Title: Healthcare Application Design – Initial prototype design

Authors: Roi Sucasas Font (ATOS)

Editor:

Reviewers:

Identifier: Deliverable # D8.4.1

Nature: Report

Version: 1

Date: 31 March 2014

Status: Draft

Diss. level: Public

Executive Summary

The purpose of this deliverable (D.8.4.1) is to provide a design of the ATOS case study healthcare application after adopting the MODAClouds solutions provided in previously work packages, including the cloud providers’ selection and the deployment of this application within the MODAClouds framework. This report will be continued and extended in the document D.8.4.2 that will be delivered due to the project month 24.

Members of the MODAClouds consortium:

|  |  |
| --- | --- |
| Politecnico di Milano | Italy |
| StiftelsenSintef | Norway |
| InstitutulE-Austria Timisoara | Romania |
| Imperial Collegeof Science, Technologyand Medicine | United Kingdom |
| SOFTEAM | France |
| Siemens Program and System Engineering | Romania |
| BOC Information Systems GMBH | Austria |
| Flexiant Limited | United Kingdom |
| ATOS Spain S.A. | Spain |
| CA Technologies Development Spain S.A. | Spain |

Published MODAClouds documents

These documents are all available from the project website located at <http://www.modaclouds.eu/>

Contents

[Introduction 4](#_Toc382320609)

[1.1 Context and Objectives 4](#_Toc382320610)

[1.2 Structure of the Document 4](#_Toc382320611)

[Achievements 5](#_Toc382320612)

[2.1 Case Study Application Design 5](#_Toc382320613)

[2.2 Role of the MODAClouds Framework 7](#_Toc382320614)

[Use of the MODAClouds Framework Design Tools 8](#_Toc382320615)

[Use of the MODAClouds Framework Runtime Tools 10](#_Toc382320616)

[Conclusion 11](#_Toc382320617)

[Bibliography 12](#_Toc382320618)

[Appendices 13](#_Toc382320619)

[Appendix A 14](#_Toc382320620)

[A Case Study Application Design 14](#_Toc382320621)

[A.1 Introduction 14](#_Toc382320622)

[A.2 System Overview 14](#_Toc382320623)

[Requirements and Goals 15](#_Toc382320624)

[A.3 Design Considerations 16](#_Toc382320625)

[A.4 Architecture Design 17](#_Toc382320626)

[Server Application, Database and Third Party Services 17](#_Toc382320627)

[Client Application 20](#_Toc382320628)

[System Architecture 21](#_Toc382320629)

[A.5 Data Design 22](#_Toc382320630)

[A.6 User Interface Design 24](#_Toc382320631)

[A.7 Security Design 25](#_Toc382320632)

[A.8 Deployment Design 26](#_Toc382320633)

[Deployment 26](#_Toc382320634)

[Monitoring Rules and Adaptation Policies 28](#_Toc382320635)

[QoS Constraints 28](#_Toc382320636)

[A.9 Case study application design with the MODAClouds IDE 28](#_Toc382320637)

[Data Model and Application Components Design 29](#_Toc382320638)

[Appendix B 33](#_Toc382320639)

[B Definitions, acronyms and abbreviations 33](#_Toc382320640)



# Introduction

## Context and Objectives

This document, the deliverable D.8.4.1, reports on an initial version of the documentation of the design activities performed in the ATOS case study, including the cloud providers’ selection and the deployment of the application within the MODAClouds framework. This document takes as a basis the previous work done in the system functionalities specification document, and in the deliverables D.7.1.1 and D.8.1, in the sections related to the ATOS case study, but it is also an updated version of some of the design elements mentioned in these previous documents.

The ATOS case study is about a daily living healthcare system composed by several applications that aim to easy the lives of people suffering from dementia disease, and also the work of their caregivers and doctors. These applications will be developed, deployed, monitored and administered in one or in several clouds with the help of all the tools that MODAClouds offers.

This initial prototype design document will present an updated and extended high-level design of the healthcare application components, including all the elements that will be developed and deployed in the context of the MODAClouds project. It aims to provide all the stakeholders a clear understanding of this system and a practical description of how the MODAClouds IDE tools will be used for this purpose.

## Structure of the Document

In order to fully document all aspects of this initial version of the case study design, this document is structured in the following way:

* The Chapter 2 presents the ATOS case study design objectives and its achievements. This includes a brief description of the case study application design and the role of MODAClouds. The selection process of the cloud provider where the application will be deployed, and the deployment of the application within the MODAClouds framework will also be described in this section.
* Chapter 3 presents the conclusions.
* The appendices present a detailed description of the achievements mentioned in chapter 2, including an introduction of the use of some of the MODAClouds design tools used for the application design.



# Achievements

| **Objectives** | **Achievements** |
| --- | --- |
| Give a description of the case study application initial prototype design. | This document first presents an introduction and a short description of the case study application high-level design that is later extended in the appendices section. |
| Describe the role of the MODAClouds in the ATOS case study application. | Description of how the MODAClouds tools will be used in the design and run-time. This includes a short description of the selected cloud provider and how the application components will be deployed using the MODAClouds tools. |

## 2.1 Case Study Application Design

In this section a brief description of the case study application design will be presented. Readers can find a much more detailed description of the application design in the *Appendix A* section.

The ATOS case study application is a distributed and flexible telemedicine system or solution for the management of people suffering from dementia. Dementia is a collective term used to describe the problems that people with various underlying brain disorders or damage can have with their memory, language, and thinking, like Alzheimer. This new system will help these patients, but also their caregivers and doctors, by providing an integrated online clinical, educational, and social network tool to support all of them. Based on a set of monitoring parameters and measuring scales, the case study application aims to detect early symptoms that predict the decline of the patient. It also aims to avoid emergencies and secondary effects. Therefore, the period that patients can remain safely cared for at home will be prolonged, no matter where they are located. Other advantage offered by this solution is the improvement of the follow-up of these patients, while the workload of the clinicians is reduced.

This telemedicine solution can be seen as two main blocks; one of them would be the server-side block (hosted in a multi-cloud environment) and the other would be the client-side block (desktop client applications). But it can also be seen as a composition of four main elements or group of elements: a **server application**, a **desktop client application** a **database** service, and other **third party services**, like an email delivery service or a multimedia repository management service, that will be added to this system in order to enhance and improve it.

One of the main elements of this solution is the **server application**. It is responsible for the management of all the application logic and all its components will be distributed in the same or in different public or private clouds. This server element has two main components (but has the ability to include more in the future). The first one is a SOA based web services application that is responsible for the communications and transactions with the application database. It also acts as an interface with the other system components managing the connections and operations with other third party services (email, multimedia repository, etc.). Finally, this web services application is also responsible for part of the application logic, processing the data sent by the patients and managing the warnings. The other main component of the server application is a web graphical user interface application where the administrators and clinicians can do all their tasks, like managing the users, assigning exercises to the patients, assigning questionnaires to caregivers, monitoring the patients, and other kind of tasks. These server components will be implemented, deployed, and monitored using the MODAClouds framework.

In this telemedicine solution there is also a **desktop client application**, used by the patients and their caregivers, that collects all the patients’ measurements and parameter data that are sent later to the server. All this data is later processed and stored by the server application.

There are also a set of **third party services** that will be used by the application components, like an email delivery service to handle the communications with the system users, or like a multimedia repository to manage all the videos, audios etc. that will be available to the patients and caregivers. They will be used to enhance the functionalities of the application, making it easier for the developers to concentrate in the core elements of the healthcare system.

And finally, the last main element is a **database** used by the healthcare system to store all the information managed by the different components, i.e. the personal and medical data. This database can be a proprietary database hosted in a private cloud or can also be a database offered by a third party services provider. Depending on the solution, we have the option to use a public database offered by a cloud provider, which will also be designed, created and managed by the MODAClouds tools.

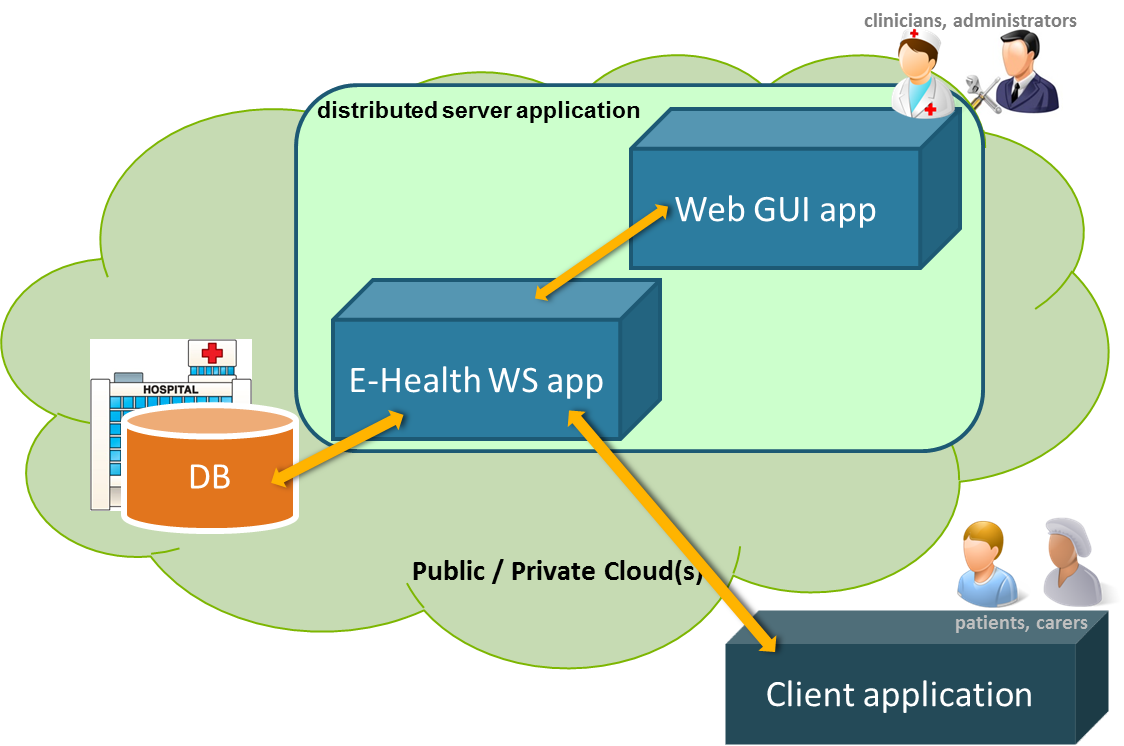


Figure 2.1.1: Daily living healthcare application main components

The architecture design of this flexible solution is based on a decentralized distribution of all the server-side block components in the cloud, so that this system can take advantage of the benefits that the multi-cloud environment offers. The server application components, the third party services, and the database will be distributed among one or several public or private clouds after the selection of the cloud provider. On the other side, the desktop client application will be out of the cloud infrastructure, accessing only the web services component of the server application.

The design of this solution has to take into account the kind of data (private data from patients, caregivers, and clinicians) that is managed by the different components, the communications between these components and how this data is stored. These communications will be secured and in some cases will also be encrypted. It will be explained with more detail in the appendices section (Data Design and Security Design).

All these components hosted in the cloud will also need to be administered and scaled up or down depending on the needs. It is expected to achieve all these tasks with the use of the MODAClouds tools and framework.

## 2.2 Role of the MODAClouds Framework

The objective of this section is to describe how the use of the MODAClouds tools will help in the design, deployment, monitoring, and governance of the healthcare solution. Although this document is focused in the design-time, the run-time aspects will also be briefly mentioned. An initial approach to the use of the modelling and design tools will be presented in the last points of the *Appendix A* section.

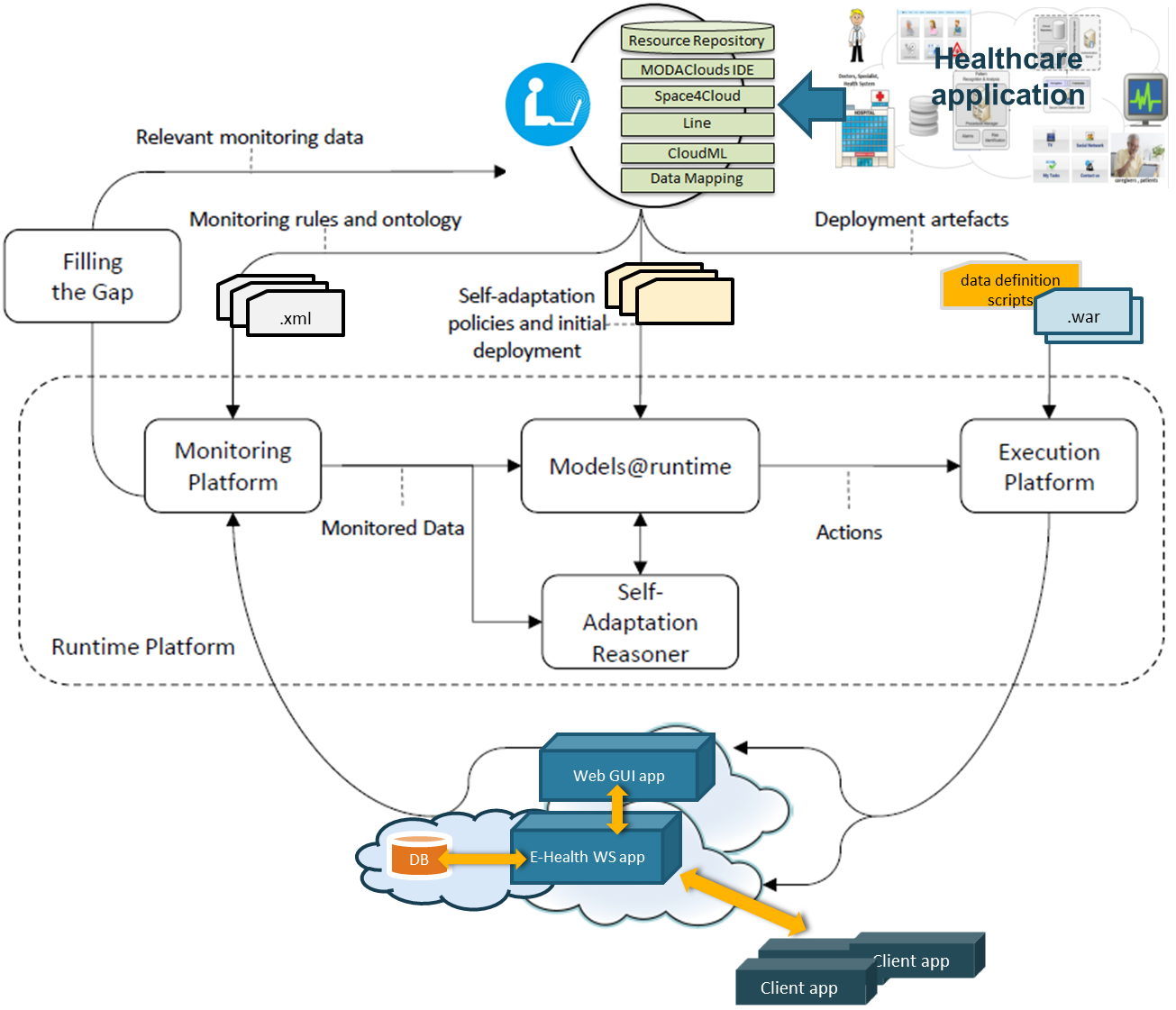
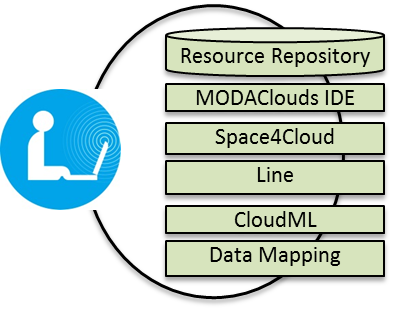


Figure 2.2.1: Application components within the MODAClouds architecture

Why use the MODAClouds framework instead of a manual approach? The cloud or clouds providers’ selection, where the server-side block components will be deployed, but also the implementation of these components and the governance of them will leverage both the MODAClouds design and run-time tools. First, these tools will help us in the design of these components and in the definition of the constraints, governance policies, QoS, etc. in a cloud transparent way. These above tasks, but also the data design (in a cloud agnostic way), and the cloud provider selection will be made with the MODAClouds framework design tools, like the MODACloudML and Decision Support System (DSS) tools. On the other hand, the runtime tools will be used first to do the deployment of the different components, and then to do the follow-up of these components, verifying and ensuring the accomplishment of the constraints, monitoring rules, and policies defined in the design time. These run-time tools are also responsible for reacting according to defined rules and constraints, migrating or scaling up or down the components when needed.

Designing and implementing the server-side components in a cloud agnostic way, will help to make the list of available cloud providers’ candidates much longer and on the other hand it will make much easier the migration of these components between the different clouds. Also, the monitoring of these components will make it possible to react properly when needed, according to a set of defined monitoring rules and adaptation policies.

Use of the MODAClouds Framework Design Tools

This section will describe the ATOS case study needs and how the MODAClouds tools (design time) or its functionalities will help solve these needs.

During the elaboration of this document, some of the design tools were still not finished, but the use of these tools, its functionalities and its main characteristics have been described in previous documents. Therefore the use of these tools will also be included in this section.

#### Application components design

The healthcare application components that will be deployed in the cloud have to be designed in a cloud agnostic way in order to be as adaptable as possible to the different clouds environments (for deployments, migrations etc.). The ATOS case study application has several elements that need to be deployed in the cloud: the database and the server application components. It is expected that the other third party services used by the healthcare solution will be provided by the selected cloud provider, so they will not be implemented by the MODAClouds tools, but will be defined with the design tools.

The functional modelling environment (**MODACloudML**) component tool and the others IDE tools will allow us to do this, defining a cloud agnostic architecture of the server components in various levels of abstraction (CCIM, CPIM and CPSM models) so that the portability between dissimilar cloud providers is ensured. With these tools we will create a set of models, artifacts and scripts that will describe the healthcare server side services, their interactions between each of them, their deployments, and the cloud environments where they will be hosted.

A more detailed description of the use of the MODACloudML tool will be presented in the last sections of the *Appendix A*.

#### Application data design

As it was mentioned before, one of the main elements of the ATOS healthcare solution is a database where all the data needed for the application will be stored, i.e. the personal and medical data of the patients. This database will also be defined and designed in order to run in a cloud environment. As the ATOS case study offers a great flexibility with this component used by the application (it can be a private database hosted in a private cloud, but could also be a database offered by a public cloud provider), the MODAClouds **Data Mapping Component** will be used in order to design how the working data can be stored through cloud data storage services, also in a cloud agnostic way.

#### Cloud provider selection

It is also needed to have an efficient tool in order to select a proper cloud environment where to deploy the server application components. The cloud provider selection will be made after a feasibility study of which cloud provider can support the runtime of the application components. Taking into account the specific needs of the application, its architecture and given a set of parameters and requirements, like costs, communication requirements, the programming language, the software components, the geo-location of the servers, etc., the **Decision Support System** (DSS) tool will offer a list of cloud providers’ candidates that match these parameters. From this list of providers we will be able to select the ones that satisfy our needs.

Because the tools needed for the feasibility study of the available providers that match our needs are not yet available, we have taken a manually approach to select a proper cloud provider to host the healthcare application components.

After a technical support and characteristics analysis of various platform providers, as well as a cost and other functionality aspects analysis, we have selected the Heroku platform for an initial deployment of the application components.

From a technical point of view, this provider supports the technical requirements of the application components that will be hosted in the cloud, like support for java web applications (1.6 or higher). It also offers a set of storage services, like MySQL or PostgreSQL.

This provider also offers a lot of add-ons. These add-ons are third party services like databases, email services, monitoring services, multimedia services etc. From all the third services offered by Heroku, we have selected three: a database (mentioned before) where all the application data will be stored, an email services application in order to handle the delivery of emails and a multimedia repository to store all the videos and media that can be accessed by the patients.

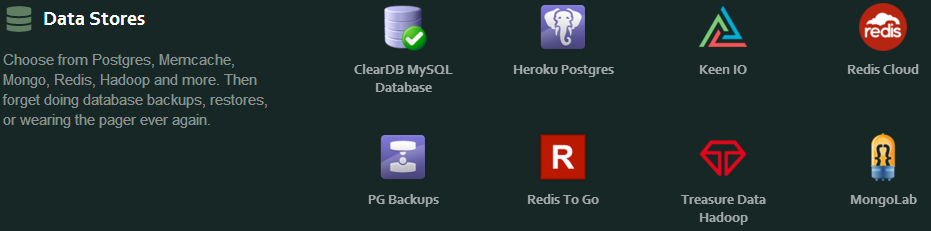


Figure 2.2.2: Heroku add-ons (offered third party services)

There’s also one reason why we have chosen this platform. It’s cheaper (for a short and medium period time of use of the resources we need) than most of the platforms we have analysed.

So the criteria selection would be the following:

* support for java web applications
* database services offered
* costs
* third party services offered
* PaaS provider

#### Monitoring rules, QoS, and adaptation policies definition

In order to make that the MODAClouds runtime tools do a proper and expected governance of the healthcare application components, first we will need to describe in some manner what we want to monitor, which parameters we need to monitor, when we want to scale up or down the different components and also when we want to migrate these components to other cloud providers.

In order to do that, MODAClouds offers the tools (**QoS modelling and analysis tool**) needed to define in the design-time these sets of QoS, monitoring rules and adaptation policies. Some of the rules, policies and definitions taken for this initial prototype will be described in the last section of *appendix A*, after the design components description.

Use of the MODAClouds Framework Runtime Tools

With the previously described needs and solutions offered by the MODAClouds tools, the monitoring and adaption at run-time will be possible. As it was briefly mentioned before, the MODAClouds run-time tools (**Monitoring Platform** and the **Self-Adaptation Reasoner**) reply to the need to verify and enforce the application constraints and policies at runtime. When the system detects a violation of any of the constraints or policies defined, the system is expected to react properly triggering user policies.

The MODAClouds run-time tools will also allow reducing the operational overhead with multi-cloud application management providing unified API to govern the application components (included the deployment and the un-deployment of them) with uniformed metrics across all deployed applications allowing comparing and contrasting between applications performance across clouds.



# Conclusion

This document (D.8.4.1) presented an initial version of the prototype design of the ATOS case study. It tried to introduce the use of the MODACLouds design-time tools in the development of the healthcare application. It is expected that after the other MODAClouds design-time and run-time tools are available, the forthcoming report will present a more refined version of the ATOS case study design. This forthcoming document, D.8.4.2, will be delivered in month 24.

# Bibliography

Marcos Almeida, Antonin Abhervé, Nicolas Ferry, Santo Lombardo, “MODACloudML IDE - Proof of concept”, *MODAClouds Deliverable 4.3.1*, 2013

*Case Studies requirementes - Initial Version, MODAClouds Deliverable 7.1.1*, 2013

*Case Study Design Analysis, MODAClouds Deliverable 8.1*, 2013

*MODACloudML development - Initial version, Deliverable 4.2.1*, 2013

*MODACloudML QoS abstractions and predictions model specification - Initial version, Deliverable 5.2.1*, 2013

# Appendices

2. Case Study Application Design

## A.1 Introduction

The *appendix A* presents a more detailed description of the healthcare application design in the context of the MODAClouds project. This appendix section is structured in the following way. Section A.2 presents the system overview, including a brief description of the main functional requirements in order to have a better understanding of the application components and its design, explained later. Section A.3 presents the main design considerations that have been taken into account. The following sections will present a high level design of the system architecture, the data, the security and the interface. Then the deployment design will be described, including the identification of the monitoring rules and adaptation policies needed to do a correct governance of the server components. And finally, the last section will present an introduction of the models generated by the MODAClouds IDE tools, trying to identify the correspondence between the different application elements and the expected MODAClouds IDE models.

## A.2 System Overview

The ATOS case study consists of a telemedicine solution for patients suffering from dementia disease that will benefit not only them, but also their caregivers and the clinicians that do the follow-up of these patients.

Taken as a basis a monolithic web application, this new system will be designed for a multi-cloud environment and deployed later in this environment with the help of the MODAClouds tools.

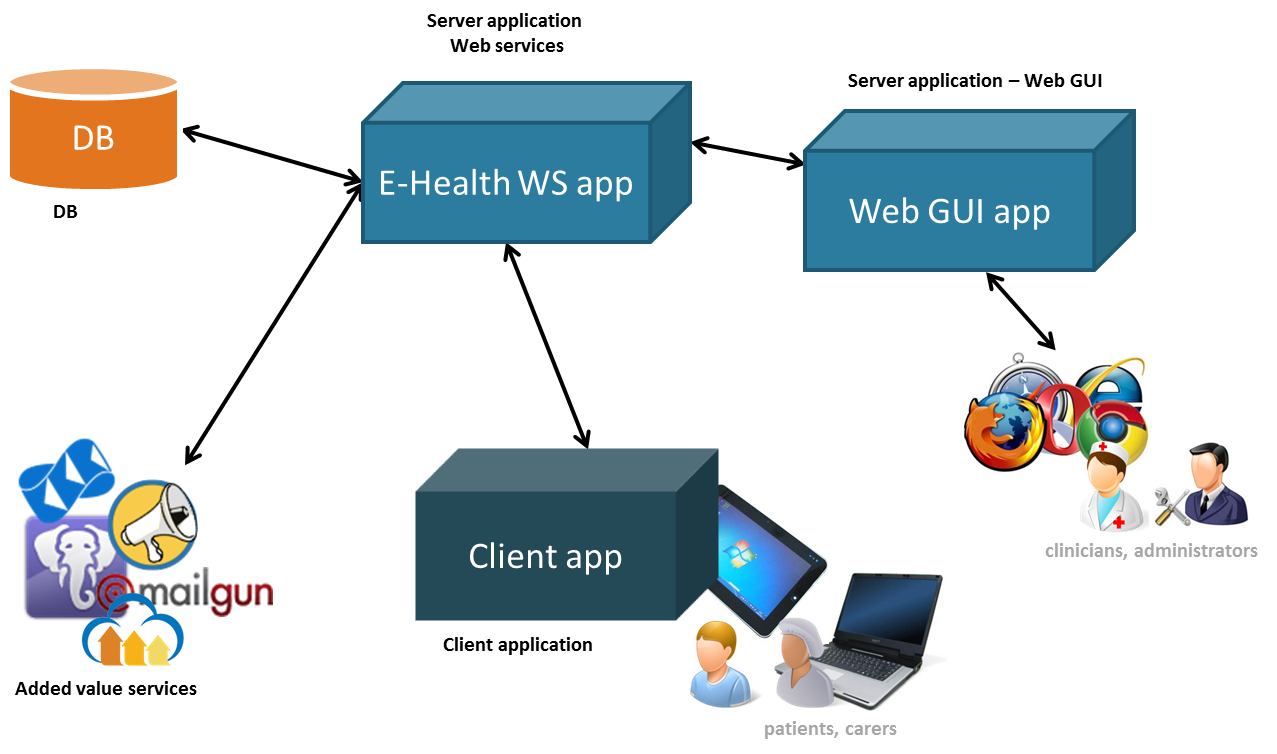


Figure A.2.1: Daily living healthcare application elements overview

This solution consists of two main blocks, a multi-cloud server-side block and a client-side block. The first one consists of a database, a server application and a set of third party services. The server application is also composed by two ‘big’ components, a web GUI application and a SOA based web services application. There are also a set of third party services that will be added to the application, like a multimedia repository and an email delivery service, in order to enhance the solution capabilities. The other main block, the client-side block, consists of a desktop client application that will be installed in multiple patients’ devices or computers.

The services or applications of the server side block will be hosted in a multi-cloud environment, so they will be designed, deployed and governed by the MODAClouds design and run-time tools.

On the other hand the client application will be deployed outside this multi-cloud environment, but will use the services available in the server-side block.

### Requirements and Goals

Before going into detail, a brief summary of the goals and functional requirements of the case study application will be listed.

* Governance of the application components in the cloud (done by the MODAClouds run-time tools)
* (Un-) Deployment of the application components
* Monitoring and management of these services or application components
* Migration of these services to other clouds
* Enforcement of the service level agreement
* General requirements
* Privacy and protection of the data managed by the application
* Administration and usage of the application
* Management of the application users profiles
* Monitoring and follow-up of the patient’s progress (daily activities and mental health of these patients)
* Management of the patient’s tasks (including the questionnaires)
* Management of the alarms and warnings

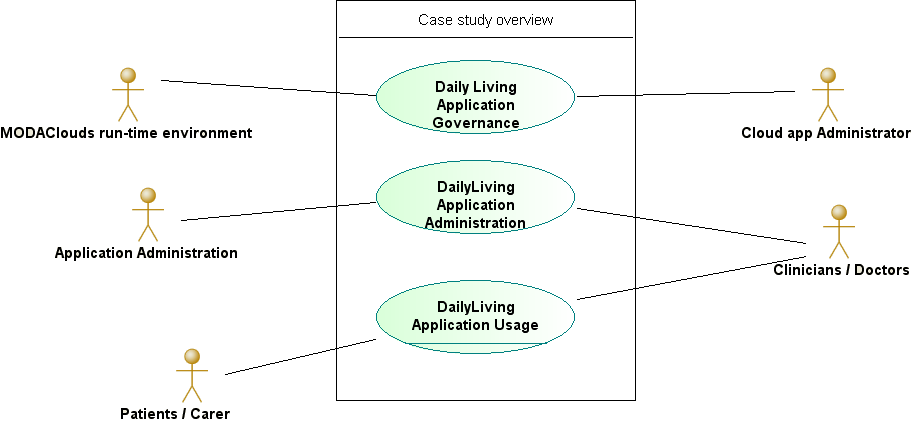


Figure A.2.2: Daily living healthcare application use cases overview

The functionalities described in the governance of the system components are also the main goals of the case study in the context of MODAClouds (run-time tools). The other functionalities refer to the application requirements.

## A.3 Design Considerations

This section describes some of the issues which needed to be addressed or resolved before attempting to devise a complete design solution of the healthcare system in a multi-cloud environment.

**Related software**

Because of the deployment in a multi-cloud environment and in order to benefit from the cloud environment properties, the server application has been divided in two sub-applications, a Web graphical user interface application and a web services application. The first one is a typical web application for the management and monitoring of the patients. The other one is an independent web services application that manages the communications with the database and other third party components. All the components that will be hosted in the cloud have to be independent of each other so they can be governed by the MODAClouds run-time tools separately one of each other.

Also because it exists the possibility of adding new features in the future, like other third party services or own components not listed in this document, the web services application has been designed taken that into account, including a possible split of this component in more independent sub-components.

**Operating systems / frameworks**

The desktop client application has been made using .NET, so all the devices used by the patients and their caregivers have to support .NET applications.

On the other side, the server components are java web applications or components that can be deployed in servers like Tomcat or similar. That means that one of the technical requirements that the cloud provider has to accomplish is the support for java web applications.

**End-user characteristics**

Also, because of the end-user characteristics, patients that suffer from dementia disease, the desktop application has an interface specially made for them, making it easier for them the navigation between the different screens and menus.

Another main characteristic that the end-user interfaces have to accomplish is the support of multiple languages. So the application has to be designed taken into account the need of multilingual support.

**Possible changes in functionality**

The web services sub-application could be divided in more components, depending on the different third party services that will be added in the future, and depending too on how they will interact with the other components of the healthcare solution. Meanwhile all these third party services will be managed by the web services application.

**Persistence**

Data persistence will be addressed using a relational database and J2EE s Object Relational Mapping capability. This relational database can be a proprietary database hosted in a private cloud or can be a database offered by a third party services provider. In a first approach, in this initial prototype design, the database will be a third party service offered by the target deployment cloud.

**Security requirements**

The system must be secured because of the kind of data that are managed by the different solution components. These data are all related to the privacy of patients and the other users of the healthcare application. Personal and medical data have to be protected in this cloud environment. In order to fulfil these requirements the system has to do the following:

* Provide security in the communications between the different components that are deployed in a public cloud.
* Confidentiality of the data stored in the database. Sensitive data will be encrypted.

**Technical requirements**

The healthcare system components have the following technical requirements:

* Desktop client application
  + .NET Framework 2 or higher
* Web services application
  + Java 1.6 or higher
  + CXF web services, spring
  + Hibernate
  + Jasypt (data encryption)
* Web GUI application
  + Java 1.6 or higher
  + CXF web services client
* Database
  + Relational database (MySQL)

## A.4 Architecture Design

The architecture of this solution is based upon a decentralized, service-based multi-cloud environment, which will benefit from a service-oriented approach that is based on Web services technologies. This architecture leverages the multi-cloud paradigm and will be organized according to the Service-Oriented Architecture (SOA) and structured in two main building blocks, whose elements communicate each other over the internet through secured SOAP calls. These two main blocks are the multi-cloud server-side block and the client-side block.

The elements that will be deployed in the multi-cloud environment will be presented first, and then the desktop client application will be described.

### Server Application, Database and Third Party Services

The server-side block consists of a database, two server application components and a set of third party services. The server application is also composed by two applications. Each of these elements are depicted in the following image.

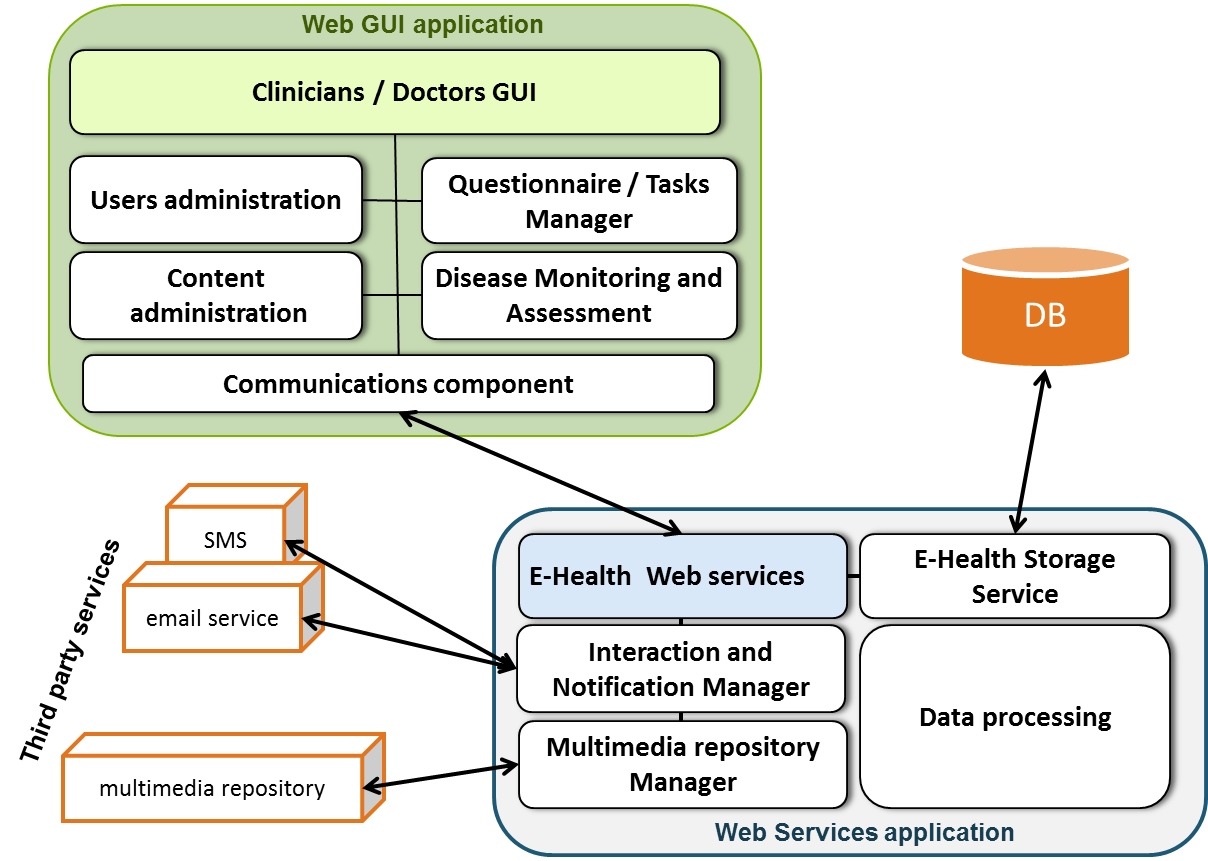


Figure A.4.1: Healthcare server application architecture with the database and the third party services

#### Web GUI application

The Web graphical user interface application is a self-contained server component that manages the monitoring of patients, the administration of the system users and also the management of the tasks and questionnaires assigned to the patients that suffer from dementia disease.

The main functionalities of this application are the following:

* Users’ administration.
* Monitoring and follow-up of the patients by the clinicians.
* Management of tasks and questionnaires.

This application is composed by several modules, responsible for doing these tasks:

* The GUI module is the interface with the administrators of the healthcare system and the clinicians. This module interacts with the other modules of the Web GUI application.
* The communications component module is a web services client module responsible for the secure communications between this application and the web services application.
* The questionnaire and task manager will do all the management (creation and edition) of the tasks and questionnaires assigned later to the patients.
* There’s also a disease monitoring and assessment module responsible for showing the clinicians the data (or warnings) needed for a correct follow-up of the patients.
* The users’ administration module is responsible for the management (creation, edition, roles management) of the application users.
* The content administration module is responsible for managing the content (external services and videos) shown to the patients in their client applications.

#### Web services application

The Web services application is also a self-contained server component that acts as an interface between all the healthcare solution components. It’s a SOAP web services application that will interact with the database, other third party services, and also the client applications.

The main functionalities of this component are the following:

* connections and transactions with the system database.
* interface with other third party services, like an email delivery service or a multimedia repository, offering the correspondent services to the healthcare solution components.
* Main logic of the application: process the patients’ data, warnings, messages, etc.

This component consists of the following modules:

* A set of web services offered to all the other components. The communications with the client applications, the database and the Web GUI application need to be secured because of the kind of data that is managed by these components.
* A storage service used to manage the connections and transactions with the database. It also manages the security and the encryption of these data in a transparent way.
* The notification and interaction manager is the responsible of managing all the warnings and other type of messages that are sent to the application users via e-mail (or SMS).
* A multimedia repository manager that is the responsible of the interaction between the application and the third party service that offers this repository.
* And a data processing module that is responsible of collecting and processing the patients’ data.

Because this component is the most important and sensible of all (it’s the node where all the communications go through, and all the others depend on it in order to run properly), it is the most susceptible to be scaled up or down by the MODAClouds run-time tools, depending on the needs. For this purpose, the expected monitoring rules and adaptation policies will be defined while working with the MODAClouds IDE tools.

#### Database

The database used by the application will be a third party service offered by some of the service providers that are included in many of the cloud platform providers. Because it will be needed to provide some security to this element, the server application will take care of this aspect, encrypting the data before storing them. It will be described in the Data Design and in the Security Design sections.

#### Third party services

The other third party services used by the application in this initial approach will be an email delivery service and a multimedia repository. In the future also a SMS gateway could be added to this set of third party services.

The email service will be used by the web services application in order to send emails to the doctors and administrators of the application. On the other side, the repository manager will be used to store videos and other type of multimedia content that can be useful to the patients (they can access this media content with the desktop client applications).

### Client Application

The client application consists of a .NET desktop application used by the patients and their caregivers. This application will be used by both of them to do the tasks and questionnaires assigned to them. This application will also be used to collect the measurements used to detect changes in the patients. It will also connect to multimedia content (like videos or games) or other kind of external services.

The main functionalities are the following ones:

* Validate the identity of the patient / caregiver.
* Offer access to services, like videos or games.
* Collect and register the data and measurements.
* Management of warnings or requests sent to the clinicians.

This application has the following main modules:

* A graphical user interface specially made for this kind of patients and the devices where the application will be installed in.
* A client logic component that is the responsible for all the application intern logic.
* A monitoring data processor that will collect all the patient measurements and process them before sending them to the server application (database), where they will be analyzed.
* A questionnaire player that will present the questionnaires to the patients, and will later process all the responses.
* A browser component used to access web pages and services that can be useful to these patients.
* A communication component that is the responsible for the communications between this application and the server application.

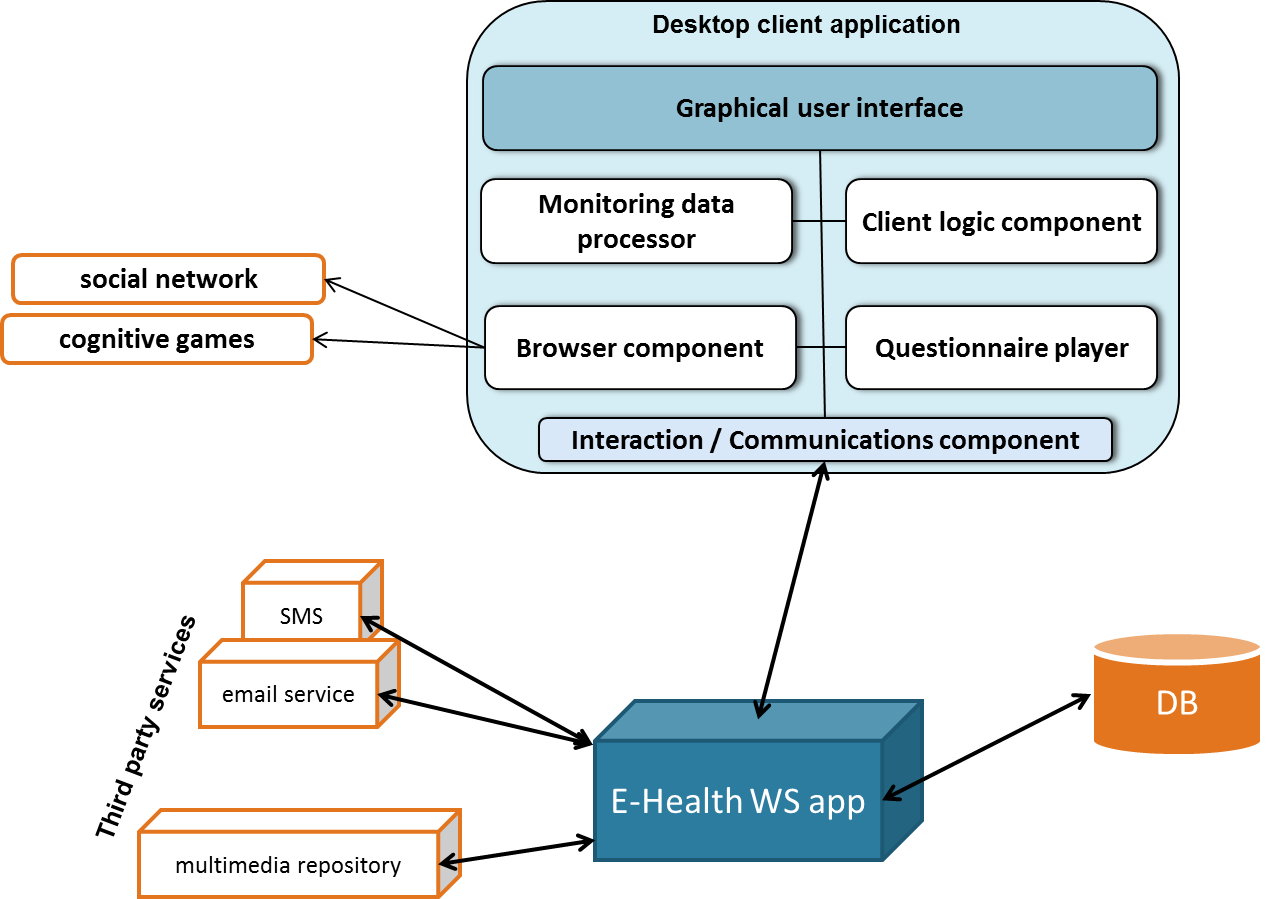


Figure A.4.2: Healthcare desktop client application architecture with the database and the third party services

### System Architecture

In this architecture we can clearly identify which is the most important component that will need to be monitored and carefully governed by the MODAClouds run-time tools. It is the **web services application**, which is responsible of the transactions with the database and also the interface with the other third party services. All the communications between the components of the healthcare solution go through this component. It may need to be scaled up or down depending on the circumstances. In some cases also the web GUI application could also be also scaled up or down. It will be detailed in the last sections of the appendix

The following image depicts a high level architecture of all the system (main components and main sub-modules) and the communications of the before described components.

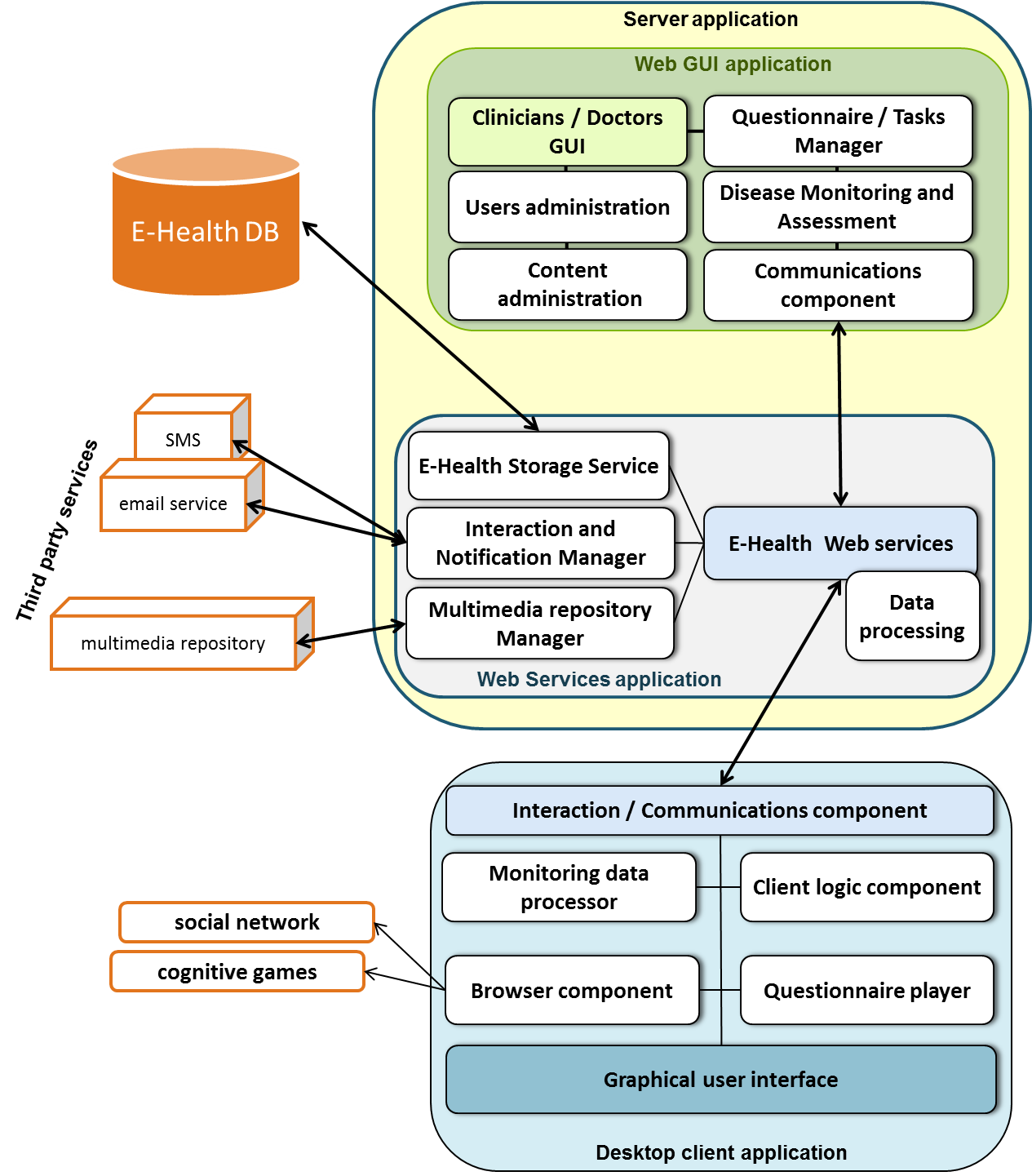


Figure A.4.3: Daily living healthcare application architecture

## A.5 Data Design

The healthcare telemedicine system will largely rely on a relational database to perform day-to-day operations and storing the patient’s data. Data will be stored in a MySQL DBMS (but could also use other database management systems) and manipulated by the E-Health storage service module (server application), which will ensure data integrity and consistency. The database will be provided by a third party service provider and hosted on a public server. The database shall be accessible without interruption regardless of time or day. The data stored in the database will include:

* + - Personal data of patients, caregivers and doctors, i.e. passwords, private data, contact data etc.
    - Medical data (measurements of the different kind of tasks) of patients
    - Questionnaire data (patients’ responses)

Because of privacy reasons and the relevance of the data that will be stored in this database, all the relevant data will be encrypted by the server application in a transparent way, independent from the DBMS.

Next picture depicts the database schema.

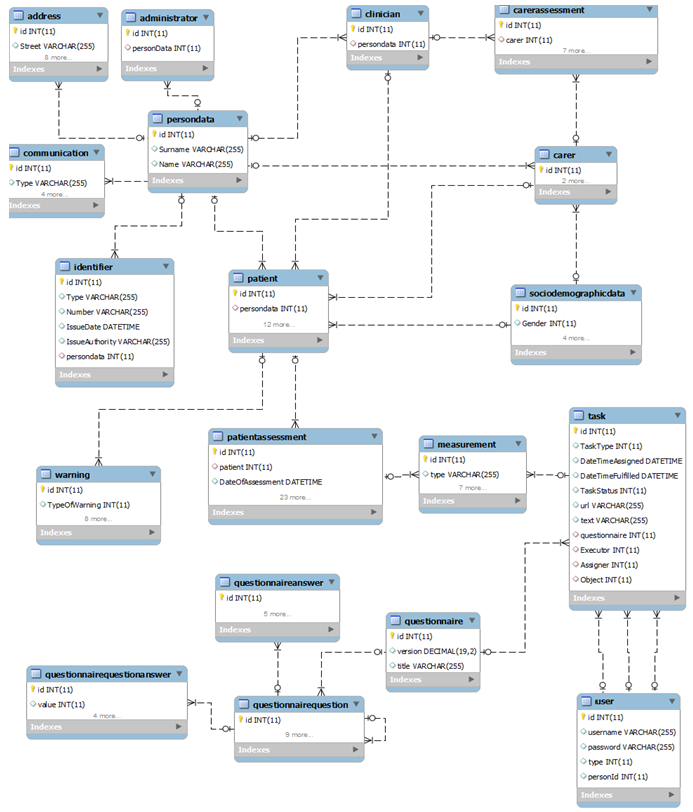


Figure A.5.1: Daily living healthcare database schema

## A.6 User Interface Design

As it was mentioned before, this application is made for the people that suffer from dementia disease, their caregivers and also their doctors and clinicians. Each kind of user has its own roles, functions and needs, so the application has to be adapted to each of these users. This is the case of the client application that has been designed taking that into account. It has an interface that is adapted to mobile devices and their users, mostly of them old people.

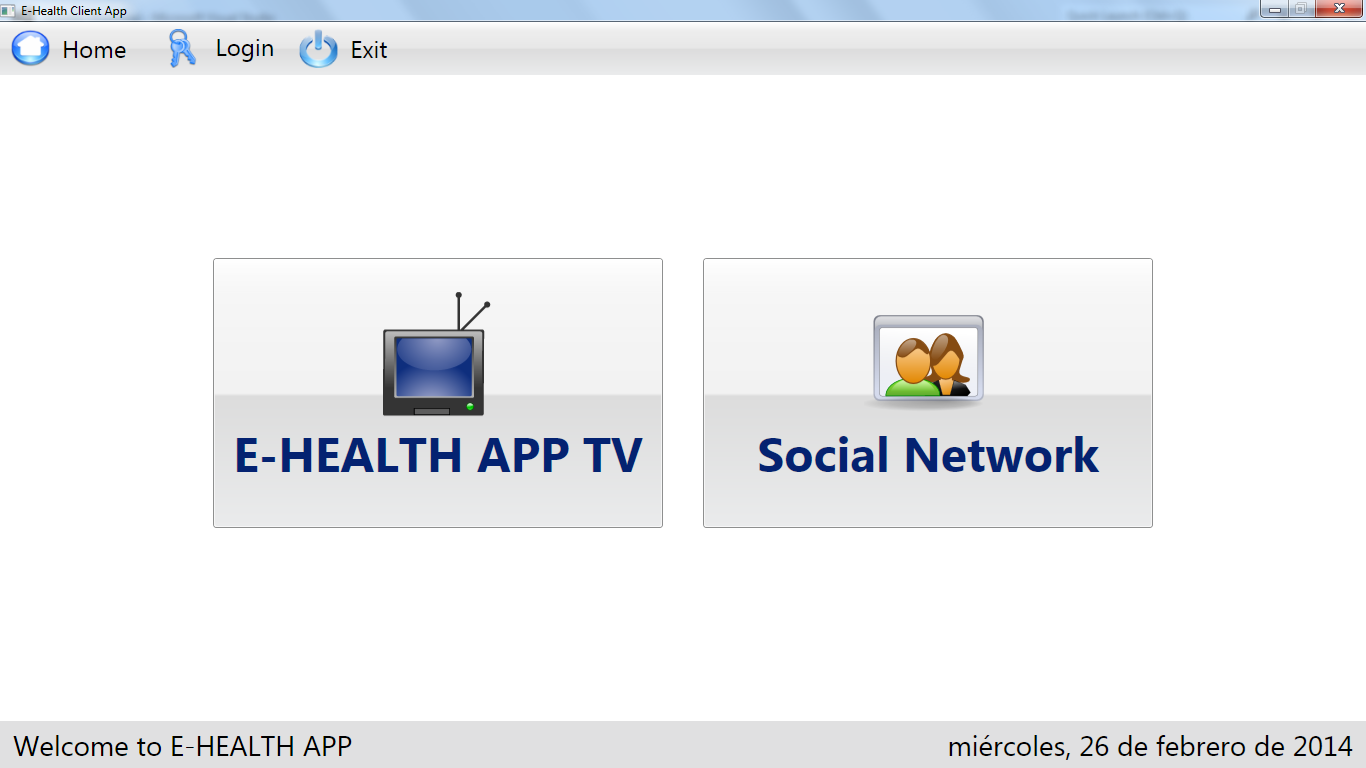


Figure A.6.1: Desktop client application interface example

On the other hand, the Web GUI application has a functional design, made to ease the use of this application by the clinicians and the system administrators.



Figure A.6.2: Web GUI application interface example

Both the Web GUI application and the desktop client application will have multilingual support.

## A.7 Security Design

One of the main design concerns is the security design of the communications between the different components. The privacy and protection of the data that will first be manipulated and sent between the system elements, and then will be stored in a database, needs to be designed and implemented.

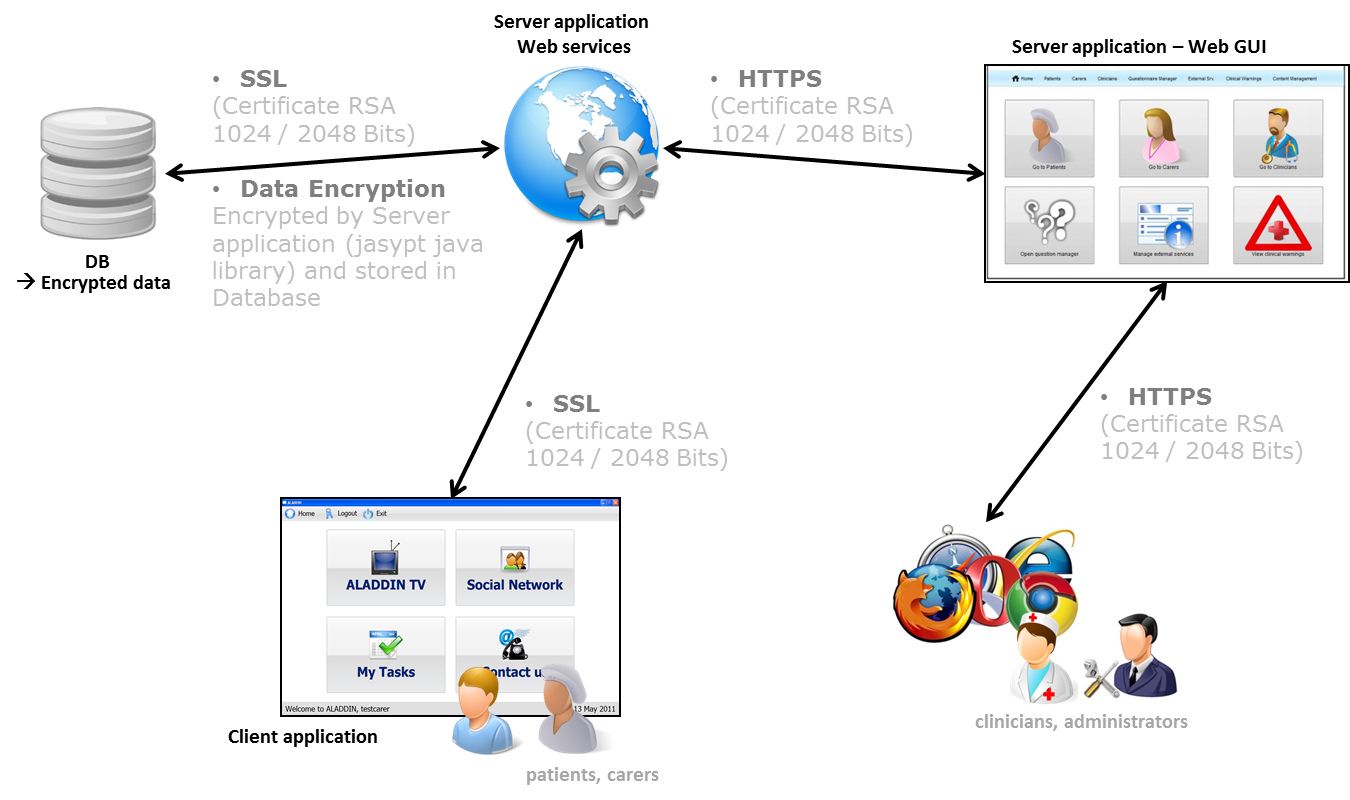


Figure A.7.1: Daily living healthcare application components - communications – security design

Regarding to this, there are two main aspects where some kind of security needs to be added:

- The communication between all the main elements of the healthcare solution where this data will go through

- How this data is stored in the database used by the healthcare system

In order to ensure an acceptable level of security and protection of the data in the communications between the different components, SSL / TLS will be implemented in these connections. To do that both the web server application component and the web GUI application component will be configured to be used as HTTPS services. Certificates with public keys of 2048 bits will be used for that purpose. Other element that has to be configured to use SSL / TLS is the database service.

Also the place where the data will be stored needs to be secured. In order to ensure that the data are stored in a proper (secured) way, the private data of the users and also the medical data of the patients will be encrypted when stored. This will be made by the web services application in a transparent way to the database.

## A.8 Deployment Design

This section shows how it is intended to deploy the different components in a multi-cloud environment. Taken into account the deployment model and also the other design aspects described in this document, we will be able to identify some of the most important monitoring rules and adaptation policies that the healthcare system will need while being governed by the MODAClouds run-time tools.

### Deployment

The healthcare solution server-side components will be deployed in Heroku, the selected cloud provider. Both the web GUI application and the web services application will be deployed as Heroku isolated instances that will be governed by the MODAClouds runtime tools. Each of these instances can be scaled up or down, depending on the circumstances.

Next picture shows the server application elements deployed in a cloud provider.

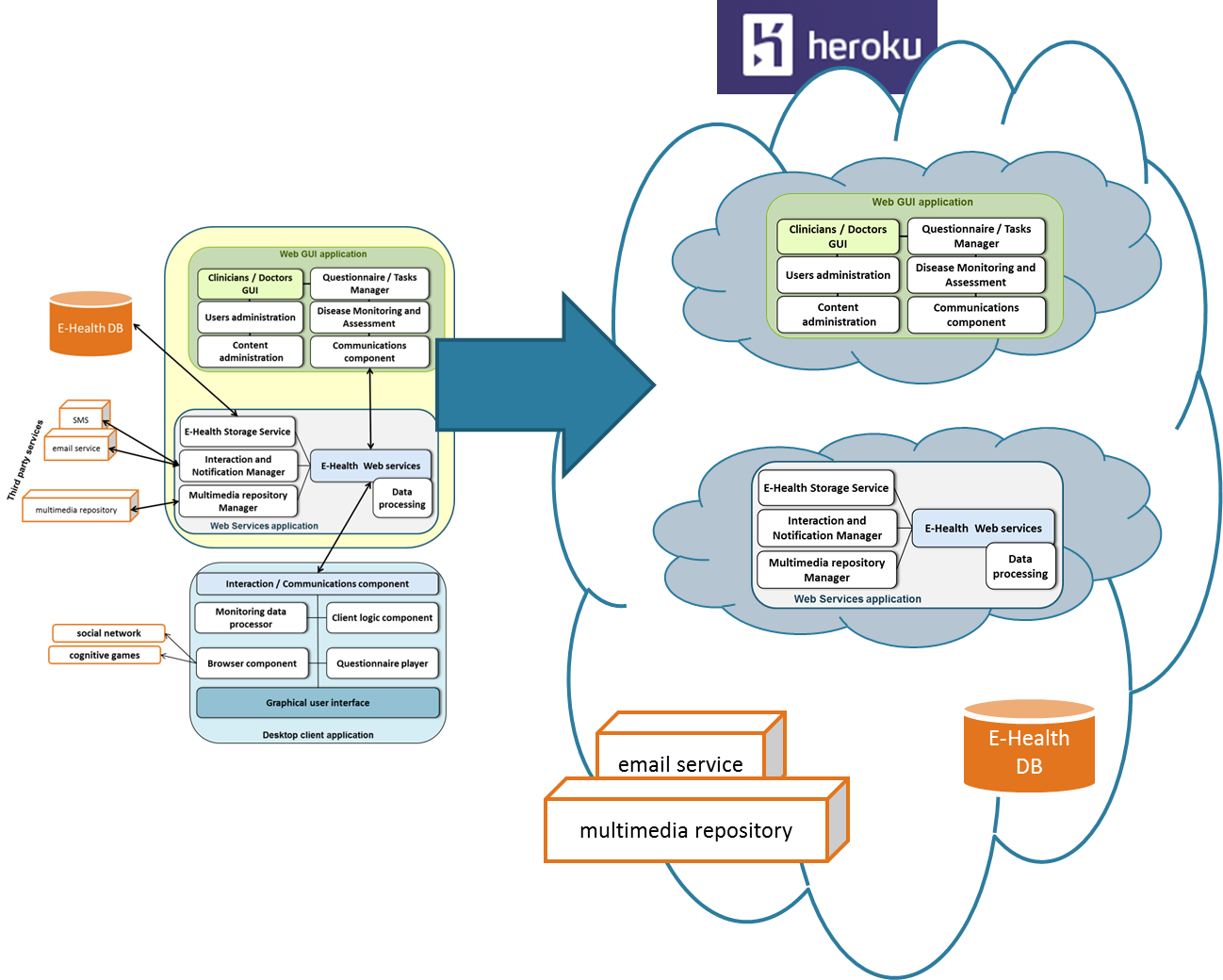


Figure A.8.1: Design to deployment transition

On the other hand, the third party services, including the system database, will be taken also from Heroku. These services can also be scaled up or down.

In the next picture the communications between these deployed elements is depicted.

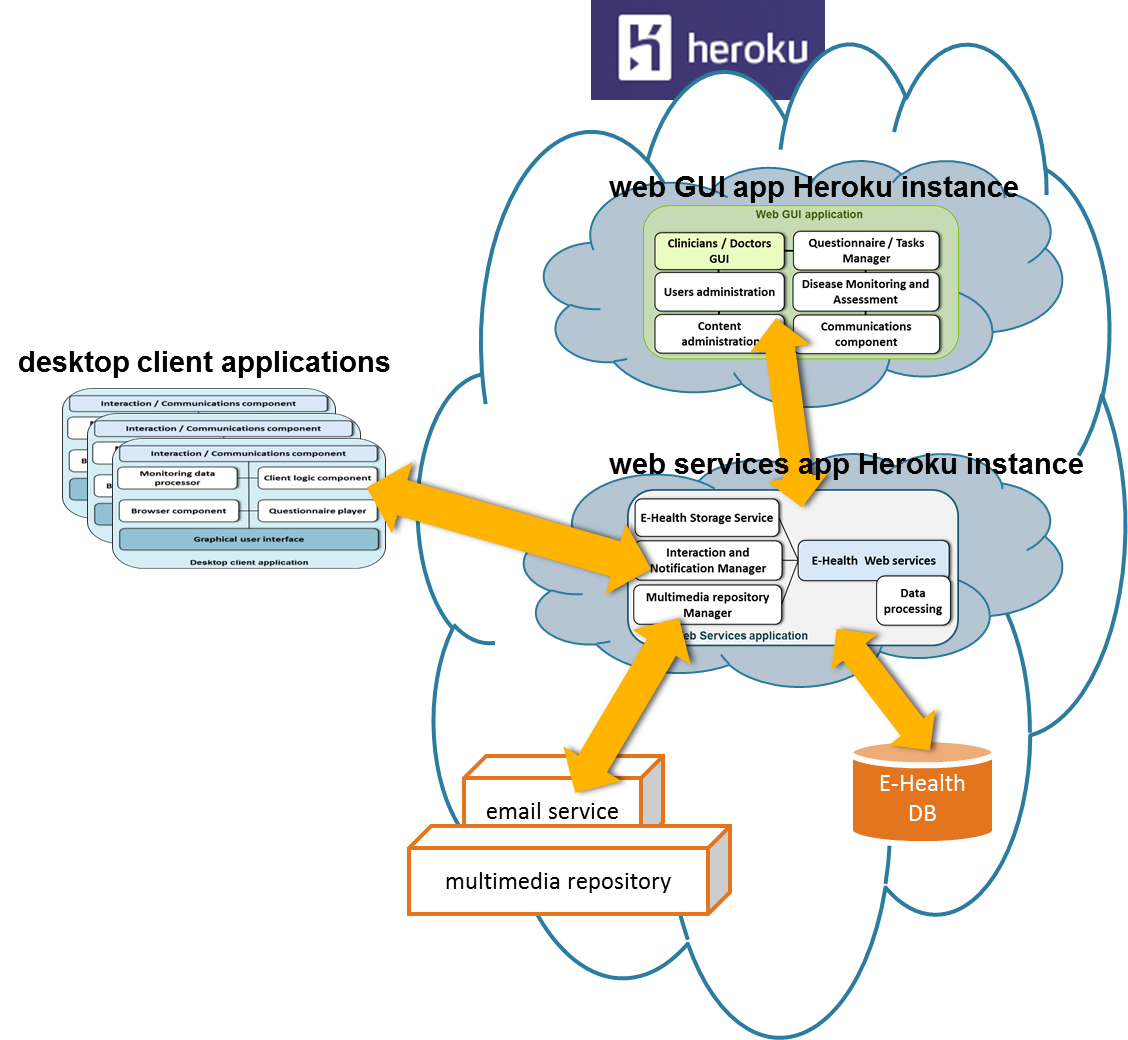


Figure A.8.2: Communications between the different components of the healthcare solution

From this communication picture we can identify the instances that are more likely to be scaled up or down. Or in some cases the components that will need to be migrated if the SLA’s or defined policies can’t be satisfied.

|  |  |  |
| --- | --- | --- |
| Healthcare application components | Instances | Comments |
| Web GUI app | 1 or more | Governed by MODAClouds runtime tools |
| Web Services app | 1 or more | Governed by MODAClouds runtime tools |
| E-Health DB | 1 | ? |
| Email Service | 1 | ? |
| Multimedia Repository | 1 | ? |

The third party services, including the database, will also need to be scaled. But for the moment it’s not clear how this will be done.

### Monitoring Rules and Adaptation Policies

In this first approach to the MODAClouds IDE a few monitoring rules and adaptation policies will be defined. They will be extended in the document D.8.4.2, where a final version of this prototype design will be presented. A possible migration of the components is not covered in this initial prototype. The following are the rules and policies we need to define with the MODAClouds IDE in order to ensure a correct governance of the different healthcare application components. For each server component a list of rules and adaptation policies will be presented (triggers and actions):

* web services application
  + scale it up when higher volumes of traffic are needed
  + scale it down when only few HTTP requests are needed
  + send warnings when some SLA or QoS is not satisfied
    - Define a response time value of the service
* web GUI application
  + scale it up or down depending on the traffic volume
  + Send warnings when some SLA or QoS is not satisfied
    - Define a response time value of the service

### QoS Constraints

* web services application
  + Short response time (average and percentile value): an average response time will be defined. Also 90% of the services should respond faster than a given time.
  + High availability (percentage value): the availability of this server component should be over 95%.
* web GUI application
  + Short response time
  + High availability

Other QoS could be added in the future.

## A.9 Case study application design with the MODAClouds IDE

In this section, the design of the ATOS case study application with the MODACloudML tool will be introduced. This is an initial approach that will try to show the design elements described in this appendix section but from another point of view, the models that will be generated by the MODAClouds IDE tools. As it was mentioned in the achievements chapter of this document, these tools will be used in these four areas:

* application components design
* application data design
* cloud provider selection (DSS MODAClouds tool)
* definition of the monitoring rules, QoS, adaptation policies etc.

The first of the MODAClouds design tools available to the developers is **MODACloudML**, based on MODELIO, an open source modelling tool. This tool covers some of the four areas mentioned before.

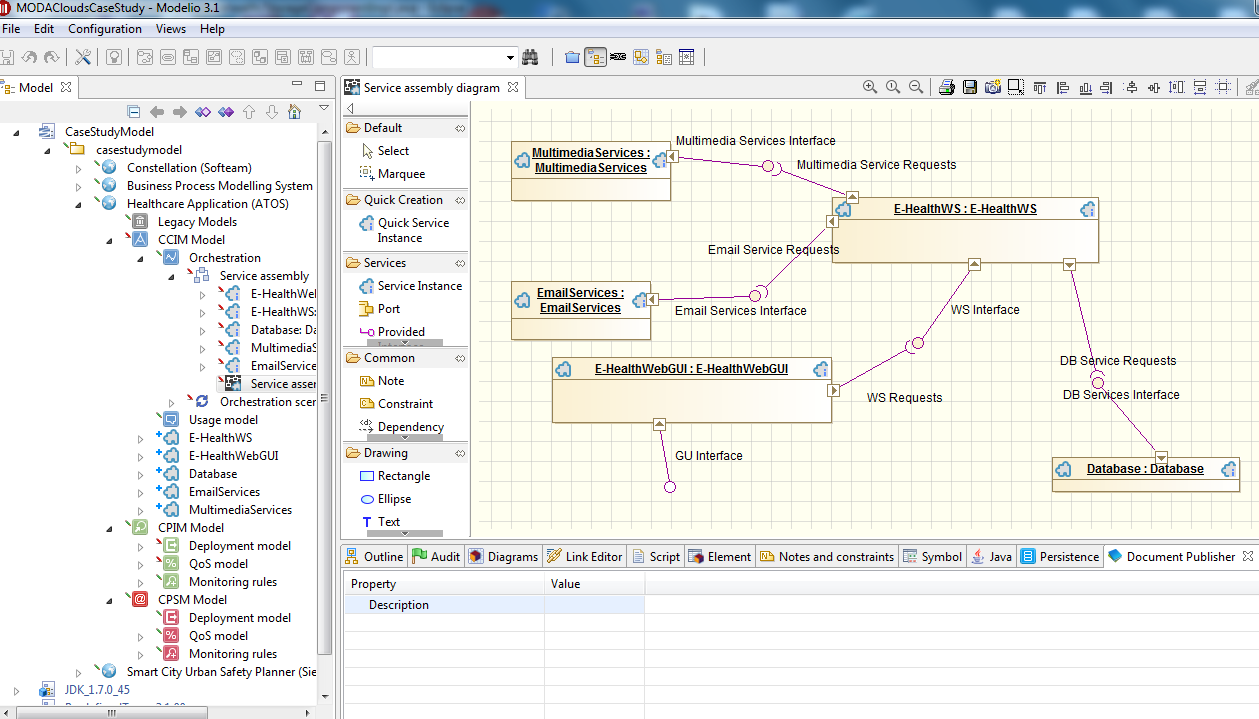


Figure A.9.1: MODACloudML (Modelio) – working with the CCIM models of the server components

With this tool we will define the functional models of the server-side block components and also the data used by the application (in the CCIM, CPIM and CPSM abstraction layers).

### Data Model and Application Components Design

First we create the CCIM models. Then, the CPIM models will be created taken the CCIM models as a basis. And finally the CPSM models will be created.

In the context of the MODAClouds framework (CCIM abstraction layer), the third party services used by the healthcare solution, the web services application, the database service and the web GUI application will each of them be defined as a service in the MODACloudML IDE tool (CCIM abstraction layer). The desktop client application will be omitted as it is a component that is independent from the cloud. It only accesses the web services application when needed.

These are the services defined with MODACloudML:

* Web GUI application
* Web services application
* Database service
* Email delivery service
* Multimedia repository service

For each one of these services, a set of provided and required interfaces will also be defined. These interfaces are how the relation between components is declared. Each of these interfaces will also have a set of operations.

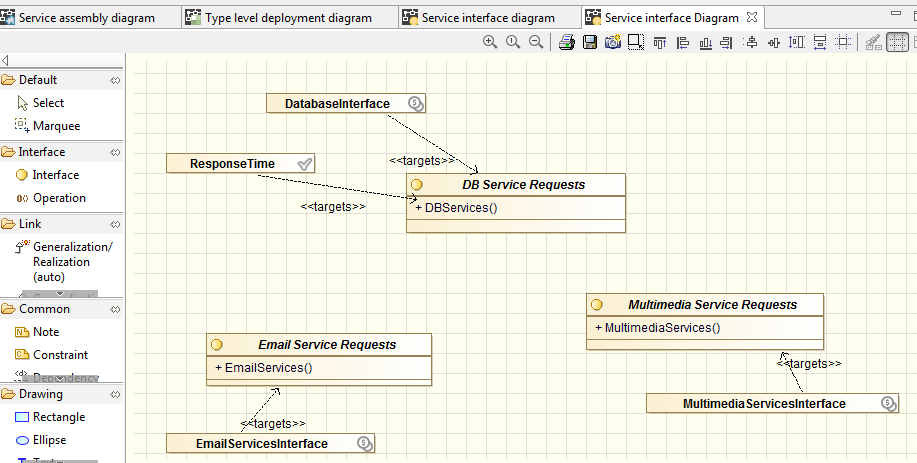


Figure A.9.2: MODACloudML – Web services application (service) required interfaces

The data model used by the healthcare application will also be designed with this tool. A high level diagram of the data model will be generated. Next image shows an example of the data high level design in the CCIM abstraction layer.

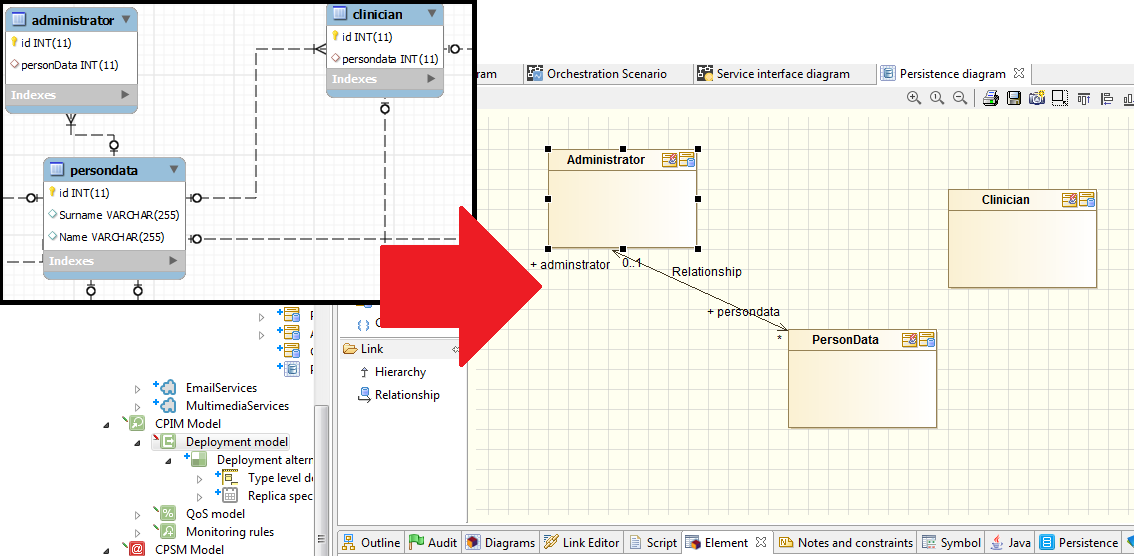


Figure A.9.3: MODACloudML – Data modelling

After defining the services and the data model, we define how these services cooperate using the orchestration scenario models. First we define the instances that exist at runtime and how they interact with each other. And then we define the different interaction scenarios at runtime. The next image shows the diagram that defines the instances at run-time, the service assembly diagram.

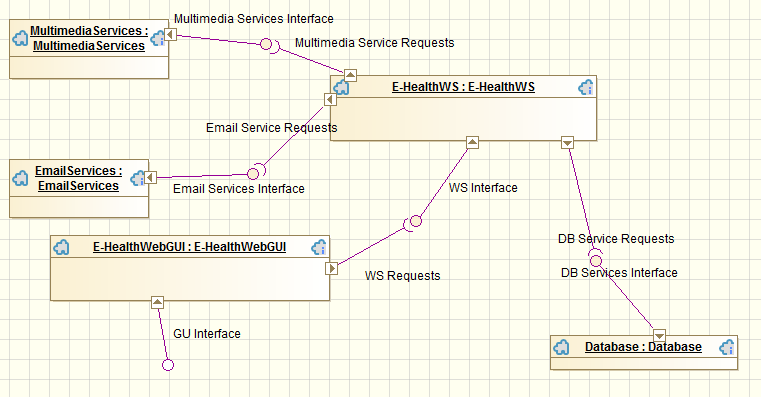


Figure A.9.4: MODACloudML - Service assembly diagram

The next image depicts the transition from the components design model, shown in the previous sections, to the correspondent MODAClouds CCIM layer services model.

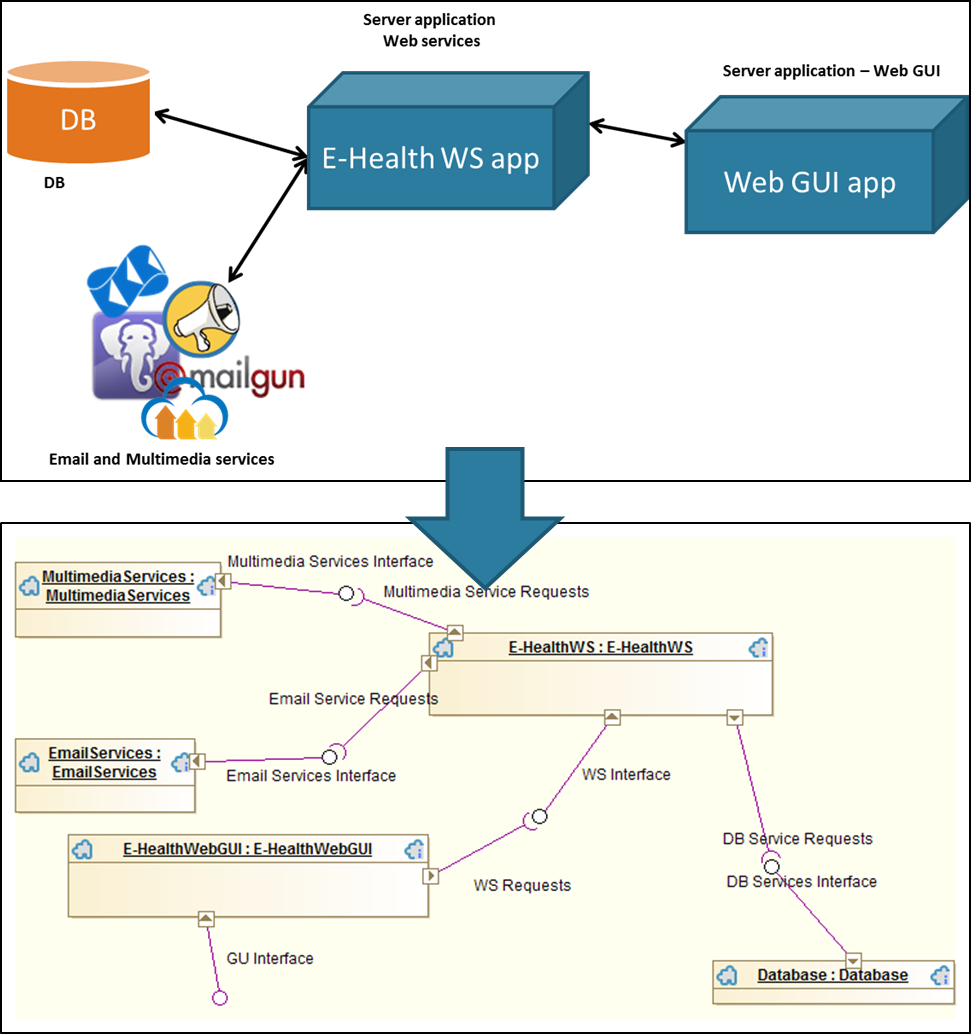


Figure A.9.4: MODACloudML – CCIM service assembly diagram of the server components

The business (functional requirement of each service) and the QoS requirements for each of these services will also be defined. For this initial prototype, only the ‘ResponseTime’ and the “Availability” QoS constraints will be defined for both the web services application and the web GUI application.

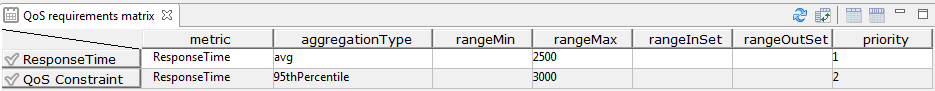


Figure A.9.5: MODACloudML – Defining the QoS of a service

After this set of CCIM models is created, we can continue with the next abstract layers models, the CPIM and the CPSM models.

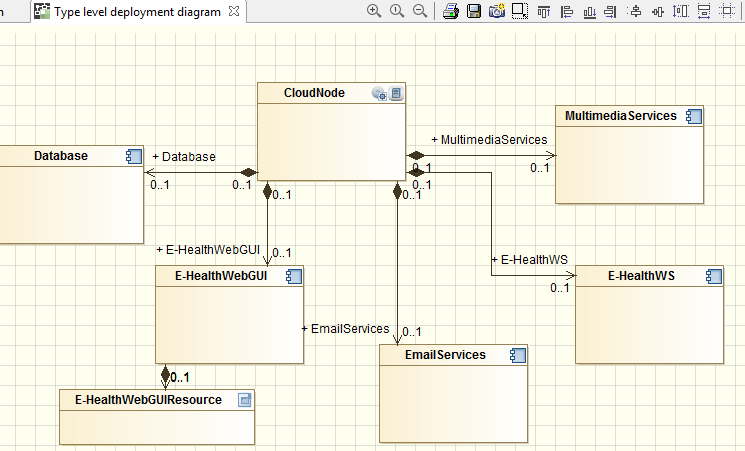


Figure A.9.6: MODACloudML – Type level deployment diagram

The first of them, the CPIM models, will continue with the design of the application in a cloud agnostic way, but will also be more specific than the CCIM layer models. Then, the CPSM models will be specific to the selected cloud provider.

Next deliverable will present a more detailed description of these steps.

With the MODACloudML tool we define a set of models that define the healthcare server-side application components in a cloud agnostic way.



B Definitions, acronyms and abbreviations

|  |  |
| --- | --- |
| **SOA** | Service-Oriented Architecture |
| **GUI** | Graphical User Interface |
| **DBMS** | Database Management System |
| **SLA** | Service Level Agreement |
| **DSS** | Decision Support System |
| **SSL / TLS** | Secure Sockets Layer / Transport Layer Security |
| **CCIM** | Cloud-enabled Computation Independent Model |
| **CPIM** | Cloud-Provider Independent Model |
| **CPSM** | Cloud-Provider Specific Model |