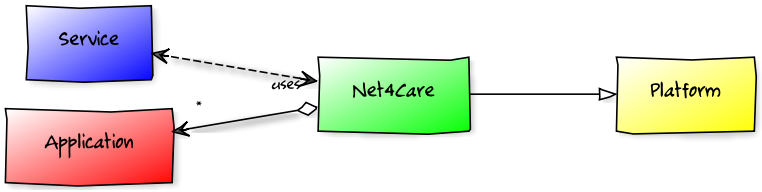
The Net4Care Platform – Version 0.3

Software Architecture



http://www.net4care.org

dev@net4care.org

December 11, 2012

Version history

Version	Date	Author	Comments
0.1	2012-09-06	KMH	Initial version of description based on the template of Rozanski and Woods (2011). Description of scope and context
0.2	2012-09-11	KMH	Stakeholders explained. Scenarios described. Glossary updated and scope definition revised
0.3	2012-09-19	KMH	Brief design approach. Goals and constraints, principles described based on Technical report 3. Functional view described
0.4	2012-09-26	KMH	Information view, concurrency view, development view described
0.5	2012-10-03	KMH	Final views described. Consistency considered
0.6	2012-10-10	KMH	Security and performance & scalability perspectives applied. Security based on notes by KM. Architecture backlog in document established
0.7	2012-10-12	KMH	Revisions based on review by HBC
0.8	2012-10-25	KMH	Availability & resilience and evolution perspectives applied. Additonal backlog items added
0.9	2012-12-11	KMH	Updated to include reference to Net4Care version number

Contents

Chapter 1

Introduction

1.1 Purpose and scope

The healthcare system in most western countries including Denmark is under pressure. The population is aging: in 2040 there will be 100% more elderly in Denmark. Furthermore, approximately 2/3 of elderly persons have at least one chronic disease. This, coupled with a development towards fewer and thus more centralized health centers, lead to a need for restructuring the healthcare system including the implementation of telemedicine systems.

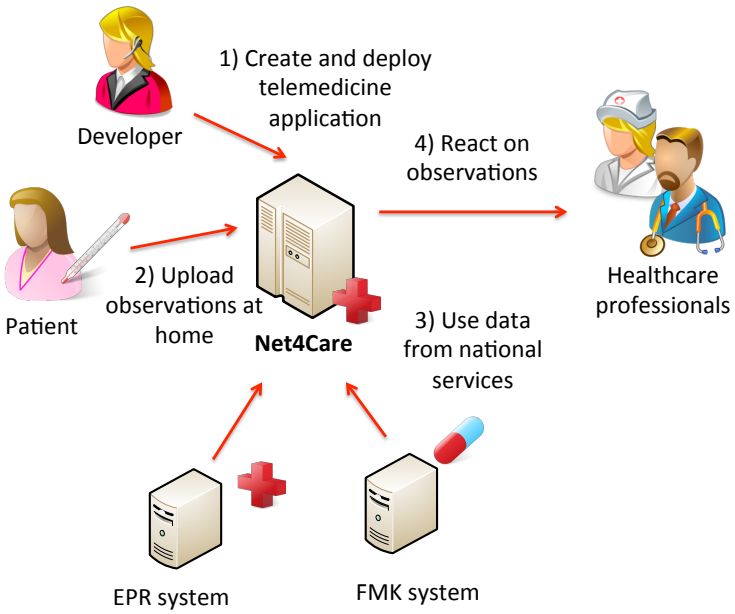


Figure 1.1: Context of Net4Care (see Figure 5.1 for a formal context diagram)

There have been many examples of locally developed, innovative telemedical systems including RRS[1] and myRecord (Andersen et al., 2011). However, these are typically not integrated with regional and national healthcare services such as Electronic Patient Records (EPRs) or the Danish "shared medicine card" (FMK) and are thus hard to deploy at a large scale.

In order to address these issues, the *Net4Care* platform aims to provide an open source *software ecosystem* for telemedicine in which developers are supported in creating standardized applications by a framework as well as learning ressources. Figure 1.1 informally illustrates the scope of Net4Care. The figure also illustrates the two categories of users of the Net4Care platform:

- *End users* who use Net4Care-supported telemedicine applications. These are patients and medical professional
- *Developer users* who create telemedicine applications using components of the Net4Care platform

Figure 1.1 informally illustrates the scope of Net4Care in relation to these users. In particular, the scope of the initial version of Net4Care includes

- Patients may upload telemedical observations through Net4Care. Net4Care provides a convenient interface for observations and stores these observations in an Cross-Enterprise Document Sharing (XDS) repository in Personal Healthcare Monitoring Report (PHMR) format[2].
- Patients and healthcare professionals may query telemedical observations
- Integration with healthcare information systems is supported through XDS
- Developers may build and deploy telemedical components that include new observation types
- Net4Care provides a step-wise development platform: from standalone, over integrated to deployed telemedical applications are supported
- International standards and specifications are observed. In particular we develop according to IHE XDS, HL7 PHMR, and Continua Alliance specifications

On the other hand, the initial scope of Net4Care excludes

[1] `http://www.alexandra.dk/dk/projekter/Sider/Remote-Rehabilitation-Support.aspx`

[2] See Section 2 for a glossary

- Other parts of HL7 CDA than Observations (such as Acts) are not supported
- The protocol for device to server communication will not follow Continua Alliance specifications
- Net4Care will not be deployed on the Danish Healthcare Data Network (SDN)
- Only Java is supported as an implementation language for telemedicine components. External protocols may be used from other languages

1.2 Audience

The primary audience of this document are developers of the Net4Care platform and developers of applications that use the Net4Care platform.

1.3 Status

This architecture description is a draft version; revisions will be gradually released in connection with the “Software Design and Software Architecture” course. The Net4Care platform is operational and in deployment as a prototype.

1.4 Architectural design approach

This architectural description follows the approach of Rozanski and Woods (2011). Following and overview of architectural requirements and forces (Chapter 3 and Chapter 4), we describe the architecture as a set of *views* (Chapter 5), i.e., representations of structural aspects of the architecture that illustrates how concerns of stakeholders are addressed. Next (Chapter6), we describe how Net4Care meets its required system quality properties.

The architectural design process has been based on *architectural prototyping*, i.e., development of executables demonstrating solutions to stakeholder concerns Bardram et al. (2004). The results of architectural prototyping have then been implemented in the Net4Care framework codebase. Finally, the resulting software architecture of the Net4Care framework has been described in this report.

1.5 Acknowledgements

The Net4Care project[3] has been funded by the European Union and the Central Danish Region through Caretech Innovation[4].

[3] http://www.net4care.org
[4] http://www.caretechinnovation.dk

Chapter 2

Glossary

Term	Definition
Continua Health Alliance	An industry group that establishes end-to-end standards for personal health management including telemedicine
EPR	Electronic Patient Record. Systems vary from region to region in Denmark
FMK	Flles MedicinKort (“Shared Medicine Card”) is a Danish, national service that allows access to all medicine information about and for patients
HL7	Health Level Seven International. Defines standards for interoperability of health information technology
MedCom	MedCom is a co-operation between authorities, organisations, and businesses related to Danish healthcare. Contributes to standards, testing, and certification. Runs SDN
HL7 CDA	HL7 Clinical Document Architecture. An XML-based format for clinical documents based on an object-oriented model of clinical information
NFIT	NFIT is a faculty coorporation between the IT units within the departments under the Faculty of Science at Aarhus University
OSGi	A Java-based module system and service platform
PHMR	Personal Healthcare Monitoring Report. An XML format intented for data produced by personal healthcare monitoring devices. It is described as an extension of HL7’s Continuity of Care Document (CCD) specification

SDN	The Danish Healthcare Data Network (SDN). A Danish VPN (or MPLS) network for transport security in healthcare information systems
SMB	Small- and Medium-sized Business. In Europe, businesses with fewer than 250 employees and a turnover of less than 50 million €
Telemedicine	The delivery of healthcare in which patient and healthcare professional are not co-located
XDS	Cross-Enterprise Document Sharing. Allows for registration, distribution, and access of health records across enterprises

Chapter 3

System stakeholders and requirements

3.1 Stakeholders

The primary stakeholders categories and roles related to Net4Care are listed and explained below.

- *Caretech Innovation (and eventually regions and municipalities)* pay for the system and are thus acquirers
- *Patients, healthcare professionals, and SMB developers* are eventual users of Net4Care
- *Net4Care project developers* construct and deploy the system based on its software architecture. Furthermore, they test the system to ensure that it is suitable for use
- *The Alexandra Institute* will be maintaining and supporting Net4Care, as well as production engineering (i.e., designing and managing the software and hardware environment on which Net4Care is to be deployed). Finally, the Alexandra Institute will administrate the system when it is operational
- *NFIT* will be supplying hardware, network, and software infrastructure for Net4Care
- *Net4Care project members and Caretech Innovation* communicate the results of Net4Care
- *MedCom and/or specialized component assessors* are intended to assess Net4Care and supplied components for adherence to standards

3.2 Overview of requirements: System scenarios

The overview of requirements for Net4Care are given as system scenarios.

3.2.1 Functional scenarios

In the following, we outline the main functional scenarios of Net4Care, i.e., what the system must do in response to external stimuli.

Scenario reference	FS-U1. Sending Telemedical Observations
Overview	How Net4Care handles telemedical observations
System state	Patient has been registered with healthcare professional (steward) and equipment (e.g., a spirometer) has been assigned to patient
System environment	The deployment environment of Net4Care and device are operating normally
External stimulus	A patient has made a telemedical measurement (e.g., a spirometry measurement) at home and chooses to send the measurement
Required system response	The measurement is securely transported to the Net4Care storage, confirmation that data has been received is obtained

Scenario reference	FS-U2. Telemedical Observations Used by Healthcare Professional professional
Overview	How Net4Care allows healthcare professionals to access information
System state	A healthcare professional is responsible (steward) for the treatment of a patient that has sent telemedical observations
System environment	The deployment environment is operating normally
External stimulus	A healthcare professional needs access to the latest telemedicine observations of a patient
Required system response	Net4Care provides access to observations as needed

Scenario reference	FS-D1. Net4Care Component Implemented by SMB Developer
Overview	How Net4Care allows SMB developers to create components for the platform
System state	The organization of an SMB developer is registered as a contributor to Net4Care
System environment	Development ressources for Net4Care are accessible
External stimulus	The SMB developer wants to develop a component for a Net4Care-based application based on an existing telemedical device (e.g., for video chats on spirometry data)
Required system response	The SMB developer is able to access development ressources for Net4Care and implement a component for the device. The component is implemented and tested locally on the developer's machine as well as in a sandboxed distributed environment

Scenario reference	FS-D2. Net4Care Component Uploaded by SMB Developer
Overview	How an SMB developer can upload a component to Net4Care
System state	An SMB developer has developed and tested a component (e.g., for video chats on spirometry data) for Net4Care
System environment	The component repository for Net4Care is operating normally
External stimulus	The SMB developer wants to upload the finished component to the Net4Care repository
Required system response	The component is uploaded and queued for review

3.2.2 System quality scenarios

The following system quality scenarios define how Net4Care should react to a change in its environment. The scenarios have primarily been elicited through Quality Attribute Workshops (Barbacci et al., 2002) with stakeholders in telemedicine projects. Only prioritized scenarios that add understanding of quality requirements have been fully defined.

Scenario reference	QS-U1. Telemedical Observations Sent
Overview	How Net4Care behaves under a normal load of telemedical observations sent (performance & scalability)
System environment	The deployment environment is working correctly
Environment changes	1,000 telemedical observations are sent in an hour to Net4Care
Required system behavior	Observations are received, stored, response sent; latency less than 5 seconds in 95% of the time

Scenario reference	QS-U2. Querying Telemedical Observations
Overview	How Net4Care behaves under normal load of querying by healthcare professionals (performance & scalability)
System environment	The deployment environment is working normally
Environment changes	20 healthcare professionals query observations in an hour under a normal observation sending load (cf. QS-U1)
Required system behavior	Queries are executed and responses sent in less than 1 second in 95% of the time

Scenario reference	QS-U3. Doubling the Load of Telemedical Observations
Overview	How Net4Care handles a load above what it is dimensioned for (performance & scalability)
System environment	The deployment environment is working normally. Users are causing a normal load (cf. QS-U1 and QS-U2)
Environment changes	Users are producing twice the amount of observations (2,000 in 1 hour)
Required system behavior	The first half of the observations are handled as in QS-U1. For the remaining half of the observations a response is returned saying that they must be resent later

Scenario reference	QS-U4. Availability of Services (availability & resilience)
Overview	To which extent Net4Care services are available
System environment	The deployment environment is working normally
Environment changes	The Net4Care server becomes unavailable for hardware or software reasons
Required system behavior	Service is reestablished. Availability over a rolling one year period of 99.5 %

Scenario reference	QS-U5. Failure of Server, Handling on Client
Overview	How telemedicine client (e.g., a spirometer and user interface) handles Net4Care server failure (cf. QS-U4) (availability & resilience)
System environment	The deployment environment is working normally
Environment changes	The Net4Care server becomes unavailable
Required system behavior	Telemedicine client is able to continue operating for at least a day, sending observations when server becomes available

Scenario reference	QS-U6. Attack on Telemedical Data Store
Overview	How Net4Care handles attempts at security breaches (security)
System environment	The system is in overloaded or normal state (cf. QS-U3)
Environment changes	Attacker attempts to read telemedical data without credentials
Required system behavior	Attack is prevented, performance in overloaded mode is not degraded

Scenario reference	QS-U7. Integration of Read-Only Service
Overview	How interoperability with external (read-only) services are handled (evolution)
System environment	At development time, developer knows API of external service
Environment changes	A developer needs to integrate data from an external, read-only service (e.g., `min.medicine.dk`)
Required system behavior	A component integrating the data is developed and tested in less than one day

Scenario reference	QS-D1. Buildability of Components
Overview	How developers may build components for Net4Care (developer ressource, evolution)
System environment	The development ressources of Net4Care are operating normally, SMB developer has no prior knowledge of Net4Care
Environment changes	An SMB developer wants to develop a component (e.g., for spirometry)
Required system behavior	The SMB developer is able to develop and test a component on a single machine in less than one day

Scenario reference	QS-D2. Component Store Availability
Overview	How available the component store is
System environment	The component store is operating normally
Environment changes	The component store fails
Required system behavior	Component store is returned to service, total availability over a rolling year of more than 95%

Scenario reference	QS-D3. Secure Component Store
Overview	The extent to which the component store can withstand attack (security)
System environment	The component store is operating normally
Environment changes	An attacker tries to store (or get) components without proper credentials
Required system behavior	Access is denied and logged. Performance of component store may degrade

Chapter 4

Architectural forces

4.1 Goals and constraints

The major business strategy in the context of Net4Care is based on the national action plan on telemedicine (Fonden for Velfærdsteknologi, 2012) in which the need for telemedical solutions is stressed. In the context of Net4Care the main business driver is the aging population resulting in an increasing number of persons with chronic diseases (as also outlined in the action plan). This leads to a twofold business goal on i) allowing for secure telemedical observation storage and query (in a multitude of telemedical applications) and ii) allowing third-party developers to reduce their investment needed to produce telemedical applications. Central business standards related to Net4Care are the act on processing of personal data [1] and the health act [2].

At the highest solution-focused level, the strategy of healthcare IT in Denmark (Digital Sundhed, 2008), aims among others for common, national health IT services. The has among others lead to the Net4Care technology driver of NSP and a decision to have as a technological goal that Net4Care becomes the platform for a software ecosystem of telemedical applications. Also, international standards such as Continua Alliance, XDS, and HL7 constrain the project (cf. P2 below).

Finally, the main real-world constraint of Net4Care is that design and development takes place within a research project with limited resources and that needs to end ultimo September 2012.

[1] "Persondataloven", LOV nr 429 af 31/05/2000

[2] "Sundhedsloven", LBK nr 913 af 13/07/2010

4.2 Architectural principles

Principle reference	P1. Open source
Principle statement	The components of Net4Care are open source. External open source components is used when appropriate with respect to functionality and quality
Rationale	Open source software better supports a collaborative approach to software development (cf. P6). Open source can help SMB developers in understanding Net4Care. Using external open source components allows for independence of further development of these
Implications	• An infrastructure for collaboration on Net4Care must be established (including license(s), software configuration, and policies) • Learning resources must be produced to allow developers to understand Net4Care • A licensing scheme must be decided
Further information	See `http://www.net4care.org`

Principle reference	P2. Open standards
Principle statement	Net4Care should follow open standards, in particular within telemedicine
Rationale	The use of open standards increases interoperability with other telemedical system components and may further integration of components developed by others (cf. P6)
Implications	• There will be a learning period in order to be able to use open standards effectively • Continua Alliance will be central; this will have implications on the architecture of Net4Care (e.g., in terms of protocols used)
Further information	Continua Alliance, HL7, XDS

Principle reference	P3. Quality over functionality
Principle statement	When there is a choice between adding extra functionality to Net4Care versus maintaining or achieving quality, we value quality higher
Rationale	Many existing open source (tele)medicine are of low quality, which arguably hinders uptake. Furthermore, the medical domain is critical and thus high-quality systems are important
Implications	• There will be iterations/sprints in which no (or very little) functionality is added • The scope of Net4Care will be reduced, but the end product will be of high quality • Quality attributes (e.g., reliability and performance) should be testable

Principle reference	P4. Three tiers
Principle statement	There are three main tiers of Net4Care: the home with a Net4Care-enabled device, the server with the Net4Care server-side running, and a storage (cf. P5)
Rationale	The Net4Care home tier is an implication of the concept of telemedicine. A server tier will allow us to utilize computing resources efficiently. A data tier follows from P5
Implications	• The architecture will be separated into three tiers/physical layers • On the home tier, there will be focus on efficient protocols and simple software • On the server tier, there will be focus on performance and flexible software • The data tier will not be under our (complete control), here adapters will be important

Principle reference	P5. No medical data storage
Principle statement	Net4Care will not in itself provide longterm storage of medical data
Rationale	Healthcare data is typically owned and stored by producers of the data. Owners of data are responsible for controlling and monitoring access to data
Implications	• Data will have to be cached (temporarily) for performance reasons • Legal implications of connecting data from separate sources must be analyzed • A dedicated telemedicine storage may have to be established (outside of Net4Care)

Principle reference	P6. Towards a software ecosystem
Principle statement	In architectural decisions, we favor decisions that will enable third-party developer to add components to Net4Care
Rationale	Telemedicine is complex and expensive for SMBs to develop. Net4Care should provide a framework for developing as well as a platform for deploying SMB components
Implications	• The architecture should focus on compositional techniques, e.g., based on composition of services rather than configurations • A main part of the Net4Care project will be non-technical (business and organizational) aspects of a software ecosystem • Third parties will have to be involved in the design of Net4Care
Further information	(Christensen and Hansen, 2012)

Chapter 5

Architectural views

5.1 Context view

5.1.1 Context diagram

Figure 5.1 shows the environment in which Net4Care operates and enumerates the most important internal and external system with which it interacts. Internally, the component repository allows developers to store components that may be deployed in the Net4Care platform (this is shown in Figure 5.2).

XDS.b implements both a registry (for registering references to telemedical observations) and a repository (for storing telemedical observations). Integrations with external health informatic systems (EPR systems, FMK, and NPI) is done through PHMR documents stored in an XDS.b repository. The interaction of patients and healthcare professionals with Net4Care, facilitated by XDS.b, is shown in Figure 5.3.

5.1.2 Interaction scenarios

Figures 5.2 and 5.3 show interactions with the Net4Care system and its context.

5.2 Functional view

Figure 5.4 shows an overview of the functional model of Net4Care. We focus on the runtime for end users (rather than developer users). The main architectural pattern (Avgeriou and Zdun, 2005) applied is that of Client-Server in which the Forwarder is the Client and the Receiver is the Server. Furthermore, the external XDS.b repository that XDSRepository wraps is the Repository of a Shared Repository. XDSRepository, DataSource, and

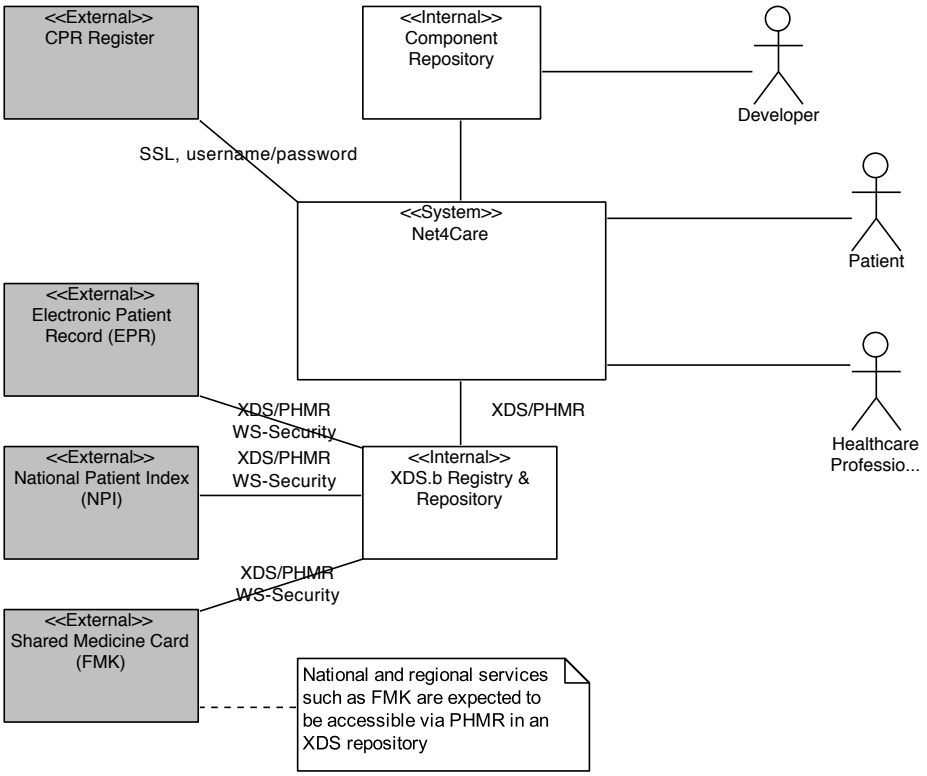


Figure 5.1: System context

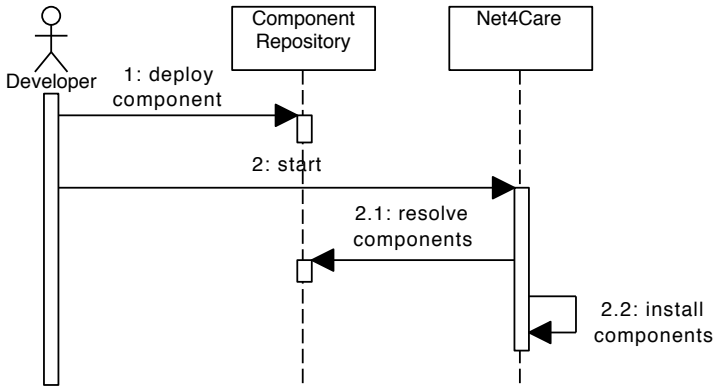


Figure 5.2: System context. Developer interaction scenario

ObservationCache are shown as abstract components since they configuration of components can be changed to, e.g., allow for testing. For example, an SQLiteXDSRepository implements a local XDS.b repository using

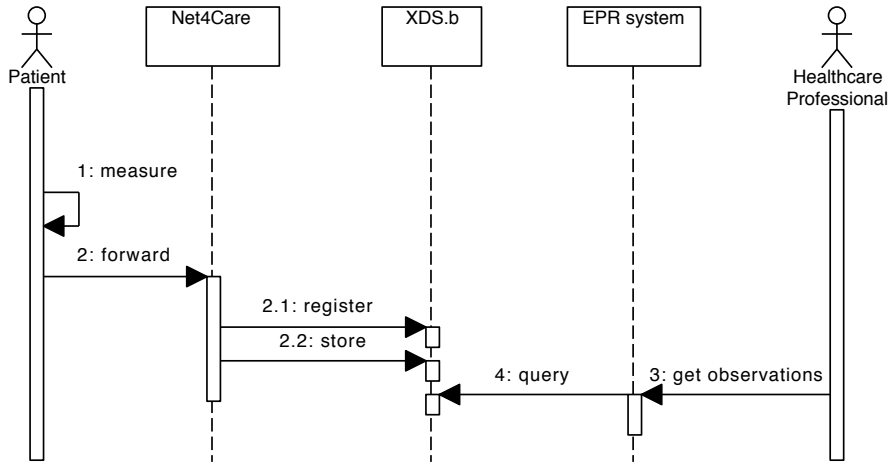


Figure 5.3: System context. Patient and healthcare professional interaction scenario

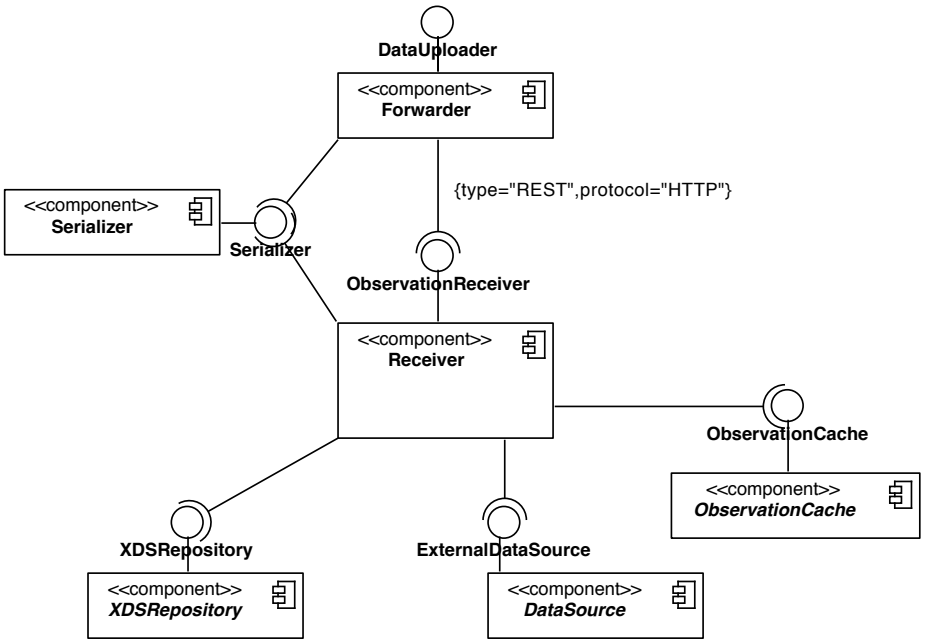


Figure 5.4: Functional model – overview

the SQLite database. Figure 5.5 shows Net4Care in a full configuration in which an external XDS.b repository (“MS Codeplex”) is used.

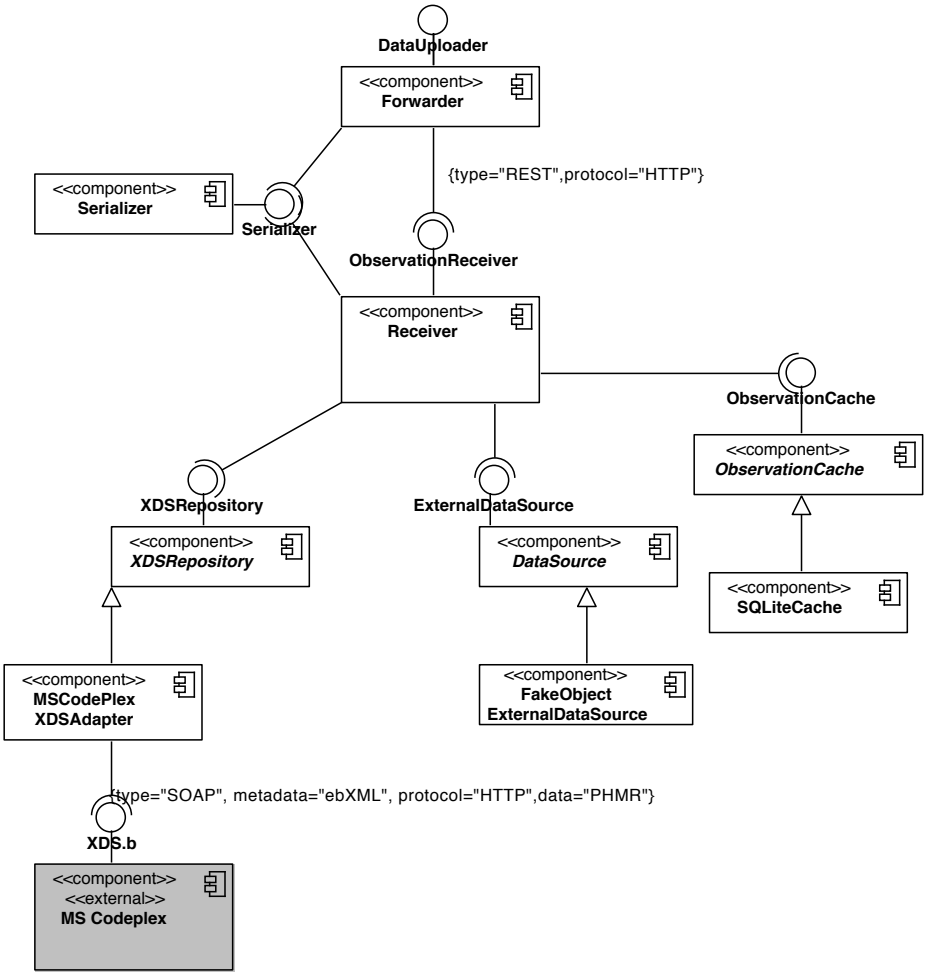


Figure 5.5: Functional model – full configuration

5.2.1 Functional elements

We here define the responsibilities and interfaces that are required and provided by each functional element.

Element name	Forwarder
Responsibilities	• Forwarding of telemedical observations to a Receiver • Abstract network protocol • Resolve queries on stored telemedical observations through a Receiver
Interfaces – required	Serializer, ObservationReceived
Interfaces – provided	DataUploader

Element name	Serializer
Responsibilities	• Serialize telemedical observation objects into the on-the-wire format used in network communication (concretely, this is JSON format) • Deserialize on-the-wire formatted data to telemedical observation objects
Interfaces – required	None
Interfaces – provided	Serializer

Element name	Receiver
Responsibilities	• Receive telemedical observations • Resolve queries for telemedical observations • Create PHMR documents based on received observations and data from DataSources • Forward observations to an XDSRepository and ObservationCache • Resolve observation queries using an ObservationCache or XDSRepository
Interfaces – required	ExternalDataSource, XDSRepository, ObservationCache
Interfaces – provided	ObservationReceived

Element name	XDSRepository
Responsibilities	• Register and store PHMR documents using metadata (e.g., CPR, timestamp, HL7 observation codes) • Retrieve document sets based on XDS queries
Interfaces – required	None
Interfaces – provided	XDSRepository

Element name	DataSource
Responsibilities	• Query external data sources for person data based on identification of person (concretely via a CPR number) • Query external data sources for treatment data based on identification of treatment
Interfaces – required	None
Interfaces – provided	ExternalDataSource

Element name	ObservationCache
Responsibilities	• Cache telemedical observations based on metadata (e.g., CPR, timestamp, and HL7 observation codes) • Resolve XDS queries if the result is cached
Interfaces – required	None
Interfaces – provided	ObservationCache

5.2.2 Functional scenarios

We here present the two main end user scenario: i) storing telemedical observations and ii) querying for telemedical observations. Scenario i) is shown in Figure 5.6 and scenario ii) is shown in Figure 5.7.

5.2.3 System-wide processing

The main concerns of system-wide processing are i) delivery of observations across nodes and ii) exception processing across nodes. Issue i) is dealt with through (de)serialization of a common class (StandardTeleObservation) to an on-the-wire format (JSON currently). Issue ii) is dealt with through definition of a Net4Care-specific exception type (Net4CareException) that is propagated through the network protocol (HTTP).

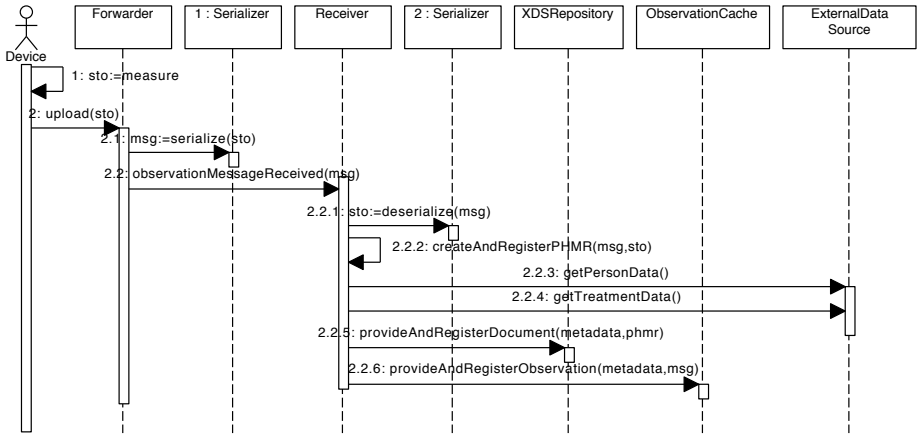


Figure 5.6: Functional scenarios – storing observations

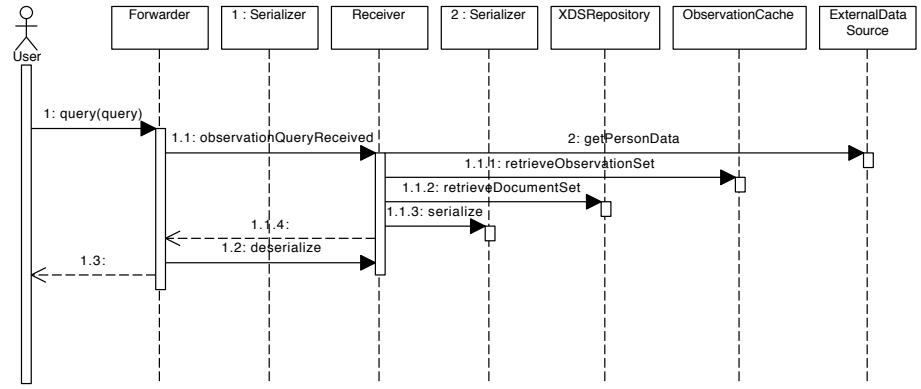


Figure 5.7: Functional scenarios – retrieving observations

5.3 Information view

5.3.1 Data structure

The static information structure model of Net4Care is shown in Figure 5.8. Objects of grayed classes are retrieved from external data sources, objects of white classes are generated by telemedical observations. The objective of the data model is that valid PHMR documents can be created from instances of it.

5.3.2 Data ownership

The systems owning data in the context of Net4Care are i) Net4Care itself, ii) external data sources, iii) XDS registries and stores, and iv) EHR/medical

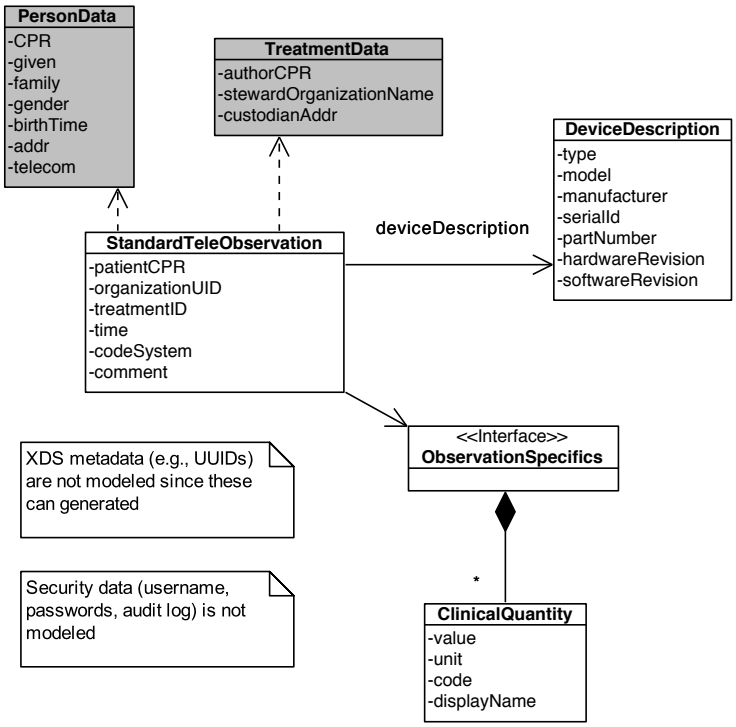


Figure 5.8: System data structure

information systems. The ownership relations between these systems and the data of Section 5.3.1 is shown in Table 5.1.

Entity/System	Net4Care	External Data Source	XDS	EHR system
Observation	creator, reader	none	master	reader
Person Data	reader	master	none	reader
Treatment Data	reader	master	none	reader

Table 5.1: Information ownership

5.3.3 Information lifecycles

The lifecycle of observations is shown in Figure 5.9. Observations may never be deleted. This leads to a need for annotating/referencing existing

observations to improve the quality of stored observations. An example of an observation that has a quality that needs to be improved would be an observation that is inadvertently made by a person distinct from the patient.

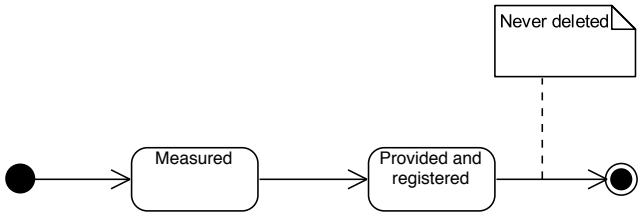


Figure 5.9: Observation life cycle

5.3.4 Timeliness and latency

We delay considerations on this until the performance & scalability perspective has been applied.

5.3.5 Archive and retention

Archiving and retention is outside the scope of Net4Care and will be handled by XDS stores and EHR systems.

5.4 Concurrency view

5.4.1 Concurrency model

Figure 5.10 shows how the functional elements of Net4Care are mapped onto operating system processes and (Java) threads. Of particular interest is the thread pool in the server process that is assumed to be implemented by the HTTP server used.

The server process is mostly stateless (i.e., request can be serviced independently). An exception is the (SQLite)cache for which resource contention handling is needed. A possible implication is that a different cache implementation may be needed.

5.4.2 State model

The state model of Net4Care is shown in Figure 5.11. The top states show a simplified OSGi lifecycle for the complete system.

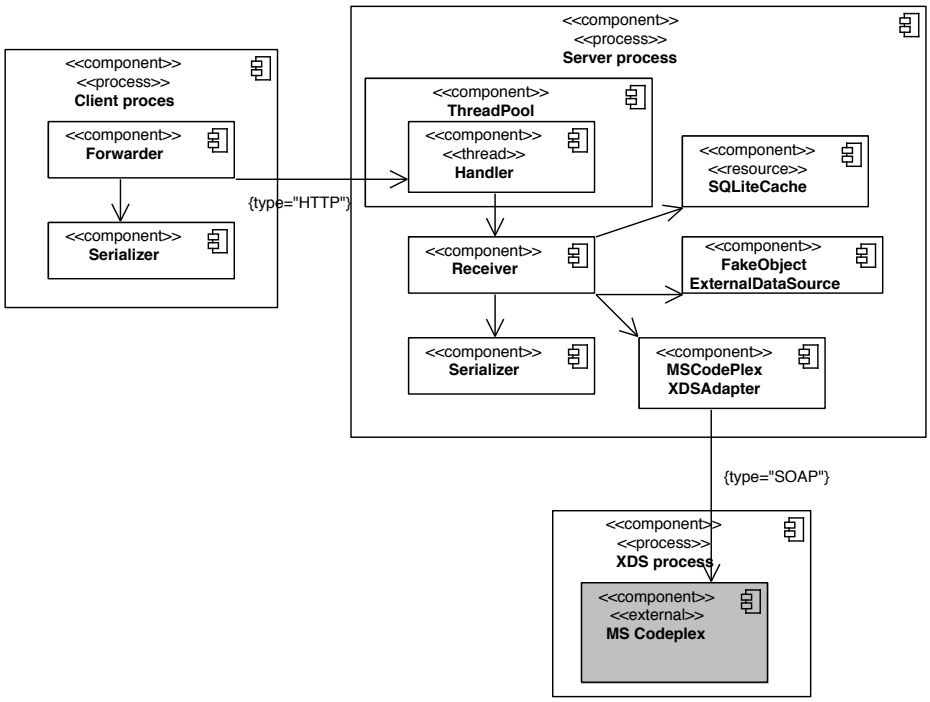


Figure 5.10: Concurrency model

5.5 Deployment view

5.5.1 Runtime platform model

Figure 5.12 shows the current deployment of Net4Care. The Net4Care Server and XDS.b Server are deployed in virtual machines at NFIT. Each internal artifact (white) correspond to a functional element in the functional structure model (cf. Section 5.6.2, page 30). On the Net4Care Mobile Client, Net4Care artifacts are compiled into the Android APK, i.e., the Net4Care deployment artifacts must be compatible with Android (2.1).

5.5.2 Software dependencies

In the following, we only describe the dependencies of XDS.b server node. This is done since a) only assumptions on virtual machines are done on other nodes and b) software that artefacts are dependent on are described in the POM file of each artefact.

The XDS.b server has the following software dependencies (Microsoft, 2012):

- Microsoft SQL Server 2008

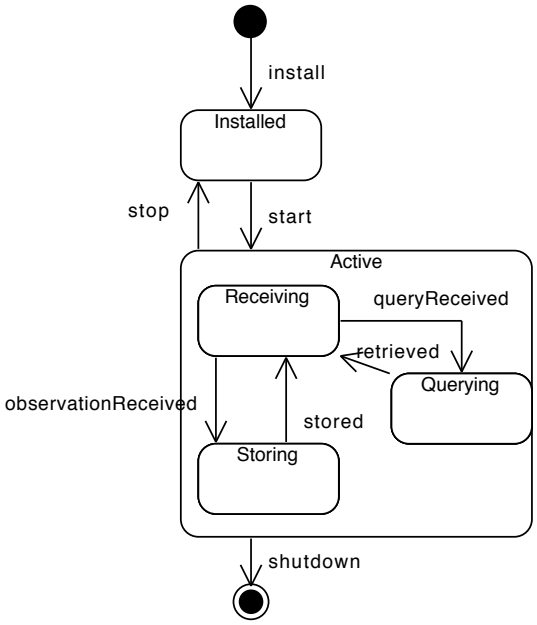


Figure 5.11: State model

- Microsoft Visual Studio 2010
- .NET Framework 4.0
- Enterprise Library 5.0 – May 2011

5.5.3 Network model

The network deployment of Net4Care uses the network infrastructure of NFIT. This includes that the Net4Care Server runs behind a firewall. This is illustrated in Figure 5.13.

5.6 Development view

5.6.1 Module structure

Figure 5.14 shows the module structure of Net4Care organized into layers.

5.6.2 Common design

In the following, we outline common design decisions:

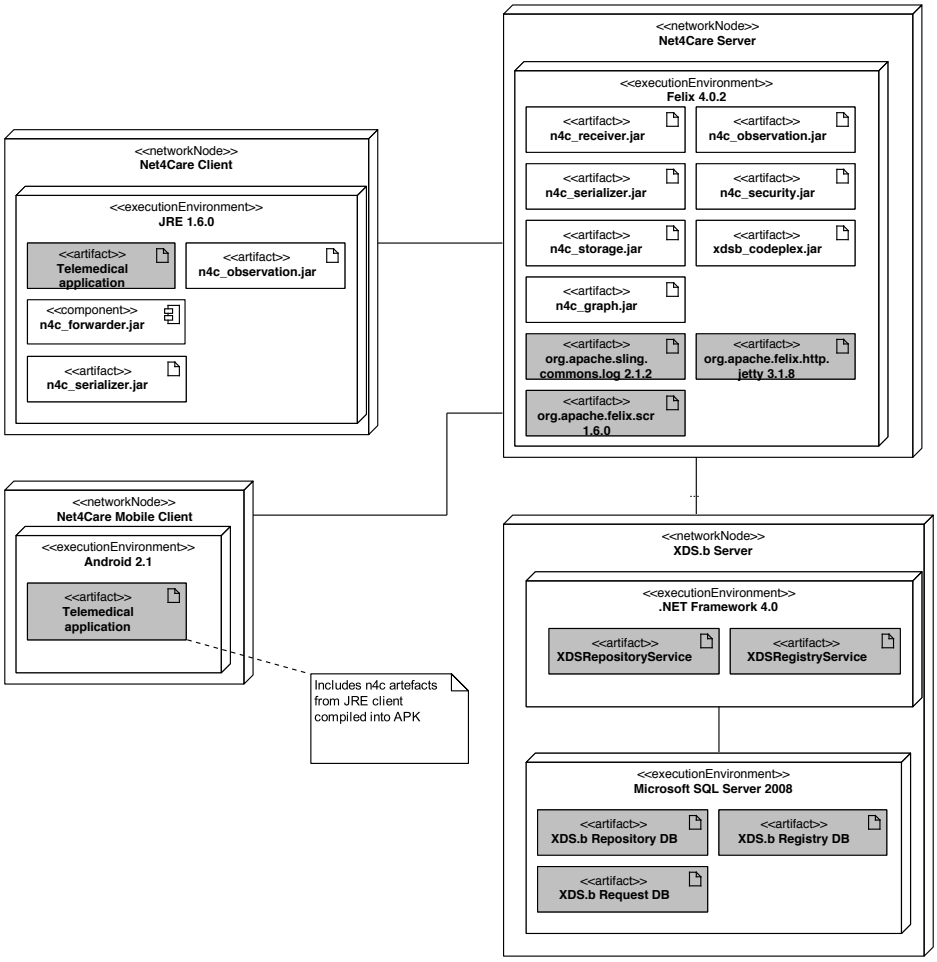


Figure 5.12: Net4Care deployment model

- Functional dependencies are resolved via *dependency injection* using OSGi. For testing purposes, functional components must be deployable in a command line, non-OSGi environment.
- *Logging* is done in each component using Log4J, logging important events according to severity levels
- *Authentication* must be performed by all HTTP Servlets using HTTP Basic Authentication

5.6.3 Standards for design, code, and test

Briefly, the following standards should be followed in developing for Net4Care:

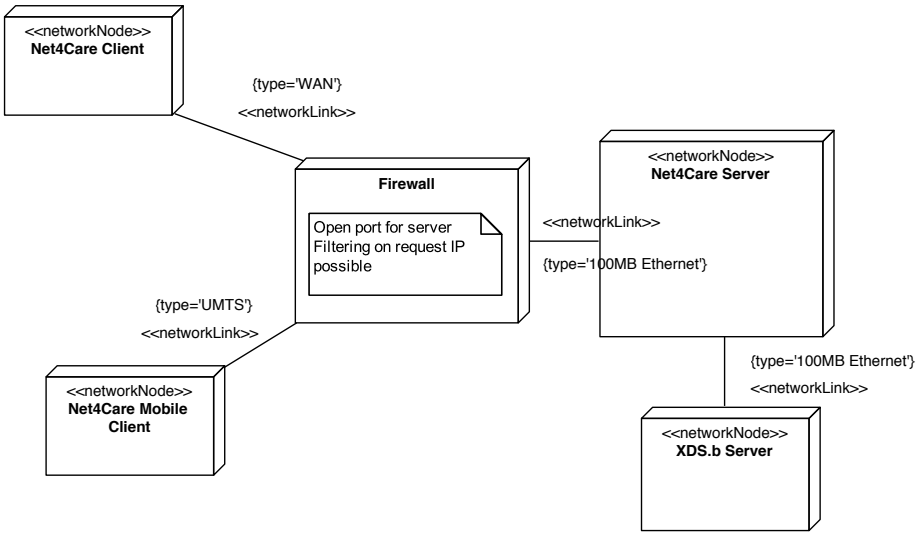


Figure 5.13: Net4Care network model

- Functionality should be defined through *interfaces*. Given these, *functional elements* are implemented as OSGi components implementing these interfaces according to OSGi's Declarative Service specification
- *Building* is done using Apache Maven with a project for each OSGi bundle. In this context, POM files are used to *resolve dependencies* to libraries
- Externally used software must be compatible with the Apache *license*
- Functionality should have *automated unit tests*, using JUnit
- Subversion is used for *configuration management*. There is a trunk for ongoing development and branches corresponding to each release (e.g., a branch for release 0.2)

5.6.4 Codeline organization

The codeline organization is:

```
net4care/
  n4c_osgi/
  n4c_example/
  n4c_site/
```

In `n4c_osgi`, bundles are defined. `n4c_example` contains example applications demonstrating Net4Care functionality, and `n4c_site` contains the source of the Net4Care website including learning material.

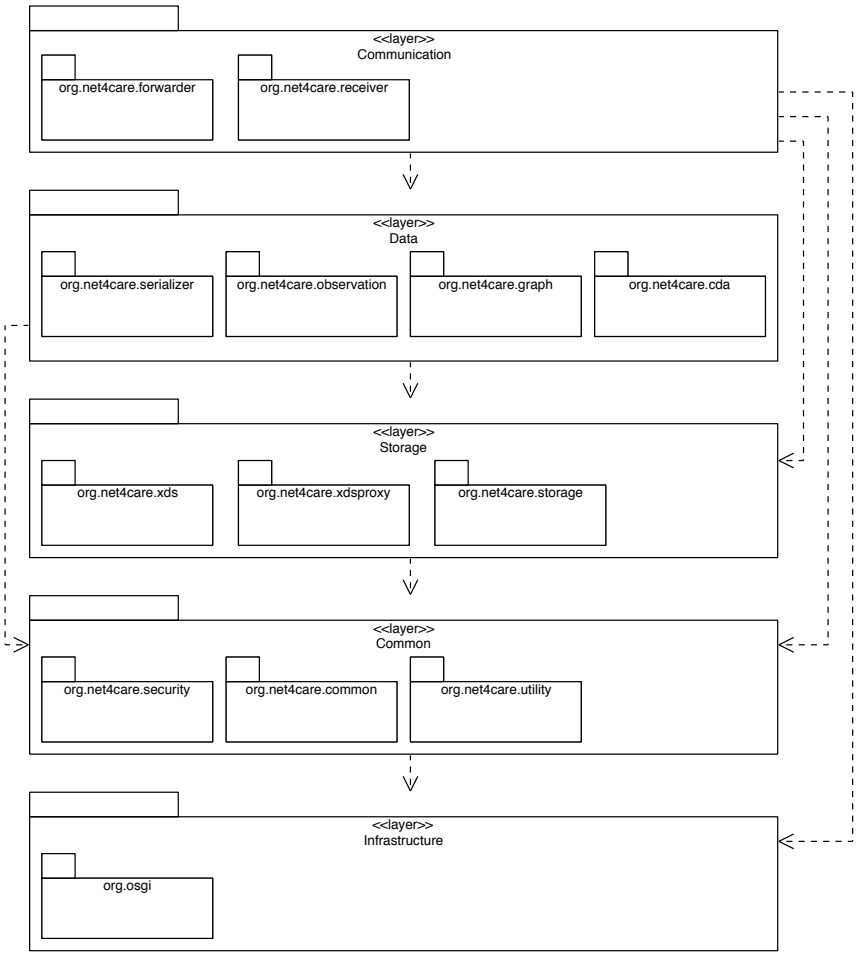


Figure 5.14: Module structure diagram – layers

For bundles, the codeline organization is:

```
n4c_<name>/
  src/
    main/
      java/
 pom.xml
```

Unit test code is put in the `n4c_test` bundle.

5.7 Operational view

5.7.1 Installation and migration

The following steps are required to install the complete Net4Care platform:

1. Install and configure Microsoft's XDS.b reference implementation (see (Microsoft, 2012))
2. Configure Net4Care's XDS proxy to point to the location of 1. (This is currently done as a compile-time configuration in the `n4c_xdsb_codeplex` bundle)
3. Download, build, and install the Net4Care server-side bundles (see `http://www.net4care.org` for a description). This can be done using Maven from the command line
4. To test the installation, build and test any of the Net4Care examples or test cases (see `http://www.net4care.org` for a description). This also explains in general how to set up client applications
5. Set up authentication if needed (see `http://www.net4care.org` for a description)

It is assumed that there is no existing system to migrate.

5.7.2 Operational configuration management

We identify the following configuration groups:

- *XDS.b parameters*. These are as XDS.b-specific, Microsoft IIS, and Microsoft SQL Server 2008 parameters, cf. Microsoft (2012)
- *Net4Care HTTP server parameters*. The documentation for the Felix HTTP bundle[1] describes these

No configuration sets have been identified, and thus the configuration groups are not defined further.

5.7.3 System administration

A primary monitoring facility is logging in the Net4Care server (cf. Section 5.6.2). A tool such as Apache Chainsaw can do log monitoring. Also, Log4J can be configured to send emails in the advent of error.

[1] `http://felix.apache.org/site/apache-felix-http-service.html`

The Net4Care server should be run on a machine that can be connected to via SSH such that standard command line tools may be used for control. For the XDS.b server, Remote Desktop should be used for administration.

No further elements of the management model are defined. This for when the platform has reached a sufficient number of users. Furthermore, management of client applications is not considered for now.

5.7.4 Provision of support

Application developers are the only category of stakeholders that will receive support through Net4Care. SMBs providing telemedical applications on top of Net4Care, handle support themselves.

Support is provided through Redmine[2] and email to `dev@net4care.org`. Framework developers are responsible for support on a best-effort basis.

[2] `http://redmine.net4care.org/projects/net4care/issues`

Chapter 6

System qualities

6.1 Performance and scalability

The main performance and scalability requirements for Net4Care are QS-U1, QS-U2, and QS-U3 (see Section 3.2.2, page 11).

A performance model related to QS-U1 and QS-U2 is shown Figure 6.1. Round-trip latencies have been estimated as worst-case estimates on a WiFi or local area network. Processing times for the XDS.b, Receiver, and SQLiteObservationCache are based on performance tests. In total we get that:

- Response time for QS-U1 is approximately 4 seconds
- Response time for QS-U2 is also approximately 4 seconds (data not available in cache)

While QS-U1 may be fulfilled, QS-U2 is clearly not in the case in which query result data is not cached. Further work on caching is needed.

Also, QS-U3 is not supported as-is. Monitoring of number of requests and subsequent state change for request handling is needed.

6.2 Security

Table 6.1 shows an analysis of the sensitive resources of Net4Care with a focus on end user applications. For access to these sensitive resources, the Net4Care security policy is outlined in Table 6.2.

The identification of sensitive resources and security policy leads to the follow threat analyses in the form of attack trees:

Goal 1: Change health status (“patient”, “next-of-kin”)

1. Tamper with telemedical device

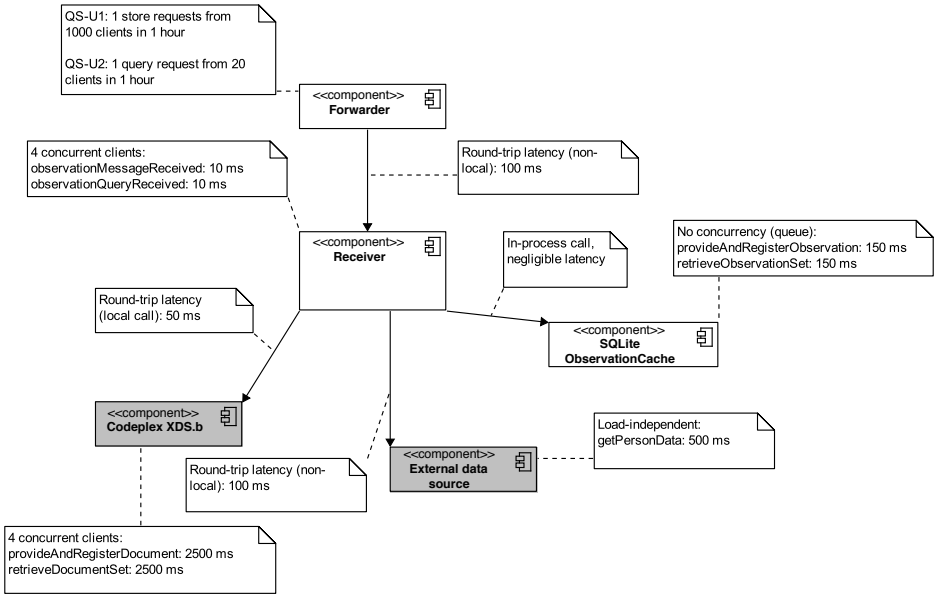


Figure 6.1: Performance model

2. Change observations on client
3. Intercept observations and change
4. Change data on server
   (a) Change cache database
   (b) Change XDS.b database
5. Produce false readings and send

Goal 2: Read patient/treatment data (“journalist”)

1. Read patient/treatment data by technical means
   (a) Attack client device
   (b) Man-in-the-middle attack
   (c) Attack on server
2. Read patient/treatment data via principal
   (a) Via patient
   (b) Via healthcare professional
   (c) Via administrator

Goal 3: Access patient reference data (“criminal”)

1. Access external data source
2. Access cached data on server

Resource	Sensitivity	Owner	Access Control
Patient reference data (CPR, address etc.)	Personal information of value for, e.g., identity theft	CPR register	Read-only access by system
Treatment data (steward, custodian etc.)	Defines treatment, if changed could lead to harm	Healthcare professional	Access to creation by authenticated principal, accountability
Observation data (clinical quantities, devices etc.)	Defines state of patient, potential for privacy invasion and harm	Patient user	Access to creation by authenticated principal
System data (passwords, configurations)	Must be controlled to maintain integrity	Net4Care administrator	Access to modifications by authenticated principal

Table 6.1: Sensitive resources – end users

3. Get reference data from principal

We finally outline the security measures that are implemented in Net4Care with references in brackets to the attack tree attacks that the measure is trying to prevent. The results of these design decisions are found in the models of Chapter 5.

	Patient reference data	Treatment data	Observation data	System data
Healthcare professional	Read on treated patient	Read on treated patient	Read on treated patient	Modify on own data with audit
Patient user	Read own data	Read own data with audit	Create and read own data with audit	Modify on own data with audit
Net4Care administrator	Read-only with audit	Read-only with audit	Read-only with audit	Full with audit

Table 6.2: Security policy – end users

1. It is assumed that medical devices are sufficiently tamper proof and that the communication to client devices is secured. (In the case of Bluetooth, e.g., this is provided through secure pairing) [attack 1.1]
2. Android and PC clients should be implemented using security mechanisms available on the platform [attack 1.2]
3. The Net4Care server is behind a firewall in a production environment. Access through the firewall is only via SSL [attack 1.3, 1.4, 2.1.a, 2.1.b]
4. Connections to external systems are protected using their security implementations. For the external CPR data source this is, e.g., TLS/SSL and username/password (CPR-kontoret, 2009). XDS.b connections should be protected with SSL and WS-Security [attack 2.1.c]
5. Users are authenticated and authorized. Authentication is via HTTP Basic
6. An audit log is kept for actions related to security [attack 1.5, attack 2.2]
7. Personal reference data is not used on the (untrusted) clients [attack 3]
8. Passwords are stored in hashed form (adding a random salt when hashing to prevent use of rainbow tables) [2.1.c]

9. Servers (XDS.b and Net4Care) are kept up-to-date with security updates. Device owners are asked to keep devices up-to-date [attack 1.4, 2.1.c, 3.2]
10. Security training should be provided to users (in particular Net4Care administrators) [attack 2.2, 3.3]

6.3 Availability and resilience

The main availability and resilience requirements for Net4Care are QS-U4, QS-U5, and QS-D2 (see Section 3.2.2, page 11). QS-U3 is related to resilience to increasing loads.

For the purpose of the requirements, the following service types, levels of service, and operational service levels can be identified:

- Types of services
  - Observation store and query: for this there is no interval during the day when no service is allowed. No storage (and only query based on cache)
  - Component store and install: This is only needed during working hours (and not outside normal working days)
  - (Statistics: could be a batch job run at night)
- Levels of service (for individual service types)
  - Observation store and query: normal, no storage (XDS.b down), no service
  - Component store and install:normal, no install (update cannot be performed), no service
  - (Statistics:normal,noservice)
- Operational service levels (for the whole system)
  - Normal operation
  - Degraded operation (no storage or no install)
  - Noservice

A statistics service types (for statistics on stored observation in the XDS.b store) has tentatively been added to consider batch-oriented service types.

Given these service types, Figure 6.2 shows a possible availability schedule for Net4Care.

In the following, we focus on the observation store and query service type (since QS-U4 is the central and most stringent availability requirement) and do a simplified availability calculation.

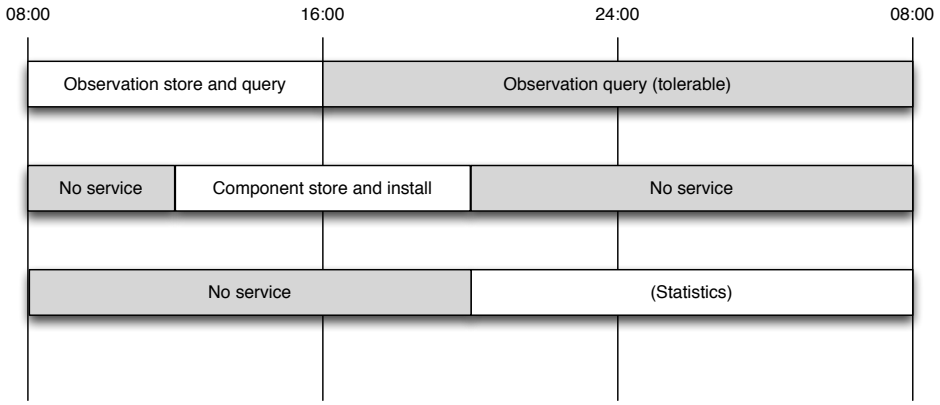


Figure 6.2: Availability schedule

We first assume that the main hardware failure type is disk failure. Assuming that the MTTR (Mean-Time-To-Repair) for NFIT including potential restore from backup (and logs) is 1 day and that the MTBF (Mean-Time-Before-Failure) for a disk (no RAID) is 10 years and that the faliures of the Net4Care server disk and XDS.b server are independent, we get the following hardware availability:

$$\text{availability}_{\text{hw}} = \frac{MTBF}{MTBF + MTTR} = 99.95\%$$

This is well within the QS-U4 requirements, however, failure rates for system software are higher. Assuming that MTTR for a system software failure (e.g., repair through reboot) is 4 hours, and that Windows (XP; XDS.b) has a MTBF of 1000 hours and Linux (RedHat; Net4Care server), system software availability becomes

$$\text{availability}_{\text{ssw}} = 99.60\% \times 99.95\% = 99.55\%$$

Table 6.3 shows an analysis of incidents leading to failures and their recovery consistent with the above.

In conclusion, these numbers leave little room for planned downtime during operation. An assumption is that we will not invest in better/more reliable hardware, so the availability schedule may have to be changed or that MTTR may have to be shortened (e.g., through better monitoring).

6.4 Evolution

Based on system quality scenarios (QS-U7 and QS-D1; see Section 3.2.2, page 11) and remaining features on Redmine[1], Table 6.4 shows relevant

[1] `http://redmine.net4care.org/projects/net4care/issues`

Incident	Impact	Remedial action	Time to repair
Hardware (non-disk) failure, server	No service (if server) or no storage (if XDS.b)	Replace component, restart	4 hours
Disk failure	No service	Replacement of failed disk, restore from backup and/or log	1 day
Non-transient network failure	Temporary service outage, loss of throughput	Reestablish network	Not quantified
Operating system crash	No service	Reboot (but maybe failure is systematic)	4 hours
Data corruption	Storage and query service unavailable	Recover data from backup and observation log	1 day
Application software failure	Storage and/or query service unavailable	Restart application/component	Not quantified
External service failure	CPR not available, only available for cached data leading to storage service failure	None	Not quantified

Table 6.3: Incident recovery scenarios

types of evolution, magnitude estimate (Low/Medium/High), likelihood of being necessary (Low/Medium/High), and the timescale on which the evolution is expected to happen (Long/Medium/sHort).

Evolution types that are of high magnitude, high likelihood, and expected to occur on a short timeframe are to be prioritized in architecting. However, a guiding principle (cf. architectural principle P3, page 16) is that we to a large extent pay for evolution cost when evolution is needed. The highest priority evolution types are

- Addition of external services. This is facilitated via OSGi services, however, no support for integration per se (e.g., caching, fault tolerance)
- New REST protocol. A Java Facade shields from this (on client and server)

- New query types. This is handled as needed

	Type	Magnitude	Likelihood	Timescale
Integration of new service	Integration	M	H	H
New component	Functional	L	H	M
New query types	Functional	L	H	M
New protocol (REST)	Functional	M	H	H
New protocol (XDS)	Platform	H	L	L
Add areas of HL7	Functional	H	H	L
Support twice the number of end users	Growth	H	M	L
Support twice the number of developer users	Growth	M	L	L

Table 6.4: Evolution requirements ranking

Appendix A

Architecture backlog

The following briefly lists outstanding architectural decisions that need to be made or effected in Net4Care:

1. Context view: update with security information for external systems
2. CPR security (cf. (CPR-kontoret, 2009)) should be implementable: need to store system-to-system credentials
3. XDS.b should be configured to support WS-Security
4. REST protocol should be redesigned to not use CPR numbers. A mapping from username to CPR must be made on server
5. State as a development principle that CPRs are not on client
6. Operational view: XDS.b and Net4Care servers should be kept up-to-date wrt. security
7. Operational view: client device owners should be told to keep device up-to-date
8. Operational view: security training should be described
9. Information view: add security information
10. Concurrency model: update to reflect security
11. Design authorization support
12. Security perspective: consider threats, policies, and implementation for component repository
13. Development view: mention need to use security mechanisms available on clients
14. Functional view: update protocol to HTTPS

15. Functional view: determine if getTreatmentData function of ExternalDataSource is realistic (i.e., can we assume that external data source will have such an interface)
16. Cache external data (e.g., CPR)
17. QS-U2 is not fulfilled, 4 seconds response time for a query is too high. Optimize XDS.b database or better caching mechanism
18. Model that system changes state after too many requests (QS-U3)
19. Define disaster recovery plans
20. Check the backup made by NFIT (and restore from backup)
21. Transaction log (check the completeness of this) should be logged to other disk
22. Update deployment to include monitoring (e.g., Pingdom or Chainsaw)
23. Client-side buffering/caching. External service (CPR) caching
24. Support overload mode
25. Create updated roadmap to better estimate evolvability
26. Automated build to deploy production system
27. Rollback of deployment
28. Retain development environment (e.g., downloaded JAR files, machine images)

Bibliography

Andersen, T., Bjørn, P., Kensing, F., and Moll, J. (2011). Designing for collaborative interpretation in telemonitoring: Re-introducing patients as diagnostic agents. *I. J. Medical Informatics*, 80(8):112–.

Avgeriou, P. and Zdun, U. (2005). Architectural Patterns Revisited - A Pattern Language. In *EuroPLoP*, pages 431–470.

Barbacci, M., Ellison, R., Lattanze, A., Stafford, J., Weinstock, C., and Wood, W. (2002). Quality Attribute Workshops, 2nd Edition.

Bardram, J., Christensen, H. B. r., and Hansen, K. M. (2004). Architectural Prototyping: An Approach for Grounding Architectural Design and Learning. In *WICSA*, pages 15–24.

Christensen, H. B. and Hansen, K. M. (2012). Net4Care: Towards a Mission-Critical Software Ecosystem. In *Proceedings of WICSA 2012*.

CPR-kontoret (2009). CPR Direkte Personnummer. Beskrivelse af grænseflade for OFF4. `http://cpr.dk/cpr_artikler/Files/Fil1/4285.pdf`.

Digital Sundhed (2008). National strategi for digitalisering af sundhedsvæsenet 2008-2012. http://www.ssi.dk/Sundhedsdataogit.

Fonden for Velfærdsteknologi (2012). National handlingsplan for udbredelse af telemedicin. http://www.digst.dk/.

Microsoft (2012). Xds.b reference implementation build and deployment guide.

Rozanski, N. and Woods, E. (2011). *Software Systems Architecture: Working with Stakeholders using Viewpoints and Perspectives*. Addison-Wesley Professional, second edition.