

Charles University, Prague
Faculty of Mathematics and Physics

Master Thesis



Petr Kalina

*Design and implementation of PAC
system*

Department of Software Engineering

Advisor: RNDr. Ing. Jiří Peterka

Study Program: Computer Science

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze dne

Petr Kalina

.....

Table of Contents

1	<i>Project Outline</i>	9
2	<i>The Specification</i>	12
3	<i>Organization of this work</i>	15
4	<i>PACS and Imaging Informatics</i>	16
4.1	<i>What is a PACS</i>	16
4.2	<i>PACS Components and Workflow</i>	17
4.3	<i>Industrial Standards concerning PACS</i>	20
4.3.1	<i>DICOM</i>	20
4.3.2	<i>HL7</i>	28
4.3.3	<i>IHE</i>	30
5	<i>Basic design choices</i>	34
5.1	<i>Environment</i>	34
5.2	<i>PACS model</i>	34
5.3	<i>Image archive platform</i>	37
5.4	<i>Reliability</i>	40
5.4.1	<i>DB dump (hot-copy/mirror) and rsync</i>	41
5.4.2	<i>Distributed data storage</i>	42
5.4.3	<i>HSM – storage of tarballs on tape</i>	44
5.4.4	<i>DICOM forwarding</i>	46
5.5	<i>Integration</i>	47
5.5.1	<i>Internal Interfaces</i>	48
5.5.2	<i>Workflow Management Interface</i>	48
5.5.3	<i>Interface to access image data from within and outside of the hospital</i>	50
5.6	<i>Platform</i>	53
5.6.1	<i>Network</i>	53
5.6.2	<i>Hardware, Operating System, File System and RAID</i>	53
5.6.3	<i>The Database</i>	54
6	<i>Implementation</i>	56
6.1	<i>Implementation overview</i>	56

6.2	<i>Image Archive Implementation</i>	56
6.2.1	<i>Dcm4chee image archive architecture</i>	57
6.2.2	<i>Dcm4chee data storage system</i>	63
6.3	<i>Reliability Implementation</i>	64
6.3.1	<i>Hot-swap and replication</i>	66
6.3.2	<i>Reliability emergency plans</i>	68
6.4	<i>Integration Implementation</i>	68
6.4.1	<i>Internal Interfaces and Scheduled Workflow Support</i>	69
6.4.2	<i>Interface to access image data from within and outside of the hospital</i>	71
6.4.3	<i>DcmMIT toolkit</i>	72
6.5	<i>Maintenance and Administration</i>	75
7	<i>Summary</i>	77
8	<i>Future</i>	80
8.1	<i>Back-end logic improvements</i>	80
8.2	<i>Integration Improvements</i>	80
8.3	<i>Image Display Improvements and Other tasks</i>	81
9	<i>References</i>	82

List of Figures

<i>Figure 1.1 – Hindering factors for adoption of new healthcare technologies in Europe</i>	10
<i>Figure 2.1 – Basic concept of PACS work</i>	12
<i>Figure 4.1 – Old and new Diagnostic Workstation [HUA04]</i>	16
<i>Figure 4.1 – Siemens Biograph PET CT Scanner</i>	17
<i>Figure 4.3 – Display Workstation at the department with adjacent RIS Workstation</i>	18
<i>Figure 4.4 – Image Archive Server of the department</i>	18
<i>Figure 4.5 – DICOM Standard Architecture [DCM]</i>	22
<i>Figure 4.6 – DICOM Image IOD – an example of a Composite DICOM object [DCMCOO]</i> 23	
<i>Figure 4.7 – DICOM Dataset Example</i>	24
<i>Figure 4.8 – Dataset Encoding Example</i>	24
<i>Figure 4.9 – Example Conformance Statement [PHILCS]</i>	28
<i>Figure 4.10 – Example HL7 ORM^001 (Order Entry) Message</i>	28

<i>Figure 4.11 – IHE Integration Statement Example [DCM4CHE]</i>	32
<i>Figure 5.1 – Standalone PACS Model</i>	35
<i>Figure 5.2 – Client-Server PACS Model</i>	36
<i>Figure 5.3 – Web PACS Model</i>	37
<i>Figure 5.4 – Reliability via Database Dump and Rsync</i>	42
<i>Figure 5.5 – Reliability via Database Dump and Distributed Filesystem or Intelligent I/O layer</i>	43
<i>Figure 5.6 – Reliability via HSM module</i>	45
<i>Figure 5.7 - Reliability via DICOM forwarding</i>	46
<i>Figure 5.8 – PACS Integration Interfaces</i>	48
<i>Figure 6.1 – Implementation Overview</i>	56
<i>Figure 6.2 – dcm4chee Database Model</i>	58
<i>Figure 6.3 – XMLElement properties managed via the JBoss JMX Console</i>	59
<i>Figure 6.3 – dcm4chee Services in the JBoss Application Server’s JMX Console</i>	60
<i>Figure 6.4 – dcm4chee web interface</i>	62
<i>Figure 6.5 – Individual servers and their location</i>	65
<i>Figure 6.6 – Example output of the backup checking routine</i>	66
<i>Figure 7.1 – Implementation Summary</i>	77

Název práce: *Návrh a Implementace systému PAC*

Autor: *Petr Kalina*

Email autora: petr.kalina@jlabs.cz

Katedra (ústav): *Katedra softwarového inženýrství*

Vedoucí diplomové práce: *RNDr. Ing. Jiří Peterka*

E-mail vedoucího: jiri@peterka.cz

Abstrakt:

PAC (Picture Archiving and Communication) systémy slouží k archivaci a správě digitálních obrazových dat produkovaných medicínskými přístroji na radiologických a jiných odděleních. Efektivní implementace PACS musí řešit řadu netriviálních otázek - jak jsou data reprezentována a archivována, jak jsou komunikována, jakým způsobem je systém integrován s ostatními nemocničními subsystemy, jak podporuje workflow příslušného oddělení, jakým způsobem zajišťuje adekvátní robustnost, spolehlivost a eventuální rozšiřitelnost.

V této oblasti existuje několik velice komplexních industrialních standardů, které návrh může respektovat.

Cílem této práce bylo navrhnout a implementovat PAC systém schopný dlouhodobého produkčního nasazení na PET Centru Nemocnice Na Homolce v rámci existujících požadavků a omezení, který bude zároveň co nejstandardnější z hlediska zmíněných standardů.

Klíčová slova: PACS, DICOM, Elektronický zdravotní záznam

Title: *Design and implementation of PAC system*

Author: *Petr Kalina*

Author's email: petr.kalina@jlabs.cz

Department: *Katedra softwarového inženýrství*

Supervisor: *RNDr. Ing. Jiří Peterka*

Supervisor's e-mail address: jiri@peterka.cz

Abstract:

PAC (Picture Archiving and Communication) systems enable for storage and management of digital picture data produced by various modalities on radiological and other hospital departments. An effective implementation of a PAC system has to face number of challenges – how the picture data and corresponding metadata are represented, how are they communicated, how the system integrates with other department or hospital subsystems, how the workflow and basic tasks are managed or how scalability, robustness and reliability is achieved.

There are several industrial standards affecting this area – and all of them are quite complex.

The aim of this work was to design and implement a PAC system that would be capable of long-term production use at PET center of Na Homolce hospital under existing requirements and limitations and that would be very standardized in respect to the existing industrial standards.

Keywords: PACS, DICOM, Electronic Health Record, Medical Imaging

1 Project Outline

The goal of this project was to implement a PAC system at PET center of Na Homolce Hospital, which would replace current unfitting solution.

The term PACS refers to Picture Archiving and Communication Systems. The pictures mentioned here are the digital picture data produced by various examination modalities – a PET (Positron Emission Tomography) scanner, a CT (Computed Tomography) scanner, MR (Magnetic Resonance) scanner etc.

A PACS system is a subsystem of a Radiology department that is concerned with management, communication and storage of the data produced by examination modalities present at the department. As such it has to feature appropriate communication means, appropriate capacity and reliability measures. As the data stored in the system are part of the corresponding patient's eHR (Electronic Health Record), the system also has to provide an interface for the HIS (Hospital Information System) and/or RIS (Radiology Information System) that enables for integration of the systems. A production PAC system also has to offer means for system configuration, administration and management of lifecycle events such as disasters, failures of various parts of the infrastructure and scheduled maintenance of underlying hardware and software.

Last but not the least the complexity of PACS design is also a result of non-trivial amounts of data it has to contain: a single radiology patient examination can result in 700 512x512 slices with 24-bit color depth, which, even when compressed by some lossless compression amounts to about 50MB size. Big radiology department can have multiple modalities and perform up to 100 examinations per day, which means a production of 5G per day!

The technological background behind PACS is very complex and difficult to master for a number of reasons. The main obstacles making the mastering of Digital Imaging and PACS areas of expertise difficult are:

- ***The complexity of the main standards*** – most of the communication and data manipulation inside PACS is carried out by means of DICOM (Digital Imaging and Communication in Medicine) standard. DICOM defines common data format for the picture data, facilitates a full blown application level communication protocol with various services the applications can provide or use and provides means for reporting and work flow management. Other standards in play are HL7 (Health Level 7) – a medical messaging protocol that for application level information exchange – and IHE (Integrating Healthcare Enterprise), which is a complex set of documents explaining how the two above mentioned standards should be used in cooperation to achieve fully integrated standardized environment. Together the standards amount to thousands of pages of complex, structured and technical information that is far from easy to read.
- ***Very specialized industry driven area*** – Radiology information systems and PACS are very specialized areas of expertise. The development is driven by the actual needs

of the healthcare industry and mostly done by the big companies in the field, where most of the experts can be found. The expertise in Medical Imaging and PAC systems is thus not very common in the academic arena. Thus it is very difficult to come by a decent piece of literature mapping the field or by a person with university background that is an expert in the field.

- **Quite new area** – some of the standards in the field are only few years old. The IHE (Integrating Healthcare Enterprise), for instance, was first presented to public in 1999 and it is only the last few years that it is becoming the status quo in healthcare systems integration.
- **America/Asia drives the development** – most of the research activity is done in America and Asia. As mentioned in [HUA04], there is number of factors hindering the adoption of new healthcare technologies in Europe (see table 1) – and they are all the more apparent in post-communist countries, which are technologically still very much trailing behind. Therefore the expertise here concerned the new emerging technologies is very low – and mostly out of academic ground. However the trends in healthcare, radiology and imaging informatics are clear and rapid – according to IHE homepage 90% of major healthcare system vendors already conform to IHE and the number of applications in Europe is growing.

Favorable Factors in the US	Hindering Factors in Europe
Flexible investment culture	Preservation of workplace culture
Business infrastructure of healthcare	Social service oriented healthcare
Competitive environment	Government control
Including PACS Expert consultants	Do it yourself mentality
Technological leadership drive	If it's not broken, don't fix it approach

Figure 1.1 – Hindering factors for adoption of new healthcare technologies in Europe

Even after understanding the principles of PACS and PACS integration, to roll a PACS implementation into production is never easy. Even though corresponding integration standards and guidelines exist, it is very common that vendors of various parts of hospital infrastructure do not conform to them, use various proprietary extensions etc. and therefore an extra effort – both programmatic and diplomatic – has to be spent on integration issues.

This project thus required various different skills and touches many areas. It started as an experimental pioneering effort into new technological area – quite unique on Czech university grounds – which gives it a scientific edge. It operates with very wide technological background. Eventually it moved to production-ready solution design, including the

reliability, efficiency and administration requirements typical for large scale in-production systems.

2 The Specification

As mentioned in section [1. Project Outline](#), the goal of this project was to implement a PAC system at PET center of Na Homolce Hospital, which would replace previous unfitting solution.

The PET center of the Na Homolce Hospital is historically the first molecular Imaging department in the Czech Republic. It features some unique characteristics - it is the biggest molecular imaging department in Czech Republic as far as the amount of produced data goes and it features some of the most advanced examination technology. It is well respected even in European scope.

The basic concept of work of the previous PAC system at the department is shown on Figure 2.1.

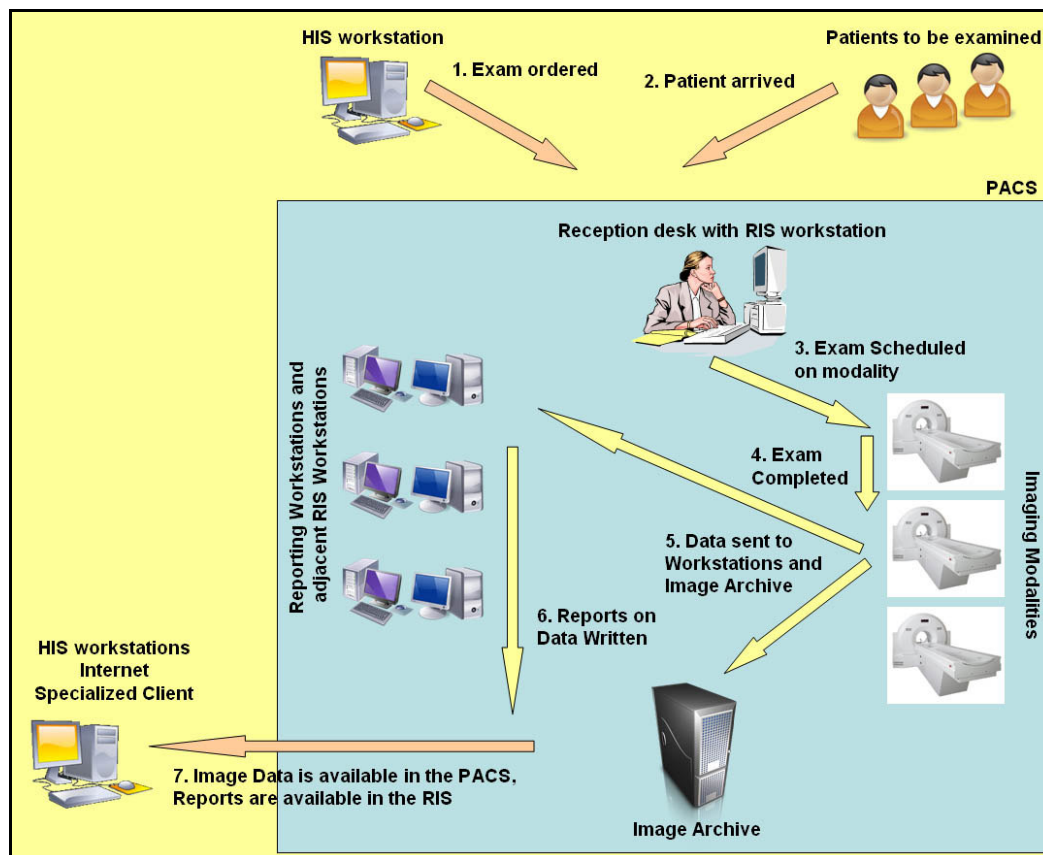


Figure 2.1 – Basic concept of PACS work

The examinations were ordered by requesting physicians from other hospital departments or at the main reception desk of the hospital for patients coming from other hospitals. The examinations were ordered in the HIS (Hospital Information System), which passed corresponding information to the RIS (Radiology Information System). At the department the incoming patients and corresponding ordered exams were managed by a RIS workstation at the main reception desk of the department. Once the patient arrived to the department,

appropriate examination was scheduled on the corresponding modality (examination device). The examinations were carried out on three modalities. Two of the modalities used DICOM (Digital Imaging and Communication in Medicine [DICOM]) as the data representation format and communication protocol, the third modality used a proprietary non-standard means of representing and communicating the data, which the central image archive supported. From the modalities the data were either directly pulled to diagnostic display workstations or pushed to the image archive and pulled to the workstations from there (depending on the modality). There the physicians analyzed the data and wrote diagnostic reports to the adjacent RIS workstations. Then the data were sent to permanent storage to the image archive (in case this was not done before).

There were no reliable policies to ensure correct linking of the actual image data to patient's eHR. Under some circumstances the exam data arriving to the image archive were identified correctly by the exam ID generated when the exam was ordered, but many times they were not. The patient IDs were not checked in any way for completeness and validity. Moreover there was no convenient way of displaying the data from outside of the department.

Thus the specification was based mainly on disadvantages and limitations of current system:

- ***The capacity of the current system was inappropriate*** – The image data so far produced by the modalities at the department in five years of operation amounted to over 100000 studies, over 30M individual image slices and around 1TB of compressed data. The system was at the limit of its capacity. The amount of data produced by the department was growing – new modalities with higher resolution produce much larger data than the old ones.
- ***Responsiveness was barely sufficient*** – with growing amount of data the system was beginning to be insufficiently responsive.
- ***Integration issues*** – the system did not integrate well with the rest of the hospital information system. There was no way how to access the data from hospital information system, there was no reliable way how to bind the image data to the eHR (electronic Health Record) of the corresponding patient. It was not possible provide remote access to the data to other healthcare organizations other than a CD with exported data – a feature that was crucial for this very specialized department as most of its patients are actually in care of other hospitals and only come to be examined at the department.
- ***Remote work*** – the system didn't support remote mode of operation with physicians displaying the data and reporting on it from home.
- ***Insufficient level of reliability*** – there were no policies as to how to solve the various service/crisis situations that may occur. There was no periodical checking of data consistency. The data was backed up on mounted NFS disk and one redundant archive was ready at hand, but there was no checking, whether the backup is actually working. The level of surety that everything would work should the potential failure of the main server occur was generally low – also no one was really sure about the

time in which the full operation could be restored. There was no documentation describing what to do in crisis situations.

- ***Insufficient service/consulting support*** – the support provided by the system vendor was limited, slow and expensive. As there was no PACS expert within the hospital staff, the consulting services were also lacking to tune up the solution.

Thus the following characteristics formed the specification of the new system that was to be designed and rolled into production:

- ***Interoperability with the rest of the department equipment*** – some parts of the new PACS will remain the same – the modalities, the diagnostic workstations. It was necessary that the new system is able to communicate with all existing equipment of the department in appropriate way.
- ***Sufficient capacity for all current data and expansibility*** – the system had to ensure smooth operation with all current data, enough space for near future (2-3 years) and a plan how to increase the capacity later while still preserving appropriate responsiveness.
- ***Appropriate reliability measures*** – the data had to be backed up safely and strategies had to exist of how to restore system function in expectable failure situations.
- ***Integration with HIS*** – the system had to provide access to the image data from the HIS, which required ensuring correct linking of the image data to the eHR of the patient and providing quality integration interface.
- ***Remote data sharing*** – the system had to provide means for secured remote access to the data to enable for data sharing with other healthcare institutions. The system should also provide means to support the remote work mode for physicians.

The specification evolved as the project went on. Originally the project was started merely as an inquiry into new technological area and eventually gained momentum of its own. No specification other than this was ever available.

3 Organization of this work

As mentioned in section [1. Project Outline](#), the area of expertise that this project deals with is very specialized, very difficult to master and quite new. Therefore we find it appropriate to spend some time explaining PACS basics and the theoretical background of PACS implementation e.g. the basic description of industrial standards in play etc.

The rest of the work is divided into 6 chapters:

- **4. PACS and Imaging Informatics** – a brief introduction to PACS and imaging informatics areas of expertise
- **5. Basic design choices** – outline of the most important design choices that had to be faced and reasoning behind the chosen solutions
- **6. The Implementation** – more detailed look at the implementation
- **7. Summary** – evaluation of achieved success
- **8. Future** – in what areas the project can be further improved
- **9. References**

4 PACS and Imaging Informatics

PACS stands for Picture Archiving and Communication System. The origin of the term is healthcare and the pictures mentioned here are the picture data produced by various medical examinations.

The history of PACS goes as far as early 1980s when the radiology community first envisioned the concept of PACS. PACS consists of image acquisition devices, storage archiving units, display workstations, processing units and databases all integrated by communication network with some data management policies. With the benefits of digitalizing the data and introducing computer based data storage and management systems together with the advance in computer technologies, the PAC systems have matured rapidly, and their applications have gone beyond radiology to the entire health care delivery system. In that period the continuous development the process has converged towards some well respected practices and industry standards – namely the *DICOM* (Digital Imaging and Communication in Medicine) and *HL7* (Health Level Seven) communication protocols and *IHE* (Integrating the Healthcare Enterprise) initiative, that now form the framework for many successful PACS implementations in America and Europe, as well as in other continents.

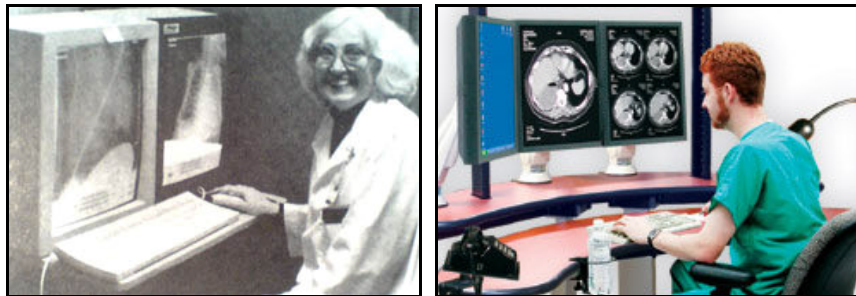


Figure 4.1 – Old and new Diagnostic Workstation [HUA04]

Together with other activities aiding digitalization of the whole healthcare delivery system the PACS systems aim to the target of paperless/filmless computer managed healthcare facilities, both increasing the efficiency of health care delivery and forming the basis for many advanced data management and analysis applications.

4.1 What is a PACS

As noted before, the origin of PACS are radiology departments and their striving towards digitalization. A digital radiology department has two main components – a **Radiology Information System (RIS)** and digital imaging system – or the PACS. The RIS is a subset of the **Hospital Information System (HIS)** or Clinical Management System (CMS). All these systems contribute to information stored in the **Electronic Health (or Medical/Patient) Record (eHR/eMR/ePR)** and supply management functionality.

The PACS involves picture acquisition, archiving, communication, retrieval, processing distribution and display – the RIS is more concerned with the tasks associated with the patient

documentation and tasks concerning administrative and management routines of the department, such as ordering and managing patient visits, managing textual parts of the patient documentation produced at the department, supporting department management etc. The two systems communicate through some well defined interface and often are supplied by two different vendors.

A PAC systems range from very simple scenarios – for example a film camera, a digitizer and a simple few gigabyte image archive – to very complex ones – like hospital and cross-hospital enterprise PAC systems that communicate with tens of healthcare facilities, managing hundreds of terabytes of data.

4.2 PACS Components and Workflow

There are many types and scales of PAC systems. The general concept referring to a typical middle scale PACS consists of following components:

- **Imaging modalities** – the facilities capturing the images. There are many types of modalities based on various physical principles – ranging from simple photography do highly sophisticated devices like Positron Emission Tomography (PET), Computed Tomography (CT) or Magnetic Resonance (MR) scanners.



Figure 4.1 – Siemens Biograph PET CT Scanner

- **Acquisition Gateways** – sometimes also called Modality Consoles – these are computer stations controlling the modalities and providing an interface to the modality operators to perform examinations on the modality. The digital imaging modality nearly always comes with acquisition gateway supplied by the vendor of the modality. The communication interface between the modality and the gateway is proprietary for the modality. On the other hand the gateway should provide standardized interface for communication with the rest of the system.
- **Display Workstations** – these are computers equipped with display and processing software that enables for display and human or machine performed analysis of the data acquired by the modalities. Here the physicians review and evaluate the data and

write the findings into the RIS/HIS database. Sometimes the display workstations are referred to as either **Reporting Workstations** – the computers at the department where the image data is diagnosed and reports entered to the RIS – or **Clinical Workstations** – the computers at other departments, where the data is only displayed.



Figure 4.3 – Display Workstation at the department with adjacent RIS Workstation

- **PACS Controllers and Image Archive Servers** – these are the backbone components of the PACS. All the data produced by the department are stored on image archive servers and are eventually retrieved from here. The PACS controller provides interfaces to communicate with both underlying (workstations, acquisition gateways) and superior (RIS, HIS, application servers, web servers) systems.



Figure 4.4 – Image Archive Server of the department

- **Application servers and Web Servers** – provide access to the data using various protocols. On the back end they access the archive servers and the PACS controllers to query and retrieve the data, on the front end they provide client interfaces customized for various research, educational or other purposes.
- **System Networks and Network Communication Protocols** – inseparable part of any PACS is the topology and nature of underlying network and the protocols used for

communication between various entities. On the top level the communication protocols in a modern PACS are DICOM and HL7 - both of which will be explained in section [4.3 Industrial Standards concerning PACS](#) – which form the basis for interoperability between various PACS subsystems that can eventually be supplied by different vendors.

- **Modality Operators** – these are the people who operate the modalities and supervise the actual examinations. They are the users of acquisition gateways. These people usually check the produced data and send it to the PACS archive or to workstations for analysis.
- **Physicians** – the professionals who analyze the data at the workstations and write findings to the patient’s eHR in the HIS/RIS.
- **Administrative staff** – people managing administrative flow of the department like scheduling of patient visits and accounting. They usually work with the RIS/HIS workstation, which communicates with the PACS controller and various HIS services.

Being a part of daily routines of a facility/department a PACS has to guarantee fluency and automation of basic operations performed. The typical work cycle of a radiology department consists of following steps.

- Patient registers in the HIS. Patient is ordered for a radiology exam. A unique exam ID is generated.
- The HIS notifies the RIS about this event and the exam is scheduled at the RIS. When patient arrives to the radiology department, the RIS in cooperation with the PACS controller/image archive server schedules appropriate exams on target modalities.
- The operator readies the modality for the exam and the patient is examined.
- The operator verifies acquired data. The data is sent to appropriate workstation(s).
- At the workstation the data is analyzed by the physicians and appropriate findings are entered manually to the RIS (eventually through some dictation system with subsequent transcription – which is a common praxis in English speaking countries)
- The data is sent to the image archive and a report is generated in the RIS/HIS saying that the exam was completed

The actual routines may differ depending on the specific needs of the department, the interfaces and architecture of individual HIS, RIS and PACS implementations.

The standardized communication protocols like HL7 and DICOM make it possible for independent vendors to offer PACS solutions that can be adopted into existing infrastructure with realistic effort – the main connection interfaces being well defined. This of course assumes that the host hospital information subsystems have a standardized interface – which is very often not the case.

4.3 Industrial Standards concerning PACS

A typical situation in a hospital is that various vendors contribute to various parts of the infrastructure. Usually there are many different modalities throughout the hospital each from a vendor specializing in that particular area, also the HIS and RIS and their subsystems are often heterogeneous and not supplied by a single vendor. It is impossible for one vendor to cover all the expertise needed by such a complex system.

It is very important for the hospital to be able to choose the building blocks for its system independently and fittingly and to be able to upgrade or develop new solutions in places where the system is lacking without the need to replace the whole system. That is why it was necessary at some point to create vendor independent standards facilitating the necessary communication interfaces and protocols between the parts of the infrastructure.

For PACS these are **DICOM** (Digital Imaging and Communication in Medicine) and **HL7** (Health Level Seven), which form a basis for vendor independent communication of image and textual data and for managing basic workflow necessary for PACS operation.

The nature of both of the standards is such that it does not advocate how they should be used – it merely gives the implementers some means of communication and rules that an entity conforming to some part of the standard should obey. Nor do they specify how the subsystems using them should be designed or how the two standards should be used in cooperation to facilitate some higher level system like PACS. This made the designing of a PACS very difficult and the lack of expertise in this higher level logic of applying the standards led to many proprietary ways of integration, which – even though using standard communication means – were not standard, agreed upon by wider public and were not exploiting the full potential of the standards. This is a reason why **IHE** (Integrating Healthcare Enterprise) initiative was started – to become a place where such expertise as to how the individual subsystems in healthcare delivery should interact using the above mentioned standards could be developed, tested and maintained.

All of the standards are relatively new – the first version of DICOM emerged in 1985, the first version of HL7 in 1987 and the first large scale application of IHE was first demonstrated in 2000. Even though they are all well established standards, there are still some modalities not conforming to DICOM, many other medical messaging systems than HL7 and many hospital systems and subsystems not conforming to IHE integration profiles.

4.3.1 DICOM

In 1982 the ACR-NEMA (American College of Radiology and the National Electric Manufacturers Association) created a committee to develop a set of standards to serve as the common ground for various medical imaging equipment vendors. The goal was mainly to create a common platform for information exchange and image sharing in PACS environment. The first version of the standard – then called ACR-NEMA 1.0 – emerged in 1985 to be followed by the second release in 1988. The second version brought both

underlying hardware definitions and software protocols for communication, as well as a standardized data dictionary to use for the communication. The third version, published in 1992, has brought significant changes – namely in adoption of object model for data representation and utilization of existing network protocols for data communication - and was therefore named differently than the previous two versions: Digital Imaging and Communication in Medicine (DICOM 3.0). Since then no new version was released – even though there have been quite a few additions and enhancements. The standard is backwards compatible with all the previous versions.

The current DICOM standard (2007) includes 18 parts:

- Part 1: Introduction and Overview
- Part 2: Conformance
- Part 3: Information Object Definitions
- Part 4: Service Class Specifications
- Part 5: Data Structures and Encoding
- Part 6: Data Dictionary
- Part 7: Message Exchange
- Part 8: Network Communication Support for Message Exchange
- Part 9: Point to Point Communication Support for Message Exchange (Retired)
- Part 10: Media Storage and File Format for Media Interchange
- Part 11: Media Storage Application Profiles
- Part 12: Media Formats and Physical Media-for-Media Interchanges
- Part 13: Print Management Point-to-Point Communication Support (Retired)
- Part 14: Gray Scale Standard Display Functions
- Part 15: Security and System Management Profiles
- Part 16: Content Mapping Resource
- Part 17: Explanatory Information
- Part 18: Web Access to DICOM Persistent Objects (WADO)

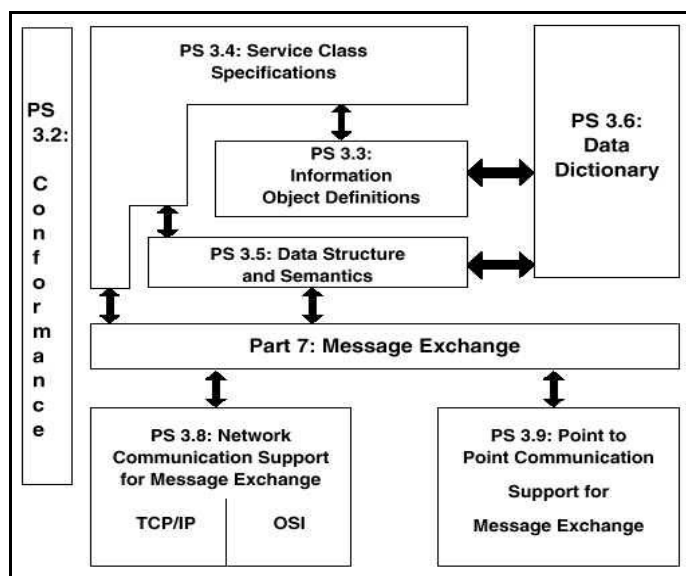


Figure 4.5 – DICOM Standard Architecture [DCM]

This amounts to about 4000 pages – DICOM is not simple, nor is the standard easy to read.

In the simplest abstraction the DICOM standard defines an object model for representing the data and entities in the PACS (the DICOM Model of Real World) and set of services to do operations with them over the network. A definition of a real world object in DICOM is referred to as the Information Object Definition (IOD).

These are some important objects defined in the standard (in DICOM terminology these are Information Object Classes):

- **Patient**
- **Study** – represents a single patient visit at the department.
- **Series** – represents a single examination at one modality – a part of the study.
- **Image (Instance)** – usually represents a single slice of a possibly 3D multi-slice series. The actual image can be from any type of modality – a CT (Computed Tomography) image, MR (Magnetic Resonance) image, CR (Computed Radiography) image or PET (Positron Emission Tomography) image – also it can be a waveform like ECG (Electro Cardiography) waveform from an ECG device. The image can actually even represent metadata – as is the case with Structured Reports which enable for linking of some textual data to the actual measurement data. The Patient->Study->Series->Instance is the default abstract data model in DICOM.
- **AE** – (Application Entity) represents a DICOM enabled node in the network
- **Modality** – an examination device

There are some of the important services defined in the standard (in DICOM terminology these are Service Classes):

- **Storage Service Class** – enables for storage of DICOM objects (Patients, Studies, Series and Instances).
- **Query/Retrieve Service Class** – enables for querying DICOM objects stored at some DICOM node and their retrieval.
- **Modality Worklist Service Class** – enables for storage and management of Modality Worklists which contain information about exams scheduled in the PACS.

4.3.1.1 DICOM Information Object Classes and Data Format

As mentioned before, the DICOM defines its model of the real world in which it defines real world objects in the clinical image arena (e.g. Patient, Study, Series, Image, etc.) and their relationships within the scope of the standard. These are referred to as the Information Object Classes or Information Object Definitions (IODs).

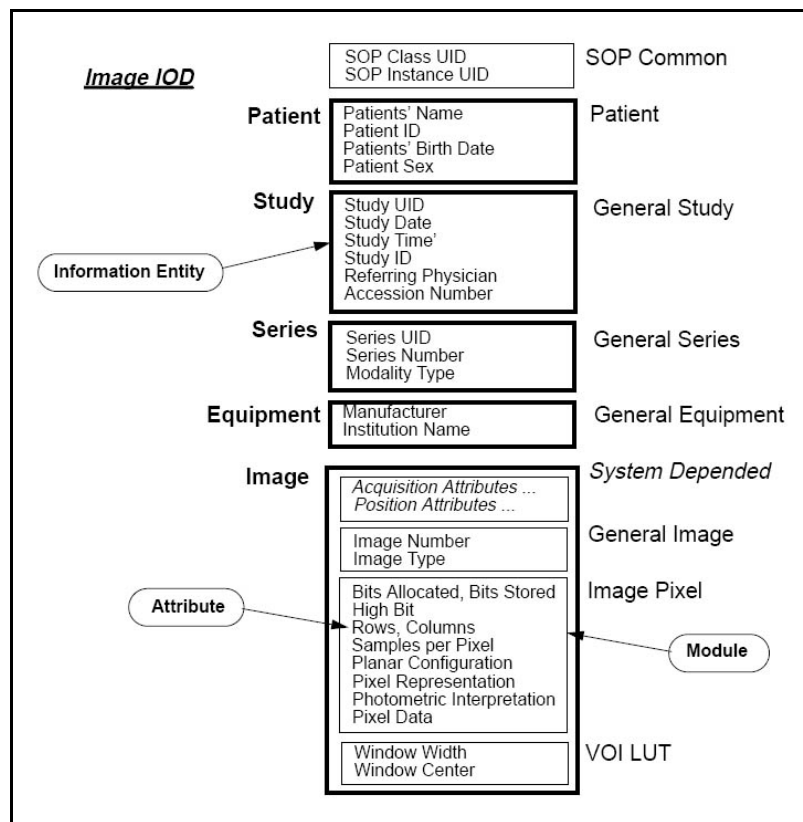


Figure 4.6 – DICOM Image IOD – an example of a Composite DICOM object [DCMCOO]

Each DICOM object is represented by its dataset. The DICOM dataset is a list of attributes of corresponding DICOM object, like patient's name, patient's ID, study ID, study accession number etc. – these attributes are defined in DICOM data dictionary and form the vocabulary of DICOM. For each attribute the standard defines its name, value representation (data type)

and its DICOM tag – a 32 bit long integer. The actual pixel data stored in some DICOM Instance object are represented by a Pixel Data tag.

0008	0005	Specific Character Set	CS	10	1	ISO_IR 100																																						
0008	0008	Image Type	CS	30	4	ORIGINALPRIMARYTOMOEMISSION																																						
0008	0016	SOP Class UID	UI	26	1	1.2.840.10008.5.1.4.1.1.20																																						
0008	0018	SOP Instance UID	UI	44	1	1.3.12.2.1107.5.6.1.6272.7.0.295426217501362																																						
0008	0020	Study Date	DA	8	1	20071206																																						
0008	0021	Series Date	DA	8	1	20071206																																						
0008	0022	Acquisition Date	DA	8	1	20071206																																						
0008	0023	Content Date	DA	8	1	20071206																																						
0008	0030	Study Time	TM	6	1	145219																																						
0008	0031	Series Time	TM	6	1	145219																																						
0008	0032	Acquisition Time	TM	6	1	145219																																						
0008	0033	Content Time	TM	6	1	145219																																						
0008	0050	Accession Number	SH	16	1	8203160064003052																																						
0008	0052	Query/Retrieve Level	CS	5	1	IMAGE																																						
0008	0054	Retrieve AE Title	AE	9	1	JAPECUBE1																																						
0008	0056	Instance Availability	CS	6	1	ONLINE																																						
0008	0060	Modality	CS	2	1	NM																																						
0008	0070	Manufacturer	LO	10	1	SIEMENS NM																																						
0008	0090	Referring Physician's Name	PN	0	0																																							
0008	1010	Station Name	SH	6	1	ESOFTA																																						
0008	1030	Study Description	LO	42	1	Test Study																																						
0008	103E	Series Description	LO	18	1	Test Series																																						
0008	1050	Performing Physician's Name	PN	0	0																																							
0008	1070	Operator's Name	PN	0	0																																							
0008	1090	Manufacturer's Model Name	LO	4	1	IP2																																						
0008	2218	Anatomic Region Sequence	SQ	--																																								
<table border="1"> <thead> <tr><th colspan="7">Items</th></tr> <tr><th>1</th><th>Group</th><th>Element Name</th><th>VR</th><th>Length</th><th>VM</th><th>Value</th></tr> </thead> <tbody> <tr><td></td><td>0008</td><td>0100</td><td>Code Value</td><td>SH</td><td>8</td><td>1</td><td>T-11000</td></tr> <tr><td></td><td>0008</td><td>0102</td><td>Coding Scheme Designator</td><td>SH</td><td>6</td><td>1</td><td>99SDM</td></tr> <tr><td></td><td>0008</td><td>0104</td><td>Code Meaning</td><td>LO</td><td>8</td><td>1</td><td>Skeletal</td></tr> </tbody> </table>							Items							1	Group	Element Name	VR	Length	VM	Value		0008	0100	Code Value	SH	8	1	T-11000		0008	0102	Coding Scheme Designator	SH	6	1	99SDM		0008	0104	Code Meaning	LO	8	1	Skeletal
Items																																												
1	Group	Element Name	VR	Length	VM	Value																																						
	0008	0100	Code Value	SH	8	1	T-11000																																					
	0008	0102	Coding Scheme Designator	SH	6	1	99SDM																																					
	0008	0104	Code Meaning	LO	8	1	Skeletal																																					
0010	0010	Patient's Name	PN	22	1	Petr Kalina																																						
0010	0020	Patient ID	LO	10	1	8203160064																																						
0010	0030	Patient's Birth Date	DA	8	1	19980904																																						
0010	0040	Patient's Sex	CS	2	1	F																																						
0010	1010	Patient's Age	AS	4	1	004Y																																						
0018	0015	Body Part Examined	CS	16	1	WB																																						

Figure 4.7 – DICOM Dataset Example

The DICOM defines how a DICOM dataset should be stored to a file defining the format of the file, enabling for different encoding modes, compression modes for the pixel data etc.

DICOM Tag, Value (Tag Name)	Binary Coding (Implicit Value Representation, Little-Endian)
0008,0000, 726 (Identifying Group Length)	08 00 00 00 04 00 00 00 D6 02 00 00
0008,0005, ISO.IR 100 (Specific Character Set)	08 00 05 00 0A 00 00 00 49 53 4F 5F 49 52 20 31 30 30
0008,0016, 1.2.840.10008.5.1.4.1.1.2 (SOP Class UID)	08 00 16 00 1A 00 00 00 31 2E 32 2E 38 34 30 2E 31 30 30 30 38 2E 35 2E 31 2E 34 2E 31 2E 31 2E 31 2E 32 00
0008,0060, CT (Modality)	08 00 60 00 02 00 00 00 43 54
0008,1030, Abdomen.1abdpcelvis (Study Description)	08 00 30 10 12 00 00 00 41 62 64 6F 6D 65 6E 5E 31 61 62 64 70 65 6C 76 69 73
...	...
	E0 7F 10 00 00 00 00 00 00 00 00 00 00 00 00 00




Figure 4.8 – Dataset Encoding Example

In general there are two fundamental groups of object classes – the Normalized Object Classes and the Composite Object Classes. Normalized Object Class is an object class that only contains attributes inherent to a single real world entity in scope of the standard – such as Patient or Study. Composite Object Class on the other hand contains attributes inherent to multiple real world entities. A typical example of composite object is any real image by any modality – like Nuclear Medicine Image of Positron Emission Tomogram – these images contain not only information about the single image, but also about the corresponding series, study, patient, hospital department, medication used during the examination like contrast agents etc. This situation well is illustrated on figures 4.6 and 4.7.

The previous example also depicts the typical communication entity when transferring data from modalities. The objects sent and received by the peer DICOM nodes and also the only DICOM files present on DICOM servers, are in fact the actual Instances (images, slices) – composite objects containing data inherent to Instance normalized object (pixel data, slice thickness, pixel spacing, position in the 3D stack of images etc.), but also containing additional information inherent to referenced Series, Study, Patient etc. normalized objects.

4.3.1.2 DICOM SERVICES

DICOM services are used to perform operations with the objects – like sending, storing, printing, displaying etc. over the network.

DICOM network consists of DICOM nodes (DICOM servers). Each node conforms to some set of services. A DICOM node is listening on some port for incoming association requests. The association request specifies what service the peer node is requesting (e.g. Storage Service) and affected object class (PET image). If the service on this object class is supported by the called peer, the association is accepted and further negotiation can take place as to how eventual data will be transferred and which special features inside the service definition are requested or supported by the peers. Then the actual action inherent to the service can take place (e.g. images are sent from one node to the other).

The DICOM services are designed in a fashion that the two communicating entities have two distinct roles – one acts as the Service Class Provider (SCP), one as the Service Class User (SCU). Usually it is the user SCU that initiates the service operation.

The SCU/SCP relationship can be illustrated on already mentioned service classes:

- ***Storage Service Class*** – the SCU of this class is able to send images to a SCP of this class. An image archive is a typical SCP of this class, the acquisition gateways are typical users.
- ***Query/Retrieve Service Class*** – the SCU of this class is able to query the database of images stored at the SCP. If the SCU is also a Storage Class SCP, it can initiate the retrieval of the images from the SCP. Image archive is a typical SCP of this class, workstations are typical SCUs.

- **Modality Worklist Service Class** – the SCP of this class maintains a list of scheduled exams – worklist items. The SCU of this class is able to query the worklist items stored at the SCP. The typical SCP of this class is the image archive, typical SCUs are the modality gateways.

The services are built on top of DICOM Message Service Elements (DIMSEs), which are the basic DICOM commands. DIMSEs are divided into operations working with composite object classes and operations working with normalized object classes:

- Normalized DICOM Message Service Elements
 - **N-EVENT-REPORT** – notification of information object-related event
 - **N-GET, N-SET** – get or set a single attribute (patient name) of an instance of target DICOM object
 - **N-CREATE, N-DELETE** – create/delete target object
 - **N-ACTION** – perform object-related action
- Composite DICOM Message Service Elements
 - **C-ECHO** – verification of connection to target Application Entity
 - **C-STORE** – storage target object instance
 - **C-FIND** – query target Application Entity for information about composite object instances
 - **C-GET** – transmission of object instance information via third party application process
 - **C-MOVE** – similar as get, with the option that the receiver is not the initiator of the operation

DICOM services use existing network communication standards for image/information transmission. The underlying network stack can be TCP/IP or ATM.

4.3.1.3 DICOM Workflow Management

The **Modality Worklist Service Class (MWL)** allows for maintaining worklists (schedules) on DICOM servers that contain scheduled exams the modalities (their gateways) are able to read. This service class provides means to place orders registered in the HIS/RIS to the gateways of target modalities and thus to manage the department workflow.

A worklist item among others contains information about the target patient, target modality, the type of exam and the ID of the exam that can be used to bind the resulting image data to the eHR of the target patient in the HIS.

The **Modality Performed Procedure Step Service Class (MPPS)** is a complementary service class to Modality Worklist Service Class. In SCU role it enables the modality to create/update information about a performed examination describing the details of the performed procedure (status, beginning, end time etc.). The information is stored inside the SCP of this class - the Image Archive, or any relevant entity of the PACS configured to manage MPPS items. This

service allows for the modality to better coordinate with image storage servers by enabling them to actually trace the status of the corresponding exam and trigger events on eventual update of the status of the exam (completion, cancelling), including pre-fetching of previous patient exams to reporting workstations.

The **General Purpose Worklist Service Class (GPWL)** is a replacement of MPPS and MWL introduced in 2001 that allows for better integration of reporting workstations into the workflow mechanism. Its functionality is very similar as that of MWL and MPPS combined, in addition it enables for locking of individual exams, so it offers means to prevent the situation when the same exam is reported to simultaneously on two reporting workstations.

4.3.1.4 DICOM Structured Reporting

DICOM **Structured Reporting (SR)** enables for transmission and storage of clinical documents. The SR classes fully support both conventional free-text reports and structured information. In addition, the SR standard provides the capability to link text and other data to particular images or waveforms and to store the coordinates of findings. The value and self-containment of information stored inside the PACS can be significantly improved when using SR. Structured Reporting is defined in supplement 23 of the DICOM standard.

4.3.1.5 DICOM WADO

DICOM **WADO (Web Access to DICOM Objects)** enables for pulling data from a WADO enabled DICOM server via HTTP protocol using standardized interface based on GET parameter passing method. The requested GET parameters to obtain an image (instance) stored in the archive are the UID (Unique Identifier) of the image and optionally the DICOM Transfer Syntax UID, which specifies in what format the image should be returned. Depending on actual archive capabilities the possible formats can range from JPEG, DICOM in various compressions to uncompressed DICOM file format.

4.3.1.6 DICOM Conformance

A DICOM enabled application advertises its DICOM related capabilities in a DICOM conformance statement. In this statement it declares its conformance to various Service Classes – the roles (SCU/SCP) that it supports and objects, on which it is able to operate including the actual compression of eventual pixel data and encoding of passed values. For example the application can declare its support for DICOM Storage Service Class in SCP role on Computed Tomogram Objects with pixel data encoded in JPEG LS Lossless using Explicit Value Representation and Big Endian encoding. The list of all supported capabilities forms the DICOM Conformance Statement of the application.

No official testing suite to verify the DICOM conformance ships with the standard. However some products are available in this area – most significantly the DICOM Validation Toolkit [DCMVDT] by Phillips pushed recently into the free software arena.

SOP Class		User of Service (SCU)	Provider of Service (SCP)
Name	UID		
Transfer			
Computed Radiography Image Storage	1.2.840.10008.5.1.4.1.1.1	Yes	Yes
Digital X-Ray Image Storage – for Presentation	1.2.840.10008.5.1.4.1.1.1.1	Yes	Yes
Digital X-Ray Image Storage – for Processing	1.2.840.10008.5.1.4.1.1.1.1.1	Yes	Yes
CT Image Storage	1.2.840.10008.5.1.4.1.1.2	Yes	Yes
MR Image Storage	1.2.840.10008.5.1.4.1.1.4	Yes	Yes
Secondary Capture Image Storage	1.2.840.10008.5.1.4.1.1.7	Yes	Yes
General ECG Waveform Storage	1.2.840.10008.5.1.4.1.1.9.1.2	Yes	Yes
Grayscale Softcopy Presentation State Storage	1.2.840.10008.5.1.4.1.1.11.1	Yes	Yes
X-Ray Angiographic Image Storage	1.2.840.10008.5.1.4.1.1.12.1	Yes	Yes
X-Ray Radiofluoroscopic Image Storage	1.2.840.10008.5.1.4.1.1.12.2	Yes	Yes
Nuclear Medicine Image Storage	1.2.840.10008.5.1.4.1.1.20	Yes	Yes
Positron Emission Tomography Image Storage	1.2.840.10008.5.1.4.1.1.128	Yes	Yes
RT Image Storage	1.2.840.10008.5.1.4.1.1.481.1	Yes	Yes
RT Structure Set Storage	1.2.840.10008.5.1.4.1.1.481.3	Yes	Yes
RT Plan Storage	1.2.840.10008.5.1.4.1.1.481.5	Yes	Yes
Query/Retrieve			
Study Root Query/Retrieve Information Model – FIND	1.2.840.10008.5.1.4.1.2.2.1	Yes	Yes
Study Root Query/Retrieve Information Model – MOVE	1.2.840.10008.5.1.4.1.2.2.2	Yes	Yes
Workflow Management (Scanner Only)			
Modality Performed Procedure Step	1.2.840.10008.3.1.2.3.3	Yes	No
Modality Worklist Information Model – FIND	1.2.840.10008.5.1.4.31	Yes	No

Figure 4.9 – Example Conformance Statement [PHILCS]

4.3.2 HL7

HL7 (Health Level 7) was established in 1987 as a user-vendor committee to develop a standard for the exchange, management and integration of electronic healthcare information.

As the core abstraction of HL7 is a message, it is also often referred to as a healthcare messaging protocol. The “Level 7” in the name refers to the 7th layer in the OSI (Open Systems Interconnection) network communication model. This indicates that HL7's scope is the format and content of the data exchanged between the applications, rather than how the messages are transmitted between computers or networks.

```
MSH|^~\&|RIS|NNH|DCM4CHEE_Server|PetrK|8203160064||ORM^O01|1|P|2.3.1|
PID|0001|8203160064||KALINA^PETR^KARLIKOVA|19821316|M||1
|Na_Pruhonu_654^Cimice^Praha_8^^18100^Czech_Republic^|603982837|603982837|CZ^|U|AGN|
PV1|||||||2007120600000001|||||||PET|
ORC|NW|2007120600000001|2007120600000001|IP|^|^200712061515|^|^KALINA^PETR|
OBR|1|2007120600000001|2007120600000001|^PET-FDG-Torzo|||||||20||||
RAD|||1^^200712061515|
NTE|||The Last Patient!
```

Figure 4.10 – Example HL7 ORM^001 (Order Entry) Message

The range of information HL7 messages can represent is much wider than that of DICOM. HL7 contains an exhaustive set of definitions enabling for communication of all major subsystems present in healthcare facility, ranging from eHR representation, hospital workflow management to billing and accounting.

The standard in current status (as of version 2.5.1 approved 2/2007) contains following chapters:

- **1. Introduction**
- **2. Control** - rules for creating, communicating and processing of messages.
- **3. Patient Administration** – messages inherent to patient administration.
- **4. Order Entry** – messages inherent to ordering and management of various medical exams.
- **5. Query** – messages enabling for querying medical (or other) information
- **6. Financial Management** – messages inherent to patient accounting financial transactions.
- **7. Observation Reporting** – communication of structured patient-oriented clinical information.
- **8. Master files** – refer to common shared information like patient indexes, staff indexes etc.
- **9. Medical Records/Information Management** – management of medical documents related to care provided to a patient.
- **10. Scheduling** – scheduling of appointments for services or usages of resources.
- **11. Patient Referral** – messages inherent to referring to patients and the set of information inherent to them between various healthcare facilities/entities.
- **12. Patient Care** – messages inherent to communication of problem-oriented information concerning patient care, such as decisions, problems, goals etc.
- **13. Clinical Laboratory Automation** – messages inherent to interfacing and integration of various devices or procedures into the LIS (Laboratory Information System).
- **14. Application Management** – messages inherent to management of HL7-supporting applications over the network.
- **15. Personnel Management** – messages inherent to transmission of new or updated administrative information about healthcare personnel.
- **Appendices A-E** – Contain data definition indexes, information about possible underlying lower level protocols, define a language for describing HL7 messages and provide glossary and index of terms used in the standard.

There are several main characteristics of HL7 as a communication protocol [IW]:

- **Event Driven** - Real-world events, such as the admission of a patient, cause messages to flow between applications. In other words, an application that encounters a real-world event sends a message to other applications that need to be aware of this event.

- **Application to Application** - It defines a communication between two independent applications, rather than between closely coupled, client/server applications. The scope of interest for HL7 is the message exchange between the applications, rather than the specific role of each application in the healthcare delivery process.
- **Point to Point** - HL7 is a messaging format that is independent of its transport method. However, it is typically used in a client/server environment for employing some form of a point-to-point protocol. It is not normal for HL7 messages to use a non point-to-point protocol, where the client listens for 'broadcasts' from a particular server.
- **Data Exchange** - HL7 specifies the way data exchange between applications will be accomplished. It does not specify how applications store or process this data.

4.3.2.1 *HL7 and PACS*

Most of basic radiology department internal procedures are covered by DICOM services and the PACS – the MWL (Modality Worklist) for managing department workflow, the Storage and Query/Retrieve services for managing the image data flow and storage. It is where the PACS meets the RIS and the HIS where HL7 can be useful.

The range of HL7 messages relevant to PACS is thus very limited – either notifying the PACS about significant event in the hospital such as patient being ordered to be examined at the department or the patient information has been updated etc. or notifying the RIS/HIS about significant events inside the PACS, like an exam being finished, data becoming available etc.

4.3.3 **IHE**

IHE (Integrating Healthcare Enterprise) is an initiative of healthcare professionals originally initiated by HIMSS (Healthcare Information and Management Systems Society) and RSNA (Radiological Society of North America) in 1998 to propose integration profiles based on common industrial standards such as DICOM or HL7 for various healthcare subsystems, with first public demonstrations at RSNA at 1999 and 2000.

Originally the committee was mostly concerned with radiology and adjacent fields, recently the area of interest is gradually widening to the whole healthcare delivery process.

The IHE was constituted in reaction to widening trend of misusing the industrial standards and resulting incompatibility of various systems leading to limited customer choice, huge budgets spent on integration issues and sub-optimal integration. The main reason for having standardization documents describing the interfaces and use-cases for various actors in the healthcare delivery process is thus customer choice, quality of integration, standardization and cost cutting. The process has proven well worthy, with most of the major vendors now conforming to IHE and indeed taking part on the development and sponsoring it.

The IHE documents take form of Integration Profiles and Technical Frameworks. Each Integration Profile describes one specific area of integration e.g. Cardiac Cath Workflow (CATH) or Request Information for Display (RID) or Patient Information Reconciliation (PIR) in rather abstract terms, being targeted to less technically skilled personnel like physicians.

IHE is organized across a growing number of clinical and operational domains. Each domain produces its own set of Technical Framework documents, in coordination with other IHE domains. Committees in each domain review and republish these documents annually, often expanding with supplements that define new Integration Profiles. Technical frameworks contain detailed technically oriented look at the integration profiles targeted to developer community.

Various entities inside the systems implementing certain set of functions are referred to as Actors. An actor is defined by a set of supported transactions. A transaction is one required operation like Patient Registration or Image Retrieve. The support for an individual transaction inside an integration profile can be optional or required.

Thus a software product can declare itself as implementing one or more IHE actors – the document summarizing the capabilities of an application in scope of IHE is referred to as IHE Integration Statement. For example a DICOM server enabling for storage of medical image data can declare itself as being the implementation of IHE Image Archive actor in Patient Information Reconciliation integration profile, Scheduled Workflow profile, Access to Radiology Information profile etc. – meaning that it supports all transactions required of Image Manager Actor in each of the profiles. If it does not support all the transactions, it can still define the subset of all requested transactions and claim limited compliancy.

		IHE Integration Statement	
Vendor	Product Name	Version	Date
dcm4che.org	dcm4chee	2.10	Sep 2006
This product implements all transactions required in the IHE Technical Framework to support the IHE Integration Profiles, Actors and Options listed below:			
Integration Profiles Implemented	Actors Implemented	Options Implemented	
Patient Information Reconciliation	Image Manager/Archive	none	
	Performed Procedure Step Manager	none	
	Report Manager	none	
Scheduled Workflow	Image Manager/Archive	Availability of PPS-Referenced Instances	
		PPS Exception Management	
		Performed Work Status Update - Receive	
	Performed Procedure Step Manager	none	
Importation Reconciliation Workflow	Image Manager/Archive	none	
	Performed Procedure Step Manager	none	
Cath Workflow	Image Manager/Archive	Availability of PPS-Referenced Instances	
		PPS Exception Management	
		Intermittently Connected Modality	
	Performed Procedure Step Manager	Cardiac Cath	
	Performed Procedure Step Manager	none	

Figure 4.11 – IHE Integration Statement Example [DCM4CHE]

The most relevant document for this implementation is the Radiology Technical Framework (Vol I. - Integration Profiles, Vol II. – Transactions and Vol III. Transactions – continued), which contains following integration profiles:

- Radiology Scheduled Workflow (SWF)
- Patient Information Reconciliation (PIR)
- Consistent Presentation of Images (CPI)
- Presentation of Grouped Procedures (PGP)
- Access to Radiology Information (ARI)
- Key Image Note (KIN)
- Simple Image and Numeric Report (SINR)
- Charge Posting (CHG)
- Post-processing Workflow (PWF)
- Reporting Workflow (RWF)
- Evidence Documents (ED)
- Portable Data for Imaging (PDI)
- Nuclear Medicine Image
- Cross-enterprise Document Sharing for Imaging (XDS-I)
- Mammography Image

- Import Reconciliation Workflow (IRWF)

The relevant integration profiles and actors from this technical framework are mentioned and explained if needed in appropriate places in the text; the full reference can be found in the framework itself.

The IHE already proved to be valuable and it is becoming the main and most respected source of standardization in healthcare in America. Europe, being traditionally slower to react, will have to join the train soon – as all the major healthcare vendors already do so. Many hospitals in Western Europe already began to build IHE compliant environment and require IHE compliant solutions. In the end it is the only way to keep maintainable and replaceable solutions in increasingly more complicated integration scenarios, enabling for customer choice and product specification.

5 Basic design choices

5.1 Environment

The slightly favored environment for all parts of the implementation was Java. There were several reasons for it:

- It is free – the runtime, IDEs and web containers.
- It is the default environment for most of the server applications at the hospital – most importantly of the application servers underlying the RIS implementation at the department.
- It is an interpreted language, so it is platform independent, removing the worries about specialties of the flavor of Linux that it will be eventually deployed to or the architecture of underlying hardware etc.
- It is advanced language, it is object oriented, there is a lot of resources, available libraries etc. for it.

5.2 PACS model

There are three basic models an implementation of PACS can adopt ([HUA04]):

- **Standalone PACS Model** – the images acquired on modalities/acquisition gateways are sent to the image archive. Then they are either automatically forwarded to workstations, or are pulled from the archive by the workstations. The most important thing here is that the workstations in this model have to be able to temporarily store images and to query and retrieve data from the archive. The workstations in this model are usually independent full fledged DICOM nodes.
 - The biggest advantage of this model is that the department is temporarily able to continue to perform exams even when the image archive goes down – on the basis that images from gateways can be sent directly to workstations and stored there until the main image archive functionality is restored.
 - The disadvantage is the cost of full-fledged diagnostic workstations that can act as long term DICOM storage nodes.

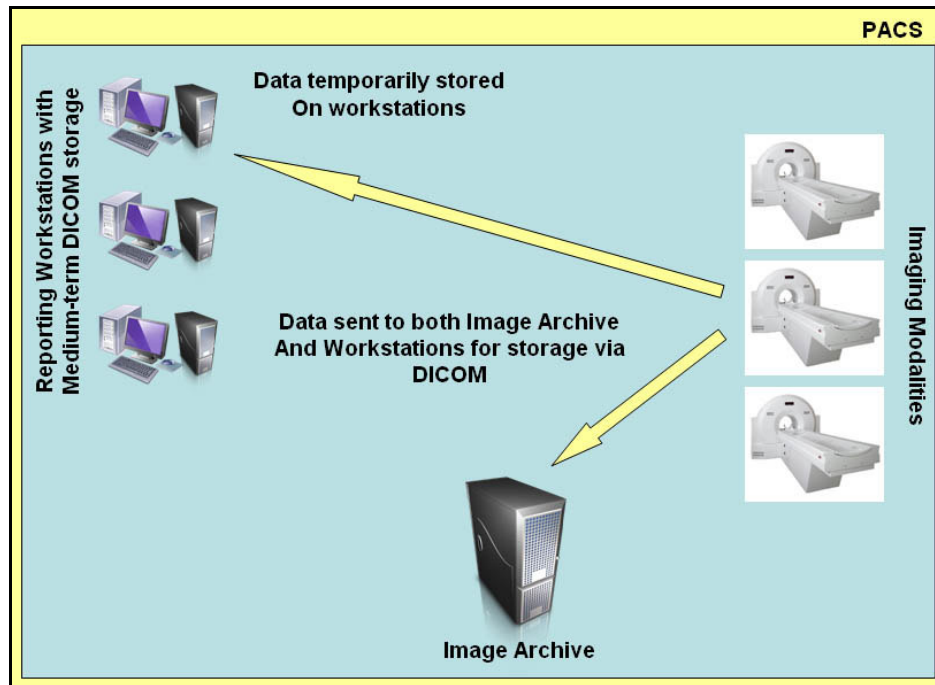


Figure 5.1 – Standalone PACS Model

- **Client/Server Model** – the images are stored solely and centrally on the image archive. In this architecture the workstations are only able to pull data from the server for a single patient at a time that is flushed before next patient is loaded, with no possibility to independently receive and store images. The patients to be retrieved are usually selected using the list of scheduled exams in the image archive worklist database.
 - The advantage of this architecture is that the list of exams to report on is visible from all workstations and thus it is simple for the physicians to find and retrieve their patients' data.
 - The disadvantage is that the image archive is a single point of failure - when it goes down or only the worklist service becomes unavailable, the system breaks down.

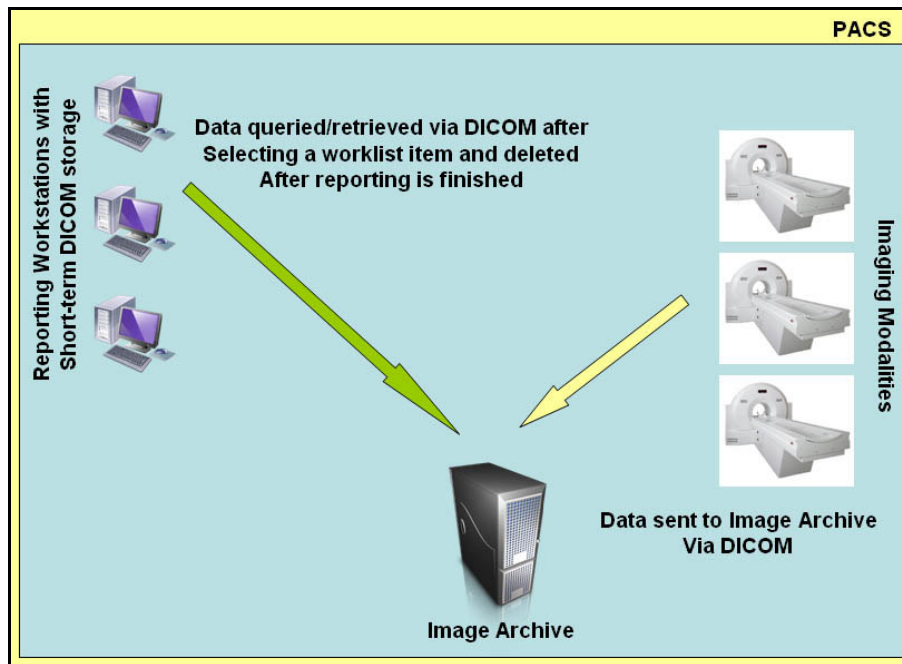


Figure 5.2 – Client-Server PACS Model

- **Web-Based Model** – the images are stored solely and centrally on the image archive and are displayed at the workstations using rich web client.
 - The advantage is that the workstations are really just clients, with no proprietary software, which saves both cost and complexity. With minimal effort this model also enables for very efficient ways of publishing images for out-of-hospital display.
 - The manipulation with the images can be very non-trivial. On classical standalone diagnostic workstations a big part of necessary image transformations is done by powerful graphic adapters and is very complex as far as the size of the pixel data and the requested speed of the operation. As it is impossible to do these transformations by in the web client itself, they have to be done by some corresponding server module of the image archive/PACS server, and the resulting images transferred to the client, which leads to high network traffic and high demands on computing power of the image archive/PACS server. This implies much higher requirements on the hardware underlying the Image Archive.
 - Another disadvantage is that the image archive is a single point of failure that, if failure occurs, paralyzes the whole department.
 - This model also couples the back-end (storage etc.) and front-end (display, diagnostic tools) logic, which means that the vendor of such system has to have corresponding expertise in both areas and that the two parts cannot be upgraded independently.

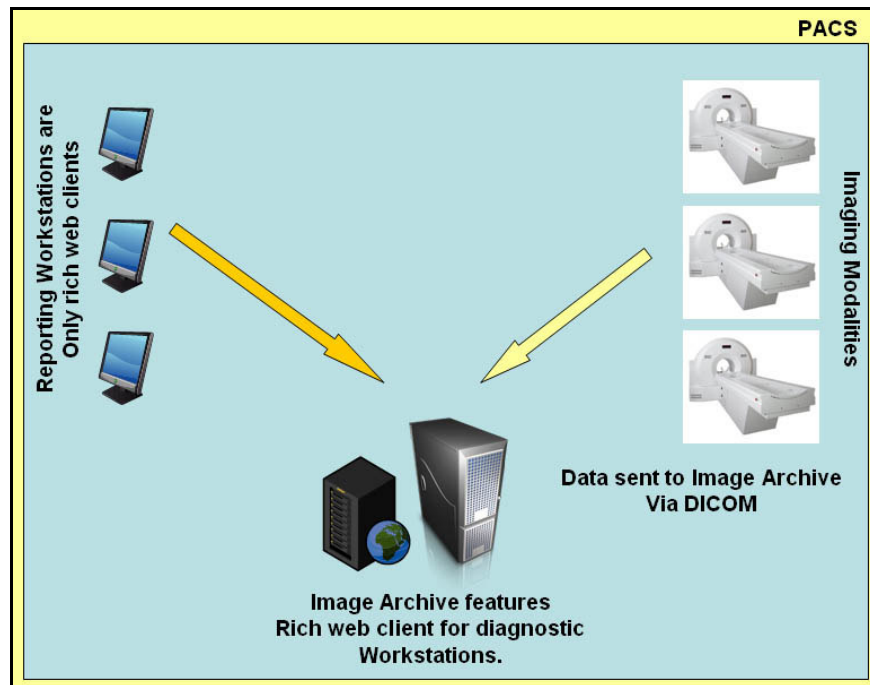


Figure 5.3 – Web PACS Model

By analyzing the needs of the department the Standalone model was picked for following reasons:

- It offers the greatest level of reliability and fitted well with the rest of the design strategy.
- The investments into full-fledged diagnostic workstations/tools were already made and the staff was used to using these. It was the easy way to start enabling to concentrate at the backend logic first.
- This model enabled for safe in-production testing during development. As long as the data were properly backed up, the department was able to survive eventual failures with minimal gaps in its schedule.

5.3 Image archive platform

The image archive is the most important building block of a PACS implementation. It has to feature a full fledged DICOM interface that the acquisition gateways and workstations communicate with. It also has to feature some interface for integration with the RIS/HIS – HL7 or else. The back-end logic inherent to the storage and management of the data is also very non-trivial – the amount of managed data is huge, many operations can be long running and the production ready solution has to be very stable, reliable and fault tolerant.

In other words it is unrealistic to implement a production ready image archive of required scale from the scrap in single-person team in realistic time. Therefore it was necessary to look for affordable – open-source in ideal case – solution.

The most important criteria to determine the fittingness of the products were:

- **Well tested** – the code has to be production ready.
- **Fitting and full featured** – how well the product suits to be the base for image archive implementation. The more required features it offers that are production ready for the required high-load scenario, the better.
- **Documentation** – no product in this range is simple, so quality documentation is extremely valuable.
- **Java** – as mentioned before in section [5.1 Environment](#), Java was a slightly favored environment.
- **License** – the product has to be available under some acceptable license.

After investigation the DICOM open-source products, following ones emerged as the most interesting candidates:

- **DCMTK DICOM toolkit** ([DCMTK])
 - Fundamentally a collection of libraries written in C/C++ that implement various parts of the DICOM standard. It offers sample implementation of various DICOM services in form of command line utilities.
 - **Well tested** - DCMTK has a decent academic background - it has been around for quite a long time. It was used at numerous DICOM demonstrations to provide central, vendor-independent image storage and worklist servers.
 - **Fitting and full featured** – there is a sample implementation of image archive built on top of DCMTK libraries – the CTN (Central Test Node) – but its main purpose is to be a test node in sample DICOM scenarios and as such it does not offer appropriate means to handle massive load. Significant effort would have to be spent to build image archive of required scale on top of the toolkit.
 - **Documentation** – the documentation of the command line utilities was very accomplished. The libraries however were not very well documented, which would make the eventual usage of them quite complicated.
 - **License** – the toolkit is available for both commercial and non-commercial use
- **jDCM** ([JDCM])
 - An implementation of DICOM in pure Java in a form of library API.
 - **Well tested** – the jDCM site contains no references to users of this library and there is no support forum. Nor there are many references to jDCM applications put in production on the internet.
 - **Fitting and full featured** – there is no image archive implementation in jDCM. It is only API for basic DICOM operations. The whole image archive stack would have to be built on top of it.

- **Documentation** – there is a decent user/developer manual available. The API is quite comprehensive at first sight as well.
- **License** – for \$990 the toolkit is available for unlimited commercial use. Evaluation usage is free and time limited, development usage is \$99.
- **Pixelmed DICOM toolkit** ([PIXELMED])
 - Another implementation of DICOM in pure Java in a form of library API. Also a simple workstation/node implementation was available.
 - **Well tested** – the code was written by David Cluine – a pioneer of DICOM and well respected member of digital imaging community (also one of the authors of [PACS DR] and the author of [DCMSR]). There was respectable user community behind with a support forum that was lively.
 - **Fitting and full featured** – no full-fledged implementation of image archive was available. The workstation implementation was a good inspiration for the implementation effort however, as it already contained implementation of the key DICOM services.
 - **Documentation** – there was a quality programmer’s documentation available.
 - **License** – the toolkit ships with BSD like license, making it possible to use the toolkit on commercial basis.
- **Dcm4che and dcm4chee** ([DCM4CHE])
 - Another implementation of DICOM in pure Java. The stack composed of dcm4che14 and dcm4che2 DICOM toolkits and of dcm4jboss (later renamed to dcm4chee) – a full fledged image archive implementation in j2ee stack built on top of dcm4che14.
 - **Well tested** – There were good references to the toolkit on the internet. The quality of the code was very good (even though lacking any comments or Javadoc doclets). There was a small user community behind with not much life in the forums.
 - **Fitting and full featured** – there was a full fledged image archive implementation available built on top of the toolkit.
 - **Documentation** – there was virtually no documentation whatsoever – code level or any other level.
 - **License** – MPL (Mozilla Public License) / GPL (General Public License) / LGPL (Lesser - GPL) triple license.

Finally, the dcm4chee (former dcm4jboss) image archive and clinical data manager was chosen to form the basis for the implementation. At the time this was a hard decision – the image archive was implemented in J2EE stack and ran inside JBoss Application Server [JBOSS], there was no documentation neither of the image archive, nor the underlying DICOM implementation and the support available from user forums was also very low. The initial learning process thus was very difficult.

It was an inspired decision however; unlike all the other projects mentioned above the dcm4chee project was growing from then on and now it is probably the best open source product in the field ([VASQ]). The user and development community has widened from that time as did the documentation. A project homepage and wiki eventually emerged. I became an active member of the user community with over 100 contributions to the user forums, several contributions to the community wiki and many found bug/improvement issues.

5.4 Reliability

One of the most important aspects of the new PACS implementation was the level of reliability it would provide – how the backup of the data would be handled and how costly would be the eventual restore.

One important choice concerning reliability was already mentioned here – the standalone PACS model as described in section [5.2 PACS model](#) allows the department to temporarily continue daily work even without the main archive server. However, without the worklist service that the image archive provides, the operation is less fluent and the capacity and reliability of the workstations where the data are temporarily stored is limited. In case of failure, the operation should ideally be restored as soon as possible.

Several strategies emerged as possible solutions to achieve high reliability. These are the most important criteria that were used for evaluating various backup strategies:

- **Probability of data loss** – the timeliness and reliability of the backup process
- **Time to restore functionality** – time from failure detection to functionality restored
- **Time to restore whole system** – time from failure to restore of original reliable state
- **Backup operation complexity** – the machine time spent on backup, the impact on responsiveness of the solution due to backup
- **Administration effort** – administrative and organizational requirements of restore process and of basic administration of the solution (scheduled maintenance of servers etc.)

In the following text following terminology is used: **front-end** denotes the DICOM image archive that the department communicates with, **back-end** or **backup** denotes any backup solution containing the duplicates or dumps of the data.

The estimated times and assumptions are based on experiments with a dcm4chee archive running above a MySQL [MYSQL] database that contains all the data produced so far at the department on production machines that are described in detail in section [6.2 Hardware, Operating System, File System and RAID](#). The produced data amounts to over 1.5TB of compressed files and over 50 million rows in the database. The size of the whole db is well over 20GB.

5.4.1 DB dump (hot-copy/mirror) and rsync

This is probably the simplest way of handling backup – also very general and often used. On daily basis – usually during off time – scripts are run by means similar to cron daemon to create a dump of the application database that is stored on some remote server and to synchronize application files to some remote location.

The obvious benefit of this solution is the ultimate simplicity. The database vendors typically provide means for dumping and restoring the database. The means for file synchronization are also easily available – on Linux it is the rsync shell utility.

With some database engines it is possible to create live database mirrors. The live-copy/mirroring solutions can provide very convenient and fast way of backing up the database. The tradeoff is the performance overhead inherent to maintaining the mirror and potential risk of both databases being corrupted at the same time by some general failure that affects both master database and the mirror. With most database servers mirroring is primarily intended for increasing database availability - therefore this method is not discussed here and was not considered as a possible solution for database backup.

Some database servers [MySQL] provide hot-copy utilities to copy databases much faster than the dump-restore process allows. The hot-copy functionally falls into same category as dump-restore – the tradeoff for enhanced recovery speed is the compatibility problems occurring more likely with eventual upgrade of the database engine version.

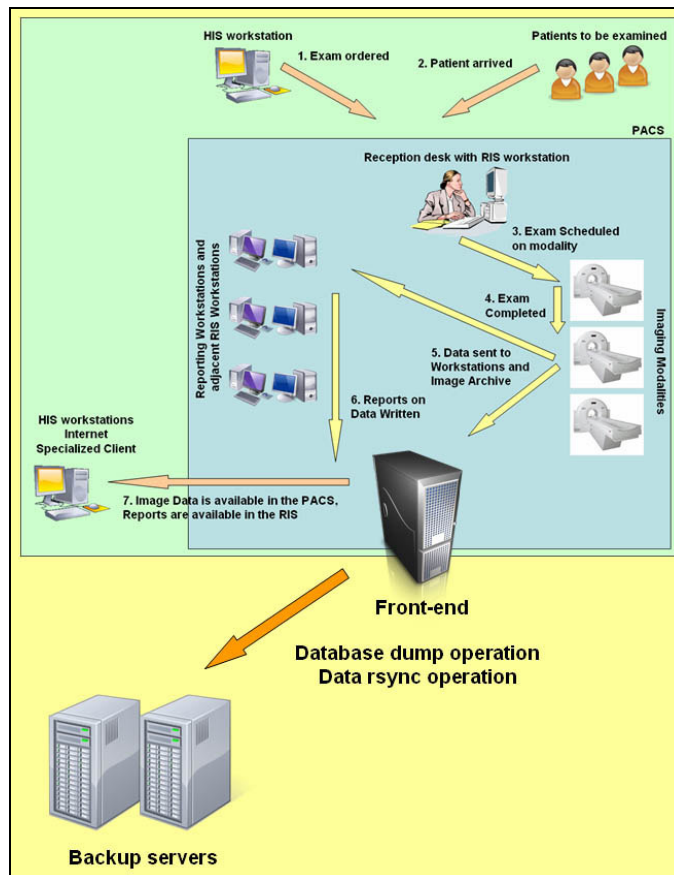


Figure 5.4 – Reliability via Database Dump and Rsync

- **Probability of data loss** – to prevent possible loss of data an additional periodic check has to be run to ensure that the daily backup was successfully created. Some data still may get lost – in case the front-end breaks down during the on time, the recent data might not be backed up. Also there is no checking of the consistency of the backup against potential disk failure. Having two backups instead of one would improve the level of reliability here.
- **Time to restore functionality** – comprises of setting up a new machine, copying the backed up data there and restoring the database. This can result in very significant time – the copying of files may take over a day, the restore of the database takes over two hours. Some time could be saved here by not copying the file data but mounting the backup directory to the systems via NFS or other networked file system instead.
- **Time to restore whole system** – is the same as above, only the actual copying of the files is necessary here.
- **Backup operation complexity** – depends on implementation. To run rsync on entire folder with the data results in huge overhead. Using the specific structure of the stored data in which the new data can be easily located the complexity can be reduced dramatically.
- **Administration effort** – depends on actual implementation. The restore of a machine from backed-up data can easily be automated. Both routines – db dump/restore and rsync – are very simple, so the knowledge base necessary for managing backup solution based on these operations is minimal.

5.4.2 Distributed data storage

Another way to back up the data is to use some kind of distributed data storage mechanism. Two possible solutions emerged:

- To use some distributed filesystem
- To implement (or find) an intelligent I/O layer that would handle the distribution by its own means.

Both ways share one ultimate disadvantage – both could only be used with the files, not with the database. On the other hand the file and database backup are very independent problems, so actually finding an optimal strategy for files only would still be great. For the database the dump/restore approach still could be used.

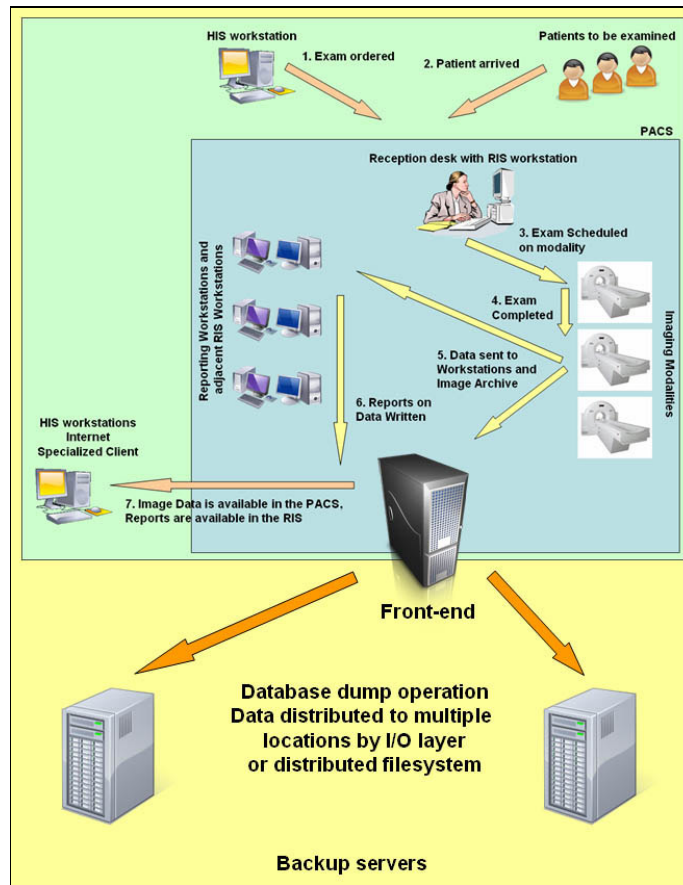


Figure 5.5 – Reliability via Database Dump and Distributed Filesystem or Intelligent I/O layer

For the distributed filesystem way following analysis applies:

- **Probability of data loss** – the distributed filesystems are very complex applications. For an expert administrator who understands some similar application in full depth it's possible to rely on such solution in production use, guarantee correctness of its operation and estimate/minimize any possible risks. No such expertise or management skills were at hand, so after cautious thought, this solution was dropped as unfitting. The other characteristics of this method also mark it as suboptimal.
- **Time to restore functionality** – depends on actual implementation and on the kind of failure. Potentially the system could continue work even if local underlying disk infrastructure (not the system disk of course) goes down.
- **Time to restore whole system** – depends on implementation. Almost certainly it would require a non-trivial administration effort.
- **Backup operation complexity** – depends on implementation; however it would surely slow down each operation of the server requiring disk access. As the volume of data stored on the disk is huge as is the number of files the application manages, this would come very costly.

- **Administration effort** – would be substantial – this would be one another complex system to be understood and administrated.

The analysis for implementing a distributed file storage system or using some legacy implementation:

- **Probability of data loss** – for a simple and robust implementation the level of surety could be acceptable. The system could maintain several copies of the files on several locations and check periodically whether they exist and are consistent.
- **Time to restore functionality** – could be minimal. The data could be read from multiple locations depending on availability.
- **Time to restore whole system** – could be minimal or rather not significant. The routines responsible for distributing the data to multiple locations could run independently of the main application (DICOM server) and thus failure of single location could be dealt with automatically and transparently. With more than two locations for a single file the reliability of the system would not be compromised in case of failure of any single location, thus the time to restore previous state would be of less relevance.
- **Backup operation complexity** – the solution would certainly have some impact on the I/O throughput.
- **Administration effort** – depends on implementation. However the complexity of the whole solution would certainly rise, which would inevitably complicate the management process.

The various redundant RAID types also fall into this category. They will be discussed later when describing the actual platform underlying the archive. They can hardly be thought of as providing high level of reliability – it is rather an enhancement for the reliability of a single machine than a storage solution.

5.4.3 HSM – storage of tarballs on tape

HSM (Hierarchical Storage Management) is policy-based management of file backup and archiving in a way that uses storage devices economically and without the user needing to be aware of when files are being retrieved from backup storage media.

The dcm4chee archive offers a HSM implementation. Individual studies are packed using tar and sent to external storage such as tape manager. When a study is requested and it is not present at the local file system, the HSM module is able to fetch it from the external storage to local disk. The reliability issues thus are not resolved by HSM, but merely transferred from front-end to back-end.

It is worth noting that the HSM system of dcm4chee relies on the database, in which all the information about stored objects has to be present. Thus this backup strategy requires some independent backup mechanism for the database.

The external storage location can be a redundant disk-based system or a tape manager. One apparent disadvantage of this approach is that the legacy storage systems or tape managers are usually expensive.

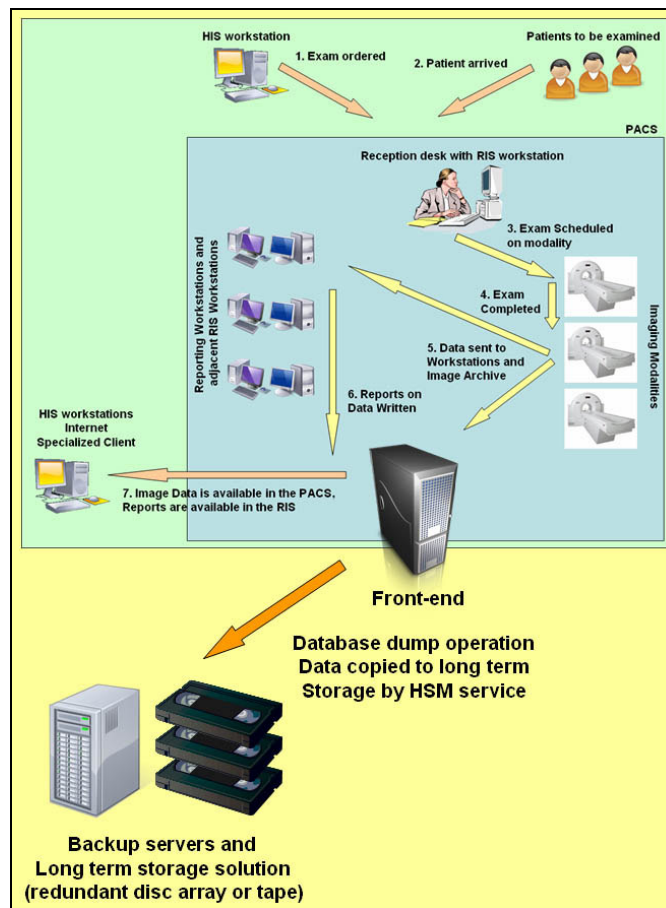


Figure 5.6 – Reliability via HSM module

- **Probability of data loss** – depends on back-end solution – however either disk based system or a tape manager are destined to reside on one place, unless they should be even more expensive, which increases the possibility of damage by catastrophe. The tapes also do age and the data eventually get corrupted – and there is no way how to easily check data for errors and how to replace the storage media for new reliable ones (which is possible with the disks). Also it is usual in HSM scenarios, that the data are pushed to external storage only after some time – after aging etc. – so some complementary short term backup mechanism has to be applied for data that were not yet moved to external storage. In case of failure of the external storage system there is no backup during the service/repair time, unless some additional strategies exist.
- **Time to restore functionality** – depends on database backup mechanism. A clear box with restored production database is enough to restore functionality, as any requested data will be pulled back from the external storage.
- **Time to restore whole system** – the same as the time to restore functionality.

- **Backup operation complexity** – the overhead of packaging tars and moving them to external storage.
- **Administration effort** – could be minimal. There is one problem though, and that is, that the connection between external storage and the archive is proprietary and depends on the image archive implementation. Should something change in the logic of the corresponding module of dcm4chee in future versions, the upgrade could become difficult.

5.4.4 DICOM forwarding

DICOM forwarding is a very common concept in DICOM networks. A node receives data for storage, stores them and then forwards them to number of pre-configured locations. Dcm4chee also provides this functionality.

A backup scenario can be setup easily using DICOM forwarding – the data received on front-end are forwarded to the back-end – both running dcm4chee archive.

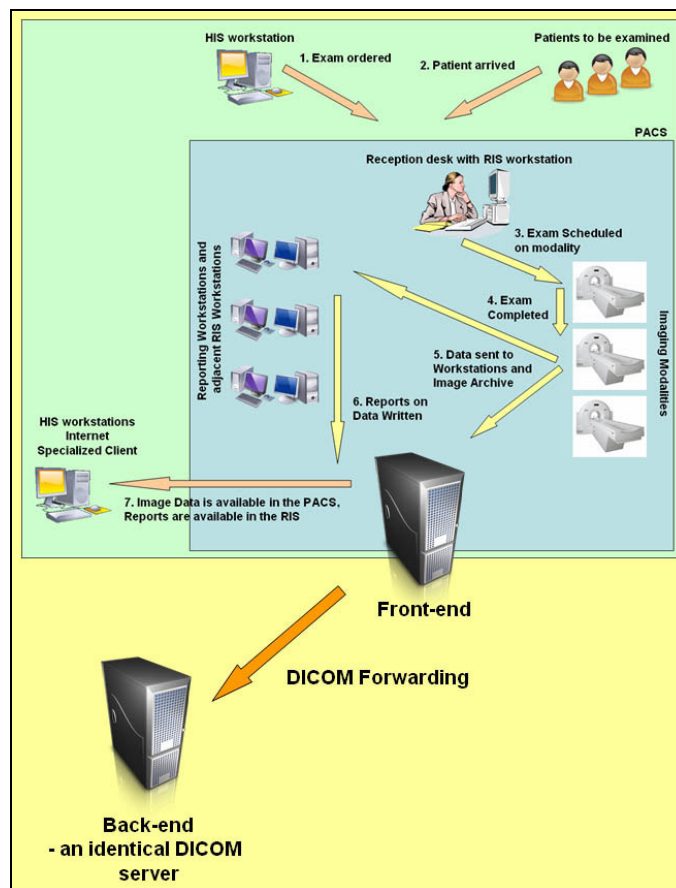


Figure 5.7 - Reliability via DICOM forwarding

- **Probability of data loss** – a daily periodic check has to be set up to check that the daily data were properly backed up and that the backup is up and running. Because

dcm4chee comes with a service that allows for checking of consistency using md5 checksums of stored files, the consistency of the backup can be verified periodically. The database exists in two instances, as the backup itself is a full-fledged dcm4chee archive.

- ***Time to restore functionality*** – With appropriate policies it can be minimal, as the back-end is a live copy of the front-end and can immediately take over.
- ***Time to restore whole system*** – to restore the system original state in case of failure of either back-end or the front-end server a mirror of the remaining server has to be done. This operation however can be automated and basically amounts to the time necessary for database backup and restore and for transfer of files stored at the system.
- ***Backup operation complexity*** – every stored object is sent to the backup. This amounts to non-trivial performance overhead of the send operation. However the send operation is far less costly than the receive operation in terms of database load, so it is not a big problem. In case the server is under too heavy a load, the send requests can be postponed and processed later.
- ***Administration effort*** – is minimal. The connection between the boxes is standard DICOM, so it is not implementation specific. The replication operation in case of server failure can be automated.

In the end the DICOM forwarding based model was chosen as a base for the reliability solution. It combines a very simple design with solid reliability. Both database and the data are backed up in a separate independent instance (actually there are two backup servers in the final implementation) of the archive with minimal chance of single failure causing the crash of both front-end and back-end instance. The solution provides minimal off-time when failure occurs. Also compared to the HSM approach, which came second best in the evaluation, it is much cheaper with similar or higher level of reliability.

There is one more positive aspect about the forwarding based solution, and that is the fact it is very homogenous and self contained. The front and back-end servers are almost identical – any additional routines can be set up on all machines identically – with no other machines contributing to the functionality. Having almost identical machines that are being able to be swapped one for another leads to intriguing concepts of hot-swap or even hi-availability environment.

5.5 Integration

There are several integration interfaces for a PACS implementation:

- Internal interfaces between image archive and modalities and workstations inside the system
- RIS/HIS interface to support department scheduled workflow
- HIS interface to enable for query and display of stored image data from the HIS

- Interface for sharing documents with other healthcare facilities

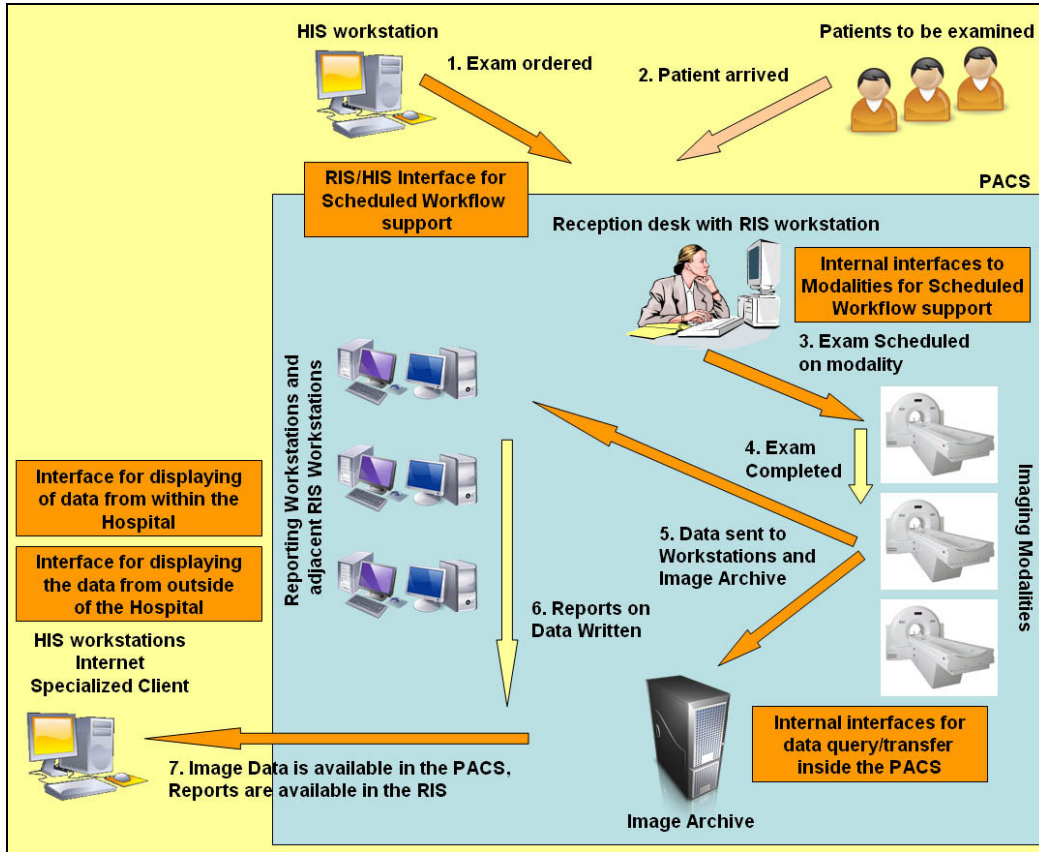


Figure 5.8 – PACS Integration Interfaces

5.5.1 Internal Interfaces

Most of the modalities and workstations nowadays are DICOM compliant, so the communication interfaces between them and image archive is standard DICOM interface. However some older modalities may require specific non-DICOM interface for their worklist and data transfer services.

There is one such modality in the target environment at the PET Center of the Na Homolce hospital - the Siemens Ecat-Exact PET scanner. This scanner does not support DICOM communication, therefore special means of communication had to be created for this camera.

5.5.2 Workflow Management Interface

The patient visits to the department are ordered from outside the department – by requesting physicians from other departments of the hospital, or by administrative staff when a patient who is in care of other hospital comes for radiology examination to be carried out at the department. In both cases the exam is ordered in the HIS.

The HIS fills the information about requested examination to appropriate database table of the RIS system. At the reception desk of the department the administrative staff of the department is able to manage (create, modify, reschedule, delete) existing orders at the RIS workstation and eventually – upon patient arrival to the department – initiate corresponding examination.

The RIS notifies the PACS about examination order that should be carried out at some of the modalities. The PACS creates corresponding records in the image archive DICOM worklist database (or, in case of DICOM non-conformant modality uses other means to schedule the exam). The modalities (more accurately their acquisition gateways) periodically query the worklist database and update the list of scheduled examinations. The modality operators select an examination from the list of scheduled examinations and start the actual procedure. Each record in the worklist bears information about corresponding patient (patient name, ID, birth date etc.) and about the exact procedure to be carried out (accession number, procedure ID, procedure description etc.). The DICOM service class facilitating the worklist database and the communication channel through which the modalities are able to pull their scheduled exams from the database is the DICOM Modality Worklist Service Class.

The data produced by the modality as a result of processing a single worklist item is identified by the information contained in the item – this is how the linking between eHR of corresponding patient in the HIS and the image data stored in the PACS is maintained.

In the IHE technical framework this flow of operation corresponds to Scheduled Workflow Integration Profile (SWF).

There are some special cases in which the basic workflow is not sufficient and special measures have to take place. These are the typical situations when this is necessary:

- A patient arrives with an invalid ID – this is quite common with the foreigners, who sometimes only get identified by their name and short ID corresponding to their date of birth. The PACS does not accept such IDs and schedules the exams for a patient with proprietary generated unique ID. To facilitate the linking of the data to patient eHR the data has to be verified later by additional routine and the patient ID in the PACS database is eventually updated with some appropriate value. This corresponds to IHE Patient Information Reconciliation (PIR) integration profile.
- An additional examination takes place possibly on a different modality than the one on which the exam was originally scheduled. In some special cases it is possible that the physicians at the department decide to take another additional examination other than was scheduled in the HIS/RIS. The common praxis is that they do so by manually initiating the examination at the modality – not by generating a new worklist item. Such examination thus does not get a valid identification – any information contained in the data is human filled through the console adjacent to the modality and thus has to be verified and eventually corrected by additional routine. This case corresponds to Image Acquisition Completed without Scheduling at Department System Scheduler/Order Filler case of the IHE Patient Information Reconciliation (PIR) integration profile.

- The worklist mechanism fails – the worklist service is temporarily out of order – a means have to be ready to reconcile the produced data and link them to appropriate eHRs of corresponding patients. In the standalone PACS model it is possible that the department continues work even if the image archive worklist service is down – due to failure of the image archive itself or some problems in the HIS-RIS-PACS communication chain. Ultimately this results to identical situation as the one above.
- With DICOM non-conformant modalities some of the identification metadata can be lost during conversion process to DICOM file format. Such is the case with the non-conformant modality in the target department. Again the metadata have to be updated later by additional routine to ensure correct linking to eHR.

There are more possible approaches to above mentioned issues.

The IHE offers recommendations for these situations, however a different approach was chosen here. In general it can be noted that the IHE advocates much closer integrated scenario in which most of the communication between the PACS and the appropriate HIS and RIS modules is two way and the modules feature some support for typical situations that can occur.

In the target environment at the PET Center however emphasis has been put rather to facilitate such means, that would enable for very autonomous function of the department with minimal or no communication feedback to the superior systems. The fundamental reasoning behind this decision was that the changes are easier (technically, but also politically) made inside the department and can be done independently on other systems.

Another important aspect when choosing appropriate integration strategy was that the timeliness of the operations such as patient reconciliation or update of study identification information is not critical and can happen on daily basis during off-time after the work hours.

Thus an alternative strategy was chosen, that combines the support for all mentioned cases with the advantage of being much less demanding in the amount of supported communication interfaces by the RIS/HIS and modalities. A special service was setup to be run on daily basis after work hours, that pulls all the daily exam orders from the RIS and processes all daily studies verifying that data corresponding to all daily exam orders are present at the image archive, are complete and well identified, eventually updating their attributes in the image archive. The routine produces a daily report with eventual unresolved issues to be printed out each morning and to be checked by the administrative staff that could initiate eventual manual corrections to be made.

5.5.3 Interface to access image data from within and outside of the hospital

Once the data are stored in the image archive and are well identified, it is necessary that they can be accessed in a convenient way from within and outside of the hospital.

The DICOM is designed to be run in a secure department subnet and to be accessed only by few trusted peers. The security and authentication management services available in DICOM are limited and do not allow for user-role based approach typical for more advanced access policies. Also the Query Retrieve Service Class of DICOM is designed in a way, that the client (Application Entity Title) requesting the retrieval of the data has to be known to the DICOM server on which the data is stored. Thus in general it is not optimal to have DICOM client at every clinical workstation throughout the hospital querying and retrieving the images from the image archive of the corresponding PACS. A concept of some kind of proxy server or web server is usually used to access the data from outside of the department that provides the necessary security and shields the image archive from the rest of the world. A corresponding client is integrated into communicated peers, that accesses the proxy to actually retrieve and display the images at target clinical station.

There are several design options concerning the way in which the proxy element accesses the image archive:

- **Database and proprietary access** – the proxy mechanism can be proprietary for the individual image archive implementation and access directly the database and filesystem underlying the implementation. In many cases the proxy mechanism is incorporated into the image archive solution itself – some systems feature rich web clients to query and display data stored on them – in which case the tight integration is appropriate and very usual. But often this “database to database” interface is used also by separate systems or by systems proprietary written to extend the functionality of the image archive to support the proxy mechanism. The disadvantage of this approach is that with the change of the image archive implementation the whole mechanism has to change.
- **DICOM access** – the proxy mechanism can connect to the image archive using DICOM means. The DICOM supports some query mechanisms using DICOM Query/Retrieve Service Class which enables querying of stored DICOM objects, the actual data can be pulled from the DICOM server using the DICOM WADO (Web Access to DICOM objects) Service, which enables for retrieve of images stored in the archive via HTTP protocol. DICOM WADO supports a range of formats (in DICOM these are called Transfer Syntaxes) in which to acquire the images ranging from DICOM uncompressed files to standard JPEG files. The obvious advantage is that such proxy mechanism can easily be used on any DICOM server that further supports DICOM WADO. Therefore this solution is much more general, reusable and eventually allows for more coexistent PACS systems of the institution to be accessed in a uniform fashion.

The implemented proxy mechanism uses the DICOM access mechanism. The data pulled from the server in one of supported formats using this way are served to a specialized client using proprietary communication protocol. The proxy element in the process is represented by the application server inside the department that accesses the Image Archive using DICOM compliant way. The client is actually a specialized tool for accessing and managing medical

patient documentation, for which a specialized module for communication and displaying radiology image documentation was written.

The advantage of using specialized encrypted protocol for transferring data into this client is that it can be easily used from within and outside of the hospital. Thus the same strategy can be used for accessing data from other affiliated hospitals that send their patients to target department for examination. This solution also enables for working-from-home mode of work for reporting physicians.

In case some specialized display capabilities are requested by the physician requesting the images, it is possible to download a zip-ball of target study in uncompressed DICOM which can be imported into any standard DICOM viewer that the physician is used to work with.

The one missing piece in the scenario is the access to the images via hospital intranet. The above mentioned solution based on the specialized client application for accessing and managing medical patient documentation covers most of the use cases; however enabling the intranet based access for the simplest way of access is still a task that needs to be solved to achieve complete integration. Very recently (beginning of December 2007) a new web-viewer extension to dcm4chee became available that could prove valuable enabling for image display via internet.

IHE makes use of the XDS (Cross-enterprise Data Sharing) and XDS-I (Cross-enterprise Data Sharing for Imaging) standards for image data sharing. Without doubt it is the most sophisticated way to share images and medical image related data, offering most complete feature set. The XDS is very complex standard understanding of which is another challenging task. Currently a new (and most likely the first and only) open-source XDS registry/repository making use of dcm4che platform was released by the MARiS project team ([MARIS]). Further inquiries into XDS are in plan, however currently the developed approach is preferable – it integrates seamlessly with the rest of medical documentation, it is well established and is the simple way forward. The open source repository was only released as recently as 9.7.2007!

It is good to note that the service for enabling cross-enterprise sharing of medical images can also be purchased. Both source and target hospital need to connect into existing infrastructure provided by the service vendor – this usually requires a dedicated proxy node inside the PACS (such is the case with the biggest provider of such service in Czech – the ePACS project ([EPACS])). This way can be considered an alternative to offering a full fledged secure interface for foreign users. In current situation however this approach is rather disadvantageous – it does not allow for the complete eHR of the corresponding patient to be accessed (even though partial information could be added to the images using DICOM Structured Reporting). Other issues are the cost, possible further integration issues, dependency on the vendor of that service and ease of use – as some of the physicians viewing the images may not have a DICOM enabled full fledged clinical workstation at hand.

5.6 Platform

Some choices had to be made as for the platform, on which the PACS solution was to be built – namely the hardware, RAID model, operating system, filesystem and database underlying the image archive, and network topology and placement of front and back-end servers. Some of the original choices are based on limited testing and will be potentially revised when refreshing the servers in near future.

5.6.1 Network

There were not many doubts about what network stack to use – the chosen image archive platform only supports TCP/IP. The department already had gigabit ethernet infrastructure into which the servers could be plugged.

Some changes were necessary due to choice concerning the actual physical location of the servers – the main front-end server and one of the backup servers were placed in two separate rooms of the department with fire protection and permanent power supply. The second backup server was placed to adjacent facility next to the hospital into which there was a connection via optical fiber. As all the servers needed to be in the department subnet, appropriate changes had to be made to the hospital network switches, routers and firewalls.

5.6.2 Hardware, Operating System, File System and RAID

As the runtime environment of the image archive platform was Java, there was a relative freedom concerning the underlying operating system. The RHEL 4 (Red Hat Enterprise Linux 4) was chosen – mostly because most of the servers in the hospital run this system, so there is a lot of expertise and administration tools available.

The hardware purchased to run the archives on were 64 bit 2 processor machines on AMD Athlon 64 X2 Dual Core Processor 3800+, with 4G of RAM.

For choosing the RAID model and filesystem the following facts were important:

- The most time consuming operation is the storage of files. The most time consuming sub-operations during storage of a single instance (file – image slice) are the database query to make sure that such instance is not in the archive and the eventual compression of the image data.
- The machines contain 8-disk array each – the probability of eventual failure of some disk is very high. Due to hot-swap capability of the backup servers and the fact that there are two redundant servers it is very easy to perform a scheduled maintenance of any of the servers – even with one of the servers out of operation there is still a full-fledged backup setup. An expected outage (such as the replacing of a failed disk after failure is reported by RAID) is thus trivial to manage. On the other hand an unexpected outage can be a serious problem. The swap of the servers is quite simple

operation; however it should be performed by knowledgeable personnel, which might not be available at the time. Therefore some level of redundancy was preferred.

- No tests were made with different configurations – in close future the servers are going to be refreshed - the RAM will be upgraded to 8G, the operating system to RHEL 5. This will be a place to further experiments, especially with the filesystems.

The RAID chosen for the underlying disk array was RAID-6 with both the database and files on the raid volume. The ability to continue operation even after two disc crash is very welcome. The raid is software-managed.

For filesystem the ext3 filesystem was chosen. It offers relatively slow access, the tradeoff being relatively little load on CPU and fast fsck times. The application is both CPU and I/O demanding, so it is an acceptable compromise. No tests were made with other filesystems and further optimizations attempts regarding filesystem choice will be made when refreshing the servers.

5.6.3 The Database

The database underlying the image archive is the performance bottleneck of the whole application. The maintained database tables are huge – depending on the chosen database engine the size of the database ranges between 10 –30GB, with over 50M records in all tables.

As the application runs in the J2EE container and uses CMP (Container Managed Persistence) model as the default data access model, the database sources are transparent to the application and virtually any JDBC enabled datasource can be used – or could be; the situation is complicated by separate db mappings for some special query commands, that use the native sql queries. The engine for generating the queries is abstracted in the way that it supports several datasources – Oracle, MySQL, PostgreSQL, DB2 and MSSQL.

A significant effort was spent on trying to enable support for Firebird SQL database engine, and eventually succeeded after having to implement appropriate extension for the JBoss CMP engine (some features, such as auto-increment behavior, are database specific and the container has to know how exactly to use them) and modifying the SQL builder class underlying the dcm4chee Image Archive implementation.

The three datasources that were considered to be used with the dcm4chee Image Archive implementation were Firebird [FIREBIRD], MySQL [MYSQL] and PostgreSQL [POSTGRE].

The first out of equation was Firebird. The Firebird database engine is based on full fledged commercial InterBase [IB] database engine. It offers an accomplished set of features, such as great set of tools for db administration or versioning support for transaction isolation, which make it one of the best-featured free databases. What ruled firebird out were the permanent issues with the JDBC driver, which could not cope with the number of connections opened by the pooling service of JBoss. Even after many configuration attempts and switching from classic server release to super server release the issues did not stop. Furthermore the solution

was compromised by the necessity to patch each update of image archive to enable firebird support. It was interesting mission though proving, how much truth there is about database transparency in a J2EE container.

The final decision had to be made between PostgreSQL and MySQL engines. Both engines offer similar and rich feature set. Thus the most important criteria to distinguish between the two engines were fittingness for high-performance production use, quality of JDBC support and performance in target conditions. Out of the two the MySQL is clearly the one considered more production ready for large scale usage. It features much wider user community and has much more references to large-scale enterprise applications [MYSQL]. When comparing the performance of the two databases in target environment (sending one year of department data production for storage to an Image Archive running dcm4chee over one and than the other database engine) the results were nearly identical (less than 10% in 12 hour lasting operation in favor of MySQL). Most importantly MySQL seems to be putting much more emphasis on its JDBC driver, with more frequent updates and full JDBC 4 support (unlike the PostgreSQL). So in the end it was MySQL that was picked up as the engine of choice.

6 Implementation

6.1 Implementation overview

Following picture summarizes the design choices described in the [previous chapter](#).

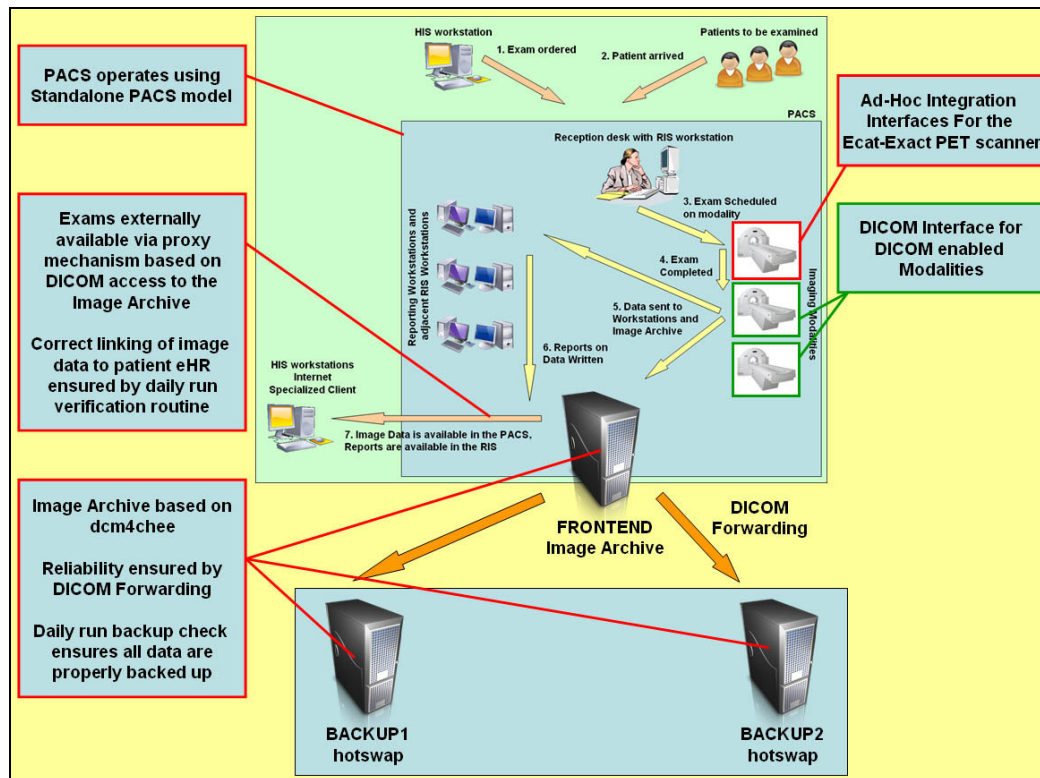


Figure 6.1 – Implementation Overview

6.2 Image Archive Implementation

As mentioned in [5.3 Image archive platform](#), the image platform of choice was the dcm4chee implementation.

Dcm4chee is part of family of DICOM related open-source products developed by Gunter Zeilinger and team of associated developers hosted at SourceForge [SF], which further contains dcm4che-1.4 – a DICOM implementation and toolkit and a later rewrite of dcm4che-1.4, the dcm4che2 toolkit. The “che” part of the name refers to Che Guevara pointing out the revolutionary nature of open-source. It is a riddle though – the “che” pronunciation is very similar to “j”, and the “4j” suffix is a very common way of naming Java targeted products.

Dcm4che-1.4 was first conceived back in 2000. Originally, the dcm4che-1.4 was designed with the intention of submitting it to Sun as a JSR (Java Specification Request) for a standard Java-based DICOM API. The JSR never happened, but the project started gaining in popularity among the Java DICOM development crowd. Shortly after, the dcm4jboss archive

was developed. This sub-project implemented the IHE Image Manager and Image Archive actors. Recently the dcm4jboss project was renamed to dcm4chee - the extra "e" referring to its enterprise usage.

In the past few years the project has been rapidly growing in the rankings of similar products and now it is possibly the most accomplished DICOM related open-source product in many aspects ([VASQ]). As a part of this development a dcm4che.org website and wiki were founded, where significant documentation effort is being made to make the products available to wider audience.

Most of the projects mentioned earlier in section [5.3 Image archive platform](#) as possible candidates to form the platform for this project are not experiencing much growth recently, with very few updates and limited user community. The Dcm4che products on the other hand - and dcm4chee in particular - are experiencing rapid development. Again it has to be said, that picking dcm4chee (dcm4jboss at the time) was a very inspired decision.

6.2.1 Dcm4chee image archive architecture

As noted before in section [5.3 Image archive platform](#), at the core of the dcm4chee stack is the dcm4che-1.4 DICOM toolkit, which is an implementation of DICOM. As such it offers an object model to represent DICOM model of real world - the DICOM objects, means to manipulate DICOM datasets, API to facilitate DICOM network communication and implementations of all DICOM services.

The image archive itself is an enterprise Java application built on top of the dcm4che14 toolkit designed to run in the JBoss Application server. At the core of the application stack there is the database model and Entity Beans façade to represent DICOM objects (Patients, Studies, Series, Instances, Modality Worklist Items and many more) stored in the archive.

Also mentioned before in section [5.3 Image archive platform](#), are the several database options available for the archive, including Oracle, MySQL, DB2, PostgreSQL or MSSQL.

The EJB façade further contains Session Beans that facilitate the main functionality – the DICOM service functionality such as storage of DICOM objects and also all the other help functionality the services of the server use.

On top of the EJB façade is the actual archive implementation, which is represented by a set of XMBeans (JBoss JMX Model Beans implementation). The XMBeans are objects that are managed by the JBoss application server XMBEAN Container via JMX (Java Management Extension [JMX]), which allows for invocation of their exported functions and setting of their exported properties via JMX (or via RMI adapter of the server).

MBean Name: **Domain Name:** dcm4chee.archive
service: StoreScp
MBean Java Class: org.jboss.mx.modelmbean.XMBean

[Back to Agent View](#) [Refresh MBean View](#)

MBean description:
DICOM Storage SCP. Provides a DICOM storage service to receive DICOM objects from remote DICOM applications. Typically these objects are images received from modalities. You can also configure what DICOM SOP classes are accepted by the StoreScp service using the **AcceptedImageSOPClasses** and **AcceptedOtherSOPClasses** attributes. The dcm4chee.archive=DCMServer must be deployed first because the StoreScp registers itself against this server.

List of MBean attributes:

Name	Type	Access	Value	Description
CalledAETitles	java.lang.String	RW	JAPECUBE1\BACKUP1\B#	List of accepted called AE titles, separated by \ (backslash). For example, JAPECUBE1\JAPECUBE1
CallingAETitles	java.lang.String	RW	ANY	List of accepted calling AE titles, separated by \ (backslash). ANY = accept any.
WarnForCoercedAETitles	java.lang.String	RW	NONE	List of the AE titles for the Storage SCUs for which a warning status, B000, is returned if data elements were coerced. Separate multiple values by \ (backslash). NONE = no calling AE titles.
AcceptMismatchAffectedSOPInstanceUIDCallingAETitles	java.lang.String	RW	NONE	List of the AE titles for the Storage SCUs from which storage requests with Affected SOP Instance UID in the command differs from the SOP Instance UID are accepted. Separate multiple values by \ (backslash). NONE = no calling AE titles.
AcceptMissingPatientID	boolean	RW	<input type="radio"/> True <input checked="" type="radio"/> False	Accept storage of objects/images without patient ID

Figure 6.3 – XMBEAN properties managed via the JBoss JMX Console

Each XMBEAN implements a single core application service. Some of the services correspond to a DICOM or HL7 services – like the StorageSCP service that contains functionality enabling for storage of received objects corresponding to DICOM Storage Service Class in the SCP (Service Class Provider role) or MwlSCP, which provides interface for the modalities to query the worklist items stored in the server’s worklist database that corresponds to the DICOM Modality Worklist Service Class in the SCP role; other services supply internal, management or utility functionality, such as the MD5CheckService that allows for periodical checking of stored files for consistency using MD5 checksums, or the FilesystemMgt service that manages underlying storage.



Figure 6.3 – dcm4chee Services in the JBoss Application Server’s JMX Console

The core service is the DcmServer service – the application server listens on configured TCP port for DICOM communication and dispatches appropriate calls to other services.

The main supported DICOM service classes the implementation offers are:

- **Storage Service Class SCP and SCU (Service Class Provider, Service Class User)** – the SCP enables for storage of DICOM objects, the SCU enables for sending objects to be stored in some other DICOM node supporting the SCP role of this service, effectively enabling the copy (in DICOM terminology rather incorrectly referred to as “move”) operation or the forwarding functionality.
- **Query/Retrieve Service Class SCP** – this service class enables the SCU to query the objects stored in the archive using a subset of metadata information stored in the

DICOM objects (such as Patient Name, Study ID, Study Date etc.) and eventually retrieve found results using Storage Service Class.

- **Modality Worklist Service Class SCP** – the SCU of this class (the modalities) is able to query the information stored in the worklist database of the server. The modality worklists the modality operators select items from when initiating any individual examination are updated using this service. The bad thing is that DICOM does not offer means to place an item into the modality worklist – such operation has to be done by different means.
- **Storage Commitment Service Class SCP and SCU** – the SCP of this class can inform the SCU about successfully committed storage of DICOM objects. This class is used to verify successful storage of items sent between two DICOM peers – successful commitment often triggers deletion of the data on the sender or marking some move order as successfully finished.
- **Verification Service Class SCU and SCP** – a basic DICOM ping functionality verifying a valid DICOM connection.
- **DICOM WADO** – is not exactly a DICOM service class as the communication protocol of this particular service is not DICOM. The WADO service enables for accessing the DICOM objects stored in the archive using HTTP protocol. A server exposes an HTTP interface on some HTTP URL which accepts image unique identifiers and DICOM transfer syntax as GET parameters in predefined format and returns the corresponding DICOM image in the response. Alternatively the WADO can return JPEG images of predefined parameters.

The implementation also offers limited HL7 interface via the HL7Server service that listens on some TCP port for HL7 communication. Some of the messages consumed/emitted by the application are:

- **ORM^001 (Order) message type** – refers to a patient being ordered for an exam. Specific segments of the message contain information about the patient and about the exam. This message is sent to the PACS by the RIS when a patient is ordered to be examined or when the exam order is updated or should be canceled. On receive of this message the PACS schedules/updates/deletes appropriate exam in the worklist database.
- **ADT^XXX (Admission Discharge and Transfer) message types** – refers to a patient event in the hospital. These can be used to manage patient information/data stored in the archive database.
- **ORU^R01 (Observation) message type** – refers to an observation made by physician that refers to some data. The observations are usually stored in the HIS in the eHR of corresponding patient and accessed by HIS applications; however they can be also accessible directly from PACS using DICOM Structured Reporting.

So far none of the HL7 interfaces of the server is used. The only interface that would be useful in current PACS design would be the interface to schedule exams in the worklist

database – however a different client was used. The introduction of more HL7 communication and enabling more integration interfaces towards RIS and HIS is a task for further iterations.

The archive provides a web user interface (UI). The UI is accessible via role-based permissions and allows for browsing, querying and administering the data stored on the server including items in the worklist database. The query functionality enables for querying the data based on number of keys. The administrative functionality on the data includes ability to create, modify (metadata such as Patient Name or Study Description – not the pixel data) and delete items stored at the server – e.g. Patients, Studies, Series and Images. It also allows for sending of items to other DICOM nodes.

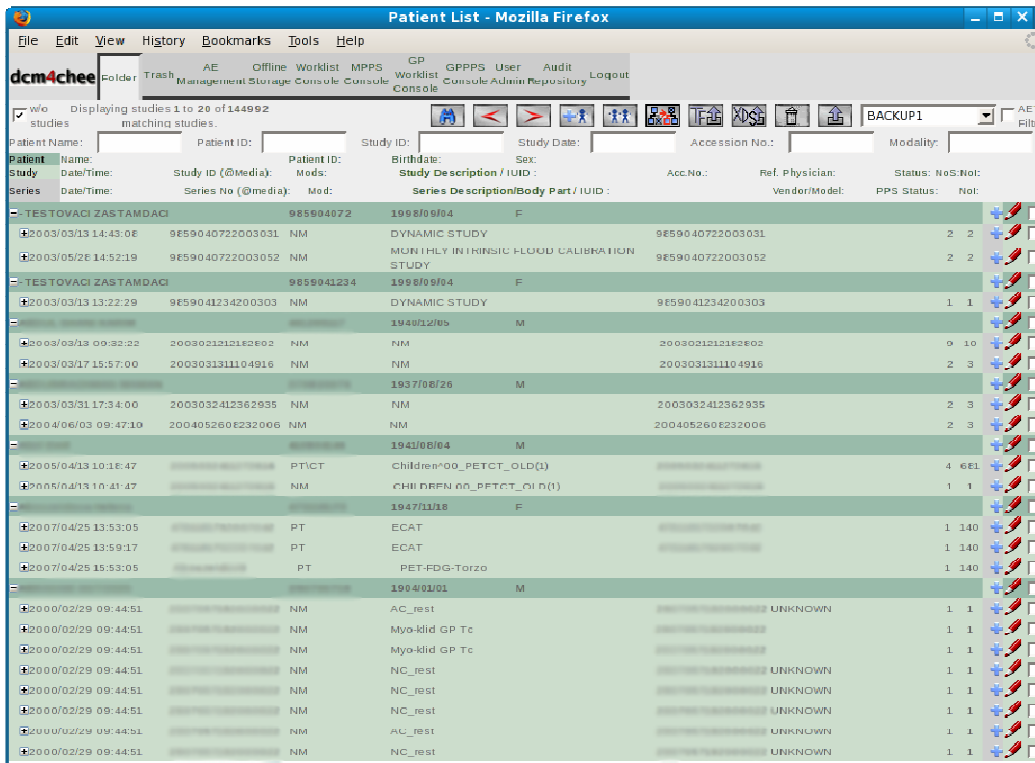


Figure 6.4 – dcm4chee web interface

The management and configuration of the archive itself is possible via the JMX console of underlying JBoss application server or via JMX client (such as twiddle.sh utility that ships with the JBoss AS) or via various deployment descriptors or configuration items (such as XSL stylesheets) on the filesystem.

The version of the archive that the current PACS implementation is using is dcm4chee-2.11.0; the most recent version is 2.13.0. The PACS architecture and implemented automation of management routines allow for relatively simple upgrade of individual servers, so the archive is continuously updated with newer versions when some important feature or bugfix is released.

6.2.2 Dcm4chee data storage system

The primary role of the archive is the storage of the data produced by the department, thus a short description of dcm4chee data storage subsystem follows.

The DICOM data produced by the department modalities take form of DICOM object hierarchy – namely the patient-study-series-instance hierarchy. Study DICOM object refers to single examination session of the patient, series refers to an individual examination (a study can contain multiple series) and single series comprises of one or many instances, which represent slices of a 3D scan. As explained in section [4.3.1 DICOM](#) the actual communicated objects are composite DICOM objects corresponding to instances (slices), but containing segments of metadata (DICOM attributes) inherent to each level of parent DICOM hierarchy.

Thus an examination of one patient resulting in one study with two series each containing 600 instances, after being sent to the image archive for storage, results in 2 times 600 composite DICOM objects being sent to the archive, each containing all the information about the actual slice it represents including the actual pixel data, but also about corresponding series, study and patient.

Both database and the filesystem contribute to storage of the data contained in the archive. The database holds predefined subsets of attributes inherent to each level of the patient-study-series-instance hierarchy and serves for lookup/management tasks. The received composite objects are stored on some underlying filesystem in the DICOM file format – this is where the full information including the pixel data is stored. Number of other DICOM objects can be stored inside the archive such as Modality Worklist Items or Modality Performed Procedure Step Items – all of these objects re stored in the database only (the database schema is illustrated by Figure 6.2).

The basic update tasks on patient-study-series-instance hierarchy (such as updating study accession number), initiated via the web user interface, the JMX functionality or by receiving corresponding DIMSE N-SET command (refer to section [4.3.1.2 DICOM Services](#) for more information about DICOM DIMSE commands) results in only the database records being updated - the underlying files are not updated. The information from the files and the database is merged when the item is communicated to the outside world. The information stored on the file-system thus can differ from that stored in the database. That is why it is necessary to keep a backup copy of the database as well – the DICOM files are not enough to reconstruct the archive – they contain all the information as of the moment, when the data was received, but in case any attributes were changed later using the services of the archive, this information is not present in the files!

When receiving a DICOM object, the server parses the DICOM dataset, extracts the metadata, eventually creates corresponding records in the parent hierarchy (if they do not exist already) and inserts a record to the instance table. The object itself is stored to the disk as a file under some configured directory. The directory structure of the storage directory can be configured organized following one of two basic patterns:

- Storage_dir/YYYY/MM/DD/HH/study/series/instances – where YYYY/MM/DD/HH represents the year, month, day and hour of receiving of the first instance of the hierarchy to the archive (the HH level is optional). The study, series, instance names are unique 8-character long hexadecimal hashes; the instances are the actual files containing all the metadata and the actual pixel data.
- Storage_dir/YYYY/MM/DD/HH/study/series/instances – with same semantics, only the represented date is that of study original creation. The date of receiving is different from that of creation - for example when migrating the archive to new machine using DICOM send operation, the date of receiving on the new archive is the date of the migration, the creation dates are unchanged.

Dcm4chee allows for compression of the pixel data of stored objects, the supported formats are RAW (uncompressed), JPEG Lossless, JPEG 2000 Lossless and JPEG LS Lossless [JPEG].

The compression of images is a key feature – the images are usually very homogenous with color movement in quite narrow “window” of substantial 24bit grayscale depth. A JPEG based lossless compressor is able to reduce the size of the dataset to around 10% of the original size. A JPEG LS lossless compression was chosen for the image storage as it was specially designed to fit the needs of lossless compression for medical imaging data [JPEG-LS] and because it proves to perform best in the circumstances [CLUJPEG].

It’s important to note, that having the images compressed in the storage system of the server does not mean that any potential client of the server has to support the corresponding compression format. When data is being interchanged in DICOM, the communicating peers first negotiate the support for object representation (as well as the requested and supported functionality) - in DICOM this is referred to as the Presentation Context and the actual encoding of the information as the Transfer Syntax. The dcm4chee implementation supports number of transfer syntaxes for the objects stored in the archive – one of them being the DICOM Uncompressed Transfer Syntax (uncompressed pixel data, little endian, referred to as Transfer Syntax UID 1.2.840.10008.1.2). The implementation is able to convert the data so that the target output is conformant with one of the Transfer Syntaxes supported by the communicating peer.

Thus all images produced by the department are eventually stored in the dcm4chee storage system and are available through the application’s DICOM interface via one of supported DICOM services or via HTTP through the WADO interface.

6.3 Reliability Implementation

As mentioned before in section [5.4 Reliability](#), the DICOM forwarding was chosen as the means of replicating the data to provide reliability. The front-end server is configured to forward any received data to the back-end servers.

There are three servers in the image archive cluster – the front-end, which is the only DICOM node accessed by the department and integration services and two backup servers. Also mentioned earlier, the physical location of the servers is such that it minimizes possibility of data loss of the solution in case of disaster. The main front-end server and one of the backup servers are placed in two separate rooms of the department with fire protection and permanent power supply. The other backup server is located in adjacent facility next to the hospital.

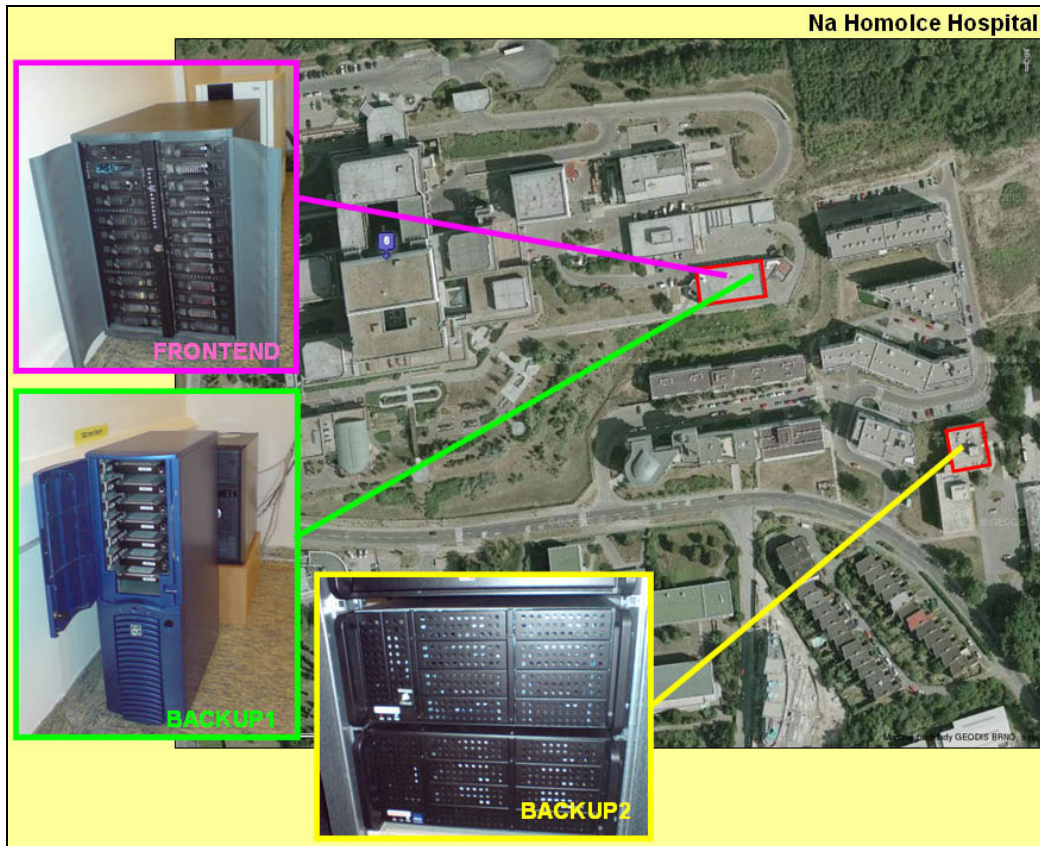


Figure 6.5 – Individual servers and their location

The data is forwarded immediately upon being received on the front-end server – the timeliness of backup is very good. Additional routine is setup to be run on daily basis, which verifies that the backup servers are up and running and that all the daily data were successfully backed up on both of them. In case of failure a mail output describing any found problem is sent to system administrators.

```

2007-12-06 11:11:11:

Errors encountered while performing daily backup check - see Status for details..

Services availability status:
~~~~~

FRONTEND (JAPECUBE1@japecube1.nm.homolka.cz): jboss (yes), mysqld (yes)
BACKUP1 (BACKUP1@japecube2.nm.homolka.cz): jboss (yes), mysqld (yes)
BACKUP2 (BACKUP2@jarchive.nm.homolka.cz): jboss (yes), mysqld (yes)

Found/Missing studies iuids:
~~~~~

FRONTEND:
found studies:
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33028
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33030
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33026
@1.2.752.37.1.1.1994044399.4.82353
@1.2.752.37.1.1.1994044399.4.82351
@1.3.12.2.1107.5.6.1.12345.30380107110808405904600000000
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33022
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33023
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33027
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33024
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33034
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33029
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33025
@1.3.12.2.1107.5.6.1.12345.30380107110809511792100000000

BACKUP1:
all studies found at frontend are also present here!

BACKUP2:
missing studies:
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33028
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33026
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33023
@1.2.40.0.13.1.1.172.20.33.1.20071025093209056.33024

```

Figure 6.6 – Example output of the backup checking routine

Further synchronization channels are setup between the front-end and backup servers to enable for propagation of any eventual updates made to the data stored on front-end to the backup servers. In case the data on front-end server are modified, the data on backup servers are modified accordingly. This is achieved by appropriate configuration of the Content Edit Service of dcm4chee. The service offers functionality allowing for updating the data stored on the server. When an update of local data is performed via the service, the service is able to issue corresponding DICOM DIMSE (DICOM Message Service Element) operations on configured servers. Thus the actual communication facilitating the update propagation is a standard DICOM communication.

6.3.1 Hot-swap and replication

Having three servers instead of more typical configuration with solitary backup device has many advantages, which are well worth the additional cost. In case the front-end goes down one of the backup servers can take over. The remaining server stays in backup role and the solution still offers well backed-up reliable environment. The same applies to maintenance – it is easily possible to take any of the servers including the front-end out of the operation for maintenance without any compromises to the reliability of the solution and with minimal administration effort. The maintainability – the possibility to perform periodical service to

keep the software and hardware up to date – is an essential condition for providing high level of reliability. The ease with which this can be done also minimizes some possible risks – it is very common that the consistency of the data is compromised not during fluent production run, but during service or update. The servers can thus be updated one at a time. Compatibility between various versions of dcm4chee (or indeed the Java, filesystem or operating system) is not an issue as the communication between the servers is facilitated using standard means. Any potential updates can be first rolled into production on one of the back-ends and after thorough testing it can eventually be applied to the front-end.

To maximize the benefits of this configuration the effort necessary for swapping two nodes in the stack has to be minimized. The fundamental step here is that the configuration of all of the three machines is identical. By exploiting the configuration features of dcm4chee a single configuration was achieved which enables the server to operate in both backup and front-end mode, without any necessary change – including the configuration of forwarding to backups, propagating changes etc. Thus it is possible to swap servers by simply changing their IP addresses.

The hack enabling for easy swapping of the servers is, that each of the servers has multiple DICOM identities. DICOM nodes are identified by a name – referred to as the AET (Application Entity Title). The three servers all have three identities (e.g. ARCHIVE, BACKUP1 and BACKUP2) – each one is able to communicate like any of the remaining two. The front-end communicates with the outside world using the main AET (e.g. ARCHIVE), and forwards the data and updates to the other two servers (e.g. BACKUP1 and BACKUP2). To prevent cyclic forwarding, appropriate conditions are applied in the server config – again identical on all of the servers. The result is that the two servers can be swapped simply by changing their IP address, thus the backup servers indeed are full-fledged hot-swaps.

The second operation necessary for providing high reliability is the replication of existing node to a new one. This operation is quite complex, therefore it was necessary to implement some kind of automation. In approximately 1 day any of the nodes can be replicated to a new machine using an easy-to-setup routine that takes care of installing and configuring the dcm4chee instance and underlying MySQL database and replicates the data from another node. The replication operation will be described in greater detail in section [6.5. Maintenance and Administration](#).

Thus all the data is backed up in reliably and timely fashion, the time necessary to restore full functionality in case of failure is minimal. The solution allows for periodic/continual service and upgrade with no outages in either availability or reliability.

The next step would be to setup some kind of specialized DICOM heartbeat service monitoring availability on the cluster and automate the hot-swap process in case of failure fully using some High Availability services like HA-Linux. This feature was not implemented though, and remains on the wish list for next iteration.

6.3.2 Reliability emergency plans

Even the most reliable, high availability and best designed systems often have one common single point of failure when put into production – and that is, that any emergency situation requires knowledgeable administrator to be actually present to solve the situation. By having more people able to perform basic administration tasks, the reliability of the solutions is further extended.

Therefore a set of wiki documents has been created to describe the steps necessary to perform in the two typical lifecycle events of the solution – scheduled maintenance and failure. The documents do not describe the operations in full detail, however they provide basic clue as to how serious the situation is and what steps should be taken immediately. The documents are intended for system administrators and system users – for example in case two of the servers go down, there is no backup of the data – the department staff should be advised not to delete any data from the department workstations until some backup is set up.

The solution is still evolving rapidly, so it is difficult to maintain a consistent documentation - also some things are and will remain complicated. The documents do not enable a non-qualified person to perform administration of the system – however they provide a clue to understand how serious the current situation is, what steps can be taken to minimize further risks and provide basic administration trail for anyone who might be doing administration. One typical situation was considered while writing them – and that is that the administration performed by one knowledgeable staff instructed by an administrator of the system over the phone. The provided automation of basic tasks together with these documents should be enough for any knowledgeable person to successfully cope with most of the typical situations.

6.4 Integration Implementation

The interoperability with various parts of hospital infrastructure is a common pitfall of implementing any application for healthcare. The IT subsystems of hospitals are often very heterogeneous and very seldom fully standardized, thus plugging another piece of application often requires implementing special interfaces. On the other hand the standardization is the only way to create well integrated maintainable environment as any non-standard solution or interface makes it even more difficult to replace any of the subsystems or integrate some new one. The hustle between simple-but-nonstandard and standard-but-complicated is on daily menu as far as integration in healthcare goes.

An extra effort has been spent on providing an easy to work with client library for the integration and management services of the PACS – the dcmMIT (DICOM Management and Integration Toolkit). This client enables for very easy development of custom applications communicating and working with the PACS system. It provides powerful and easy to use API on one side and standardized and configurable communication with the PACS on the other. It is the central piece of integration implementation concerning the described PACS solution described in this work, however it can be easily adopted to be used with other PACS solutions. It will be discussed in greater detail in section [6.4.3 DcmMIT toolkit](#).

As mentioned before in section [5.5. Integration](#), there are several integration interfaces for a PACS to implement:

- Internal interfaces between image archive and modalities and workstations inside the system
- RIS/HIS interface to support department scheduled workflow
- HIS interface to enable for query and display of stored image data from the HIS
- Interface for sharing documents with other healthcare facilities

6.4.1 Internal Interfaces and Scheduled Workflow Support

As mentioned in section [5.5.1 Internal Interfaces](#), there are three modalities at the department, two of which are fully DICOM compliant. The other modality does not feature any DICOM interfaces.

The two DICOM enabled modalities produce the data in the DICOM file format and are able to send it to the archive using the DICOM Storage Service Class. One of them supports automatic push of data after finished examination, the other one only allows for manual push operation.

The modalities are also able to query the worklist maintained on the image archive using DICOM Modality Worklist Service Class and pull their scheduled exams from there.

The one remaining modality – the Siemens Ecat-Exact PET scanner – however does not support DICOM. The data produced by this modality are not in DICOM file format and cannot be communicated using DICOM protocol; the modality also does not support the DICOM Modality Worklist Service Class – the scheduling of exams on this modality has to be carried out using different means.

The actual format of the data is the Ecat 7 file format – a proprietary Siemens data format used with older modalities (unfortunately no homepage or a Siemens page is dedicated to this format, therefore no appropriate reference can be given). Some experiments have been made with converting this format to DICOM using open-source solutions (XMedCon [XMEDCON], LONI plugins [LONI]), but the outcome was not satisfactory. Therefore an alternative solution had to be found.

The workstations at the department are all by Siemens and they support the Ecat 7 format. They also enable automatic pull operation of any newly created data from the Ecat-Exact scanner and automatic conversion to DICOM. Thus it is possible to have all data produced by the modality pulled to one of the workstations and there converted to DICOM and available through that workstation's DICOM services.

A periodically run pull mechanism making use of DICOM Query/Retrieve Service Class was setup to pull the converted data from the workstation to the image archive. Thus all data produced by any of the modalities is eventually stored in the image archive.

The worklist mechanism of the modality is able to consume exam orders stored in a special formatted text files. Such well formatted files can be put to a special folder of the modality gateway computer via FTP – eventually they are processed and appropriate exam is scheduled on the modality [ECAT]. The mechanism for worklist management was extended to support this operation.

The actual mechanism enabling for scheduling of individual exams at the department is implemented in form of an EJB (enterprise Java Bean) client accessing corresponding Session EJB deployed on the dcm4chee that manages the worklist. For the Siemens Ecat-Exact scanner a special Session EJB was implemented that is deployed to the JBoss in which the archive runs, that uses FTP to add worklist items to the worklist database of the scanner. This Session EJB is accessed by the same client application as the one managing the worklist of the dcm4chee Image Archive.

This is in contradiction with IHE Scheduled Workflow integration profile which states that the exam orders should be communicated using the HL7 ORM^001 message type. As Dcm4chee provides a HL7 interface able to consume the ORM^001 messages and scheduling if orders appropriately in the DICOM worklist that it maintains, this solution is considered sub-optimal and is to be revised in the future.

Both the server side (the webservice dcm4chee specific implementation) and the client Java API are parts of the dcmMIT Toolkit.

The future iterations of dcmMIT (described in section [8.2 Integration Improvements](#)) will provide an HL7 consumer that will allow for various implementations of the actual scheduling operations to be triggered upon receiving the ORM^001 HL7 message. One of the possible implementations will be forwarding of the message further – to the dcm4chee HL7 consumer, another putting a specially formatted file to a special folder at the Ecat-Exact modality gateway. This approach will enable for the whole PACS to be accessed in IHE compliant way.

As mentioned earlier in section [5.5.2 Workflow Management Interface](#), in a number of cases the actual examination taking place may not be a result of an exam order scheduled in the HIS. In such cases the data resulting from the examination cannot be correctly identified and have to be bound to the correct patient's eHR later. This functionality is supplied by a daily run data verification routine that was implemented to verify completeness of the data and to ensure correct linking of the data to patient eHR.

This routine is available as a web-service deployed on each of the Image Archive dcm4chee based servers (thus supporting the hot-swap architecture of the data backup solution, described in section [6.3.1 Hot-swap and replication](#)). The web-service is able to consume a String parameter containing XML encoded list of ordered exams and performs following operations:

- For each individual exam:

- Find data potentially being the result of source exam order in the PACS. To achieve this, the exam uses queries based on patient ID and date-time of the exam. Mark found studies as identified.
- Based upon provided XML configuration describing typical data produced as a result of various exams determine, whether all required items (studies, series) were found in the PACS.
- Update the identification description information of found data to match the identification of the exam order and thus to corresponding patient's eHR.
- In case not all of the required data corresponding to the order were found, remember this exam to be reported to the RIS later.
- Report all orders with missing data to the RIS along with the list of unidentified studies and their series found inside the PACS. The actual channel used for reporting missing data and unidentified studies and their series to the RIS is a configurable (via second XML-encoded parameter of the service) webservice call that passes single String parameter containing the exam order.

The routine implementation is part of the dcmMIT toolkit and uses the toolkit DICOM API to communicate with the underlying PACS.

6.4.2 Interface to access image data from within and outside of the hospital

As mentioned in section [5.3.3 Interface to access image data from within and outside of the hospital](#), the applications within and outside the hospital that are not part of the actual department PACS do not usually access the data stored in the archive using DICOM. The DICOM standard only supports sending of data between two nodes that know of each other's existence – an anonymous client requesting data is not supported inside the DICOM communication protocol, also the standard does not provide any convenient security management policies.

This is the reason why some kind of proxy-mechanism is usually used instead, that enables for access to the data from outside of the department, provides the necessary security and shields the image archive from the rest of the world. As noted in section [5.3.3 Interface to access image data from within and outside of the hospital](#), the proxy mechanism used here uses standardized DICOM means to connect to the image archive to actually query, retrieve and manipulate the data stored in the image archive.

This functionality is wrapped in the dcmMIT Toolkit. The toolkit features very simple to use yet powerful Java API to represent DICOM objects, to query data stored on the server using the DICOM Query/Retrieve Service Class and to retrieve it using DICOM WADO service of the server. The dcmMIT powered client or proxy can be simply enhanced by the hospital security policies.

The proxy element in the operation is the jDatobot [DATOBOT] application, a simple application server abstracting the data access layer and simple services that is used for various tasks in the hospital. A service for accessing PACS data has been implemented for this application that offers structured interface and security for calling applications and communicates with the underlying PACS on the other side using the dcmMIT services. It runs on a trusted server within the department network.

For displaying of the data, the Stetoskop [STETO] application is used. The Stetoskop obtains data by invoking the above mentioned jDatobot service accessing the PACS. The jDatobot application and Stetoskop application communicate using proprietary encrypted protocol. The Stetoskop application is capable of managing and displaying other parts of the patient medical documentation as well, allowing for uniform access to the complete patient eHR (Electronic Health Record).

The Stetoskop application provides very convenient way of displaying the data from within the hospital. As the communication of the Stetoskop and jDatobot applications is secure and in cooperation they are able to provide authentication service and filter the provided data accordingly based upon the security information provided in the HIS, the Stetoskop application also provides basis for sharing the image data stored inside the PACS with other hospitals and for remote reporting from home for the physicians momentarily not present at the department.

6.4.3 DcmMIT toolkit

DcmMIT (DICOM Management and Integration Toolkit), implemented as a part of the PACS system described in this work, is a toolkit that allows for standardized yet simple and programmatic approach to PACS integration and management. The DICOM in the name refers to the fact that the basic task of the toolkit is to bridge the DICOM enabled PACS environment and the non-DICOM outside world, but also to the fact that it uses standardized DICOM means wherever possible to communicate with the PACS. It is the central part of integration implementation of this solution.

The toolkit provides simple yet powerful API to represent DICOM objects and server components of a PACS system and offers client API for accessing various significant services of the PACS, such as querying, retrieving and updating data inside the PACS or managing PACS worklists.

Having convenience API is such as mentioned above is useful for a number of reasons:

- Any application wishing to integrate with the PACS can do so in convenient manner, enabling for programmatic approach. The resulting communication towards the PACS is as standardized as possible.
- Further standardization is possible within the toolkit. The underlying PACS solution itself remains intact and self contained. Anyone wishing to integrate directly with the PACS can still do so.

- The toolkit is also great for implementing internal data management services inside the PACS – the data verification daily run routine that makes sure all relevant data are present in the PACS and well identified was implemented using this toolkit.
- The toolkit is a great central place to store any new PACS knowledge that can take form of a program. By managing the namespaces well and strictly distinguishing the general and standardized from the proprietary, the toolkit can continue growing and adding to its value.
- The implementers of applications communicating with the PACS are usually not familiar with DICOM, HL7 or IHE or indeed any principles this paper is about. The toolkit enables even people like these to start working with the PACS in a standard way, secondary benefit is that while they do not necessarily need to know too many details about PACS and DICOM, they eventually begin to learn more about basic objects and services inside the PACS. Thus having a toolkit like dcmMIT helps continuous development, because it makes the collaborating side understand the area of the expertise of PACS.

The basic object model of the toolkit consists of:

- Patient/Study/Series/Instance hierarchy
 - The objects only contain the subset of attributes of corresponding objects (like patient name, patient ID, sex, weight etc. for Patient object). When querying some server for stored data it is possible to configure what subset of attributes will be returned. For the Instance (image) level only the metadata attributes are included – not the actual pixel data.
 - Thus it is possible to actually operate with the objects without having to pull the pixel data from the server. The actual pixel data (or the whole DICOM object representation) can be pulled from the server using DICOM WADO in one of supported formats (e.g. JPEG, uncompressed DICOM etc.). The important aspect is that DICOM supports returning of metadata even to anonymous client (as opposed to returning the full dataset with pixel data) as does WADO – this means that the toolkit allows for anonymous access to the data stored in the archive including the query capabilities via DICOM standard ways.
- DcmServer, Dcm4cheeServer
 - Corresponds to a single DICOM server (eventually running dcm4chee).
 - The objects offer number of clients of the services of the server:
 - DicomGateway object offers functionality to query the content of the server using DICOM Query/Retrieve Service Class.
 - WADOGateway object enables to pull the pixel-data or the whole dataset of corresponding instance from the server using DICOM WADO service.
 - Some of which are specific to Dcm4cheeServer, like:

- ContentEditServiceClient – this enables for updating of data stored in the server using the ContentEditService XMBean of the dcm4chee implementation. Updates carried out using this service are propagated to the backup servers.
 - MWLManagerClient – enables for managing DICOM Modality Worklists on target server.
 - EcatWlManagerClient – enables for managing ECAT worklists on Ecat Exact modalities.
- PACS Configuration and classes representing PACS Services
 - The ConfigurationXML manages XML based configuration of the PACS. The configuration contains declarations corresponding to the servers present in the PACS and the services available on them. It further contains definitions of services available in the PACS that the clients can access with their specific configuration
 - Each of the services that can be defined in the configuration has a corresponding service implementation in the source-code that parses the properties and offers the functionality corresponding to the service.

```

<dcm4chee-server>
  <server-id>TEST2</server-id>
  <aet>TEST1</aet>
  <host>w64c2.hq.jlabs.cz</host>
  <port>11113</port>

  <dcm4chee>
    <db-host>default</db-host>
    <db-user>user</db-user>
    <db-pass>password</db-pass>
    <db-encoding>utf8</db-encoding>

    <jmx-user>jmxuser</jmx-user>
    <jmx-pass>jmxpass</jmx-pass>

    <wado-url>default</wado-url>
  </dcm4chee>
</dcm4chee-server>

<dcm-server>
  <server-id>IMAGINARY</server-id>
  <aet>IMAGINARY</aet>
  <host>somehost.somedomain</host>
  <port>11112</port>
  <wado-url>http://somehost.somedomain:service=wado</wado-url>
</dcm-server>

<mw-l-service>
  <service-id>petMML</service-id>
  <mw-l-server-id>TEST1</mw-l-server-id>
  <mw-l-charset>UTF-8</mw-l-charset>
</mw-l-service>

<ecatwl-service>
  <service-id>petEcatwl</service-id>
  <ecatwl-manager-host-id>TEST1</ecatwl-manager-host-id>
  <ecatwl-charset>US-ASCII</ecatwl-charset>
  <ecatwl-ftp-host>host</ecatwl-ftp-host>
  <ecatwl-ftp-user>ftpuser</ecatwl-ftp-user>
  <ecatwl-ftp-pass>ftppassword</ecatwl-ftp-pass>
  <ecatwl-ftp-wl-dir>/tmp</ecatwl-ftp-wl-dir>
</ecatwl-service>

<worklist-service>
  <service-id>petWorklist</service-id>
  <mw-l-service-id>petMML</mw-l-service-id>
  <ecatwl-service-id>petEcatwl</ecatwl-service-id>
</worklist-service>

```

Figure 6.7 – Example dcmMIT configuration

Thus all of the management/communication means necessary for PACS integration with the rest of the world are available in a single Java toolkit with simple to use API and with means to capture the structure of the underlying PACS and define services, that the clients can refer to using XML based configuration.

6.5 Maintenance and Administration

A lot of effort has been spent on making the basic maintenance and administrative tasks such as installation of a new machine or replication of existing machine including data stored on that machine to a freshly installed one as simple as possible.

The Installation process is far from straightforward. This is the sequence of steps that need to be taken:

- The target machine OS has to be installed
- The disk infrastructure has to be set up
- The Java Runtime Environment and corresponding packaging system (JPackage [JPACKAGE]) has to be installed
- Other prerequisites have to be installed such as the MySQL database engine or the Sun Java Image I/O Tools package
- The database engine needs to be configured - parameters such as cache sizes, concurrency settings, table locations or logging need to be set up. Appropriate directory structure has to be created including access rights. The sysconfig has to be setup for the database engine.
- The user under which the application will be running has to be created.
- The database user the application will use to access the database, the database schema and the access rights allowing for access from preconfigured hosts have to be setup.
- The application needs to be installed in to appropriate directories including setting access rights and init scripts. The application ships as a standalone JBoss server with necessary libraries already deployed.
- The application has to be set up, including modifications to startup script and configuration of the underlying JBoss instance, database access setup, logging setup and setup of all services – which includes setting tens of properties.
- Additional Webservice/EJB archives have to be deployed to the server.
- The backup check and data verification routines have to be setup for this machine.

The replication process is also quite non-trivial and includes following steps:

- Installing dcm4chee to target clean machine and setting it up the same way as on the source machine.
- Synchronizing the file data using rsync (however it is not possible to run rsync on the whole folder containing data. A list of paths of 20M files – the first thing that the rsync exchanges – assuming 64 characters per path, amounts to over a gigabyte of

information to be kept in the memory. As the operation has to be possible even during production, this is unacceptable, so the data has to be rsync-ed by parts.)

- Synchronizing the database using dump and restore to ensure consistency
- As the operation can take part during production, data can be updated, added or deleted to the source machine during the actual replication. It is therefore necessary to resynchronize the servers after the replication is done. This operation should be run during off-time hours therefore it is managed by separate script.

To be able to maintain productivity while managing multiple servers, to prevent errors that would inevitably happen time to time when repeating this sequences of steps and to be able to provide service of the solution in timely and reliable basis, it is very desirable to automate as much of the routines as possible.

Both processes are automated using shell scripts with configuration files. Important configuration items such as the database config file, the files representing the configuration of individual XMBEans (dcm4chee services) or patched JBoss init and startup scripts are kept on the filesystem and are eventually patched to correspond the actual configuration.

Each of the servers contains an “admin” folder in which various management scripts such as script allowing for configuration of instance’s logging etc. are located along with the configuration file representing the config of the server. As the backup check and pull routines are also realized by shell scripts and use the configuration file of the server as input, the base of the solution is very self-contained and centralized around a single configuration file. The stack of front and back-end servers is thus forming autonomous self-managed entity.

Further automatization routines are available for pulling a machine out of the system for scheduled maintenance and putting it back to production that take care of reconfiguration of forwarding service of peer servers and eventual resynchronization.

The organization of the install/replicate process is such that it is easy to adopt it to updated versions of the distribution.

7 Summary

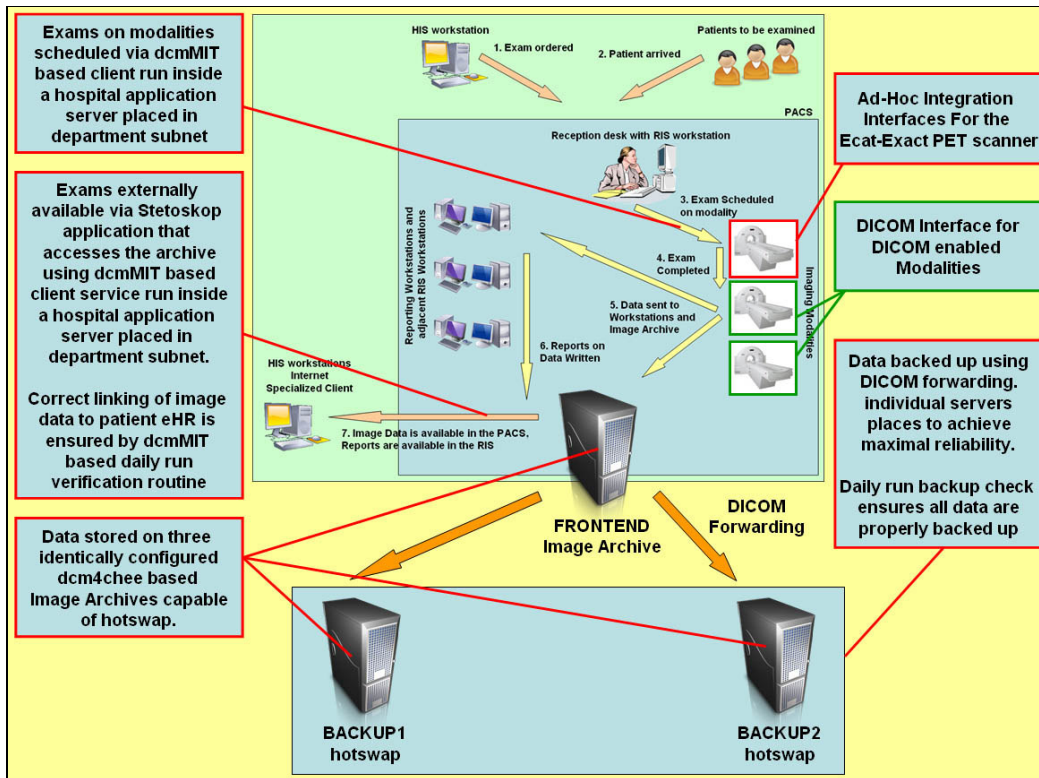


Figure 7.1 – Implementation Summary

The solution as is fulfills all the requirements of the specification. Quality of service was significantly improved in all important areas:

- In reliability, where two live hot-swap copies are being maintained and the consistency of backup is checked on daily basis. The system is stable in long running production use. Emergency plans have been documented in effort to remove the dependency of the system restore in case of failure operation on physical presence of any single person.
- In integration inside the department, where the support for scheduled workflow environment is running without outages and policies were set up to ensure correct linkage of the data to the rest of medical documentation. In integration with entities outside of the department, where a communication channels has been implemented to enable for accessing the data in a secure yet standardized way to be displayed inside and outside of the hospital together with the rest of relevant medical documentation.
- In maintainability of the system. The system is very self contained, autonomous and features automation for basic administration tasks. It supports scheduled maintenance or failure of any part of the system without any limitation to the rest of the system, which enables for smooth continuous service, maintenance and upgrade of the whole system – both hardware-wise and software-wise.

- In capacity and responsiveness where the system is appropriately swift with responses and a long term plan including eventual upgrades has been laid to ensure appropriate capacity. The solution is approximately on 50% of its capacity now that it contains all the data up to date with further potential to grow both performance-wise and in capacity.
- In level of expertise and available consulting provided to the department and the timeliness of service interventions.

In the area of standardization considerable accomplishment was achieved, however it is not the full IHE compliancy, even though the principles and operations the IHE uses are also used here and were considered as alternatives during the development. The reasons behind not choosing the way of IHE in particular situations are documented throughout the text. Eventually the solution will move towards IHE in further iterations.

The system is ready to be easily integrated to other environments; an effort has been spent to simplify the process of integration and management by introducing the dcmMIT toolkit which simplifies the creation of proprietary standardized client services greatly. In the future the dcmMIT toolkit will be further standardized, simplifying the process of adoption of IHE from the RIS/HIS side and from the PACS side – the proposed improvements to the toolkit will be discussed in section [8.2 Integration Improvements](#).

Along the way the project required quite detailed knowledge of medical imaging area of expertise. As very little is available in terms of education in this area on Czech university grounds, to get the basic orientation was a very challenging task, that required finding and studying corresponding standards, finding and investigating appropriate literature and other information sources, such as various web resources or discussion forums. The expertise covered by this work and underlying it is by no means not exhaustive – still there are many things in the Medical Imaging area of expertise this work does not profit from and that are yet to be discovered and tested and eventually. However the amount of information that had to be gathered to accomplish this project starting from zero is substantial and has some worth of its own - as has this work, which is partially mapping the subject.

The project was first conceived as purely experimental, with very low probability of success. The fact it is getting very close to be marked production ready and successful is a remarkable accomplishment, especially considering a one-man part-time team and very little assistance or previous education in the subject.

However the project is far from being finished. Constantly new ideas appear – indeed faster than it is possible to realize them. The problematic part is the number of areas the project is concerned with – from designing the hardware configurations on which the system should be run, optimizing the performance of hardware and low level services such as filesystem, RAID, database and Java, designing the actual storage strategy and optimizing the management and administration processes to designing of target integration strategies and providing management and display capabilities to the clients of the solution. The situation is

further complicated by the fact that the corresponding knowledge base is very including thousands of pages of standards that are still evolving.

Following pieces of software were written to facilitate the PACS solution described in this work:

- Shell script implementation of daily backup verification routine described in section [6.3 Reliability Implementation](#).
- DcmMIT toolkit, described in section [6.4.3 DcmMIT toolkit](#). This also contains the implementation of client and server modules enabling for management of PACS worklists and the implementation of the daily run data verification routine ensuring correct linking of image data from the PACS to the eHR (Electronic Health Record) both described in section [6.4.1 Internal Interfaces and Scheduled Workflow Support](#).
- Implementation of client services for the jDatobot [DATOBOT] application used to query and retrieve data from the PACS described in section [6.4.2 Interface to access image data from within and outside of the hospital](#).
- Shell scripts enabling for administrative tasks automation as described in section [6.5 Maintenance and Administration](#).

8 Future

The project is still far from finished. The project has potential of growth in many areas. Below, in sections [8.1 Back-end logic improvements](#), [8.2 Integration Improvements](#) and [8.3 Image Display Improvements and Other tasks](#), follows the list of possible future improvements.

8.1 *Back-end logic improvements*

The system features rock-solid backup system. Indeed this is the strongest aspect of the system. However, should the system be used in larger scale (big roentgenology departments produce over 10 times more data than the PET Center) the system would have to be extended considerably.

Obviously it would be possible to increase the power and capacity of underlying machinery accordingly, but it is not the favored way – the cost of high-performance hardware is very high.

The solution that emerges is to write an intelligent DICOM load balancer/router that will allow for multiple distinct DICOM datasources to act as a single image archive or for a group of clustered DICOM datasources to share the load and distribute the data.

Minor tasks in this area include:

- Fine tuning of MySQL – table partitioning could be the way forward for increasing and distributing database load.
- Fine tune the hardware configuration – the database should be on separate potentially striped disk system.
- Experiment with filesystems – ReiserFS could improve I/O performance.

8.2 *Integration Improvements*

The implementation of integration is sufficient under current circumstances. It is far superior to the preceding solution. However there are many areas of concern, where the solution could improve with the target of offering full feature set and easy plug-and-play integration in any environment.

The data conversion and transport channel setup for the DICOM non-conformant modality (the Siemens Ecat-Exact PET scanner) is far from ideal. A direct automated conversion and storage of the data would be preferable. However this work item is pretty low on the priority list as the modality is probably going to be replaced in the near future.

Another task is to improve the dcmMIT toolkit. The idea of integration toolkit is good. Even when the peers are not standardized, it is simple to implement powerful customized tools, that make the interfaces of the peers towards PACS standardized and thus increase the quality of the target environment while keeping the PACS uncompromised.

For the next iteration of the toolkit following architecture is envisioned:

- A web-service module will be available, that will provide interface for all the necessary operation with the data, like scheduling of exams, updating data or verifying the data based on exam orders. The actual implementation of the functions will be configurable via underlying XML configuration describing the PACS - for example scheduling of an exam can result in HL7 message being sent to a preconfigured server, or generating a specially formatted file into some folder depending on actual underlying PACS configuration or calling some service via RMI or JAX-WS.
- A HL7 consumer and sender able of receiving and sending HL7 messages will be available, that will intercept any messages sent to the PACS or sent by the PACS and trigger appropriate functions of the webservice module. It should be possible to actually plug various implementations to the framework in the underlying configuration, enabling for customizing the behavior for the target usage scenario. The two modules would work in cooperation and would form configurable interface between any PACS and any HIS/RIS system that would allow for simulation of conformant interface on any of the sides.
- The other functionality of the client, such as querying and accessing the data might be moved into the web-service tier as well to enable for uniform interface.

The next goal is to learn and employ more advanced DICOM features inside the PACS, like usage of MPPS and Structured Reporting.

Next it might also prove useful to learn what benefits could be gained from using XDS-I for cross-enterprise data sharing and eventually adopt existing solutions accordingly.

8.3 Image Display Improvements and Other tasks

The tasks in the area of Image Display include:

- Provide quality web based display interface for in-hospital display. Some initial inquiries have already been made and potential solution candidates are already in scope.
- Eventually provide secured (selective content per user) web interface for out of hospital display with possibility to download the original DICOM.
- Implement a full fledged diagnostic reporting workstation to complete the product lineup and offer a complete solution – the display and analysis features of the Stetoskop application described in section [6.4.2 Interface to access image data from within and outside of the hospital](#) are evolving quickly could form the base full-featured DICOM Reporting Workstation in the future.

As obvious, the project is still far from being finished, with many challenges still lying ahead. This work is thus just a snapshot of the solution that will further evolve.

9 References

- [CLUJPEG] Cluine D. S. (2004): The Lossless Compression of Grayscale Medical Images
(http://www.dclunie.com/papers/spie_mi_2000_compression.pdf)
- [DCM] DICOM Standard, <http://medical.nema.org>
- [DCM4CHE] Dcm4che DICOM product family, <http://www.dcm4che.org>
- [DCMCOO] Phillips, DICOM Cookbook
- [DCMSR] Cluine D. S. (2004): DICOM Structured Reporting
- [DCMTK] DCMTK DICOM Toolkit, <http://dicom.offis.de/dcmtk.php.en>
- [DCMVDT] DICOM Validation Toolkit, <http://www.dvtk.org>
- [ECAT] Bartl,J (2006): DICOM Modality Worklist for PET system SIEMENS Ecat Accel, EANM proceedings
- [EPACS] ePACS Project, <http://www.epacs.cz>
- [FIREBIRD] Firebird Database Engine, <http://www.firebirdsql.org>
- [HUA04] Huang H. K. (2004): PACS and Imaging Informatics: Basic Principles and Applications.
- [HL7] HL7 Standard, <http://www.hl7.org>
- [IHE] IHE initiative, <http://www.ihe.net>
- [IB] InterBase Database Engine, <http://www.codegear.com/products/interbase>
- [IW] InterfaceWare HL7 Manual, <http://www.interfaceware.com>
- [JBOSS] JBoss.org, JBoss Application Server documentation,
<http://labs.jboss.com>
- [DATOBOT] jDatobot, a simple application server implementation by Janiga Labs s.r.o.,
<http://www.jlabs.cz>
- [JDCM] jDCM Java DICOM Toolkit, <http://www.geocities.com/gigiobb/>
- [JMX] Java Management Extension,
<http://java.sun.com/javase/technologies/core/mntr-mgmt/javamanagement>
- [JPACKAGE] JPackage YUM repository and RPM tools, <http://www.jpackage.org>
- [JPEG-LS] JPEG-LS homepage, <http://www.jpeg.org/jpeg/jpegls.html>
- [JPEG] JPEG Homepage, <http://www.jpeg.org>
- [LONI] Laboratory of Neuro Imaging (LONI), UCLA, LONI Java Image I/O Plugins, <http://www.loni.ucla.edu/Software>
- [MARIS] MARiS Project, <http://maris.homelinux.org>

- [MYSQL] MySQL Database Engine, <http://www.mysql.com>
- [PACSDR] Dreyer K. J., Metha A., Thrall J. H. Cluine D. S. (2001): PACS – A Guide to Digital Revolution
- [PHILCS] Phillips - Conformance Statements Multislice CT Acquisition Modalities - Brilliance Workspace, <http://www.medical.philips.com/>
- [PIXELMED] Pixelmed Java DICOM Toolkit, <http://www.pixelmed.com/>
- [POSTGRE] PostgreSQL Database Engine, <http://www.postgresql.org>
- [SF] SourceForge development hosting, <http://www.sourceforge.net>
- [STETO] Stetoskop, an application for managing medical patient data by Janiga Labs s.r.o., <http://www.jlabs.cz>
- [VASQ] Vázquez A., Bohn S., Gessat M., Burgert O. (2006): Evaluation of Open Source DICOM Frameworks
- [XMEDCON] XMedCon Open Source Medical Image Converter, <http://xmedcon.sourceforge.net>