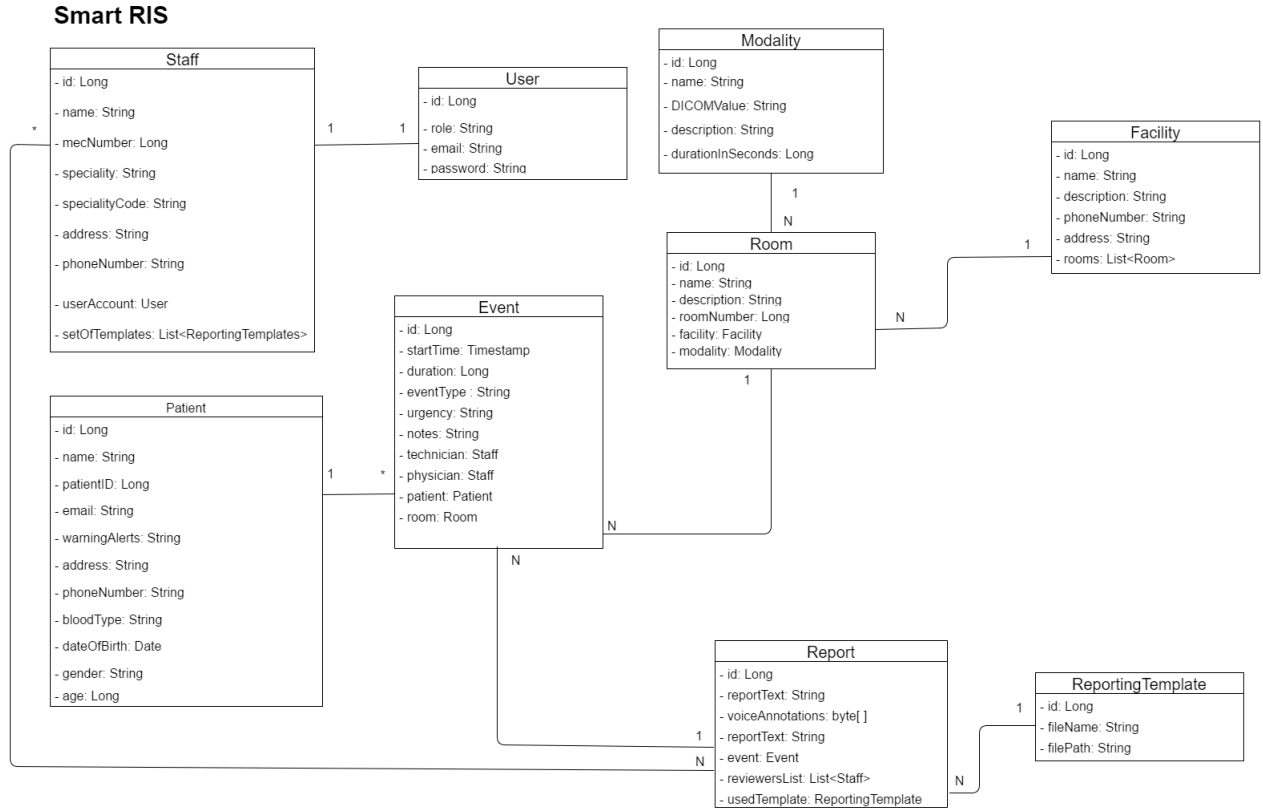


Figure 5.4: Database Model

5.5 COMMUNICATIONS

Regarding the communication architecture, it is based on the REST architectural style. It may follow a client-server architecture and relies on a stateless pattern, in other words, the server must be able to completely understand a client request, without using any server context or server session state. Almost in every implementation, the HTTP protocol is used, but since REST is not a standard, but rather an architecture style, other protocols may be used if so is needed. Currently, there are some alternatives such as Simple Object Access Protocol (SOAP) and Remote Procedure Call (RPC), but the use of REST is often preferred since it leverages low bandwidth usage, by using a smaller message format and not requiring intensive processing of the messages. Services using the REST architecture style are also easier to scale when compared with the competing alternatives, making this style widely adopted for services that are exposed throughout the Internet.

As stated in the previous paragraph, REST is usually implemented with the HTTP protocol, this being the underlying protocol used by the World Wide Web. This protocol is well-known for being the communication bridge for web servers and modern web browsers. HTTP is a stateless protocol, that offers a set of methods, such as the ones listed on Table 5.3, and provides a response in the form of status codes. A brief example of some of these response codes is listed in the Table 5.4.

HTTP Verb	Description
POST	Submits data to be processed
PUT	Modifies data on a specified resource
GET	Requests data from a specified resource
DELETE	Deletes a specified resource

Table 5.3: HTTP Methods Overview

Status code	Description
200 Ok	Default response when everything goes as expected
304 Not modified	No changes on the resource since last request
404 Not found	Requested resource not found
500 Internal server error	Error on the server while processing the request

Table 5.4: HTTP Status Code Overview

Play Framework provides a router component [49], allowing each HTTP request to be mapped to an action that will be performed on a controller. Since an application may need to specify a set of URIs, Play provides a configuration file (Listing 10) where a HTTP method and URI can be mapped to an action, to be executed on a certain controller. This file has a table like structure, where the first entry in a row is the used HTTP method, the second entry represents the URI a client may use and the last entry represents a public method to be executed in a certain application controller. Every route file is sorted in a descending order of priority, meaning that, when two entries specify the same HTTP method and the same URI, the chosen entry is the first one that appears in the configuration file.

To better illustrate the structure of a route configuration file, Listing 10 offers a brief overview of the configuration used in Smart RIS.

```

# Routes
# This file defines all application routes (Higher priority routes first)
# ~~~~

#Main service scope
GET      /                controllers.MainService.index()
POST     /authenticate    controllers.MainService.authenticate()
GET      /logout          controllers.MainService.logout()
GET      /login           controllers.MainService.login()
POST     /getMenuOptions  controllers.MainService.getMenuOptions()

#Staff Management scope
POST     /addMember       controllers.StaffManagement.addNew()
DELETE   /deleteMember    controllers.StaffManagement.delete()
POST     /updateMember    controllers.StaffManagement.update()
POST     /getMember       controllers.StaffManagement.getMember()
GET      /staff           controllers.StaffManagement.getStaff()
GET      /staffPhysicians  controllers.StaffManagement.getPhysicianStaff()
POST     /checkStaffEmail  controllers.StaffManagement.checkEmailOneness()
POST     /checkStaffMec    controllers.StaffManagement.checkMecOneness()
GET      /getStaffMec      controllers.StaffManagement.getLastMec()
POST     /getByName        controllers.StaffManagement.getUsersByName()
POST     /getRole          controllers.StaffManagement.getRoleOnSystem()

```

Listing 10: Route file example

As a result of this, a URI specified in Smart RIS can be consumed on the client side by simply performing a request to the base server URL + URI, resulting in the action associated with that specific URI being executed.

5.6 REPORTING

The ability to report a medical event is a crucial feature for every healthcare software. This is even more essential in a radiology environment, where it is possible to have a lot of outpatients performing exams. As a result, Smart RIS provides an enhanced feature for producing clinical reports.

Currently, this solution provides a way to integrate report templates written in HTML. To achieve this, at the start of the execution, an indexation is performed, mapping a template file path to a name such report may have. As a result of this, a system administrator can easily add or remove sets of templates, by simply dragging them into the folder *public/templates/RadiologyTemplates*, allowing for a custom solution to fit the institution needs. Smart RIS is compliant with IHE regarding the Management of Radiology Report Templates (MRRT), since it allows for the integration of reporting templates developed by radiology consortiums. By default, this solution is deployed with a subset of RADreport templates, consisting of two hundred radiology templates provided by Radiology Society of North America (RSNA) members [50].

When a user is prompt to fulfil a report, the ability to choose a template, whether by name

searching or by browsing a list, is provided to the clinician. After filling the report, a physician may choose to save it into the system’s database. This is achieved by gathering all input fields from the HTML form and merging them to create a JSON object, later this structure is sent to the backend where is made persistent in the database. Currently, the creator of a clinical report can also obtain feedback from other institution’s physicians, in a review form. After selecting the review list, all data are stored in the system’s database. Whenever a physician from the review list logs into the system or refresh the side menu, a red notification will draw the user’s attention. Along with a text base report, a physician can also add voice annotations or voice recordings, allowing a user to emphasise key aspects, that may later be consulted.

A reviewing feature is also included in Smart RIS, where a user may fill the report and refer a list of physicians to review it. A reviewer can update the report (if needed) and sign it.

In addition to what was previously stated, Smart RIS also has the ability to download a report in Portable Document Format (PDF) format or compliant with the DICOM SR standard, making use of the Wildcard2dcm-sr library.

Currently, PACS systems are responsible for managing all operations regarding imaging studies, while clinical reports associated with these medical images, are managed by the RIS or HIS [11]. This is the result of having files not compliant with the DICOM standard, instead clinical reports implement proprietary mechanisms and structures.

The main objective of wildcard2dcm-sr is to provide a way to generate structured reports from existing clinical reports, regardless the template or information source of such document. The process to achieve this can be decomposed into three phases: loading, mapping and assembly [51], where each phase is latter implemented by a distinct component. A brief description of each phase will now be given, in order to better elucidate the reader:

- **Loading** - this phase is responsible for loading the data associated with a clinical report, from an external application. In this case, it is responsible for getting the report data from Smart RIS.
- **Mapping** - the main objective of this phase is to translate the external concept labels into the library’s dictionary of well-known concepts
- **Assembly** - it encompasses the necessary operations to place the mapped concepts in the correct nodes, respecting the report tree. Also included in this phase is the creation of DICOM SR objects, which will later be written to a file and provided to the user.

Since the DICOM standard specifies a set of templates for each modality [52], it is necessary to also provide a way to state wich concepts should be included in the report. This is achieved using the template loader component, capable of loading the report structure from a set of configuration files. These configuration files are JSON/XML representations of the TIDs and CIDs that define the standard template being used.

As a result of the high complexity inherent to the DICOM SR object’s tree structure and the mapping of the *TIDs* and *CIDs* that is required, wildcard2dcm-sr was an incomplete solution, supporting only mammography clinical reports [53].

This library was complemented in the scope of this dissertation, adding support for other kinds of clinical reports. This was achieved by adding new JSON mappings, in order to create new tree structures of the DICOM object’s, as well as some modifications to the source code of this library.

5.7 PACS ARCHIVE

Regarding PACS components, it is essential to provide a way to visualise medical images, as well as to store such medical data. As previously stated on chapter 2, this type of data is compliant with the DICOM standard, encompassing images, wave videos, wave sound files, among others.

To store and access this type of data in Smart RIS, Dicooogle PACS was used. Developed by the bioinformatics group of Aveiro university, Dicooogle is an open source PACS [54], that provides a way to index and query DICOM objects on a set of repositories. This plug-in based system, follows a peer-two-peer architecture, appealing to scalability and interoperability between heterogeneous systems. Dicooogle provides storage services for a various set of modalities, as well as DICOM queries, providing a way to access the medical image.

Presented on figure 5.5, is one of the interfaces Dicooogle PACS provides to the user. Depicted in the same image is the result, of performing the query `PatientName:Smith`.



Figure 5.5: Query Performed on Dicooogle

A medical image on Dicooogle can be accessed by a web service call. To achieve this, it is necessary to execute a call to open a new URL, that follows the arrangement `dicooogle.com/#/image/SOPInstanceUID`, where `SOPInstanceUID` represents the UID of the requested DICOM data. As a result, the pretended data will be displayed on Dicooogle's interface, if a user has a valid session. On the other hand, if the user does not have a valid session, Dicooogle will prompt for credentials, making it a trustworthy solution to store sensitive patient data.

Since Smart RIS may be integrated on an institution where there is already a PACS, it is crucial to support the integration of different PACS systems. Currently, various PACS systems provide a set of web services to allow a user to access or visualise data, specifying only the `SOPInstanceUID` of the appropriate object. For this reason, Smart RIS also allows the system administrator to set up a variable in the config file that represents the base URL of the PACS system in use. This approach allows for a decoupling of Smart RIS from the PACS.

5.8 SECURITY

A robust authentication method is provided with this solution, ensuring that only trustworthy users can have access to Smart RIS. To implement the authentication mechanism, the class presented on Listing 11 was created, extending the default `Security.Authenticator`. Two methods on this class were overridden, in order to allow the creation of the following functions:

- `getUsername(Context ctx)` - Gets the current email stored in the browser cookie.
- `onUnauthorized(Context ctx)` - Triggered if a null email is found on a cookie.

As it stands, a null email on the cookie means that a user has no permission to use the system, meaning that the method `onUnauthorized()` will prompt the user to the log-in screen, where he can perform a valid authentication. By annotating a controller action with `@Security.Authenticated(Secured.class)`, is guaranteed that the logic contained in this action, will only be executed if a valid cookie session is detected in the browser.

```
public class Secured extends Security.Authenticator {

    @Override
    public String getUsername(Context ctx) {
        return ctx.session().get("email");
    }

    @Override
    public Result onUnauthorized(Context ctx) {
        return redirect(routes.MainService.login());
    }
}
```

Listing 11: Secure class

Currently, Smart RIS can communicate over a secure channel, increasing the overall system security. If configured, the previously stated security is guaranteed through the Hyper Text Transfer Protocol Secure (HTTPS) protocol provided by the Play framework.

5.9 GLOBAL SETTINGS AND INITIALISATION

By extending Play's class `GlobalSettings` [55], a developer can manage global operations and settings, regarding the application. As a result, it is possible to implement actions to take place at a given time, for example, communicating with a web service that the current application is now online.

Listing 12, serves as example for a very simple *Global.class*, that intercepts the application start event and the application stop.

```
import play.*;

public class Global extends GlobalSettings {

    public void onStart(Application app) {
        Logger.info("Application has started");
    }

    public void onStop(Application app) {
        Logger.info("Application shutdown...");
    }
}
```

Listing 12: Intercept global events

This functionality is implemented in Smart RIS, where at the start of the application, the template indexation phase takes place, allowing the integration of new and personalised templates in the system's database. In the same method interceptor, a default system administrator account is created on the system, granting a way to access the system.

CHAPTER 6

RESULTS

This chapter will provide the achieved results. To accomplish an organised explanation, it will be presented Smart RIS screenshots along with a brief explanation of the action being executed, to leverage the reader into understanding the application's workflow. To achieve an organised walkthrough of the application, it is divided into sections, according with user-roles in the system, offering a brief description of the operations such role can perform.

6.1 COMMON OPERATIONS

Usually in every role based application, there is a set of operations that may be available to every role in the system. Such set of operations is also presented in Smart RIS, and it will be described in sequence.

Presented in Figure 6.1 is the first interaction that Smart RIS provides to the user. It consists in a login screen, where a user is prompted to enter his credentials. If a successful login is performed, access to Smart RIS operation menu is granted to the user. On the other hand, if a user has a valid login session, he may choose to log out of the system and consequently, cleaning all the associated session cookies on his browser, resulting in a redirect to the login screen.

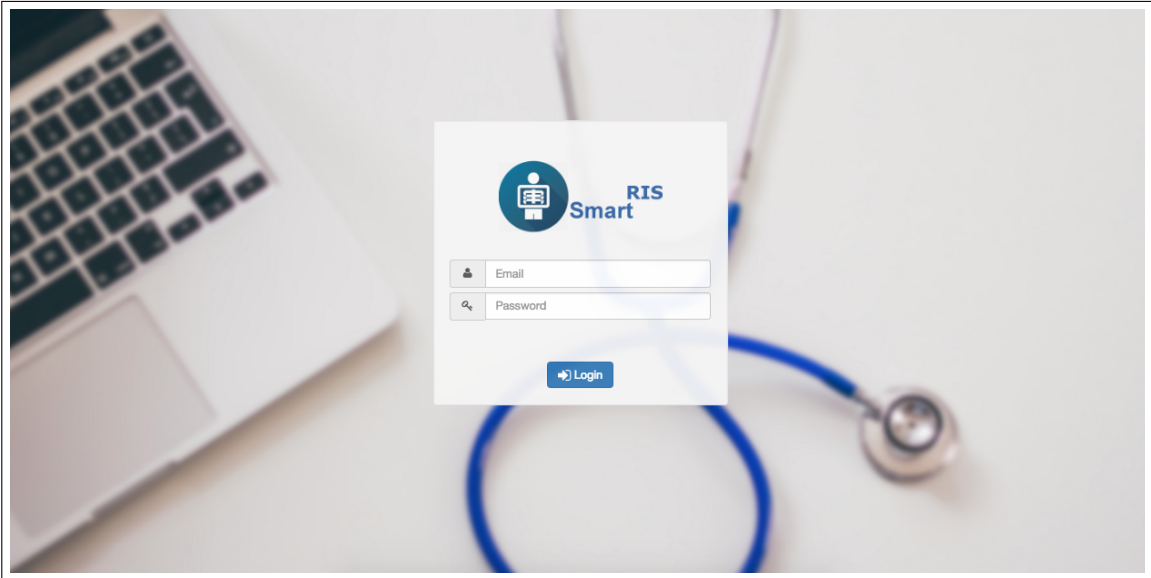


Figure 6.1: Login Interface

Every account in the system is associated with personal information, regarding the user. Considering that information like name, phone number, address, email, account password and so on, may need to be updated or modified, Smart RIS provides a way for a user to perform such amendments, regarding his personal account.

Currently to access the personal information form, a user is required to have a valid login session, from which he may navigate within the system. To access the previously stated form, it is required to select the **User Profile** operation, which is emphasised by the user icon. Depicted in Figure 6.2, is how a user can access such operation within the Smart RIS system. After selecting such operation, a form like the one presented on Figure 6.3 is displayed. From there a user may examine and update such data if so is pretended.

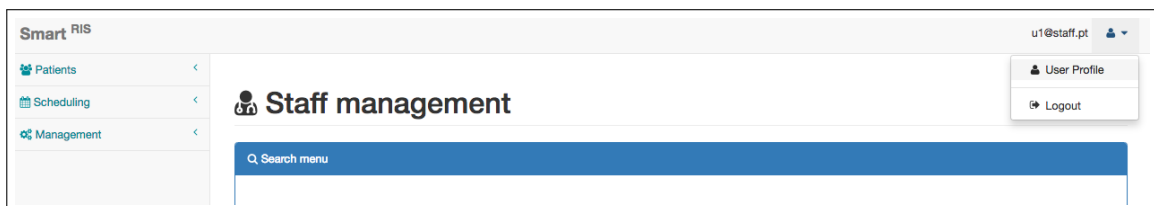


Figure 6.2: User Profile Operation

Staff management

Update member

Name
David Fontes

Mecanographic number
1

Address
3885-508 Esmoriz

Phone number
+351911911911

Email address
u1@staff.pt

Password

Re-enter Password

Figure 6.3: Personal Information Overview

6.2 SECRETARY

This role is responsible for maintaining patient records on the system, as well as managing all the schedule operations, regarding appointments and exams. As a result, such responsibilities can be shift, from the clinical staff to a secretary. Consequently, this change allows the clinical staff to only focus on providing a treatment to the patient.

As it stands, all operations that will now be presented on this subsection, required a user with a secretary role on the system, as well as a valid login session.

Figure 6.4, illustrates the form presented to a secretary, when the operation **Add patient record** is selected. From there, a secretary may enter basic information regarding the patient, such as name, identification number, address, age, gender, blood type, among others. Such information allows the system to create a thorough record, containing all the necessary data. After entering the required information, a secretary can save the record, which stores all information in the system database.

The screenshot shows a web interface for 'Patient management'. At the top, there is a blue header bar with a person icon and the text 'New patient record'. Below this, the form contains several input fields: 'Patient's full name' with a placeholder 'Enter name', 'Patient's ID' with a placeholder 'Enter patient ID', 'Patient's address' with a placeholder 'Enter address', 'Patient's phone number' with a placeholder 'Enter phone number', 'Patient's email address' with a placeholder 'Enter email', 'Date of birth' with a date picker icon, and 'Gender' with a dropdown menu showing 'Select one'.


Figure 6.4: Patient Form Overview

Every healthcare software has the need to search for patient records. As a result, a search interface is provided on Smart RIS, where a query can be performed by **Name**, **Patient id** or even **Email**. The results of such query are displayed in a table view, completed with a paging feature, along with a small property which allows specifying the number of entries to be displayed on each page, in order to ease the search for a certain record.

To access the patient search menu, a secretary must select the **Search patient** option, from the side menu, resulting in a similar form as the one on Figure 6.5. Furthermore on the previously stated figure, it is also depicted an example of a **Name** search. On this particular case, a secretary intends to find all patient records that contain **Ma** on the name field.

To achieve the best results, the **Name** search is performed recurring to regular expressions, which allow the user to obtain a wider range of values, since it will try to match the **regex** to multiple tokens, on the same name String.

When browsing the results table view, a user may select a certain row, by clicking on it, in order to visualise more information regarding the selected patient record. Such operation will display a form containing all the patient's information, which can be modified if required, or simply delete the complete record and associated data, from the system.


Patient management

Q Search menu

Search patient by:

☒ Name
☐ Email
☐ SSN

Ma

Q

Available records on the system

Show
10
entries

ID	Name	Email	Address	Phone	Gender	Age
633037760	Maria Telmo		4300-493 PORTO	+351717083904	F	27
927406272	Manuel Celestino		4300-493 PORTO	+351717083904	M	27
440773792	Renata Macedo		3800-327 AVEIRO	+351717083904	F	24
338669248	Marco Anibal		3810-315 AVEIRO	+351717083904	M	42

1

Figure 6.5: Patient Record Search

Smart RIS also provides a way to navigate throughout a calendar, that contains all information regarding event scheduling, associated with the institution. Three different views are provided by the system, annual view, monthly view and daily view. Each of the previously stated views, displays event entries in a different manner. For instance, an event entry on a calendar daily view (Figure 6.6), will display more information, such as **room** and **modality** of such event, when compared with the event presentation provided by the monthly view (Figure 6.7). Considering that an event may have three levels of urgency associated with it, all views provide a colourful feedback to the user, expressing the urgency assigned to that particular event.

As it stands, a click on a certain event will trigger a more detailed view of such event, where it is possible to update data regarding the event, as well as delete the event and all associated data, from the system.

Similar to the previous operations, a secretary can access the calendar operation, from the side menu. By default, a monthly view of the current month is provided, allowing a quick access to events that will occur in a near future.

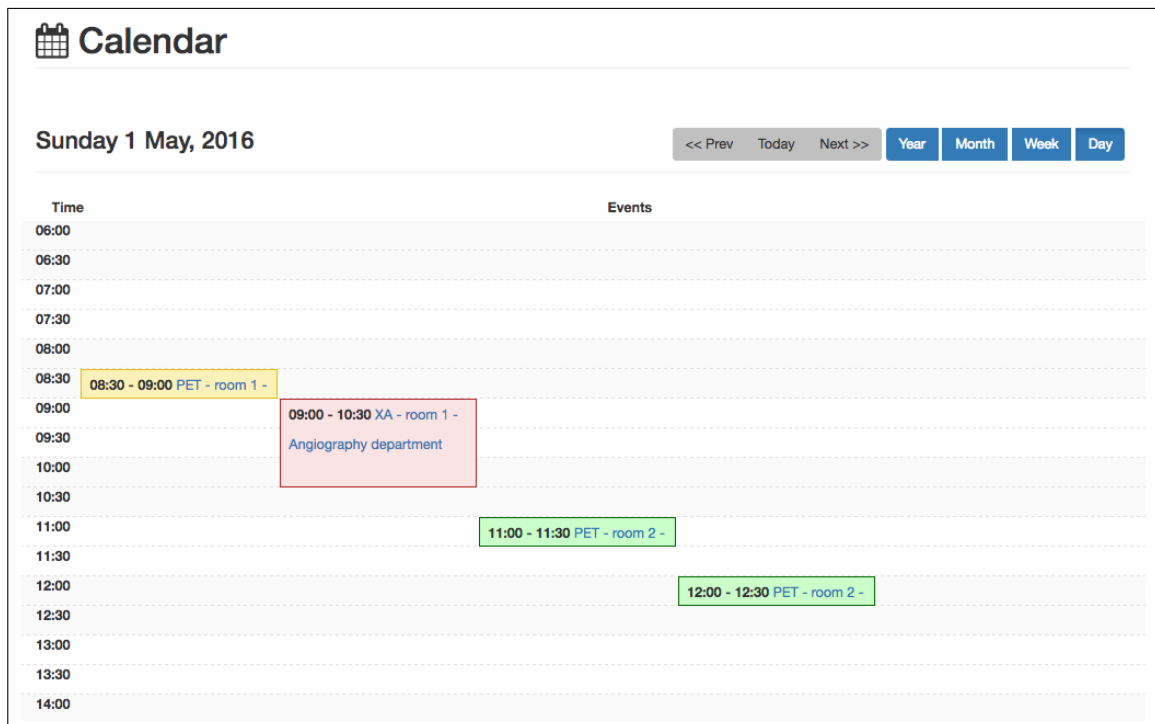


Figure 6.6: Daily View

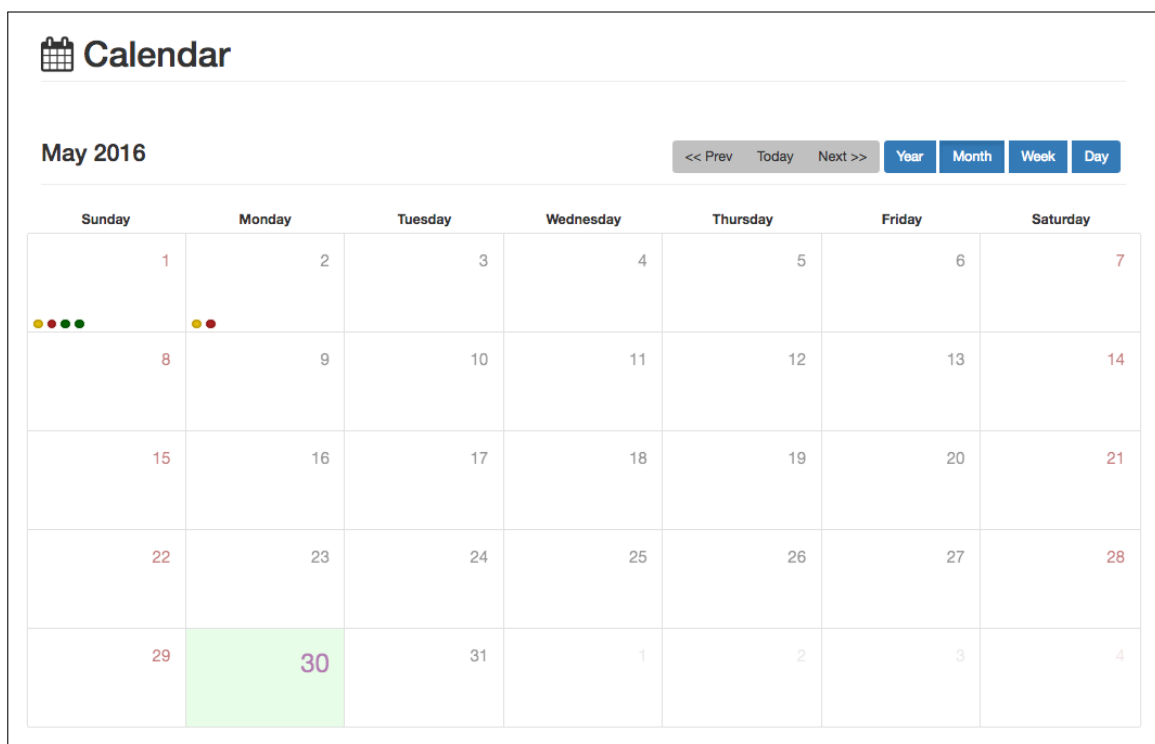


Figure 6.7: Monthly View

After selecting the calendar feature, a secretary can also schedule new events, besides the event view operation. To do so, it is required to select the pretended day and hour, by first clicking on a given

day, followed by a click on the appropriate hour, on the calendar view. These actions will result in a modal form popping on the screen, allowing to enter event related information, such as **patient id**, **physician id**, **room** where the event will take place, **urgency** of the event, among others. Illustrated on Figure 6.8 is an brief overview of the form, for scheduling a new event on the system. Currently, the date and time elements are automatically generated from the previously selected parameters. Its sole purpose is to provide a feedback to the user, in relation to the selected date and time.

The image shows a 'New appointment' modal form. The form has a title bar with 'New appointment' and a close button. The form contains the following fields:

- A date and time field showing '2016-05-03T08:30:00' with a calendar icon.
- 'Patient's ID' field with a placeholder 'Enter ID'.
- 'Patient's name' field with a placeholder.
- 'Technician staff number' field with a placeholder 'Enter MEC number'.
- 'Technician name' field with a placeholder.
- 'Reporter physician staff number' field with a placeholder 'Enter MEC number'.
- 'Reporter physician name' field with a placeholder.
- 'Event type' field with a dropdown menu showing 'Select event type'.
- 'Modality' field.

The background shows a calendar interface with a sidebar on the left displaying a time slot from 06:00 to 13:30. The main calendar area shows 'Tuesday 3 M' and navigation buttons for 'Today', 'Next >>', 'Year', and 'Month'.

Figure 6.8: New Event Form Overview

6.3 SYSTEM ADMINISTRATOR

This role was created in order to delegate the administrative burden to an entity, that has the necessary knowledge to perform the designated tasks. As a result, a System Administrator is responsible for setting up and maintaining the Smart RIS system. Since this role can be seen as the super user of a Smart RIS instance, it encompasses all the operations stated on the previous subsection 6.2, plus the ones that will now be described.

To access the Smart RIS system, a user must have a valid account registered in the system's database, as well as access to his personal credentials. As a result, it is necessary to provide a way to manage records associated with staff members. Such set of operations is available to the role System Administrator, where it can create new records, providing the necessary information, as well as to

modify or delete existing ones.

In order to create a new staff record on the system, a form like the one illustrated on Figure 6.9 is displayed to the System Administrator, if this operation is selected from the side menu. From there all information regarding the new member can be entered. Afterwards, the created record can be stored on the system. Every new staff record is created with a default password, or with one specified by the System Administrator. This can be extremely dangerous, so the new user should immediately change the temporary password, after the first login. Currently, this option can be enforced by the solution.

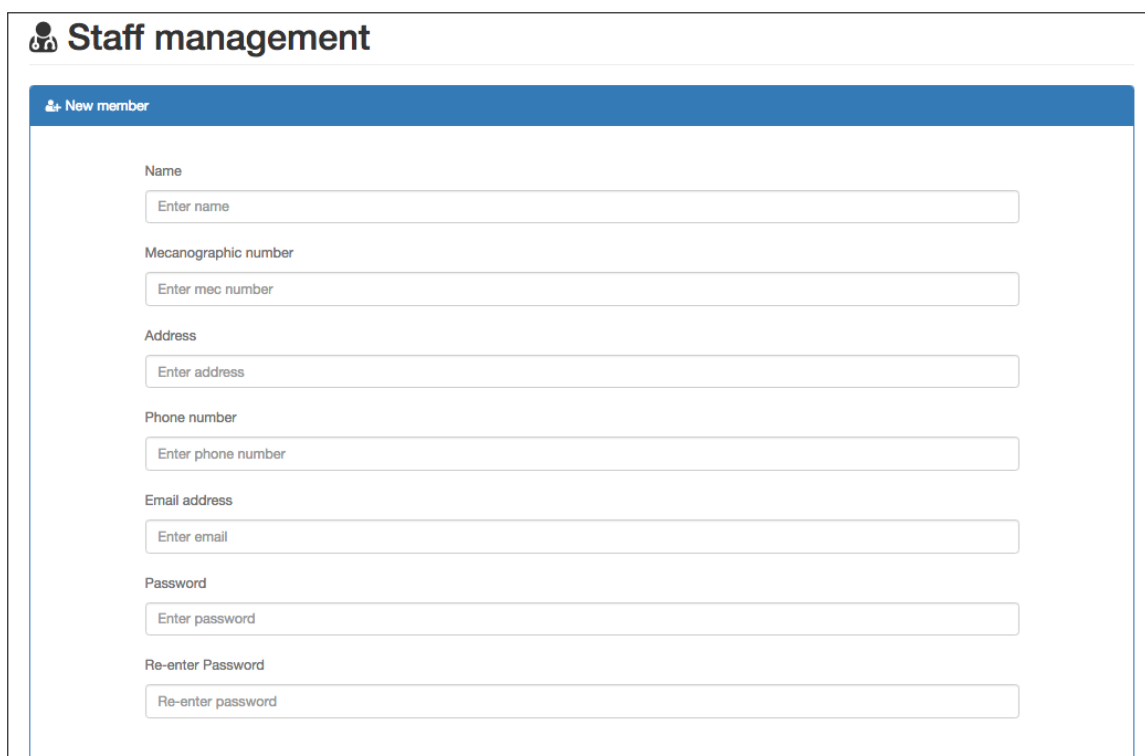

The image shows a web form titled "Staff management" with a sub-header "New member". The form contains several input fields for creating a new staff member. The fields are: "Name" (placeholder: "Enter name"), "Mecanographic number" (placeholder: "Enter mec number"), "Address" (placeholder: "Enter address"), "Phone number" (placeholder: "Enter phone number"), "Email address" (placeholder: "Enter email"), "Password" (placeholder: "Enter password"), and "Re-enter Password" (placeholder: "Re-enter password"). Each field is a simple text input box with a light blue border and a light gray placeholder text.

Figure 6.9: New Staff Record Form Overview

To encompass a full set of managing operations, a search feature is also provided for this type of records. This operation is similar to the search patient feature, using instead staff records. Currently, a search can be performed by **Name**, **Staff number** or even **Email address**. Identically to the previously stated feature, a table view is presented with the results, where a click on a certain row will display more detailed information regarding the staff record. On this new screen section, information can be updated or deleted.

Currently, the scheduling feature in Smart RIS relies on the infrastructures records, as well as modality records that exist in the system database. As a result, it was necessary to provide management for modality records, since each radiology clinic may implement a set of modalities. Depicted on Figure 6.10, is the form presented to a System Administrator, when the operation **Add modality** is performed. Afterwards, a form will appear allowing the user to enter information that characterises a modality. A modality record is then described by parameters like average duration of the procedure, brief description, name, among others. After entering all the necessary information, a System Administrator can choose to save such information in the system's database, thus resulting in the creation of a new modality record.



Modality management

[New modality](#)

Name

DICOM value

Approximate time duration


Description

Submit

Figure 6.10: New Modality Form Overview

A modality search mechanism is also provided by Smart RIS, allowing to perform a search by modality name and displaying the result into a table view, similar to the previously explained search mechanisms. Following the usability pattern previously explained, if a certain record is selected, by clicking in a table row, a more detailed view is presented to the user. From there, a System Administrator can modify an existing record or delete it.

Figure 6.11 illustrates the result of a blank search, representing a query for all modalities available in the system's database.



Modality management

[Search menu](#)

Available records on the system

Show entries

Name	DICOM Value	Description	Duration (min)
Positron Emission Tomography	PET	PET uses small amounts of radioactive materials called radiotracers, a special camera and a computer to help evaluate your organ and tissue functions	30
X-Ray Angiography	XA	Angiography is a medical imaging technique used to visualize the inside, of blood vessels and organs of the body	90

1

Figure 6.11: Modality Record Search

A clinic may have more than one facility or department. Consequently, it is necessary to provide a

way to divide responsibilities, if required. For this reason, a System Administrator can create a facility record on the system, where a form like the one pictured in Figure 6.12 is provided to the user. From there, basic information can be entered. Such data should be sufficient and clear enough, to identify a certain facility, from a universe of multiple entities.

The screenshot shows a web interface titled "Facility management" with a sub-header "New facility". The form contains four input fields: "Name" (placeholder: "Enter facility name"), "Phone number" (placeholder: "Enter facility phone number"), "Address" (placeholder: "Enter facility address"), and "Description" (placeholder: "Enter facility description"). A blue "Submit" button is located at the bottom right of the form.

Figure 6.12: New Facility Form Overview

To better schedule and manage the available resources, a System Administrator can also create a clinic room record on the system. Such record is then associated with a facility and a modality, thus allowing for a better scheduling of operations within the system, since a modality has an average duration associated with it. When such operation is selected from the side menu, a form like the one depicted in Figure 6.13, is provided to the user where basic information can be entered.

The screenshot shows a web interface titled "Facility management" with a sub-header "New room". The form contains five input fields: "Select a facility" (dropdown menu showing "Angiography department"), "Select a modality" (dropdown menu showing "PET"), "Name" (placeholder: "Enter room name"), "Room number" (placeholder: "Optional"), and "Description" (placeholder: "Enter room description"). A blue "Submit" button is located at the bottom right of the form.

Figure 6.13: New Room Form

A search facility and search room mechanisms are also provided to the role System Administrator. These mechanisms are similar to the previously stated ones, where a **Name** query can be performed, in order to find the intended record. Consequently, the search results are also displayed in a table view similar to the previously presented Figure 6.11, using infrastructure records instead. Following the same approach, the selection of a certain record will trigger a new view with detailed information regarding the selected record, that can be updated or simply deleted.

6.4 TECHNICIAN

This role includes all operations regarding a technician, on the Smart RIS system. Such role is responsible for performing exam procedures on a patient, as well as fulfilling the associated report. A technician can also navigate throughout patient records, in addition to being able to consult previous events related to the patient.

Currently, Smart RIS supports the integration of multiple templates for clinical reports, by dragging a set of HTML files to a specific folder. Consequently, when a new indexation is performed, all templates will be available to the system users.

As it stands, a technician usually only works with a certain group of modalities and associated clinical reports. In order to ease the search for a certain template, a technician may specify a subset from the universe of templates, to be his default set, thus reducing the list presented to the user in regular operations. To access this operation, a technician must navigate to the settings menu, illustrated on Figure 6.14, where a search interface is provided, allowing to perform a search by **Name** on the universe of available templates. Subsequently, a user may select as many templates as intended, in order to create his subset.

Presented on Figure 6.15 is the result of performing a search for **CT** templates, along with the obtained results and a group of selected clinical templates. If a technician chooses to save the newly added settings, all the selected templates will be associated with his profile, as the available group of clinical templates. On the same Figure is also depicted, the **Saved settings** tab. From there a technician may consult all the templates that make part of his universe of templates.

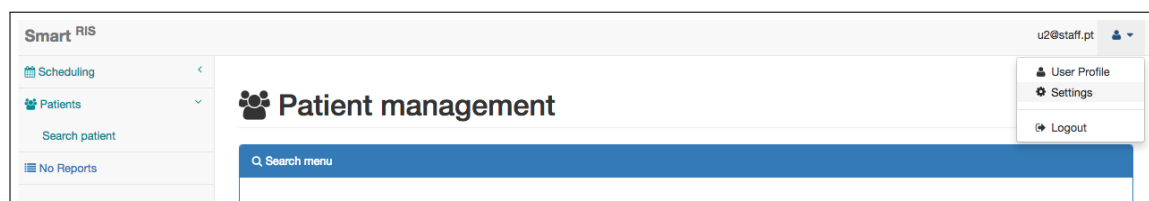


Figure 6.14: Settings Menu

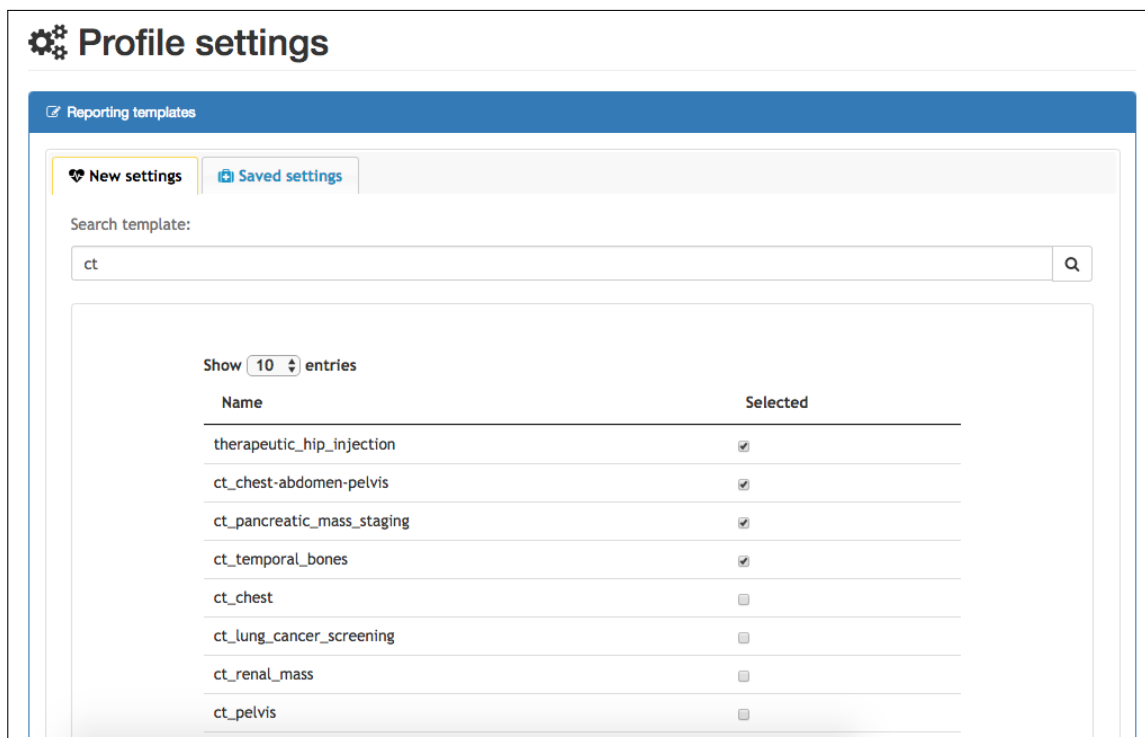


Figure 6.15: Menu for Selecting a Subset of Templates

Calendar views are also provided to this role, allowing to show events related to the currently logged-in technician. He can browse the calendar and, by selecting a certain day, all scheduled events are available to consult. This feature also allows a user to browse past events, for comparative analysis. After selecting this operation from the side menu, a calendar view is presented to the user, where he can browse throughout the calendar, similar to the feature described in subsection 6.2. Figure 6.16 shows a worklist for a certain day.

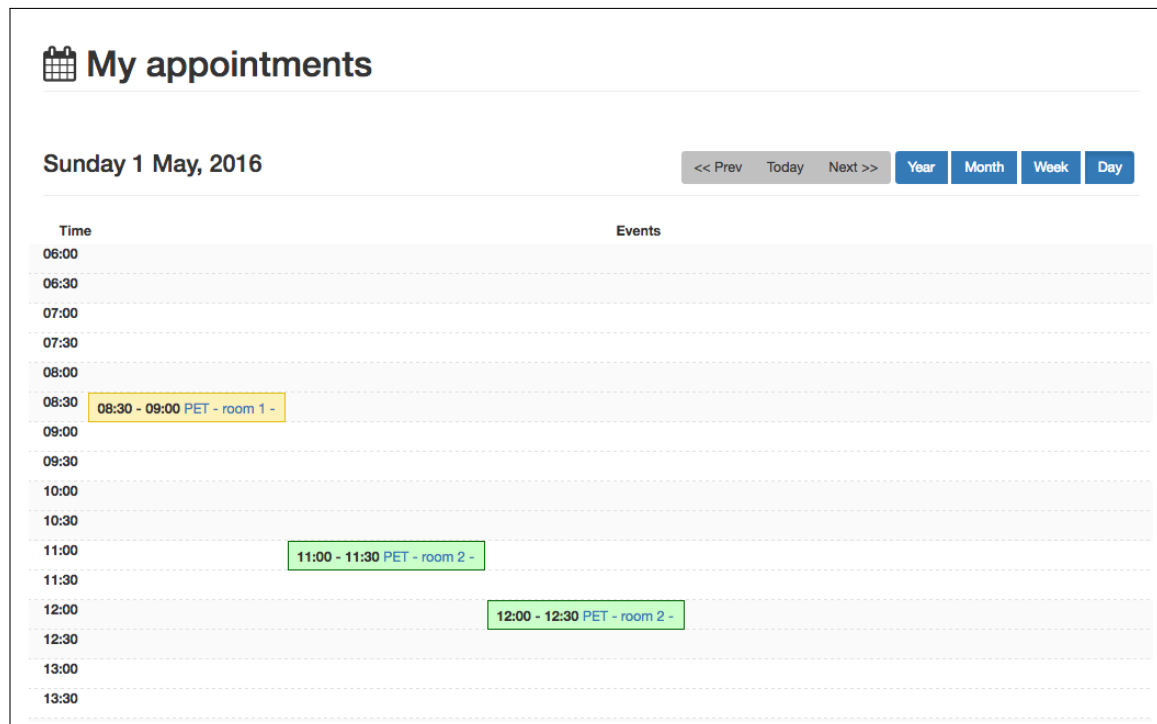


Figure 6.16: Technician's Worklist

The patient search mechanism, is also available to the technician user role. The same search parameters are available, and the results are also displayed in a table view, similar to the previously presented Figure 6.5.

If a record is selected, a tab navigation menu is presented to the technician, containing two distinct tab views. The first one, i.e. **Patient info**, shows a form similar to the one presented in the Figure 6.4, where the technician can consult patient personal data. The second tab presented a list of previous events regarding the selected patient. Figure 6.17 shows an possible event history, associated with a patient record.

Patient management

Patient info

Previous events

Previous events list

Show 10 entries

Event type	Modality	Urgency	Room	Patient ID	Date	Attachment
Exam	X-Ray Angiography	event-important	room 1 - Angiography department	934546880	April 30th 2016, 6:00:00 am	Exam
Appointment	Positron Emission Tomography	event-warning	room 1 - PET	934546880	May 1st 2016, 7:30:00 am	
Exam	X-Ray Angiography	event-important	room 1 - Angiography department	934546880	May 1st 2016, 8:00:00 am	Exam

1

Figure 6.17: Patient's Previous Events

Previous events list of results presents a summary and, by clicking on a record a more detailed view, show all information regarding that.

Regarding an event, it is possible to consult the produced **Exam** data or **Clinical report**, if such data is available. To perform this operation, a technician must select the **Exam** feature, from the detailed event view or from the previous events table, depicted in Figure 6.17. As a result, Smart RIS will open the examination image, in the PACS system viewer. Figure 6.18 illustrates an example of an exam being visualised with Dicoogle PACS.

Currently, Dicoogle PACS requires a valid login session to visualise medical data, prompting the user to enter his credentials, if one is not found.

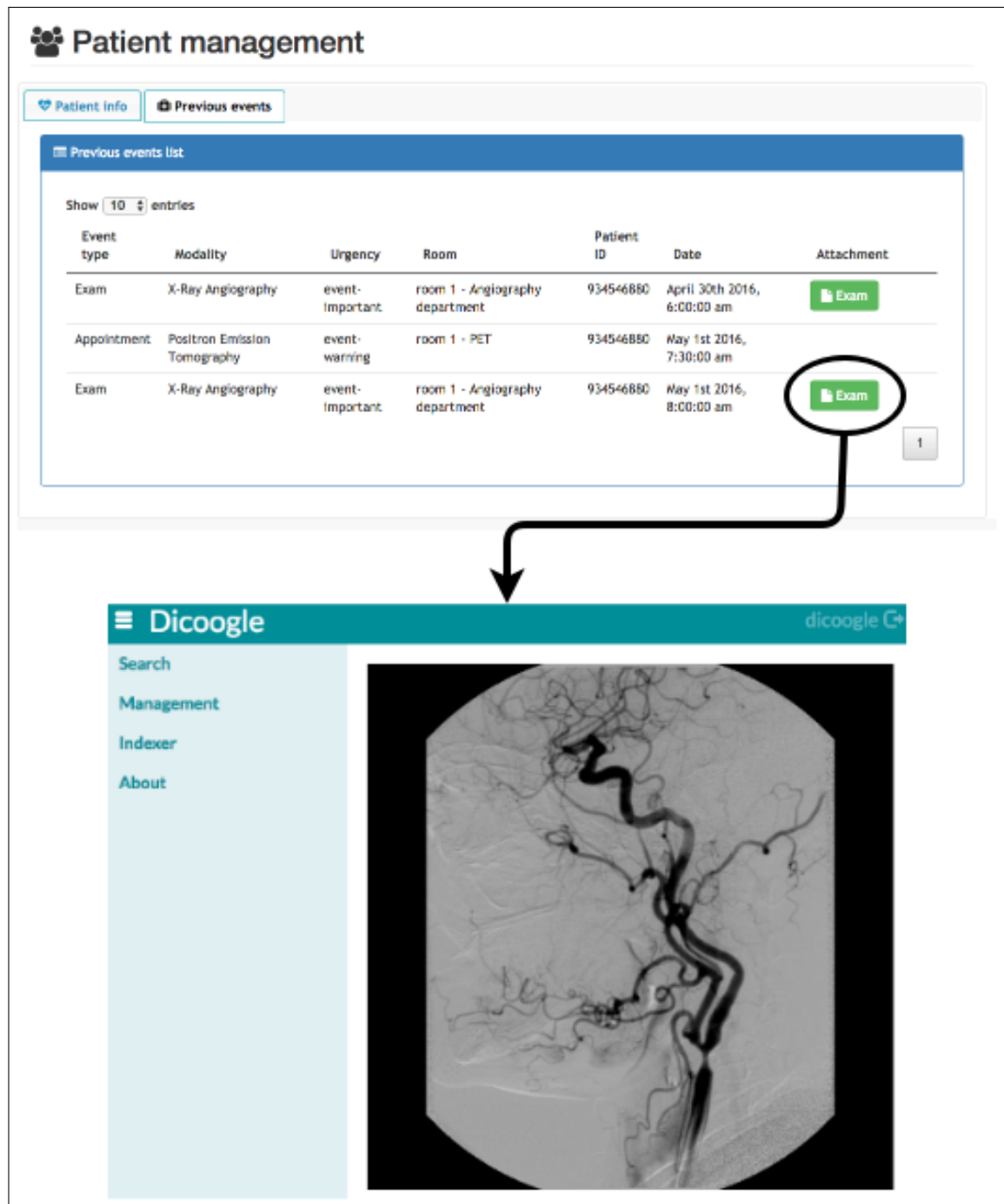


Figure 6.18: Exam Data Displayed with Dicooogle

When consulting a clinical report, Smart RIS provides a view similar to the one illustrated in Figure 6.19. From there, a technician may play existing voice annotation, associated with the report, as well as export the selected report in PDF or DICOM SR format. Clinical reports available for consulting are validated since a group of physicians have already sign it. For this reason, the technician cannot modify the displayed report, as all inputs are blocked by the system.

Reporting

Event number: 1

Modality: X-Ray Angiography

Event type: Exam

Physician: Daniel Oliveira mec: 2

Technician: Luís Duarte mec: 4

Reviewed by: Daniel Oliveira mec: 2

Notes:

Patient: Eugénio Gustavo Patient id: 934546880

PDF

DICOMSR

Procedure

Computed tomography (CT) of the abdomen and pelvis with intravenous and oral contrast performed May 30th, 2016

Clinical information

26 year-old male with history of cancer and new onset abdominal pain.

Comparison

None.

Findings

Lung bases: No pulmonary nodules or evidence of pneumonia. Cardiac: Base of heart is within normal limits. No pericardial effusion.

Figure 6.19: Clinical Report

After a technician logs into the system, Smart RIS automatically checks if the user has pending reports to fulfil, associated with his account. If that is the case, a small red alert, similar to the one depicted in Figure 6.20, is displayed to the user. A click on this alert will trigger the system, into displaying a list of events that have a pending report, for the current user to fulfil. Next, a complete list of events is displayed in a table view, where a group of important parameters, such as date, urgency, modality and others, can be intuitively displayed to the technician (Figure 6.21).

Smart RIS

u4@staff.pt


Scheduling


Patients

Reporting





My appointments

Figure 6.20: Pending Reports Notification

 **Report**

 Event selection

Show 10 entries

Event type	Modality	Urgency	Room	Patient ID	Date	Attachment
Exam	X-Ray Angiography	event-important	room 1 - Angiography department	934546880	April 30th 2016, 6:00:00 am	 Exam
Appointment	Positron Emission Tomography	event-warning	room 1 - PET	934546880	May 1st 2016, 7:30:00 am	
Appointment	Positron Emission Tomography	event-success	room 2 - PET	498312320	May 1st 2016, 11:00:00 am	
Exam	Positron Emission Tomography	event-success	room 2 - PET	498312320	May 1st 2016, 10:00:00 am	 Exam
Exam	X-Ray Angiography	event-warning	room 1 - Angiography department	498312320	May 2nd 2016, 10:30:00 am	 Exam
Exam	Positron Emission Tomography	event-important	room 1 - PET	498312320	May 2nd 2016, 4:00:00 pm	 Exam

1

Figure 6.21: Events with a Pending Report

To write a report, a technician must first navigate to the report tab. From there he can perform a search by template name, on his subset of reporting templates. Afterwards, a click will load the selected template (Figure 6.22), thus providing the inputs to be written. The report's header is automatically generated, with information regarding the event, and consists of the first rectangle presented in the above-stated image. A technician can also associate voice annotations with the report, in order to enhance the final appreciation of the stated matter.

A technician may select a list of physicians to review and sign the report. This operation is available on tab **Refer physician**, that will present a table view similar to the one illustrated in Figure 6.23.

Reporting

Event number: 4

Modality: Positron Emission Tomography

Event type: Appointment

Physician: Daniel Oliveira mec: 2

Technician: Luís Duarte mec: 4

Reviewed by:

Notes:

Patient: Paulo Rúben Patient id: 498312320

Procedure

Clinical information

Comparison

None.

Findings

Figure 6.22: Clinic Report Presented to the Technician

Select reviewers

Mec #	Name	Email	Select as reviewer
2	Daniel Oliveira	u2@staff.pt	<input checked="" type="checkbox"/>
5	Miguel Lage Valério	u5@staff.pt	<input checked="" type="checkbox"/>
6	Vasco Santos	u6@staff.pt	<input type="checkbox"/>

Save

Figure 6.23: Referring Table

6.5 PHYSICIAN

This role encompasses all operations regarding a doctor, within the Smart RIS system. It is responsible for maintaining event reports, as well as reviewing and validating clinical reports, produced by a technician. It includes all the technician's operations stated on subsection 6.4, in addition to the signing clinical reports feature, that will now be described.

As previously stated, a physician may be referred by a technician, to review a clinical report. If that is the case, a physician may open the assigned report, and if necessary, perform amendments on it.

When all information contained in the selected report is correct, the assigned physician may associate his name with it, by sign it.

Figure 6.24 illustrates the view provided to a doctor, for report reviewing. From there, a physician may modify any parameters, play voice annotations, as well as export the selected report to his local computer in PDF or DICOM SR format. Currently, this operation is provided with the same tab view, as the one presented in Figure 6.22, where the **Reporting** tab contains the fulfilled report. As a result, the assigned physician has access to important information such as medical images and patient's history, that may leverage an unbiased review of the produced report.

The reviewing method, described in the previous paragraph provides an extra level for ensuring, that only correct and reliable reports are stored in the system's database. Consequently, a clinical report is only available in the patient's history when all assigned physicians had analysed and signed the report.

Figure 6.24: Clinical Report Reviewing

6.6 OTHER ASPECTS

The choice of the technologies used in the development of this project was an importing aspect, since a solution like Smart RIS needs to be **portable**. This means that all the necessary environment to deploy the service, should not be tied to one computational system. As a result, the set of services that support Smart RIS operations can be deployed in a production system, with no regard for the hardware or operating system contained in it. Currently, the same principle applies to a client that uses the Smart RIS solution, where to access the platform a user only requires a web browser. As a result of a reactive interface design, this feature is yet more noticeable in **mobile** systems. Figure

6.25 and 6.26 shows the ability to access the platform, using an IOS device, as well as an Android smartphone respectively.

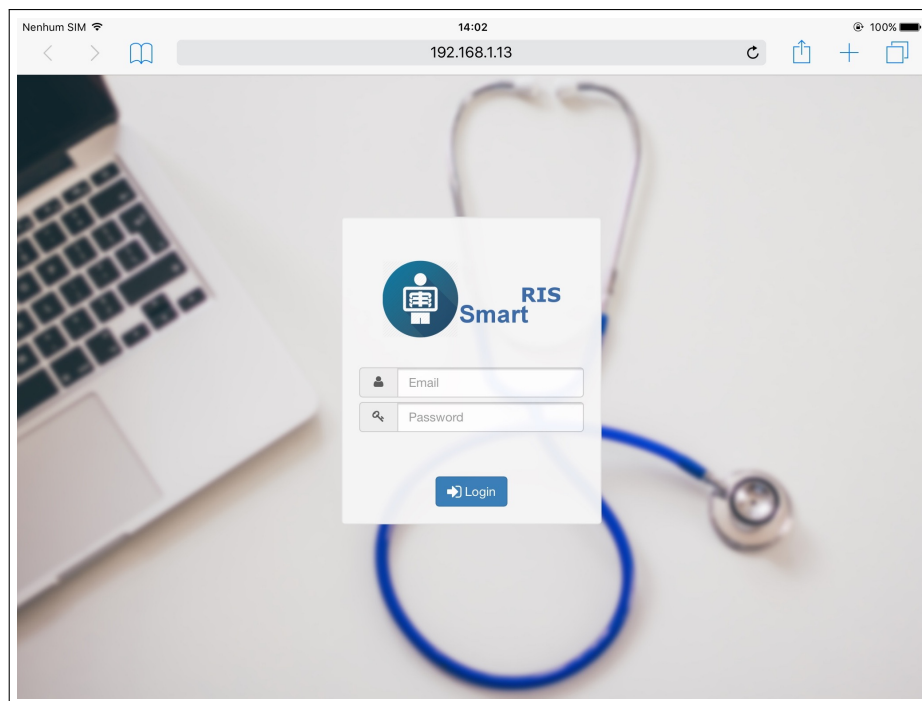


Figure 6.25: IOS tablet accessing Smart RIS

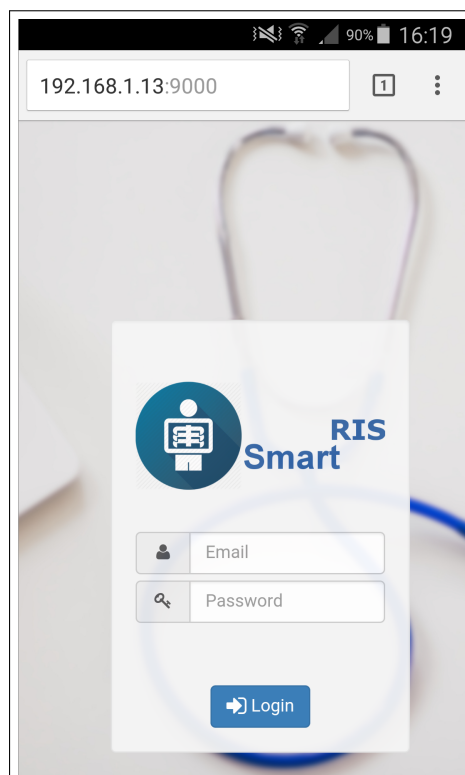


Figure 6.26: Android smartphone accessing Smart RIS

CONCLUSION

In this chapter, it is presented a brief overview of the obtained result, as well as problems that were addressed with the development of Smart RIS. It is also presented some advantages achieved with Smart RIS solution. Finally, some future work is also suggested.

7.1 CONCLUSION

Nowadays, having medical imaging capabilities is a crucial feature for every healthcare institution, since it can greatly leverage a physician into providing a correct diagnosis and treatment.

With the current massification of medical imaging modalities that are interoperable though DICOM, great volumes of data can be generated, shared and distributed in an imaging department. As a result, there is a constant search for efficient methods to manage such volume of data and workflows. PACS solutions such as Dicooogle PACS can solve part of the previously stated problems, by providing an organised environment for DICOM data management, but there is a gap between a PACS and a system capable of managing all operations in a radiology clinic. To minimise such gap, a RIS is usually deployed in the institution, providing a usable interface to the physician, as well as a set of features regarding the workflow in a radiology department.

Current open source solutions offer a set of managing features, but these features are focused on HIS operations rather than a radiology environment. These solutions follow a modular pattern and allow modifications to the source code, in order to adapt the final solution to the user needs. Since the main objective of such systems is to manage HIS operations, there is a large list of modules that is not intended to be used in a radiology environment, leading to decoupling then to the main system. As a result, the final product can become unstable, thus not being suitable to manage all operations regarding a radiology institution.

This thesis proposes an information system that was developed with free or open source technologies, it is focused in radiology departments management and overpasses the limitations of previously described systems. Smart RIS is complete with a set of features to provide management of administrative operations, as well as a role-based segregation of responsibilities. It also provides an intuitive calendar where department events can be programmed. The scheduler supports multiple facilities, rooms and modalities. Moreover, the system makes easy the establishment of relations between previous resources and its adjustment to the institution workflows.

Smart RIS can integrate any PACS and provides a set of rich reporting functionalities that are only available in some commercial solutions. It was designed to support distinct reports templates that can be easily added to an installation that is already in production, without requiring software code changes. Moreover, those reports can be exported to common PDF format and also to the standard DICOM SR. This last aspect is of extreme importance to ensure the interoperability with third systems and also the integration of reports with respective images in the PACS archive.

Smart RIS is a full web application that can run on any platform, including the mobile ones, only requiring a common web browser to access the system. The solution is supported by a robust and scalable backend that exposes all operations through web services. Moreover, it provides an abstraction for data persistence that makes possible to select distinct database engines according to with client requirements.

The design of graphical user interface was subject of special concern, aiming to achieve a simple and clean look, not overwhelming the user with too much information, as it happens with other previously tested solutions. As a result, all actions a user may perform are accompanied icons, providing visual feedback of the area of interest regarding the operation. In addition to this, every action is at most, at a distance of two mouse clicks, making a quick and intuitive to use interface.

7.2 FUTURE WORK

The modular approach implemented in Smart RIS, allows to evolve the solution in practically any direction. At the end of this project, some features have been identified as potential evolutions in future work:

- The ability to support voice commands. The final version of Smart RIS supports voice annotations on clinical reports, but does not support menu navigation or command execution, using only a user's voice. This feature will allow a user to better manage his diagnoses in conjunction with the Smart RIS navigation if so is needed.
- Implement mechanisms for a seamless integration with an existing HIS. Smart RIS focus was to provide an all in one solution for small/medium institutions, supporting administrative tasks and radiology operations. Since the necessity of this solution to be integrated with a HIS may arise, it is fundamental to have some kind of integration profile.

REFERENCES

- [1] X. W. Gao, Y. Qian, and R. Hui, “The state of the art of medical imaging technology: from creation to archive and back.”, *The open medical informatics journal*, vol. 5 Suppl 1, 2011.
- [2] B. Blazona and M. Koncar, “Hl7 and dicom based integration of radiology departments with healthcare enterprise information systems”, *International Journal of Medical Informatics*, vol. 76, 2007.
- [3] P. Miller, *Interoperability. what is it and why should i want it?*, 2000. [Online]. Available: <http://www.ariadne.ac.uk/issue24/interoperability>.
- [4] I. the Healthcare Enterprise, “Integrating the healthcare enterprise - radiology”, 1998. [Online]. Available: <http://www.ihe.net/Radiology/>.
- [5] G. Weatherburn, S. Bryan, A. Nicholas, and R. Cocks, “The effect of a picture archiving and communications system (pacs) on diagnostic performance in the accident and emergency department”, *Emergency Medicine Journal*, vol. 17, May 2000.
- [6] *Iaea radiation protection of patients - radiography*. [Online]. Available: https://rpop.iaea.org/RPOP/RPoP/Content/InformationFor/HealthProfessionals/1_Radiology/Radiography.htm.
- [7] S. S. Boochever, “His/ris/pacs integration: getting to the gold standard.”, *Radiology management*, vol. 26, no. June, 2004.
- [8] J. Watkins, “A hospital-wide picture archiving and communication system (pacs): the views of users and providers of the radiology service at hammersmith hospital”, *European Journal of Radiology*, vol. 32, no. 2, 1999.
- [9] O. S. Pianykh, “Digital imaging and communications in medicine (dicom): A practical introduction and survival guide”, Springer Science & Business Media, 2009.
- [10] P. Mildengerger, M. Eichelberg, and E. Martin, “Introduction to the dicom standard”, *European Radiology*, vol. 12, no. 4, Apr. 2002.
- [11] H. K. Huang, *PACS and imaging informatics basic principles and applications*. Wiley-Blackwell, 2010.
- [12] NEMA, *Digital Imaging and Communications in Medicine (DICOM) Part 3: Information Object Definitions*. National Electrical Manufacturers Association, 2011.
- [13] “A very basic dicom introduction - dcm4che-2.x - confluence”, 2015. [Online]. Available: <http://www.dcm4che.org/confluence/display/d2/A+Very+Basic+DICOM+Introduction>.
- [14] NEMA, *Digital Imaging and Communications in Medicine (DICOM) Part 3: Data structures and encoding*. National Electrical Manufacturers Association, 2011.

- [15] —, *Digital Imaging and Communications in Medicine (DICOM) Part 3: Message exchange*. National Electrical Manufacturers Association, 2011.
- [16] —, *Digital Imaging and Communications in Medicine (DICOM) Part 18: Web Services*. National Electrical Manufacturers Association, 2011.
- [17] —, *Digital Imaging and Communications in Medicine (DICOM) supplement 161: WADO-RS*. National Electrical Manufacturers Association, 2011.
- [18] —, *Digital Imaging and Communications in Medicine (DICOM) supplement 163: STOW-RS*. National Electrical Manufacturers Association, 2011.
- [19] —, *Digital Imaging and Communications in Medicine (DICOM) supplement 166: QIDO-RS*. National Electrical Manufacturers Association, 2011.
- [20] D. A. Clunie, “Dicom structured reporting: overview and chracteristics”, 2000.
- [21] HL7, *DICOM SR Basic Diagnostic Imaging Report to HL7 CDA Release2 Diagnostic Imaging Report Mapping*. 2007.
- [22] J. Hope, “Open source licensing”, *Intellectual Property Management in Health and Agricultural Innovation: A Handbook of Best Practices*, 2007.
- [23] GnuHealth, *Gnu health*, 2016. [Online]. Available: <http://health.gnu.org/>.
- [24] G. H. community, *Gnu health module installation documents*, 2016. [Online]. Available: https://en.wikibooks.org/wiki/GNU_Health/Installation.
- [25] OpenMrs, *Open mrs*, 2016. [Online]. Available: <http://openmrs.org/>.
- [26] JavaMelody, *Javamelody*, 2016. [Online]. Available: <https://github.com/javamelody/javamelody/wiki>.
- [27] NagiosEnterprises, *Nagios*, 2016. [Online]. Available: <https://www.nagios.org/>.
- [28] *Utf-8*, 2016. [Online]. Available: http://www.w3schools.com/charsets/ref_html_utf8.asp.
- [29] *Icd-10*, 2016. [Online]. Available: <http://www.who.int/classifications/icd/en/>.
- [30] “Snomed ct”, 2016. [Online]. Available: <http://www.ihtsdo.org/snomed-ct>.
- [31] OpenEmr, *Open emr*, 2016. [Online]. Available: <http://www.open-emr.org/>.
- [32] Clearcanvas, *Clear canvas*, 2016. [Online]. Available: <https://www.clearcanvas.ca/>.
- [33] *Clear canvas source code repository*, 2016. [Online]. Available: <https://github.com/ClearCanvas/ClearCanvas>.
- [34] C. Health, *Carestream vue ris*, 2016. [Online]. Available: <http://www.fujifilmusa.com/products/medical/radiology/ris/>.
- [35] Fujifilm, *Fujifilm synapse ris*, 2016. [Online]. Available: <https://www.carestream.com/vue-ris.html>.
- [36] A. Healthcare, *Agfa impax ris*, 2016. [Online]. Available: http://www.agfahealthcare.com/usa/en/main/products_services/ris_pacs_reporting/radiology_information_systems/impax_ris_for_hospitals.jsp.
- [37] D. L. Weiss and C. P. Langlotz, “Structured reporting: patient care enhancement or productivity nightmare? ”, *Radiology*, Dec. 2008.
- [38] C. L. Barcellos, A. von Wangenheim, and R. Andrade, “A reliable approach for applying dicom structured reporting in a large-scale telemedicine network”, in *2011 24th International Symposium on Computer-Based Medical Systems (CBMS)*, IEEE, Jun. 2011.

- [39] *Play framework*, 2016. [Online]. Available: <https://playframework.com/documentation/2.5.x/Home>.
- [40] K. Sharan, “Model-view-controller pattern”, in *Learn JavaFX 8*, Berkeley, CA: Apress, 2015.
- [41] *Jquery*, 2016. [Online]. Available: <https://jquery.com/>.
- [42] *Handlebars*, 2016. [Online]. Available: <http://handlebarsjs.com/>.
- [43] *Webjars*, 2016. [Online]. Available: <http://www.webjars.org/>.
- [44] T. Edwards, *Sweet alert*, 2016. [Online]. Available: <http://t4t5.github.io/sweetalert/>.
- [45] Serhioromano, *Bootstrap calendar*, 2016. [Online]. Available: <https://github.com/Serhioromano/bootstrap-calendar>.
- [46] Ebean, *Ebean*, 2016. [Online]. Available: <http://ebean-orm.github.io/>.
- [47] D. Yang, *Java Persistence with Jpa 2.1*. Outskirts Press, 2013.
- [48] H. Böck, “Java persistence api”, in *The Definitive Guide to NetBeans™ Platform 7*, Berkeley, CA: Apress, 2012.
- [49] *Play framework router component*, 2016. [Online]. Available: <https://www.playframework.com/documentation/2.4.x/ScalaRouting>.
- [50] *Rsna’s reporting initiative*, 2016. [Online]. Available: <http://www.radreport.org/>.
- [51] P. Matos, L. A. Bastião, T. M. Godinho, and C. Costa, “A dynamic approach to support interoperability for medical reports using dicom sr”, vol. 32, 2016.
- [52] NEMA, *Digital Imaging and Communications in Medicine (DICOM) Part 16: Content Mapping Resource*. National Electrical Manufacturers Association, 2011.
- [53] —, *Digital Imaging and Communications in Medicine (DICOM) supplement 79: Breast Imaging Report Templates*. National Electrical Manufacturers Association, 2011.
- [54] C. Costa, C. Ferreira, L. Bastião, L. Ribeiro, A. Silva, and J. L. Oliveira, “Dicoogle - an open source peer-to-peer pacs.”, *Journal of digital imaging*, vol. 24, no. 5, Oct. 2011.
- [55] *Global settings class documentation*, 2016. [Online]. Available: <https://www.playframework.com/documentation/2.4.x/GlobalSettings>.