



Harald Schaffernak, BSc

# Utilization of near-eye display devices in hospital environments

## **MASTER'S THESIS**

to achieve the university degree of

Master of Science

Master's degree programme: Computer Science

submitted to

**Graz University of Technology**

Supervisor

Dipl.-Ing. Dr. techn. Wolfgang Vorraber

Department of Engineering- and Business Informatics

Univ.-Prof. Dipl.-Ing. Dr. techn. Siegfried Vössner

## **AFFIDAVIT**

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis dissertation.

---

Date

---

Signature

## ABSTRACT

Today's healthcare sector is highly specialized and resources are limited, thereby an effective and efficient use of the given resources is indispensable. This master's thesis investigates methods to improve and support medical processes using near-eye display devices, such as Google Glass. Previous work already revealed two promising use cases which are examined in detail during this thesis. These two use cases – the Virtual Consultation, which describes a remote video conference to share competences between hospitals – the Doctor's Visit, which covers the process where the medical staff is visiting patients to get an impression of their recovery, are analyzed and discussed. The former use case has been identified as most promising. Therefore a proof-of-concept application was implemented to perform field tests in hospital environments. This application scenario has the potential to improve and support the medical care especially in areas where users need their hands to interact with the environment and can't use it to operate with additional devices.

## ACKNOWLEDGMENTS

First I would like to thank Dipl.-Ing. Dr.techn. Wolfgang Vorraber and Univ.-Prof. Dipl.-Ing. Dr.techn. Siegfried Vössner for giving me the opportunity to write this master's thesis as well as supporting me during the process. Furthermore, I would like to thank the involved medical staff and the IT department of the two hospitals, where the tests were conducted. And last but not least, I would like to thank my family and friends, especially Andrea Wiesinger and Maximilian Sachs for providing me helpful material and advice for this thesis.

# Contents

---

- 1 Introduction.....1
  - 1.1 Utilization of Near-Eye Display Devices in Hospital Environments ..... 1
  - 1.2 Statement of the Problem..... 2
  - 1.3 Related Work..... 2
  - 1.4 Structure of the Thesis ..... 4
- 2 Requirements Engineering .....5
  - 2.1 Definitions ..... 7
    - 2.1.1 Stakeholder ..... 7
    - 2.1.2 Requirement..... 8
    - 2.1.3 Business Process..... 9
    - 2.1.4 Goal ..... 10
    - 2.1.5 System context ..... 11
  - 2.2 Requirements Elicitation ..... 11
    - 2.2.1 Identification ..... 12
    - 2.2.2 As-Is Analysis ..... 12
    - 2.2.3 6-3-5 Method ..... 13
    - 2.2.4 Field Research..... 14
    - 2.2.5 Interviews ..... 14
  - 2.3 Documentation..... 14
    - 2.3.1 SOPHIST Requirement Templates ..... 15
    - 2.3.2 Unified Modeling Language - Sequence Diagram ..... 16
    - 2.3.3 Business Process Model and Notation ..... 16
    - 2.3.4 Use Case ..... 17
  - 2.4 Validation ..... 18
    - 2.4.1 Quality Metrics ..... 19
    - 2.4.2 Analytical Model..... 20
    - 2.4.3 Prototypes ..... 20
  - 2.5 Management ..... 21
    - 2.5.1 Requirements Management with Object Ids ..... 22
- 3 Quality of Service.....23
  - 3.1 Terminology..... 24
    - 3.1.1 End-to-End QoS ..... 24
    - 3.1.2 Classes of Services ..... 24
    - 3.1.3 QoS Parameters..... 25

|       |  |    |
|-------|--|----|
| 3.1.4 | Classification of Applications.....            | 26 |
| 3.1.5 | QoS Mechanisms .....                           | 27 |
| 3.2   | Protocols.....                                 | 28 |
| 3.2.1 | Protocol: IP .....                             | 30 |
| 3.2.2 | Protocol: UDP .....                            | 31 |
| 3.2.3 | Protocol: TCP .....                            | 31 |
| 3.2.4 | Protocol: HTTP.....                            | 31 |
| 3.2.5 | Protocol: RTP .....                            | 31 |
| 3.2.6 | Protocol: RTCP .....                           | 32 |
| 3.2.7 | Protocol: RTSP .....                           | 32 |
| 4     | Hospital Information System.....               | 33 |
| 4.1   | A Brief History.....                           | 33 |
| 4.2   | HIS Software .....                             | 34 |
| 4.2.1 | HIS for SAP Business One .....                 | 34 |
| 4.2.2 | Medico.....                                    | 35 |
| 4.3   | Standards.....                                 | 37 |
| 4.3.1 | HL7.....                                       | 37 |
| 4.3.2 | DICOM .....                                    | 39 |
| 5     | Head-Mounted Display.....                      | 41 |
| 5.1   | A Brief History.....                           | 41 |
| 5.2   | Head-Mounted Display Properties and Types..... | 42 |
| 5.2.1 | Head and Helmet-Mounted Display.....           | 43 |
| 5.2.2 | Arm-Mounted Display .....                      | 44 |
| 5.2.3 | Optical Head-Mounted Display .....             | 44 |
| 5.2.4 | Head-Mounted Projective Display .....          | 45 |
| 5.2.5 | Virtual Retinal Display .....                  | 45 |
| 6     | Requirements Engineering: Application.....     | 46 |
| 6.1   | Project Goals .....                            | 47 |
| 6.2   | Scenario Categorization .....                  | 47 |
| 6.3   | Requirements Management .....                  | 48 |
| 6.4   | Subproject: Virtual Consultation .....         | 49 |
| 6.4.1 | As-Is Analysis and Identification.....         | 50 |
| 6.4.2 | Potential for Improvement.....                 | 53 |
| 6.4.3 | Scenario Categorization.....                   | 53 |
| 6.4.4 | Resulting Requirements .....                   | 55 |

|       |   |     |
|-------|---|-----|
| 6.4.5 | Resulting Business Process .....                                      | 59  |
| 6.4.6 | Virtual Consultations – Conclusion.....                               | 61  |
| 6.5   | Subproject: Doctor’s Visit.....                                       | 61  |
| 6.5.1 | As-Is Analysis and Identification.....                                | 61  |
| 6.5.2 | Potential for Improvement.....  | 65  |
| 6.5.3 | Doctor’s Visit – Conclusion .....                                     | 67  |
| 7     | Architecture and Design: Virtual Consultation Proof-of-Concept.....   | 69  |
| 7.1   | System, Context and Environment of the Proof-of-Concept .....         | 70  |
| 7.2   | Communication flow .....  | 72  |
| 7.3   | Related Work.....   | 73  |
| 7.4   | Requirements for the Proof-of-Concept .....                           | 74  |
| 7.5   | Choice of Technology .....  | 76  |
| 7.5.1 | Technology Choices for Google Glass.....                              | 76  |
| 7.5.2 | Technology Choices for Windows Application .....                      | 77  |
| 7.6   | Software Design and Architecture .....                                | 80  |
| 7.6.1 | Core Component Architectural Software Design: Sinks and Sources ..... | 82  |
| 7.6.2 | Model View ViewModel Architectural Software Pattern.....              | 85  |
| 7.6.3 | The Final Software Design .....                                       | 87  |
| 8     | Implementation: Virtual Consultation Proof-of-Concept.....            | 89  |
| 8.1   | Excursus: Video and Audio Compression .....                           | 89  |
| 8.2   | General Overview .....  | 90  |
| 8.2.1 | Frameworks used in the Proof-of-Concept .....                         | 91  |
| 8.3   | The Applications .....  | 93  |
| 8.3.1 | The Google Glass Application .....                                    | 93  |
| 8.3.2 | The Windows Control Application.....                                  | 94  |
| 8.4   | Capabilities .....  | 95  |
| 8.4.1 | Video Transmission .....  | 95  |
| 8.4.2 | Audio Transmission .....  | 99  |
| 8.4.3 | Picture Transmission .....  | 101 |
| 8.4.4 | HIS Related Functionality .....                                       | 103 |
| 8.5   | Technical Limitations of the Proof-of-Concept .....                   | 105 |
| 8.6   | Possible Extensions of the Proof-of-Concept .....                     | 106 |
| 9     | Results .....   | 108 |
| 9.1   | Feedback Methods .....  | 108 |
| 9.2   | Feedback Results .....  | 108 |

|      |                                 |     |
|------|---------------------------------|-----|
| 9.3  | Conclusion .....                | 108 |
| 10   | Conclusion and Future Work..... | 110 |
| 10.1 | Limitations .....               | 110 |
| 10.2 | Future Work .....               | 110 |
| 11   | References.....                 | 112 |
| 12   | List of Figures.....            | 118 |
| 13   | List of Tables .....            | 120 |



# 1 Introduction

---

This master's thesis discusses the utilization of near-eye display devices, such as the Google Glass, in hospital environments to improve and support the medical process. It is built upon the outcome of existing research of the Department of Engineering- and Business Informatics and as a consequence, it is a refinement and enhancement of the existing work of the institute. The key aspects which are analyzed in this thesis are the use cases called Virtual Consultation and the Doctor's Visit. The first describes the use case where a doctor, wearing the Google Glass, gets remote support from a medical specialist. The second use case analyzes the process of the daily visit of patients by their doctors and how this process can be optimized using a near-eye display device. The requirements engineering as well as field tests and the evaluation was done in collaboration with two hospitals using a proof-of-concept application. The close proximity to the everyday hospital routine is intended to ensure that the results are as close to reality as possible.

## 1.1 Utilization of Near-Eye Display Devices in Hospital Environments

This section will give a short overview of the chapters as well as a brief description of the relationships between them. The first chapters of this thesis will discuss the theoretical background of the practical part. This includes four topics. First, the theory of **Requirements Engineering (RE)**, which is applied for the practical part, to gather, manage, validate and document requirements. Second, the theory of **Quality of Service (QoS)** with the focus on streaming media, which is important for the implementation of the proof-of-concept application. Third, a description of **Hospital Information Systems (HIS)** and their interfaces such as **Health Level 7 (HL7)**, because for the practical part the communication with the HIS is necessary to utilize some requirements. Finally, an overview of **Head-Mounted Displays (HMD)** will give the background knowledge of devices such as **Google Glass (Glass)**.

Subsequently, the practical part of this master's thesis will apply the RE to collect information with respect to both use cases: Virtual Consultation and the Doctor's Visit. The requirements identified in this project phase provided the basis to implement a proof-of-concept application to

validate the benefits of a near-eye display device in hospital environments. The outcome is summarized in the last chapters of this thesis and will show that the utilization of these devices would have a beneficial influence on the medical process.

## **1.2 Statement of the Problem**

The focus of this thesis is the optimization and improvement of the medical processes using a near-eye display device such as Glass. To achieve this goal the above mentioned use cases, the Virtual Consultation and the Doctor's Visit, are introduced.

The Virtual Consultation use case results from the fact that the today's healthcare sector is highly specialized. This means that for each area there are some medical specialists, which are rare and thus expensive. As a result, hospitals cooperate with each other to share e.g. their qualified specialist personnel. This implies that the specialists aren't permanent locally available in each hospital. In order to compensate this, technical aids are needed, for instance, a virtual presence of the expert. The Virtual Consultation analyzes this use case with respect to near-eye display devices.

Complementary to this, the Doctor's Visit is a fairly optimized process and there is not much room for improvement. But in the course of the RE a similar problem as for the Virtual Consultation was uncovered. Where during a visit an additional opinion from a medical expert is needed which is not available every day. In this use case the Virtual Consultation may also improve the medical process to enhance the patient care.

## **1.3 Related Work**

This thesis is embedded in a research project from Vorraber and Vössner et al. Which explores the use of near-eye display devices for medical applications, with a focus on improving the efficiency and effectiveness of clinical care. The paper "Medical applications of near-eye display devices: An exploratory study" (Vorraber, et al., 2014, p. 1266 ff.) already describes the use of Glass during surgeries to display vital parameters from the patient directly in the field of vision of the surgeon. Also the article lists further use cases such as the Virtual Consultation which will be

discussed in this thesis in detail (Vorraber, et al., 2014, p. 1267). The research project includes another master's thesis, which discussed similar topics and also implemented a proof-of-concept application that displayed vital parameters of a patient in the view of the surgeon (Sachs, 2014, p. 1 ff.).

Li & Alem (2013) discuss a similar topic as this thesis: The remote support in the medical area. The focus of the paper is slightly different than in this thesis. It mainly highlights the aspect of health workers, who provide care in patient's homes and are supported by remote experts. (Li & Alem, 2013, p. 493 ff.)

Another article mentioned the advantages by using Glass during surgeries, e.g. by displaying medical images to the surgeon, or allowing video and audio streams out of the operation room to communicate with other doctors (Whiteman, 2014, p. 1 ff.).

There are further articles which realized the benefits of using near-eye display devices such as Glass for hospital environments. For instance, the use case of medical education, which has to balance between patient's safety and the possibility for trainees to develop own medical skills. Vallurupalli et al. (2013) discussed this topic in more detail. (Vallurupalli, et al., 2013, p. 267 ff.)

Fussell et al. (2003) compares five media conditions, regarding their value for remote collaboration on physical tasks, with each other: side-by-side, audio-only, head-mounted camera, scene oriented camera and scene oriented plus head-mounted cameras. The study was performed with pairs of participants, with one participant acting as "worker" and the other as "helper" who provided guidance. The results of the study showed that the task completion times were best in the side-by-side scenario followed by scene oriented camera. However, head-mounted camera, scene oriented plus head-mounted cameras and audio-only did not differ significantly from each other. Furthermore, the work quality rated by the participants showed similar results. (Fussell, et al., 2003, p. 513 ff.)

This is in contrast to the results of (Birnholtz, et al., 2010, p. 261 ff.) where a system providing detailed plus overview information was superior to other constellations. However, this study hasn't used a head-mounted camera for the experiments.

The issue who controls the point of view in a remote assistance scenario and how this affects the user's performance and behavior is investigated in (Lanir, et al., 2013, p. 2243 ff.). Using task completion times and the number of errors the study showed that for a novice worker a helper-driven point of view is preferable. But the paper admitted that more research needs to be done if the worker has a deeper task knowledge.

## **1.4 Structure of the Thesis**

This thesis is structured in a theoretical and a practical part. The theoretical part discusses four topics which will be needed for practical sections, this includes the process of the RE (see chapter 2), QoS (see chapter 3), an introduction of HMDs (see chapter 5) and an overview of HISs (see chapter 4). The practical part contains the actual requirements engineering of a real world project to optimize the medical processes (see chapter 6). Based upon the RE a proof-of-concept application was designed (see chapter 7) and implemented (see chapter 8) to validate the requirements as well as showing advantages and disadvantages of the use cases. During the hospital daily routine several experiments were made with the proof-of-concept, which will be discussed in chapter 9. The last chapter of this thesis will conclude the whole work and will also show some limitations as well as give an outlook on possible future work (see chapter 10).

## 2 Requirements Engineering

---

At the beginning of each software project the involved persons have to deal with massive not well-organized information, so the first task for every project manager is to organize the information. To pick out the right requirements in a desired form, from the whole information pool in a project, is a very difficult task, accordingly it is a critical phase in each project. A lot of project managers are vastly underestimating the complexity of RE and therefore about 70% of software projects fail, because of poor requirements (Narciso & Verner, 2009, p. 131). Furthermore, the cost of fixing an error which is introduced in the requirements phase is increasing during the project and rises dramatically when the software development process reaches its end; this can be seen in Figure 1 (Boehm, 1976, p. 1227) (McConnell, 2004, p. 30).

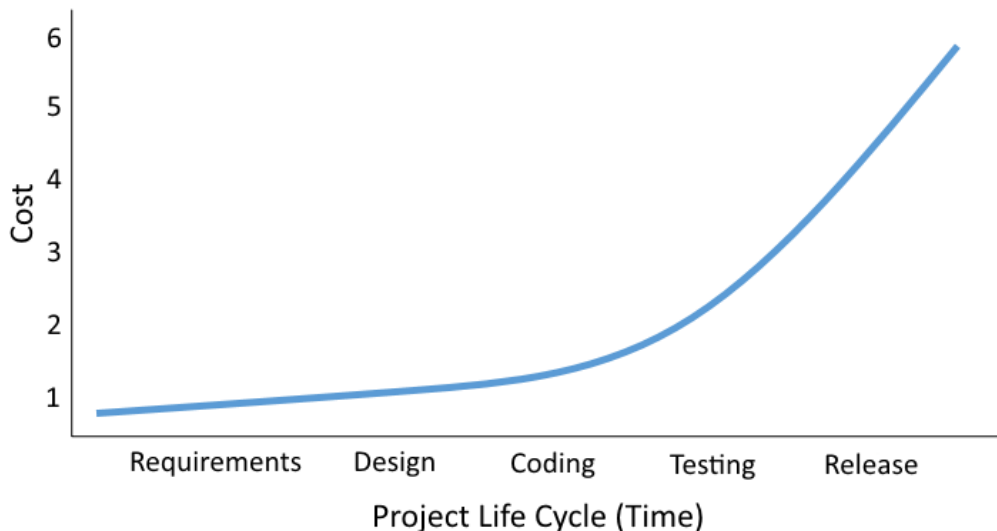


Figure 1: Cost-to-fix-an-error curve (modified from (Boehm, 1976, p. 1228))

Irrespective of the fact that RE is just a part of the software development process, it plays a major role for the success of a project. Several software development models were invented in the last decades, like the widely-known waterfall model, the V-Model or the agile development etc. In each of these models RE plays a different roll and hence might be used in a different way. This chapter will give a general overview of the topic RE and does not intent to be a guide for a special software development model.

There are numerous definitions of RE:

"Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families." (Pamela & AT&T Labs, 1997, p. 315)

Another helpful definition can be found in *Requirements Engineering and Management*:

"The discipline demarcates a systematic approach to software development beginning with the first hazy goals all the way through to a comprehensive set of requirements. [...]" (Rupp, 2009, p. 15)

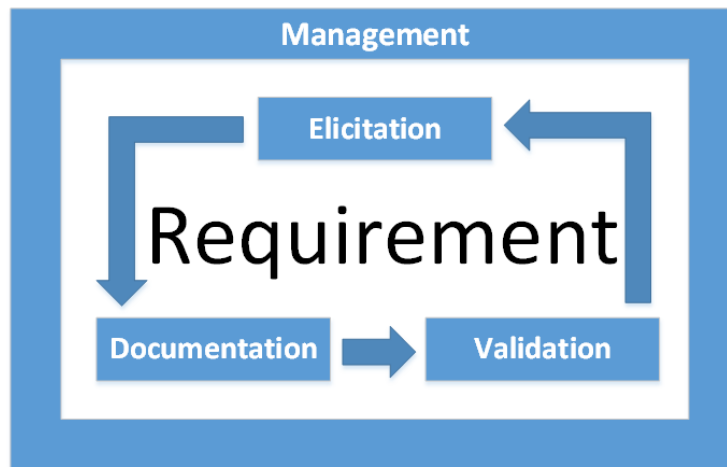


Figure 2: The iterative RE process (based on (Rupp, 2009, p. 1ff))

In simple terms, RE focuses on four different tasks: Elicitation of requirements, documenting and validation of those, as well as managing in all these areas (Rupp, 2009, p. 14). RE itself is an iterative process, hence each of those phases can be performed multiple times (see Figure 2). It is apparent that it would be very difficult to gather all required information at once, so the easiest solution is to iterate multiple times through the RE process from Figure 2 with a rising level of detail. Rupp et al. (2009) describes this with the term *specification levels* and also mentions that a simple “general-to-specific – approach” isn’t practical but difficult to put into practice. The practical procedure is more a *middle-out* process, where, depending on the situation, a refinement or an abstraction takes place (Rupp, 2009, p. 44).

The term requirements elicitation means the collecting of requirements for a system from stakeholders or other information sources. This can be done by interviews, questionnaires,

observations or through analyzing of documentation. Furthermore, documentation of requirements is an essential part of RE and can be done in different ways e.g. in prose or by use case diagrams. The documented requirements must be analyzed with respect to correctness and completeness, this can be done by reviews, test cases or consolidations. Another essential part is the management, which takes place in all steps of the RE process, key terms are for example versions or releases.

In the next sections these areas of RE, as well as definitions will be discussed in more detail, to attain a better understanding of the entire RE process.

## **2.1 Definitions**

This section will provide definitions which are frequently used in RE.

### **2.1.1 Stakeholder**

A person, a group, or an organization that has directly or indirectly influence on the requirements is called a *stakeholder* (Rupp, 2009, p. 62). Such entities have an interest in the course and result of the project.

Project manager Babou Srinivasan (Srinivasan, 2008) proposes a classification in power and interest of the stakeholder:

1. Stakeholders with a high degree of power and interest, are very important for the project and therefore the requirements engineer should work very closely and intensively with these entities.
2. Entities with high level of power, but little interest should be satisfied. The requirements engineer should keep a close contact, but shouldn't annoy these stakeholders with too many technical details.
3. Stakeholders with a high motivation but little influence are important for the success of the project. These entities can warn of occurring risks at an early stage of the project and thus it is important to highly involve these entities in the process.

4. The last group are stakeholders, who have little interest and power in the project, but should also be informed on a regular basis.

In 1983 Edward Freeman and David Reed proposed, in one of the first articles that notice the importance of this topic, two different definitions:

The first describes the stakeholder in the wider sense, so every entity that is effected or affects the achievement of an organization's objectives, is called a stakeholder. The second definition describes the stakeholder in the narrow sense, hence a stakeholder is an entity on which the organization is dependent for its survival. It can be seen that the main focus of this paper is the concept of stakeholders in an organization and not in the meaning of RE for a software project. But this can easily be transformed and extrapolated on such projects. (Freeman & Reed, 1983, p. 91)

### **2.1.2 Requirement**

A requirement in the term of RE is a functional need that a system or a process, for example a software application, must fulfill to achieve the satisfaction of a customer or client. Generally, requirements are the input of the product development and therefore a critical factor for the outcome. Requirements can be classified into two main types, functional and non-functional requirements. There are other approaches of a classification of requirements, e.g. the International Requirements Engineering Board use a subdivision in functional requirements, qualities and constraints (Pohl & Rupp, 2009, p. 17). There are further definitions and synonyms, like behavioral and non-behavioral requirements etc. but all of these definitions can be categorized in functional and non-functional requirements. In any event, the definition for the subdivision into the two types of requirements, there are major discrepancies between different sources and authors. In the paper *On Non-Functional Requirements* (Glinz, 2007, p. 21 ff.) this problem is discussed in detail. However, for this thesis the definition is based on the book *Requirements-Engineering und Management* (Rupp, 2009, p. 1 ff.) which is very similar to (Partsch, 2010, p. 1 ff.).



### **2.1.2.1 Functional Requirement**

A functional requirement defines a behavior or a function what a system or process should be able to accomplish. This type of requirement is the leading part and what a client or customer directly demands from the system or process. (Partsch , 2010, p. 27)

It is important to know that a system that satisfies all functional requirements may still be useless for the client, because it violates some not explicitly formulated quality constraints. Or in other words, some non-functional requirements. (Rupp, 2009, p. 248)

### **2.1.2.2 Non-Functional Requirement**

All requirements where the term functional requirements doesn't fit, are called non-functional requirements (Rupp, 2009, p. 18). This category of requirements is essential for the acceptance of the system or process. Non-functional requirements are often supplements or refinements of functional requirements, e.g.: A new data item shall be saved in less than 2 seconds. Additionally, functional requirements occasionally arise from non-functional ones. Rupp et al. (2009) proposed a further breakdown of non-functional requirements:

- Technological requirements limit the solution space by environmental restrictions or by given specifications.
- Quality requirements determine the quality of the system.
- Requirements to the handling and design of the user interface.
- Requirements to all parts of the product to be delivered, except the main system.
- Requirements to downstream activities like support.
- Requirements to legal frameworks.

### **2.1.3 Business Process**

A brief definition of the term business process can be found from Lewis & Slack (2003):

*“[...] a set of logically related tasks performed to achieve a defined business outcome. “ (Lewis & Slack, 2003, p. 100)*

A business process can include further business processes or start other business processes. Additionally, a business process is not restricted by department boundaries or even by the company. To get a deep understanding of a business process it is important to look at a low level of the process. So for example a used IT system is a black box and can hide essential parts of the business process and thus it is important to get a closer look at such systems. (Rupp, 2009, pp. 189, 190)

#### **2.1.4 Goal**

The SOPHIST-approach (Rupp, 2009, p. 70) defines goals as overall objectives, what the project should achieve and so they describe the direction a project should take. While this is the case, the formulation of goals should happen early in the project, for instance during the as-is analysis. Furthermore, each goal should be assigned and represented by a stakeholder. According to Rupp et al. (2009), there are several demands to the formulation of the goals:

- The goals shall describe the project objectives completely and correctly.
- Goals should be understandable for each stakeholder.
- The goals shall not describe or imply a concrete solution.
- Constraints to the solution space should be listed and well known.
- The goals should be realistic with respect to the available resources.
- Etc.

To document the goals, there are multiple options, depending on the project itself, or some constraints from the environment. The possibilities are for example: text in prose, And-Or tree's and more.

### 2.1.5 System context

Rupp et al. (Rupp, 2009, pp. 72-76) propounds the subdivision of the whole environment, the project is located in, into three main parts.

- The term *system* describes the main part of a project, and thus it is the entity what should be developed or improved. Hence, the *system* describes the scope where the project takes place.
- The *context* is the surrounding of the system and thus it defines the interfaces and how the system integrates in the whole environment. Therefore it is an important source to gather system requirements.
- What remains is the irrelevant part of the environment. This part has no influence to the project.

To achieve the classification it is important to delimit each area from each other, with as small as possible grey area between them. Nevertheless, at the beginning of the RE, grey areas are unavoidable, but in the course of the RE process they should get smaller. (Rupp, 2009, p. 75)

## 2.2 Requirements Elicitation

Initially, information sources must be found in form of people / stakeholders, documentation or existing systems, which are affected from the new system or process. This *identification* phase is crucial for all following steps. The identified information sources are initially used to find the goals and the scope of the new project. Without them, the project could run

in the wrong direction for days or even months and therefore causes unnecessary costs, or the project could fail before it really has begun (Rupp, 2009, p. 61). A simple method to find goals, is to perform an *as-is analysis*. The analysis will expose problems and thus potential for optimization and through that it is possible to detect the goals. Therefore, the following sections explain the identification phase as well as the as-is analysis in more detail. There are further requirement elicitation methods which can be applied in different situations. For instance, creativity

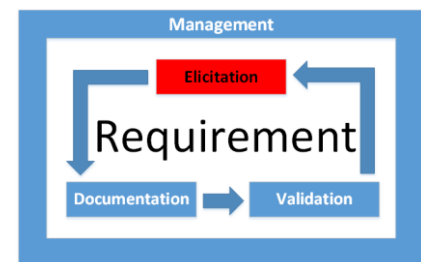


Figure 3: RE process: elicitation

techniques are used to generate new ideas, such as Brainstorming or 6-3-5 Method (see section 2.2.3). Observation techniques are used to assess the current situation, e.g. field research. Survey techniques can be used to gather information from stakeholder directly, popular methods are interviews or questionnaires.

### **2.2.1 Identification**

To get started, the first step is to find information sources. These can be found in documents, existing systems or from stakeholders (Pohl & Rupp, 2009, p. 27). As it is impossible to get all information sources at the beginning, in the course of the project further information sources will be identified.

#### **Documents**

Not only documentations of a current system are useful, also norms, legislations and bug reports are practical information sources. Paper is patient and therefore optimal to get a first overview.

#### **Existing systems**

Information can be found in current system, as well as in competing systems.

#### **Stakeholders**

Stakeholders are the most important information source and for the purpose of clarity it is recommended to record the stakeholders on a list, with name, contact, leading position etc.

### **2.2.2 As-Is Analysis**

The as-is analysis is an important step in requirements elicitation and thus often the basis for a successful project. To gather general ideas and a better understanding of the starting position, the as-is analysis can be applied for inventing new systems as well as replacing old systems. In most cases a system should support users, or is built to take over tasks from users. To see and gain a better understanding what the customer or client wants from the new system, it is very useful to understand the business process behind. Moreover, an analysis of associated business processes may also be helpful to get the whole context of the system and the relations between different processes. Hence, this reduces the possibility to overlook relevant requirements and

fosters the understanding of the system. There is also a very positive side effect obtained from the analysis, the closer look at the business process often uncovers potential for optimization and thus a better user acceptance. (Rupp, 2009, p. 63)

According to (Rupp, 2009, p. 189), analyzing the business processes is especially helpful if any of these terms apply:

- The current business process isn't supported by an IT- system beside standard software such as Word, Excel etc.
- The current business process is supported by an IT- system and the precise course of the business process has been forgotten by the users.
- If there is a reason to assume that there is potential for optimization.

To accomplish the as-is analysis, a close cooperation with stakeholders of the affected organizations, is necessary. Furthermore, this allows the users to add suggestions for improvements in the workflow or system. A method to extract the business process from the affected parties, is to design use cases and subsequently use e.g. **Event-driven Process Chain's (EPC)**, **Unified Modeling Language (UML)** - activity diagrams or **Business Process Model and Notation (BPMN)** to determine the exact business process.

To summarize, the as-is analysis captures the unbiased initial situation and will serve as the basis for further steps. Nevertheless, a too intensive as-is analysis can create the risk, to build a system that focuses too much on the current situation and overlooks better solutions.

### **2.2.3 6-3-5 Method**

The 6-3-5 method is a creativity technique to gather new ideas. In the traditional interpretation each of 6 participants writes 3 ideas down on a piece of paper. Each sheet is passed 5 times to a new person, this person has to read the ideas on the paper, complete it or develop new ones. Each turn takes 3 to 5 minutes and at the end each of the 6 participants has seen and extended all eighteen original ideas. (Rohrbach, 1969, p. 73 ff.)

## 2.2.4 Field Research

Field research belongs to the category of observation techniques. It is particularly useful if the stakeholder only implicitly knows the workflow, or the stakeholder's amount of time is very limited. With this technique the requirements engineer studies the stakeholder's day-to-day work, while the stakeholder is doing it. Meanwhile, the requirements engineer can ask questions to get a comprehensive understanding of the process. Nevertheless, this technique does have some disadvantages. For instance, it doesn't work for procedures which cannot be observed easily due to access or visibility restrictions. A second disadvantage is that the stakeholder is feeling uncomfortable, because of the presence of the requirements engineer and thus it distorts results. (Rupp, 2009, pp. 93,94)

## 2.2.5 Interviews

A simple and widely used technique to gather information is the interview. In this method a requirements engineer asks stakeholders predetermined questions and documents the answers for a later evaluation. The questions itself shall be formulated without any potential solution. Also, a good idea is to research the background of the stakeholder which should be interviewed, and ask only questions of his or her field and not things which the person cannot know (Leffingwell & Widrig, 2000, pp. 93-95). A disadvantage of this technique is the time consumption if many stakeholders are involved. Furthermore, the quality of the interview outcome often depends on the experience of the interviewer. An approach to improve the outcome of the interview is discussed in section 2.3.1. (Rupp, 2009, p. 97)

## 2.3 Documentation

A key aspect in RE is the documentation of requirements. This can be done in numerous ways. This section will present common methods to document requirements in a formal way, suitable for the different project stages.

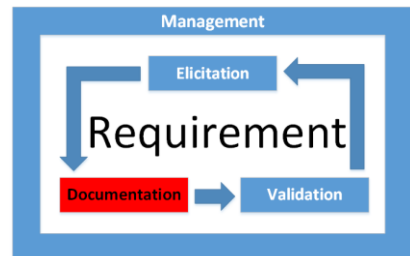


Figure 4: RE process: documentation

The documentation of requirements starts during the requirement elicitation phase. For instance, while an interview takes place, requirements engineers have to write down the important information in a suitable form. The book *Requirements-Engineering und Management* (Rupp, 2009, p. 162) proposes a formal method in natural language, to document the requirements in a desired form while, for example, an interview is in progress (see section 2.3.1). Also, the documentation of the requirements is an iterative process. During the RE the requirements will be refined or replaced, or even removed (Rupp, 2009, p. 345). It is apparent that natural language isn't the only method to document requirements, there are several other options such as: UML – use case diagrams, UML – sequence diagrams or state machines etc. Some of these methods are providing more detailed information than other, and thus some of these methods are refinements of others. (Rupp, 2009, pp. 184-245)

### **2.3.1 SOPHIST Requirement Templates**

Rupp et al. proposed in (Rupp, 2009, pp. 160-181) a formal way to document requirements in prose. Figure 5 shows the structure of a sentence which should be used to formulate a requirement. Usually people try to get linguistic diversity in their texts; this approach tries the opposite and therefore it limits the room for interpretation as well as the possibility of misunderstandings. Furthermore, words such as “shall”, “should” and “will” have a legal significance, for instance if “shall” is chosen in a requirement the supplier is legally obliged to implement this part. The word “should” describes only desirable features; however, the supplier is not legally obliged to implement those. The term <system name> has to be replaced with the system name, for example: “the Streaming-Tool” or the generic term “the system”. The term <process> has to be replaced with a so called process word, for example: “to display”, “to remove”, “to save”. Finally, the term <object> completes the process word, for example: “to save the customer”, “to display the message”. (Rupp, 2009, pp. 160-181)

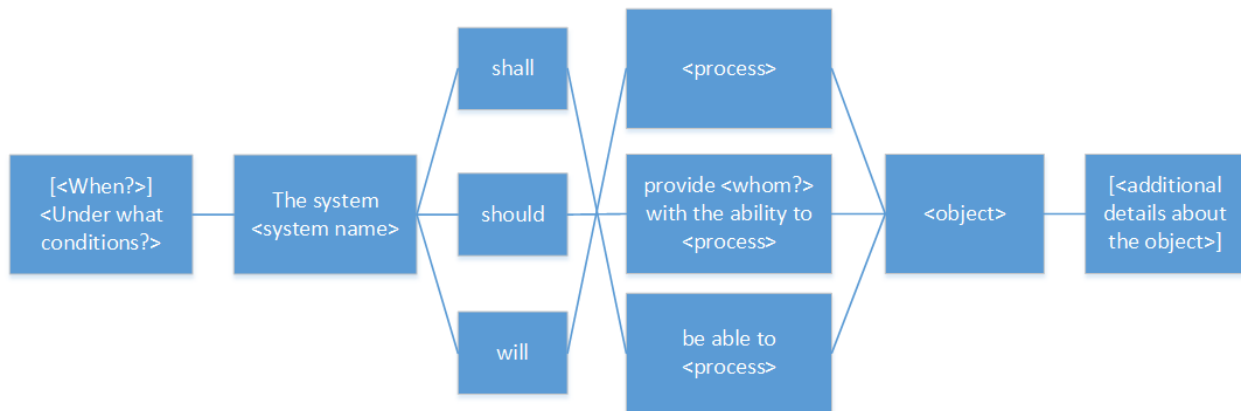


Figure 5: Requirements template from (Rupp, 2009, p. 177)





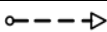

### 2.3.2 Unified Modeling Language - Sequence Diagram

The UML sequence diagram is an interaction diagram. It shows the chronologically sorted communication between entities. The major strength of this type of diagram is to visualize the communication between different systems or components, sorted chronologically. Additionally, synchronous as well as asynchronous communication can be described by the sequence diagram. It is also often used to refine a UML – use case diagram. For the RE process the sequence diagram can be used in nearly all project stages with, of course, different level of detail. A problem of this type of diagram is that it gets fast very large and thus confusingly. Hence a separation in sub diagrams is unavoidable, though this doesn't solve the problem entirely. (Rupp, 2009, pp. 220-222) (Partsch , 2010, pp. 285-288)

### 2.3.3 Business Process Model and Notation

The central theme of BPMN is to provide a graphical notation to specify the business process, further procedures and the messages that flow between them. The resulting diagrams are called **Business Process Diagrams (BPD)** and have some similarity to UML activity diagrams. The basic elements of BPMN are represented and described in Table 1.



|               |   |   |
|---------------|---|---|
| Event         |  | An Event occurs during the course of a process, for instance, the arrival of a message.   |
| Activity      |  | An Activity describes a generic term of work that a business process achieves or shall achieve.   |
| Gateway       |  | A Gateway is a decision point of Sequence Flows in the business process. So a Gateway joins, merges or forks paths in the business process. |
| Sequence Flow |  | The Sequence Flow notation describes the order in which the activities will be performed.   |
| Message Flow  |  | The Message Flow notation describes the exchange of messages between two parties.   |
| Association   |  | The Association notation (directed or undirected) links information such as text with graphical elements.                                   |

*Table 1: Basic elements of BPMN (Object Management Group®, 2011, p. 29).*

There are further notations as well as extensions and supplements of these basic elements, which can be found online at: [www.bpmn.org](http://www.bpmn.org). (Object Management Group®, 2011, p. 29)

### **2.3.4 Use Case**

A use case describes the behavior of a system or process under certain conditions. There are various forms to document use cases, for instance UML diagrams or as a simple prose text. Additionally, a use case can be formulated with different levels of detail. For example, Rupp et al. (2009) proposed a technique called Use Case Specification which is a semi-formal notation to document use cases briefly in form of a table. It uses the natural language as basis and consequently it is easy to read and understand without a special knowledge. Nevertheless, the Use Case Specification provides some mandatory fields which are frequently forgotten or neglected. This table can be extended with extra fields; for instance, for non-functional requirements which often get overlooked. Furthermore, a use case should avoid technical solutions in its formulation, because such solutions changes over time. For example, the term “user and password” is a technical solution which may be replaced in a view years with better technology. Hence the Use Case Specification provides a solid baseline to document use cases

briefly which is especially useful at an early stage of a project. In contrast, the Use Case Specification gets long and confusing if all special cases are listed. Secondly, it is designed to get a general overview and thus it is not sufficient as the only documentation technique in a project. To get a better understanding an example is shown in Table 2. (Partsch , 2010, pp. 282-285) (Rupp, 2009, pp. 214-217) (Cockburn, 2008, pp. 15-17, 55, 56)

|                          |  |
|--------------------------|--|
| <b>Name</b>              | Save document  |
| <b>Short Description</b> | The use case describes which actions are necessary to save a document to the hard drive.   |
| <b>Actors</b>            | User   |
| <b>Preconditions</b>     | The document must be changed before the next save action. Or if it is a new document it must not be empty.   |
| <b>Trigger</b>           | The user has pressed the “Save – Document” button.   |
| <b>Typical Process</b>   | Verify if the default directory exists.<br>Verify if the user has write permissions to the default directory.<br>Verify if the disk has sufficient free space.<br>Write the file to the disk.<br>Show the user a message box that the file was successfully saved. |

Table 2: Example of a Use Case Specification using the scheme proposed by Rupp et al. (2009).

## 2.4 Validation

Methods such as an as-is analysis (see section 2.2.2) at the beginning of the project and the definition of the system scope, or applying the SOPHIST requirement templates (see section 2.3.1) during interviews, are preventive measures to ensure high quality level of requirements. But the fact that different people with different skills from different fields work together,

leads in most cases to misunderstandings or other problems, which will reduce the quality of the requirements. To illustrate, during an interview it is hard to recognize that two involved stakeholders want contradictory requirements. For that reason the validation phase is conceived

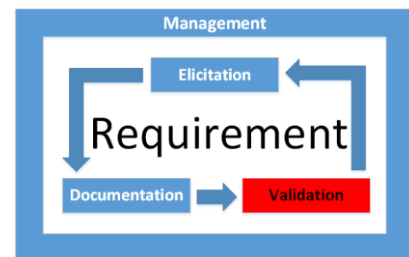


Figure 6: RE process: validation

to enhance the quality of the requirements, before the architecture or design phase starts. (Rupp, 2009, p. 290)

In EN ISO 9000:2005 the definition of quality is:

*"Quality; degree to which a set of inherent characteristics fulfils the requirements"* (STANDARDIZATION, 2005, p. 18)

A similar definition can be found in the IEEE Standard Glossary of Software Engineering Terminology:

*"The degree to which a system, component, or process meets specified requirements."* (IEEE Standards Board, 1990, p. 60)

In practice this means, how well a requirement fits the objectively measurable quality criteria. Section 2.4.1 will discuss the term quality and what a quality metric is in more detail.

### **2.4.1 Quality Metrics**

To compare the quality of multiple entities a quantification is required, i.e. the quality has to be expressed as a numeric value. This principle is the foundation for the validation of requirements. If it is not possible to measure the quality of the current situation, it is not possible to measure an improvement or deterioration. A definition for the term Quality Metric can be found from the IEEE Standards Board:

*"[...] A function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality"* (IEEE Standards Board, 1990, p. 60)

A quality metric takes as input requirements or requirement documents and outputs a numeric value, representing the degree to which these requirements satisfy a given attribute that affects its quality (Rupp, 2009, p. 314).

### **2.4.2 Analytical Model**

Rupp et al. (2009) defines the analytical model as a method which uses different diagrams, such as UML – sequence diagrams, state machines or UML – class diagrams etc. to represent the requirements in a different way. The basic principle here is to uncover missing or incorrect requirements by a redundant representation. To illustrate, prose requirements such as shown in section 2.3.1, usually need many pages and thus they get confusing and connections between them get lost. However, for example a UML – sequence diagram represent the information in a different way, and thus the relation between different requirements may become clear. Subsequently, if a mistake gets detected, the involved requirements need to be corrected. Hence there is a bidirectional connection between the analytical model and the raw requirements. That aside, the analytical model does have some disadvantages. For example, to create such diagrams there is additional knowhow necessary. Also there is a high additional expense for creating these diagrams. (Rupp, 2009, pp. 304-306)

### **2.4.3 Prototypes**

A prototype already implements a part of the requirements and verifies the feasibility and / or usefulness of functional and / or non-functional requirements. Moreover, in an early stage of the project a prototype allows to find missing or unnecessary requirements, before the actual project is realized. In addition, a prototype allows the user to get a better understanding and feeling of the final product. Furthermore, it serves as a discussion basis between the involved parties. There are different types of prototypes which can be used to gather information. For example paper prototypes are used to simulate the user interface. Despite these, prototypes are very work-intensive and thus costly, also they may be misinterpreted by customers. For instance, a prototype which simulates the functionality without doing anything in the background may lead to a wrong impression of the status of completion of the product. (Rupp, 2009, p. 298)

## 2.5 Management

**Requirement management (RM)** describes methods or activities for organizing and supporting the RE process. According to Rupp et al. (2009), there are two main assumptions in RE:

- Requirements change in the course of the project. For example, there are always stakeholders which want to change, add or delete requirements.
- Requirements will continue to be used from other people, so RE, of course, is not the end of the project management process.

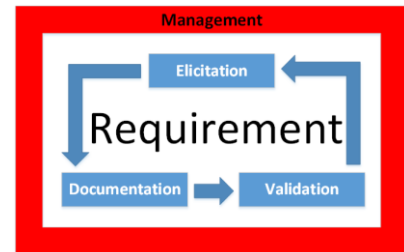


Figure 7: RE process: management

So RM should deal with these problems and help requirements engineers to avoid such common traps and manage the whole RE process satisfactorily. Commonly RM does not only manage requirements, but also definitions, diagrams, tests and much more things, which are in a direct or indirect relation to the RE process.

The RE- concept is a document which includes all relevant tools, methods, processes and procedures which are relevant for the RE process. This document should be created in an early stage of the project, but in any case before the requirements documentation starts.

According to (Rupp, 2009, pp. 349-351) there are four tasks the RE- concept document shall manage:

**Information exchange** between the involved people is necessary, because each party has to know for each requirement, where to find it, in which state it is and who has to get it.

**Flow control** of read and write activities to the requirements from the involved people is another important component. Specifically, keeping track of the changes in the requirements is a crucial point.

**Dependence management** of the requirements is an unavoidable part of RM and the more requirements there are, the more important it will become. Requirements often depend on each other, or on additional documents and these dependences must be managed.

**Controlling** with respect to RM is a tool to evaluate and control the progress of the project. For instance, it is an important information for every project manager to know how far a specific task is.

### **2.5.1 Requirements Management with Object Ids**

Each information in a project passes through many hands and will be changed by different people. To avoid confusion it is important that every requirement has a unique identifier, the so-called object Id. Object Ids stay the same if the requirement changes and must not be reused if a requirement gets deleted, to guarantee its uniqueness. Of course it is optional to use unique names as object Id's, but it is highly recommended to use sortable patterns. This makes it easy to sort and filter the requirements in e.g. spreadsheets. Further, such patterns don't relate to the content of the requirement, so if the content of the requirement changes, this has no effect on the object Id. In addition, it is useful to encode different subsystems or specification levels in the object Id. (Rupp, 2009, pp. 359-361)

To give an example: **PR-SA-042**

- **PR:** Product requirements document
- **SA:** Sample application
- **042:** Unique id.

It can be seen that for humans such object Ids are easier to assign to special areas or applications as pure numbers. Furthermore, the administration of object Ids can be done in a simple office application, but there are numerous professional RE tools which manage this task automatically and thereby guarantee the uniqueness of the object Ids.

### 3 Quality of Service

---

QoS covers the non-functional behavior of a system which is similar to the non-functional requirements discussed in section 2.1.2.2. The non-functional behavior focuses on run-time of functions, resource usage, errors etc. Furthermore, QoS does not have a common or formal definition, instead there are many different definitions depending on the field of work. This chapter will discuss QoS in the context of packet-switched networks. In packet-switched network small data packets are routed through the network using routers, switches or other network devices to communicate between entities. (Mellouk, 2009, pp. 21-24)

Nevertheless, definitions can be found in the literature, for instance from Mellouk:

- “- QoS describes requirements on the behavior of service providers.*
- QoS may be described according to multiple parameters (delay, etc.).*
- QoS means different levels of user satisfaction.*
- QoS involves the networks as well as the applications.*
- QoS involves physical devices and terminals as well as software.*
- QoS may be considered at different communication layers (physical, data link, network, transport, and application), middleware, etc.*
- QoS requires the deployment of various mechanisms (negotiation, resource reservation, scheduling, routing, etc.)” (Mellouk, 2009, p. 24)*

A more general definition from the International Telecommunication Union (ITU):

*“Totality of characteristics of a telecommunications service that bear on its ability to satisfy stated and implied needs of the user of the service.” (International Telecommunication Union, 2008, p. 3)*

Additionally the definition from Tanenbaum and Wetherall:

*“It is interesting to observe that the network has more resources to offer than simply bandwidth. For uses such as carrying live video, the timeliness of delivery matters a great deal. Most networks must provide service to applications that want this real-time delivery at the same time that they*

*provide service to applications that want high throughput. Quality of service is the name given to mechanisms that reconcile these competing demands.”* (Tanenbaum & Wetherall, 2011, p. 35)

## **3.1 Terminology**

This section will give an overview of the terminology used in the context of QoS.

### **3.1.1 End-to-End QoS**

Generally users are interested in the QoS requirements at the point they interact with the system. Thereby all nodes which are needed for the communication are involved to fulfill the QoS requirements, this is usually called end-to-end QoS. For instance, the end-to-end delay or the end-to-end jitter of an application. (Mellouk, 2009, p. 24) (Shenker & Wroclawski, 1997, p. 10)

### **3.1.2 Classes of Services**

Depending on the characteristics of an application, different guarantees of QoS are necessary. For instance, the end-to-end delay in a file transfer application is not as important as in a real-time chat application. (Mellouk, 2009, pp. 24-25)

The list of different service levels below is only an overview of some of the most important elements and thus incomplete. Furthermore, the different service levels are not disjoint i.e. there exists overlap as well as no clear dividing lines. Nevertheless, the described services are important for a better understanding of the following sections.

#### **Deterministic service**

The QoS is deterministic, this means that a desired QoS is guaranteed for the user or application. A deterministic service is needed for time-critical applications. (Mellouk, 2009, p. 25) (Tanenbaum & Wetherall, 2011, pp. 318, 319)



### **Best-effort service**

There is no assurance that a specific level of QoS is met, but the involved parties use their best effort to satisfy the user's requirements. (Mellouk, 2009, p. 25) (Tanenbaum & Wetherall, 2011, pp. 318, 319)

### **Predictive service**

QoS requirements can be achieved by tuning the network condition under the assumption that the future network load can be calculated from the current load. (Mellouk, 2009, p. 25)

### **Probabilistic QoS**

Delivers a QoS level with some probability but does not guarantee a certain quality. (Mellouk, 2009, p. 26)

### **Controlled-load**

Capacity admission control is used to assure approximately the same QoS under light network load as under heavy network load. This service level is especially useful for multimedia applications where the quality constraints outweigh the time constraints. (Mellouk, 2009, p. 25)

## **3.1.3 QoS Parameters**

QoS parameters are needed from an application to specify QoS requirements for a network (Campbell, 1996, p. 20) (Aurrecochea, et al., 1998, p. 140). Table 3 shows typical QoS parameter values for generally known applications. The list of QoS parameter types below is only an overview of some of the most important elements and thus incomplete.

### **Time related parameter: Delay**

Represents the transmission delay of an element (e.g. router) in the network. The sum over the individual transmission delays of a path through the network defines the end-to-end delay. (Mellouk, 2009, pp. 27-28) (Szigeti, et al., 2013, p. 4)

### **Time related parameter: Jitter**

Represents the variation of the period between the arrivals of packets. To illustrate, a server sends a continuous stream of packets (e.g. every ms a packet) to a client. However, the client

receives these packets with a varying period, this distortion is called jitter. (Mellouk, 2009, p. 28) (Tanenbaum & Wetherall, 2011, p. 406) (Szigeti, et al., 2013, p. 4)

**Volume-related parameter: Bandwidth**

Represents a measurement of the amount of data that can be transferred in a given time period e.g. bits per second. (Mellouk, 2009, p. 28)

**Error-related parameter: Packet loss**

The packet loss parameter provides an overview of the lost packages during a transmission. Additionally, the ratio between lost packets and total packets is called packet loss ratio. (Mellouk, 2009, p. 28) (Szigeti, et al., 2013, p. 4)

**Reliability-related parameters**

These parameters are represented by multiple names such as **Mean Time Between Failure (MTBF)**, **Mean Time To Repair (MTTR)** or simply availability. (Mellouk, 2009, p. 28)

| Application type | Example          | Bandwidth                 | End-to-end delay       | End-to-end jitter      | Los rate           |
|------------------|------------------|---------------------------|------------------------|------------------------|--------------------|
| Data             | FTP              | High bitrate is preferred | Low delay is preferred | Low delay is preferred | 0 %                |
| Real-time        | Voice            | ≤ 64 kb/s                 | ≤ 300 ms               | 10 ms                  | 0 %                |
|                  | Video (TV)       | ≤ 10 Mb/s                 | ≤ 250 ms               | 10 ms                  | 10 <sup>-2</sup> % |
|                  | Compressed Video | ≤ 2 Mb/s                  | ≤ 250 ms               | 1 ms                   | 10 <sup>-6</sup> % |

*Table 3: Typical QoS parameter values for generally known applications (modified from (Mellouk, 2009, p. 31)).*

**3.1.4 Classification of Applications**

Applications can be classified in different categories, distinguished by their QoS needs. Classes according to Mellouk (2009) would be:

### **Background Applications**

Typical examples are SMS, MMS and Email applications. A common feature is that the user is not expecting the data and thus this class is very delay insensitive. Nonetheless the delivery of the data may have some time constraints. (Mellouk, 2009, p. 30)

### **Streaming Applications**

In a streaming application a user is consuming (real-time) video and audio data. Such an application needs a low end-to-end jitter value, furthermore the maximum delay depends on the usage of the application. (Mellouk, 2009, p. 30)

### **Conversational Applications**

Typical examples are telephony and video conferences which are performed between users. Depending on the user's perception this class is very end-to-end delay sensitive. Additionally, the end-to-end jitter constraints are similar to the streaming class. (Mellouk, 2009, p. 30)

### **Interactive Applications**

The interactive class represents applications which are interactively communicating with remote services. Examples are: games, web browsers and remote terminals. Round trip delays and (depending on the application) the packet loss are important parameters for these applications. (Mellouk, 2009, p. 30)

## **3.1.5 QoS Mechanisms**

Network resources are not infinite, thus it is not possible to transport the traffic of all applications with the required bandwidth, without jitter or packet loss and with no delay. As a result, QoS mechanisms are designed to control the available network resources. The following subsections describe two QoS mechanisms.

### ***3.1.5.1 Admission Control***

Admission control regulates the admission of new connections in a computer network depending on the available resources to avoid e.g. congestions. Specifically, that means if the necessary resources are not available to guarantee an appropriate level of QoS, the new data flow will not

be admitted in the network. A similar approach is used for the telephone network. If the network gets overloaded the system refuses new connections. To calculate the needed bandwidth of calls is easy, because it is always 64 kbps. Whereas the needed bandwidth in computer networks is discontinuous and thus not that easy to estimate. For instance, if a 100 Mbps network guarantees its users a minimum of 50 Mbps, the easiest solution would be to admit only two users in the network, but this will waste resources and is not very efficient. Therefore more complex approaches are used e.g. based on statistical models. It is, of course, possible to demand further QoS requirements, beside bandwidth, to the network. (Tanenbaum & Wetherall, 2011, pp. 397, 398, 415-418) (Mellouk, 2009, pp. 36, 37)

### **3.1.5.2 Traffic Control**

Network traffic control mechanisms are responsible for a proper end-to-end services across networks in real-time. This enables applications to achieve a certain level of QoS. (Mellouk, 2009, p. 41)

#### **Congestion control**

A congestion is a overloading of the network if too much traffic is send from computers. This leads to a poor performance, packet loss and an increased delay. The congestion occurs at the network layer but it is caused from the transport layer. Thus, congestion can be controlled and avoided by the transport layer. (Tanenbaum & Wetherall, 2011, p. 530)

#### **Flow control**

Flow control is a mechanism to adjust the transmission rate, to avoid that a slow receiver is swamped with data by a fast sender. (Tanenbaum & Wetherall, 2011, pp. 34-35)

## **3.2 Protocols**

Communication protocols in the sense of computer networking are processes or rules with which the communication between entities takes place. Most communication protocols can be assigned to a layer in the **Open System Interconnection (OSI)** model which gives a better overview of the context of the protocol. However, the most common used protocol suite is not the OSI Model but

the **Transmission Control Protocol / Internet Protocol (TCP/IP)**. A mapping between the layers of these protocol suites can be seen in Figure 8. The implemented protocols of the individual layers communicate with their neighbors where each layer addresses a different part of the communication. The layers and thus the protocols of those layers are based on each other, this is called the *protocol stack*. An assignment of protocols to their layer can also be seen in Figure 8. (Tanenbaum & Wetherall, 2011, pp. 29-31, 41, 42, 48-49)

As already mentioned, **Internet Protocol (IP)** networks are the most common technology these days, but by default they only provide best effort services (Mellouk, 2009, p. 354). However, there are other technologies such as **Asynchronous Transfer Mode (ATM)**. ATM networks provide a uniform bandwidth and uniform delay. This technology supports a constant bit rate for e.g. telephony, as well as real-time variable bit rate for e.g. video conferences and non-real-time variable bit rate for e.g. movie on demand services. The remaining bit rate can be used from other applications which are not delay or jitter sensitive such as file transfer programs. Compared with the OSI model ATM is located at the link layer. Today it is only a niche technology which is used e.g. in broadband access lines such as **Digital Subscriber Line (DSL)**. (Tanenbaum & Wetherall, 2011, pp. 249, 250, 406)

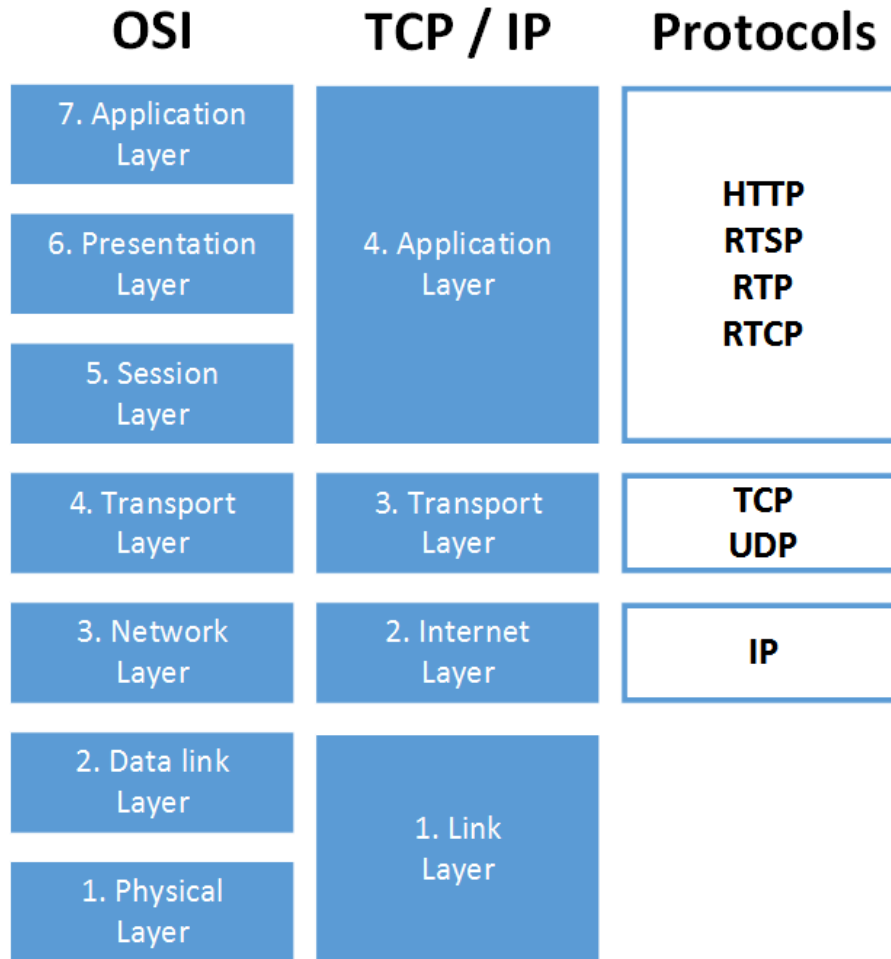


Figure 8: OSI model versus TCP / IP and related protocols (modified from (Tanenbaum & Wetherall, 2011, p. 42)).

### 3.2.1 Protocol: IP

The IP protocol is located at the network layer of the OSI model as well as in the internet layer of the TCP/IP model and provides a connectionless best effort service. The protocol is responsible for the transport of packets between entities, regardless whether these entities are in the same network or not. The IP header included the *type of service* field which allowed a specification whether the user wants low delay, high reliability or high throughput for an IP packet. However, the field was not used by routers and hence it was reassigned for **Differentiated Services (DiffServ)**. (Tanenbaum & Wetherall, 2011, pp. 436-442)

### **3.2.2 Protocol: UDP**

The **User Datagram Protocol (UDP)** is a connectionless transport protocol which encapsulates IP datagrams. UDP is a very simple protocol without any congestion control, flow control, sequenced delivery of packets or retransmission of lost packets. It only provides an interface to the IP protocol with adding as less as possible additional features. (Tanenbaum & Wetherall, 2011, pp. 541-543)

### **3.2.3 Protocol: TCP**

The **Transmission Control Protocol (TCP)** is a connection-oriented protocol which provides applications a reliable end-to-end service over an unreliable network. TCP enables applications to send byte streams in a *full duplex* mode, i.e. an endpoint is able to send and receive data at the same time. Such an endpoint is also called *socket* and is bound to a special *port*. Furthermore, the protocol supports congestion control, flow control, sequenced delivery of data and retransmission of lost packets. Due to the mentioned advantages TCP is the leading protocol in the internet. (Tanenbaum & Wetherall, 2011, pp. 552-606)

### **3.2.4 Protocol: HTTP**

The **Hyper Text Transfer Protocol (HTTP)** is the basis of the **World Wide Web (WWW)** and located at the application layer of the TCP/IP model. Normally HTTP works on the top of TCP. It enables clients to send requests to servers and server to send responses back. Hence, HTTP is used like a transport protocol to communicate between applications. (Tanenbaum & Wetherall, 2011, pp. 45, 683, 684)

### **3.2.5 Protocol: RTP**

The **Real-time Transport Protocol (RTP)** is a generic protocol designed to transfer multimedia data such as audio or video. Normally RTP runs on the top of UDP and thus there is no reserved bandwidth or particular low delay for RTP packets (besides those QoS parameters which may be manually configured in routers). RTP does not offer a retransmission of lost packets, because this

would yield in increasing jitter, though it does recognize packet loss. Additionally, the RTP packet includes further methods to reduce jitter. (Tanenbaum & Wetherall, 2011, pp. 547-549)

### **3.2.6 Protocol: RTCP**

The **Realtime Transport Control Protocol (RTCP)** is used to negotiate QoS parameters between server and clients for the RTP protocol. QoS properties such as delay, congestions or bandwidth are used to adjust streaming parameters. For instance, if a congestion occurs the server can switch to another encoding which needs less traffic. (Tanenbaum & Wetherall, 2011, pp. 549, 550)

### **3.2.7 Protocol: RTSP**

The **Real Time Streaming Protocol (RTSP)** is used to establish and control streams of continuous audio and video data. It is not designed to transport the data stream itself, instead it acts as a control mechanism for multimedia data (e.g. start, stop, seek back and forward in a video). RTSP works in a similar way as HTTP and it is running on the top of TCP or UDP (Tanenbaum & Wetherall, 2011, pp. 719, 720). Hence, RTSP controls a multimedia stream which is delivered by another protocol such as RTP, but it is worth mention it that RTSP does not require a special transport protocol. (Schulzrinne, et al., 1998, p. 5)



## 4 Hospital Information System

---

During the last decades the term HIS has gained importance in the medical sector. Nevertheless, it is often source of confusion and people frequently associate the term HIS with a single program.

A definition can be found from Rudi van de Velde et al.:

*“A hospital information system (HIS) can be defined as an open system which attempts to integrate and communicate the outside and inside flow of information within a hospital and provide the functions common for all applications.”* (Velde, 1992, p. 90)

As can be seen from the definition, a HIS is more than just a single application which allows users to query and enter data. It is more a system which connects different applications and enables them to interact properly with each other.

This chapter will discuss current HIS systems and standards such as HL7 or **Digital Imaging and Communications in Medicine (DICOM)**, as well as give a brief history on this topic.

### 4.1 A Brief History

Since the second half of the 20<sup>th</sup> century the utilization of IT-systems in hospital environments is increasing. The first systems were implemented to support special departments. These systems worked decentralized and improved only the workflow of a special care unit without communicating across the departmental boundaries, which isolated these systems from the outside world. As a consequence of such a system, there exists redundant and inconsistent data which produces problems. In the 1980s and early 1990s, first standardized interfaces were developed to exchange medical data between different applications to avoid the above mentioned issues. One of these standardized interfaces is known as HL7 which allows the users to identify patients across multiple departments and thus to store medical information of different care units clearly related to one person. (Prokosch, 2001, pp. 1, 2)

Over the following years the term HIS established quickly in the healthcare sector and products from different manufacturers reached the market. Nowadays such systems became integral part of the medical environment.

## **4.2 HIS Software**

There are numerous software solutions which are summarized under the term HIS. This section aims to provide a brief overview of current systems on the market.

### **4.2.1 HIS for SAP Business One**

*Hospital Information System for SAP Business One* (see Figure 9) supports and manages typical processes of hospitals, private clinics or laboratories etc. The system allows users to manage patient reservations and admissions, medical reports, invoicing, in-patient management, inpatient and outpatient procedures, patient records and therapies. HIS for SAP Business One is a proprietary application from *SAI - Servizi Avanzati per le Imprese* which is built on SAP Business One. The system is not for free and not open source. The application provides a consistent user interface based on single **Data Base Management System (DBMS)**. Furthermore, the system supports standards such as XML, SOAP or HL7 3.0. HIS for SAP Business One defines three main areas which are important for the health services: Administrative Area, Administrative / Medical Area and Healthcare Area. (SAI Servizi Avanzati per le Imprese srl & SAP, 2013)

#### **Administrative Area**

This area includes different modules for administrative and financial procedures. For instance, the budget module helps the employees to monitor costs and revenues. The controlling module processes business data and supports the analysis of financial and economic values. The warehouse module helps to manage medical materials such as drugs and expire dates. (SAI Servizi Avanzati per le Imprese srl & SAP, 2013)

#### **Administrative / Medical Area**

This area of the system includes administrative as well medical functionalities. For instance, a module helps the staff to manage the doctors' shifts. Another module supports the health services to reduce the patient's waiting. Furthermore, a separate module links the hospitalization process to the administrative area which allows a timely billing. (SAI Servizi Avanzati per le Imprese srl & SAP, 2013)

## Healthcare Area

The healthcare area covers the medical part of the system, which focuses on e.g. the hospitalization of inpatients to manage information such as the patient's bed, diet or the patient record. Additionally, a module includes a Radiology Information System to manage diagnostic images. Another module adds a Laboratory Information System to support the lab processes. Furthermore, the HIS enables the users to study medical records on PCs as well as on mobile devices such as tablets. (SAI Servizi Avanzati per le Imprese srl & SAP, 2013)

Figure 9: Typical view of HIS for SAP Business One (SAI Servizi Avanzati per le Imprese srl & SAP, 2013).

### 4.2.2 Medico

*Medico* (see Figure 10) is a HIS with an open architecture to simplify the integration of third party software systems e.g. SAP. The system supports processes for doctors, nursing staff, IT and the administration. *Medico* is a proprietary application from *Siemens*, thus the system is not free and not open source. (Siemens, 2014)

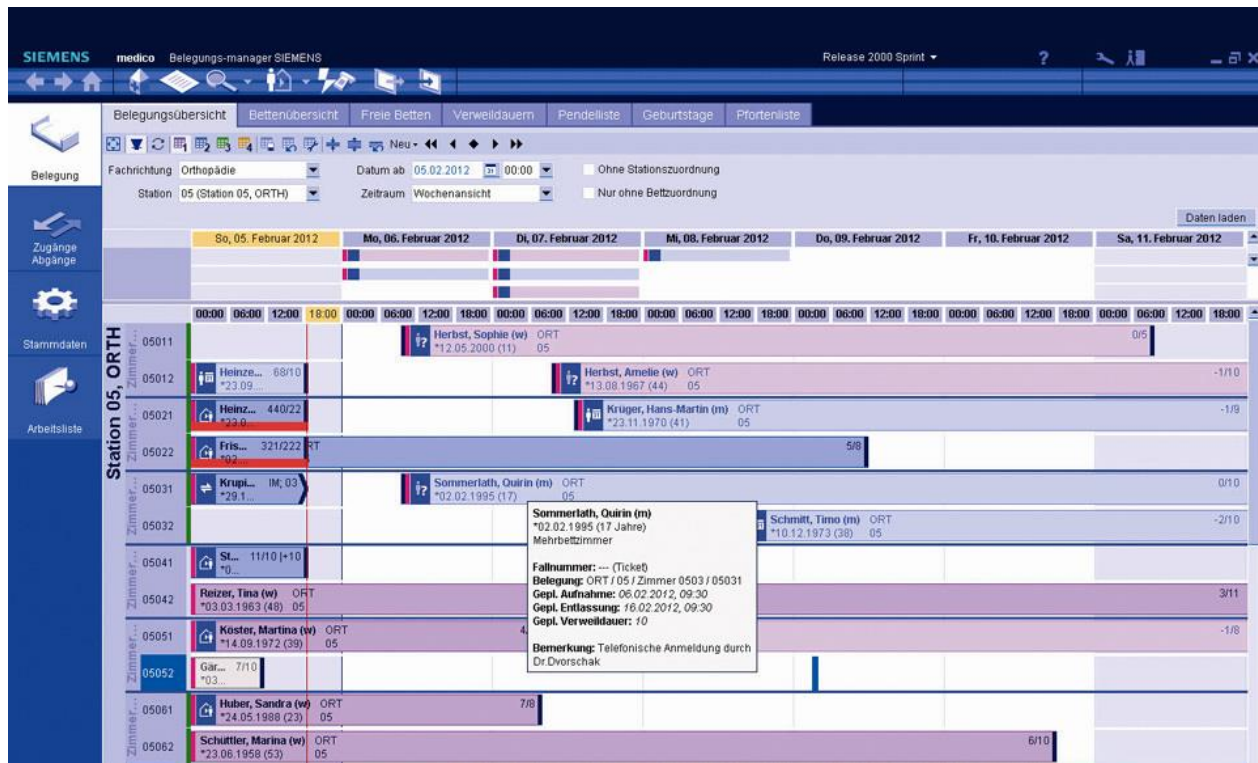


Figure 10: Typical view of Medico (Siemens, 2014).

### Supports Processes for Doctors

Medico supports doctors by creating medical documentations, operation-room organization and the optimization of workflows etc. Additionally, patient information is saved in an electronic health record on a central point, to provide easy access from different departments. (Siemens, 2014)

### Supports Processes for Nursing Staff

The system helps the nursing staff in the drafting of care planes or by accessing vital parameters from patients. Furthermore, it supports the users in the communication between different departments and the documentation of prescriptions of doctors. (Siemens, 2014)

### Supports Processes for the IT Department

Medico has an open architecture which enables the integration of further systems or the adjustment of the system to existing software. (Siemens, 2014)

## **Supports Processes for the Management**

The system manages inpatients and outpatients to document the provided medical services. This information is used to support billing and controlling. (Siemens, 2014)

## **4.3 Standards**

Most parts of a HIS are not standardized and vary from one software manufacturer to the other, nevertheless a standardization is a crucial condition for some components to e.g. simplify the communication between different applications. The following subsections describe two standardized protocols for the medical sector.

### **4.3.1 HL7**

The primary goal of HL7 is to provide a standard for exchanging data among different systems to facilitate the communication. It allows to send or receive data about:

- patient admissions / registration, discharge or transfer
- resource and patient scheduling
- orders
- billing
- medical records
- etc.

HL7 is designed to support IT systems in the health sector and does not pretend a particular architecture. Consequently, HL7 is only a standard which defines how dissimilar applications should communicate with each other:

*“It does not try to assume a particular architecture with respect to the placement of data within applications but is designed to support a central patient care system as well as a more distributed environment where data resides in departmental systems. Instead, HL7 serves as a way for*

*inherently disparate applications and data architectures operating in a heterogeneous system environment to communicate with each other.”* (Health Level Seven, Inc, 2003, p. 2 of Chapter Introduction)

Since the 1980s the standard is being continuously developed and improved by the HL7 Working Group. (Health Level Seven, Inc, 2003, pp. 1-16 of Chapter Introduction)

The encoding of HL7 messages depends on its version. For instance, HL7 version 2.x messages consist of multiple segments, where each segment refers to a special concept or entity. The limitation of this approach is that the communicating systems must agree on optional fields and the semantic interpretation of data. The transport of HL7 version 2.x messages is defined as a client-server model but messages can also be stored in a text file and transmitted with any file transfer protocol. (Beeler, 1998, pp. 151-153)

Version 3.0 introduced a new paradigm which is based on an object-oriented development methodology called the Version 3 Message Development Framework, which allows a model based specification of messages. The actual message and transport of HL7 version 3.0 is defined by the Implementable Message Specifications. As a result, there is a wide range of possible communication protocols from simple ASCII streams to object oriented interfaces such as **Common Object Request Broker Architecture (CORBA)**. (Blobe, et al., 2006, pp. 343-346) (Beeler, 1998, pp. 153-161)

#### ***4.3.1.1 HL7 Version 2.5 Message***

A HL7 version 2.5 message consists of multiple segments which can be optional or required. Each segment starts with a three character pattern. Those patterns are unique for a special segment type. The segments are separated by special characters, where each segment consists of multiple data fields. The data fields contain data of variable length and a certain data type. Similarly, data fields are separated by a special field separator character. (Health Level Seven, Inc, 2003, pp. 8-9 of Chapter Introduction)

Figure 11 shows an example HL7 2.5 message with a MSH and a PID segment. The segment separator is a carriage return. The field separator is a “|”, furthermore, “^” is a sub-field separator.

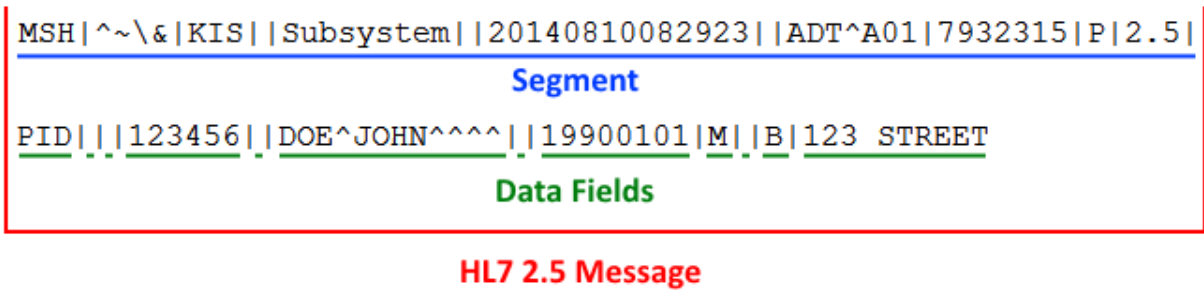


Figure 11: An incomplete HL7 2.5 message

### 4.3.2 DICOM

DICOM is a medical image processing standard which is often associated with **Magnetic Resonance Tomography (MRT)** or **X-ray Computed Tomography (X-ray CT)**, it includes definitions for:

- communication
- image structures
- diagnostic and therapeutic information
- workflow optimization / management
- image printing
- etc.

DICOM includes network services for communication between systems. It is a so called upper layer protocol which is based on TCP / IP. It enables the exchange of messages, services and information objects. A second very important part of DICOM is the specification of data structures for medical images and additional medical information associated to pictures. The data structures are based on **Information Object Definition (IOD)** (Mildenberger, et al., 2001, p. 921) which are defined for image data from e.g. MRT or X-ray CT. A header in the data includes additional information such as medical reports, patient data or performed procedures. I.e. IODs are abstract definitions of real world entities suitable for the communication between different instances. The network services which are responsible for information transfer distinguishes two roles: The

provider and the consumer of a function. For example, the provider sends an MRT image to the archiving system, which is in this case the consumer. To establish a communication between two DICOM parties there has to be a provider and a consumer and both have to support the same services (e.g. image transfer) and the same object (e.g. MRT data). (Mildenberger, et al., 2001, pp. 920, 921) (National Electrical Manufacturers Association, 2011, pp. 9-15 of Chapter Introduction and Overview)



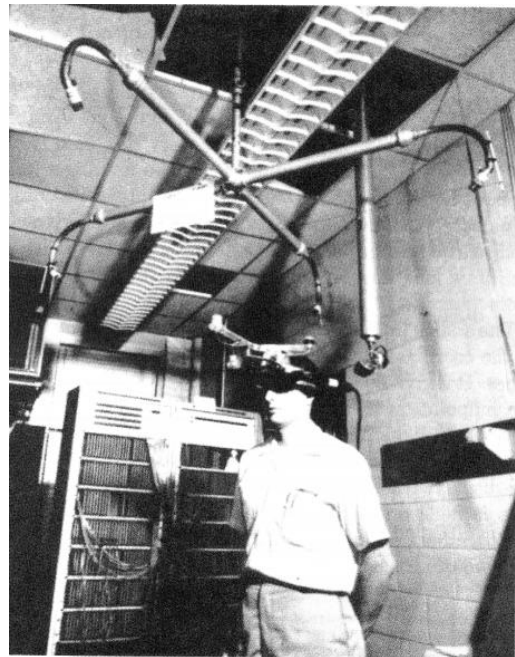
## 5 Head-Mounted Display

---

A HMD is a device, worn on the head e.g. like a glass or as part of a helmet, therefore also called Helmet-Mounted Display and is used to display information in the front of the user's eyes, or directly on the retina. Such devices are often used for the immersion into a convincing simulation of reality, or to extend the reality with additional information, known as augmented reality. The latter are also known as **Optical Head-Mounted Display (OHMD)** and allow the user to see through it. A further subdivision is monocular HMD, where the display optic is only in front of one eye, respectively to binocular HMD where the display optic covers both eyes (Rash, 1999, pp. 130, 131). Obviously for binocular HMD there is a stronger demand on stereo viewing. Today prominent representatives are Oculus Rift (Oculus VR, Inc., 2014) which is mainly designed for gaming and is fully immersive. On the other hand there is Glass (Google (a), 2014) which has the focus on extending the reality with useful information.

### 5.1 A Brief History

One of the main purposes of HMD's is stereo viewing of images. The first device that achieved this was invented by Sir Charles Wheatstone in 1838 and is called the Wheatstone stereoscope (Wheatstone, 2014). It used two pair of mirrors, each reflected a picture for the right and left-eye. In 1943 Thelma McCollum described in his patent "Stereoscopic television apparatus" the first HMD (Mccollum, 1943). The first real HMD with computer generated images was built by Ivan Sutherland in 1968. In the paper "A head-mounted three dimensional display" (Sutherland, 1968, p. 295 ff.) he presented a device called "The Sword of Damocles", shown in Figure 12.



*Figure 12: First real HMD, called "The Sword of Damocles" (Sutherland, 1968, p. 298)*

The device supported 3D stereoscopic view of simple wire frames and head tracking. The setup consisted of two miniature **Cathode Ray Tubes (CRTs)**

with special optics and two head position sensors, one ultrasonic and the other mechanical. But in these early days of computer graphics, it was even difficult to draw simple wireframe structures, because the first primitive commercial video display controller reached the market not before the 1980s. That is why this approach had several limitations. For instance, if any portion of a line was off the screen, the whole line disappeared. Since the early 1970s, military tested the helm-mounted display systems to provide information to pilots. In the early 1980s, while still in high-school, Steve Mann designed a backpack-mounted computer-imaging system that allowed him to keep an eye on a CRT while walking around (Mann, 1997, p. 26). In his later years he made many more contributions to the field of wearable computing or computer vision / graphic, for instance in 1995 he published the first paper about high dynamic rendering (Mann & Picard, 1995, p. 422 ff.). In 1985 the first **Virtual Retinal Display (VRD)** was invented by Kazuo Yoshinaka of Nippon Electric Co (Yoshinaka, 1985). Instead of projecting images in front of the eye, this technique draws the image directly onto the retina. The early 1990s were an exciting time for the computer industry. New technologies such as the internet or multi media were in the starting blocks and the first HMDs appeared on the consumer markets. From this time, one of the most popular consumer HMDs was the Forte VFX1 Virtual Reality Headgear developed by Forte Technologies, Inc. It supported stereoscopic 3D view and three **Degrees of Freedom (DOF)** head tracking (Mindflux, 2006). Despite its advanced features and high quality the product failed, because of the complex installation and the high price. In the following years numerous attempts had been made to market HMDs, but all of these failed. Now in 2015 a further generation HMDs, devices such as Glass or Oculus Rift try to enter the consumer market.

## **5.2 Head-Mounted Display Properties and Types**

The quality of a HMD does not only depend on the image quality of the device, but also on other properties such as ergonomics, costs or support etc. However, the image quality of course, is a major part of the perception of quality of a HMD. According to (Schmalstieg, 2013/14) key factors are:

- The degree of immersion describes the perception of feeling physical present in a virtual world (Bangay & Preston, 1998, p. 43). This means that the degree of immersion of a device, you can see through, is lower as from one, which veiled the entire physical world.
- The **Field of View (FOV)** describes the extent of the world that is seen from the observer. Humans have almost a FOV of 180 degrees, i.e. to get full immersion a HMD should cover as much as possible of the human FOV. (Melzer, 2011, p. 60)
- The image resolution is another important property which has a strong impact on the impression of the device. Related properties are the brightness and contrast of the image.
- The ability for binocular HMDs to display different images to each eye gives the device the potential for stereoscopic 3D imaging, so the quality of the stereoscopic vision plays a primary role for these devices. (Rash, 1999, pp. 130, 131)

Another essential part of the quality of HMDs is the ergonomics. According to (Schmalstieg, 2013/14) key factors are:

- The weight of the device, or additional cables, have a large influence on the acceptance of the device and the application field.
- Hygiene may be a requirement if a lot of people want to use the same device, or if there are specific restrictions such as in hospital environments.
- The esthetic and haptic of the device have, of course, impact to the success.

There are different types, different techniques and different fields of application for HMDs. For a better overview of the terminology these types will be discussed in the next sections in more detail.

### **5.2.1 Head and Helmet-Mounted Display**

HMD is a generic term which is often used for all kinds of display devices which are worn on the head or as part of a helmet. In the most cases it is a binocular device and has two LCD, or in the past CRT, screens plus special optics in front of the user's eyes. With this type of display the user

cannot see the real world. Therefore it is commonly used in virtual reality applications, to get a high degree of immersion. (Melzer, 2011, pp. 2, 3)

### 5.2.2 Arm-Mounted Display

An arm-mounted display, also known as **Binocular Omni Orientation Monitor (BOOM)** works like a HMD, but mounted on an articulated arm, to relieve the user from a perhaps heavy device. Hence, such constructions are able to use CRT technology to display the image (see Figure 13). Furthermore, these devices have the ability to use heavy optics and a very fast and accurate mechanical head tracking via the articulated arm. (Bolas, 1994, pp. 55-58)



Figure 13: Binocular omni orientation monitor from Fakespace Labs (Fakespace Lab, 2004)

### 5.2.3 Optical Head-Mounted Display

OHMD is similar to a HMD with the difference that it allows the user to see through it, so it isn't fully immersive. This property makes the OHMD to the perfect augmented reality device and enables applications to display information directly in the user's FOV. Accordingly, users have the ability to focus on their current work without watching to a monitor. An example of such a device is the Evaluation Kit interactive see-through HMD from Fraunhofer Research, see Figure 14. The

focus of this device is the above mentioned augmented reality, it combines the user's view with additional computer generated information using a new display technology called: bidirectional OLED microdisplay (Fraunhofer



Figure 14: The OLED based binocular interactive see-through HMD Eval-Kit (Fraunhofer Research Institution for Organics, 2014)

Research Institution for Organics, Materials and Electronic Devices COMEDD, 2012, p. 1 ff).

### 5.2.4 Head-Mounted Projective Display

This technology is often used in conjunction with OHMDs. **Head-Mounted Projective Displays (HMPD)** use a small projector that projects images to a retroreflective surface placed in front of the user's eyes. In a similar way works Glass. A mini



*Figure 15: The Google Glass (Google (a), 2014)*

projector projects the image to semi-transparent prism, which focuses the image directly on the retina (Kress & Starner, 2013, pp. 9-11).

### 5.2.5 Virtual Retinal Display

Instead of drawing the image in front of user's eyes, this technique draws the image directly on the retina. This method has the ability for a high resolution with a large FOV and a small size, thereby it does not require a display to view the image. Instead a source of photons is modulated with the image information and scanned on the retina. (Furness & Kollin, 1992, p. 1 ff.)

Currently this technique is not quite ready for the market, since there are problems like focusing the image on the retina without a tricky setup.

## 6 Requirements Engineering: Application

---

Chapter 2 above discussed the theoretical background of RE. Supplementary this section will apply the theory on a software project in the medical area. Specifically, this thesis and thus the developed software is embedded in a bigger research project (Vorraber, et al., 2014, p. 1266 ff.). The research project, including this thesis, aims to improve medical processes using near-eye display devices. It is worth mentioning that already a previous thesis (Sachs, 2014, p. 1 ff.) did some work within the research project. This thesis builds upon the existing work and expands as well as develop further aspects of this topic.

The applied RE was organized in collaboration with two hospitals. Tests as well as the RE process took place in the hospitals during the normal workday to get results and observations as close to the reality as possible.

Through prior observations in the hospitals two promising use cases were already revealed:

- Virtual Consultations
- Doctor's Visit

A closer look will be taken on these two cases in the following sections. The cases will be analyzed, discussed and evaluated in detail. In order to maintain a clear overview, the whole project will be separated in two subprojects according to the revealed use cases. Thus for each subproject the RE process will be performed independently. The structure of these sections will partly resemble the structure of the theoretical part in section 2. At first the as-is analysis should exhibit the initial situation. With this information in mind an evaluation of the current state is possible and subsequently the potential for improvement will be determined. Dependent on the evaluation the concrete requirements will be gathered. As documentation techniques the SOPHIST Requirement Template (see section 2.3.1), Use Case Specification (see section 2.3.4) and Business Process Model and Notation (see section 2.3.3) are used. Further, to manage the list of requirements during the project, the concept of object Ids is applied. In the next chapter a prototype / proof-of-concept will be used to validate the benefit of the Virtual Consultation.

## 6.1 Project Goals

As discussed in section 2.1.4 each project should have goals which describe the direction a project should take. Furthermore, the goals should be determined early in the project stage and assigned to a stakeholder. Because it is not relevant for this thesis, the names of the stakeholder will not be exposed. Additionally, the goals will only be formulated for the entire project and not for each subproject.

Since the thesis (Sachs, 2014, p. 41) was also embedded in the same research project it shares similar goals:

- Improve and support the medical processes by introducing near-eye display devices.
- Enable hospitals to share their medical knowhow across spatial boundaries.

## 6.2 Scenario Categorization

To organize and distinguish use cases of near-eye display devices in a technical way, the categorization scheme proposed by Vorraber et al. (2014) will be used in this master's thesis. The same categorization scheme can also be found in further articles and papers which are related to the same research project, which makes it easy to compare results with each other.

An example of the categorization by (Vorraber, et al., 2014, p. 1267) is outlined in Table 4 and the different factors which describe the use case will be explained briefly:

- Number of users / cases: The number of users involved in the use case.
- Data dynamics: This parameter describes how volatile the information displayed on the near-eye display device is. Static are e.g. medical records or images, dynamic are e.g. live video streams.
- Collaboration: Describes if multiple users are collaboratively interacting with the system.
- Information flow: Indicates in what direction data is transferred from the near-eye display device. Unidirectional (In) specifies that data is transferred to the near-eye display device

and unidirectional (Out) shows that data is published from the device. Bidirectional indicates that both directions are used.

- Mode of operation: Describes how or if users interact with the near-eye display device. Passive means there is no active interaction with the device. Active means that the user controls the device directly e.g. with his fingers or voice. The remote parameter indicates that the device is remote controlled by a user.
- Frequency of use: The frequency of use expresses how often the use case takes place.
- Network coverage: Indicates the distance data has to be transferred and thus the needed network type.

| Factor            | Characteristics     |                      |               |               |
|-------------------|---------------------|----------------------|---------------|---------------|
| #user/case        | Single              |                      | Multiple      |               |
| Data dynamics     | Static              |                      | Dynamic       |               |
| Collaboration     | Yes                 |                      | No            |               |
| Information flow  | Unidirectional (In) | Unidirectional (Out) | Bidirectional |               |
| Mode of operation | Passive             | Active               | Remote        |               |
| Frequency of use  | High (Daily)        | Medium (Weekly)      | Low (Monthly) |               |
| Network coverage  | Local               | Regional             | Nation wide   | International |

Table 4: Categorization scheme example to distinguish use cases, proposed by Vorraber et al. (2014)

### 6.3 Requirements Management

As already mentioned at the beginning of this chapter, to manage the requirements during the project, the concept of object Ids is used. The Ids are composed of multiple fields where each field is separated by a hyphen (-). The fields are subdivided into two parts. The first part is composed of multiple fields, where each field has alphanumeric values with multiple digits. Each of those fields describes a special property of the requirement. The second part is a global unique number for each requirement, with leading zeros. The order of the fields within the Id is important and hence must not be interchanged. The following list contains the fields of the first part of an object Id:



1. Subproject identifier:
  - VC for Virtual Consultations
2. Hardware requirement or software requirement:
  - H for hardware requirement
  - S for software requirement
  - B for software and hardware requirement
  - N for neither software nor hardware
3. Functional requirement or non-functional requirement:
  - FR for functional requirement
  - NR for non-functional requirement

An example of an object Id using this pattern is: VC-S-FR-001

## **6.4 Subproject: Virtual Consultation**

To get a second opinion is also helpful in the medical field and reduces the number of errors. Especially for things which are not common in the day-to-day work, an assistance can be very useful. Such things often need the opinion of a specialist which in the most cases is not in the same room, or building, or in the case of the Virtual Consultation even in the same city. In order to do this, the Virtual Consultation allows a doctor to get support or simply a second opinion from a specialist which isn't physically present.

Furthermore, similar use cases are surgeries, especially with young and still inexperienced surgeons, who need some kind of advice during a surgery. Even if a consulting surgeon is in the next room, she or he needs time to prepare herself or himself before entering the operation room. The reason for this is that there is a cleaning and sterilization procedure before a person is allowed to enter the operation room. By using the Virtual Consultation the surgeon can take a quick look without a time consuming preparation. (Sachs, 2014, p. 61)

Both use cases are equivalent from the user's point of view, for this thesis the focus of the Virtual Consultations won't be restricted to surgeries.

#### **6.4.1 As-Is Analysis and Identification**

This section describes the initial situation as well as the problems which were revealed during the as-is analysis process. As technique to gather the initial situation an informal conversational interview was chosen. The interviewed stakeholders were mainly medical staff or medical managers.

For the as-is analysis stakeholders of two hospitals were interviewed. Those two hospitals are geographically separated by approximately 100 km, but collaborate in certain medical areas.

For instance, the today's healthcare sector is highly specialized, so for each area there are some medical experts who work on particular fields. Those experts are hard to find and expensive. As a result, hospitals enter into cooperation arrangements between each other to share, for example, a part of their qualified specialist personnel.

In the same way this happens between the two investigated hospitals. Some specialists are only available during the week. On weekends the doctors on duty have to decide, if a medical case can wait until Monday, or if they have to transport the patient immediately to the next hospital, where such a medical professional is available. Nevertheless, often just a quick look of a specialist would help the doctor on duty to make his or her decision. For that purpose and among other things the IT department already installed a solution to exchange pictures and small videos between the two locations. The technology behind this are simple tablets which save the images or videos, taken by an employee, on a shared space where both parties have access to. In addition, to get feedback from the specialist and exchange further information an audio connection is necessary. For this purpose a simple telephone connection is used. This configuration presents the involved staff with certain usability problems. For instance, the parties have to ensure they are talking about the same picture or video. Also, often an image is not enough, e.g. if the patient has some sprain of a ligament in a joint. In the most cases such injuries change the way a person moves, and for healthcare professionals this can be a valuable information. But even if a video

was captured of the patient's movements this might not be that type of movement the expert wants to see. So the doctor in the remote hospital has to describe abnormalities and irregularities to the consulting expert, which can be, of course, in some cases difficult. Subsequently, the specialist could instruct the doctor in the remote hospital to take a video or photo, but this is time consuming and exhausting, not only for the medical staff but also for the patient. Moreover, the sprain is only an example and not necessarily the most valuable use case, but to demonstrate the current situation it is an illustrative example.

Besides, the HIS of the analyzed hospitals enables the users to save pictures and videos related to a patient. This feature is already used to document injuries or to store radiographs etc. As an example, during the hospital admission, patients often are transferred from other instances such as a family doctor. For legal reasons it is important to document how the patient's injuries were treated, before the admission to the hospital. Currently this is a two person workflow, the first one puts the object, which should be on the photo (for example a patient's leg) to the right place and the second person takes the image. Afterwards the photos are stored in the medical archive.

To get a better overview, the current process of exchanging data between the hospitals through telephone and simple pictures is outlined in Figure 16 with the use of a BPMN diagram (see section 2.3.3).

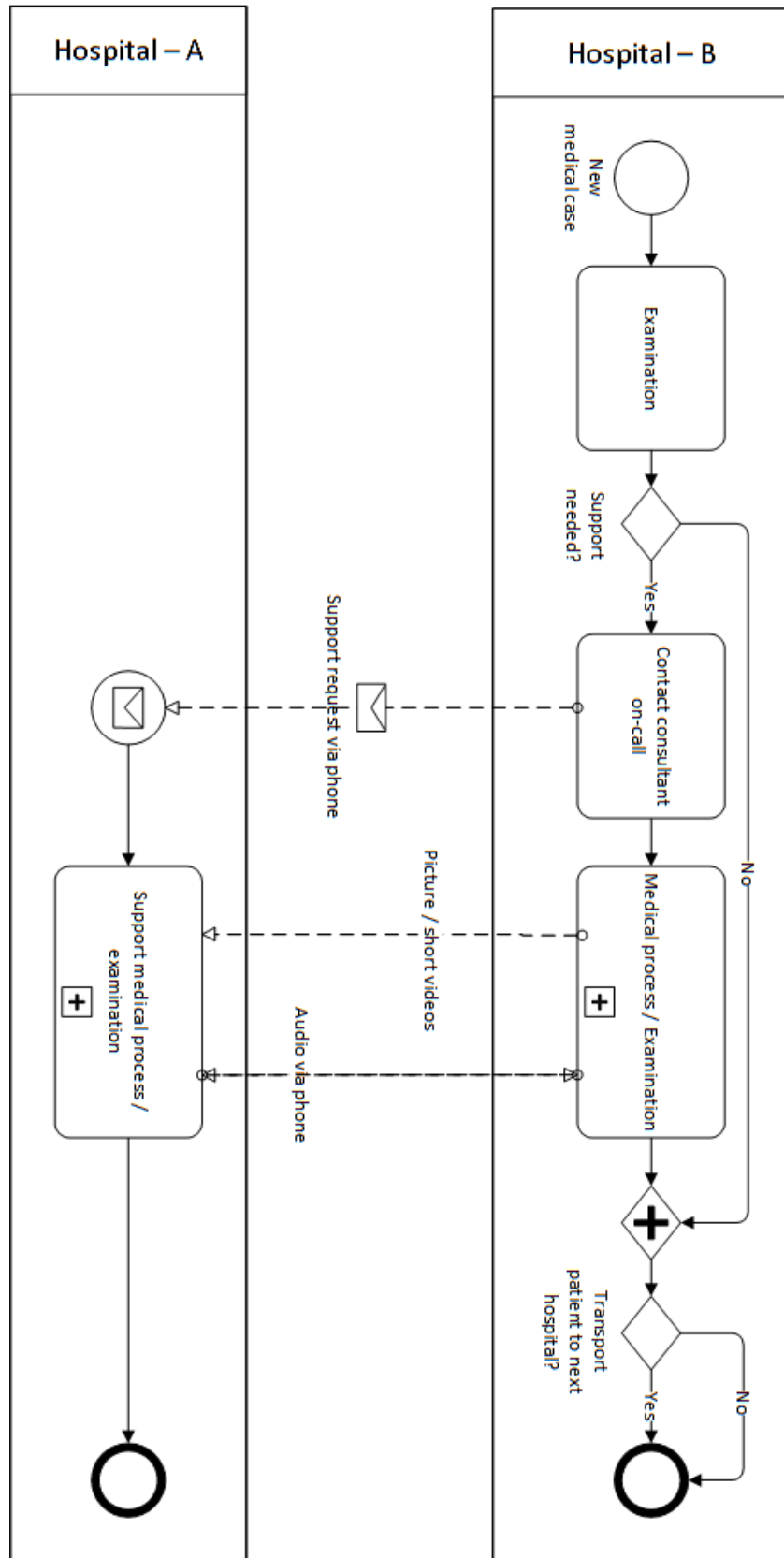


Figure 16: Overview of the current Virtual Consultation process between the two investigated hospitals, shown as BPMN diagram.

### 6.4.2 Potential for Improvement

The as-is analysis in section 6.4.1 above revealed that a strong need of an interactive communication between the hospitals is existing. The possibility of live video and audio conferences as well as simple picture transmission would improve the current situation. As well as a connection to the HIS of the hospitals, to store pictures or videos related to a patient may be an advantage. Using a near-eye display device, with an included head mounted camera, for the interactive communication, could produce major benefits for the involved parties. Especially the video signal from the first-person perspective could prove particularly useful. Discussions with the medical staff has shown, that seeing the world through the perspective of the person you are supporting may provide helpful additional information. For instance, if the doctor who wears the head mounted camera is hectic moving her or his head, she or he is maybe unfamiliar with the situation and nervous. Or it can just be used to see where the doctor is looking at. Another useful information might be, to get an impression how the patient looks at the doctor. With the first-person perspective it is easier to see if the subject is scared, in panic or even losing consciousness.

### 6.4.3 Scenario Categorization

In this section, the Virtual Consultation is outlined with the scenario categorization proposed by Vorraber et al. (2014). The matching characteristics are highlighted in Table 5.

| Factor            | Characteristics     |                      |               |               |
|-------------------|---------------------|----------------------|---------------|---------------|
|                   | #user/case          | Single               | Multiple      |               |
| Data dynamics     | Static              | Dynamic              |               |               |
| Collaboration     | Yes                 |                      | No            |               |
| Information flow  | Unidirectional (In) | Unidirectional (Out) | Bidirectional |               |
| Mode of operation | Passive             | Active               |               | Remote        |
| Frequency of use  | High (Daily)        | Medium (Weekly)      |               | Low (Monthly) |
| Network coverage  | Local               | Regional             | Nation wide   | International |

Table 5: Categorization scheme for the Virtual Consultation proposed by Vorraber et al. (2014)

#### **6.4.3.1 Existing Conferencing Software**

Applications which provide video and / or audio conferences are popular and well known not just by people in the IT sector. For instance: Skype or Google Hangouts.

- Skype is a voice over IP service combined with an instant messaging client from Microsoft. It allows simple text chats as well as voice and video chats between two users or in form of a group conference with multiple users. Additionally, Skype allows voice calls and sending SMS to recipients from the traditional telephone network. Furthermore the application is available on the most platforms, from Windows to Linux, from Android to Blackberry OS. (Microsoft (a), 2014)
- Google Hangouts is a similar application as Skype. It combines an instant messaging client with a voice and video chat tool. Available are normal pair conversations as well as video and voice conferences. Google Hangouts is linked with Google+ and can't be used without such an account. It should be noted that Google Hangouts is also available on Glass. (Google (b), 2014)

There are further consumer applications which provide a similar service, but such applications often store data on third-party servers, so there can be a problem with data confidentiality. Another problem is the availability of the service if the system is not completely under the control of the hospitals IT department. Consequently, those applications are not the best choice to achieve an interactive video and audio communication between hospitals.

Nevertheless, there are already systems specially designed for the healthcare sector e.g. (kapsch, 2013) or even companies which focus, inter alia, on this area e.g. (Vidyo, 2014). The latter is a global active company which provides solutions for governments, healthcare, finance etc. Customers are for instance, well-known names such as CERN (Vidyo, 2014). Another example for a video conference application from the professional sector is Citrix GoToMeeting (Citrix Online, 2014), which provides similar functionalities as the above mentioned systems.

#### 6.4.4 Resulting Requirements

The requirements which have been determined during the as-is analysis and further interviews with the involved stakeholders will be listed in detail. Furthermore, as already mentioned this thesis is based on results from (Sachs, 2014, p. 1 ff.) and hence some of the requirements which were already found in Sachs' work, will be adopted. As already mentioned at the beginning of this chapter, the use case will be described briefly in a Use Case Specification to get a general overview. Afterwards, the requirements will be listed in tabular using the SOPHIST Requirement Templates and organized with object Ids.

For section 6.4.4.1 and 6.4.4.2 a special terminology is applied. The following table provides an overview of this terminology (see Table 6).

| Expression                 | Definition   |
|----------------------------|--|
| The system                 | All parts of the software and hardware which establish the functionality of the Virtual Consultations. |
| Near-eye display device    | The hardware of the near-eye display device itself, as well as the software running on the device.     |
| Both parties               | The doctor who needs support as well as the medical specialist who provides the support.               |
| Consultant                 | The medical specialist who provides the support.   |
| Advisee                    | The doctor who needs support.  |
| Involved parties           | Equivalent to both parties.  |
| At any given point in time | The time period where the Virtual Consultation takes place and the system is ready to use.             |

*Table 6: Definition of the terminology which is used for the SOPHIST Requirements Template approach.*

##### 6.4.4.1 Use Case Specification

The preliminary work from section 6.4.1 and 6.4.2 is the foundation for the Use Case Specification of the Virtual Consultation, which can be seen in Table 7.

|                               |  |
|-------------------------------|--|
| <b>Name</b>                   | Virtual Consultation   |
| <b>Short Description</b>      | The use case describes the process of a remote consultation where two or more entities which are spatial separated communicate interactively with each other using the camera and microphone of the near-eye display device. This involves video as well as audio transmission in real-time.   |
| <b>Actors</b>                 | Doctor (advisee); One or more medical specialists (advisors); Patient (passive)  |
| <b>Preconditions</b>          | Network between endpoints is established.<br>Required server applications are running.<br>Near-eye display device is ready for use.  |
| <b>Trigger</b>                | During a medical process the doctor is confronted with a situation where the support from a medical specialist is needed.  |
| <b>Typical Process</b>        | <p>Arranging an appointment for a Virtual Consultation, for instance via telephone.</p> <p>At the beginning of the appointment:</p> <ul style="list-style-type: none"> <li>• Start required application on near-eye display device.</li> <li>• Doctor (advisee) wears the near-eye display device.</li> <li>• Consultant starts client application to establish the connection to the remote endpoint.</li> <li>• Bidirectional audio connection and one directional video connection (advisee to advisor) is used to discuss medical issues. <ul style="list-style-type: none"> <li>○ The advisee is using the near-eye display device to capture the video from first-person perspective.</li> </ul> </li> <li>• Pictures can be taken if needed from both parties. <ul style="list-style-type: none"> <li>○ Save pictures to local hard drive or save it in the HIS.</li> </ul> </li> <li>• When everything is settled, disconnect the client(s) and the near-eye display device until the next time the Virtual Consultation is needed.</li> </ul> |
| <b>Additional Constraints</b> | The connection outside of the hospital IT infrastructure should fulfill security constraints, such as confidentiality, integrity, non-repudiation and authenticity.  |

Table 7: Use Case Specification of the Virtual Consultation using the scheme proposed by Rupp et al. (2009).



#### **6.4.4.2 SOPHIST Requirement Templates**

To document the requirements the SOPHIST Requirement Template approach from Rupp et al. (2009) is used. The following table (see Table 8) lists all requirements related to the Virtual Consultations using the SOPHIST Requirement Template and unique object Ids. Requirements which are marked with an asterisk (\*) are adopted from (Sachs, 2014, pp. 69-71).

Because the requirements documentation step is before the actual system architecture begins, the requirements are formulated without suggesting any technical solution. This approach should keep all options open, so that the solution space isn't restricted during the following project phases.

| <b>Object Id</b> | <b>Requirement</b>  |
|------------------|---|
| VC-B-FR-01       | The near-eye display device <b>shall</b> provide only the advisee with the ability to hear the transferred audio.                                       |
| VC-B-FR-02       | The system <b>shall</b> provide the consultant with the ability to see the video from the advisee in first-person perspective.                          |
| VC-B-FR-03       | When the advisee is taking a picture the near-eye display device <b>shall</b> be able to zoom in and out.   |
| VC-B-FR-04       | When the advisee is taking a picture the near-eye display device <b>shall</b> be able to focus different parts of the scene.                            |
| VC-B-NR-05       | The system <b>shall</b> provide the consultant with the ability to watch the video stream from the advisee in a resolution of 640x480 pixels or higher. |
| VC-B-NR-06       | The system <b>shall</b> provide the consultant with the ability to watch the video stream from the advisee with 16 fps or more.                         |
| VC-B-NR-07       | The system <b>shall</b> be able to take pictures in a resolution of 1920x1080 pixels or higher.   |
| VC-B-NR-08       | Under real world conditions the system <b>shall</b> provide its services at least for 2 hours.  |
| VC-H-FR-09       | The near-eye display device <b>shall</b> provide the wearer of eyeglasses with the ability to use the device with all its functionalities.              |

|            |   |
|------------|---|
| VC-H-FR-10 | The OP-Staff <b>shall</b> be able to clean any hardware part of the near-eye display device using medical cleaning procedures.*   |
| VC-H-FR-11 | If dropped from a height of 1 meter onto solid ground, the near-eye display device <b>shall</b> continue functioning.*  |
| VC-N-NR-12 | Before a Virtual Consultation takes place the involved parties <b>shall</b> make an appointment for the Virtual Consultation.   |
| VC-N-NR-34 | When data has to be transferred over the internet the system should communicate over a <b>Virtual Private Network (VPN)</b> connection.   |
| VC-S-FR-13 | At any given point in time and if both parties are not in the same local network the system <b>shall</b> provide both parties with the ability to transfer audio to each other over the internet.                 |
| VC-S-FR-14 | At any given point in time and if both parties are not in the same local network the system <b>shall</b> provide the consultant with the ability to receive the video signal from the advisee over the internet.  |
| VC-S-FR-15 | At any given point in time and if both parties are in the same local network the system <b>shall</b> provide both parties with the ability to transfer audio to each other over the local network.                |
| VC-S-FR-16 | At any given point in time and if both parties are in the same local network the system <b>shall</b> provide the consultant with the ability to receive the video signal from the advisee over the local network. |
| VC-S-FR-18 | When data is transferred over the internet the system <b>shall</b> be able to provide a secure connection even if no VPN is available.  |
| VC-S-FR-19 | The system <b>should</b> provide the consultant with the ability to transfer the video signal from stationary cameras such as a webcam from the advisee.  |
| VC-S-FR-20 | At any given point in time the near-eye display device <b>shall</b> provide both parties with the ability to take pictures from the first-person perspective.   |
| VC-S-FR-21 | The system <b>should</b> provide external consultants with the ability to use the system in the same way as the consultant.   |
| VC-S-FR-22 | When the advisee is taking a picture the near-eye display device <b>shall</b> be able to display which parts of the scene will be on the picture.   |
| VC-S-FR-23 | The system <b>should</b> provide the consultant with the ability to use the provided functionality on an Android tablet or iOS tablet.  |
| VC-S-FR-24 | At any given point in time during a surgery the near-eye display device <b>shall</b> be able to display the vital parameters of the patient, in the FOV of the surgeon.   |
| VC-S-FR-25 | The system <b>should</b> provide both parties with the ability to save pictures in the hospital information system.   |

|            |   |
|------------|---|
| VC-S-NR-26 | When data is transferred over the internet, the system <b>shall</b> not use more than 8 Mbit/s up and download.   |
| VC-S-NR-27 | When data is transferred over the internet, the system <b>should</b> get along with 5 Mbit/s up and download.   |
| VC-S-NR-28 | If the near-eye display device loses the Wi-Fi connection the system <b>shall</b> reestablish the connection automatically.                                   |
| VC-S-NR-29 | If the connection gets reestablished and there was a video stream running before the connection was lost, the system <b>shall</b> continue the video stream.  |
| VC-S-NR-30 | If the connection gets reestablished and there was an audio stream running before the connection was lost, the system <b>shall</b> continue the audio stream. |
| VC-S-NR-31 | The system <b>shall</b> provide the consultant with the ability to use the provided functionality on a Windows 8 device.                                      |
| VC-S-NR-32 | The system <b>shall</b> provide the consultant with the ability to watch the video streams in full screen.  |
| VC-S-NR-33 | The system <b>shall</b> provide the consultant with the ability to view pictures in full screen.  |

*Table 8: Requirements for the Virtual Consultation using the SOPHIST Requirements Template and object Ids proposed from Rupp et al. (2009)*

### **6.4.5 Resulting Business Process**

Due to the newly gained information during the requirements elicitation, the business process from Figure 16 slightly changes. In contrast to the previous process, now the support request will lead to an appointment between the involved parties. If the specialist is on-call, this appointment will start immediately, but if not, the Virtual Consultation will take place at a later point in time. The updated BPMN diagram can be seen in Figure 17.

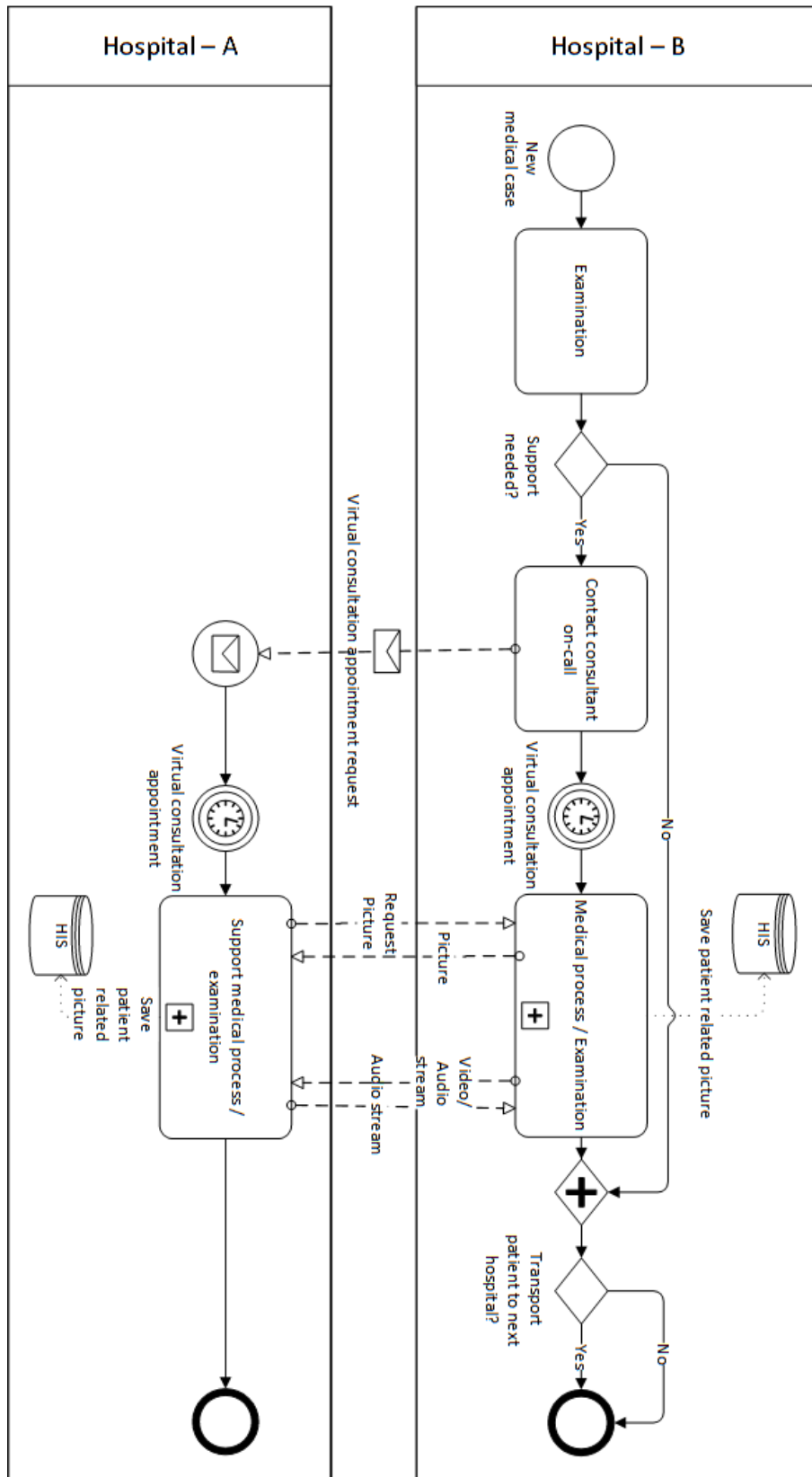


Figure 17: Overview of the updated Virtual Consultation process between the two investigated hospitals, shown as BPMN diagram.

#### **6.4.6 Virtual Consultations – Conclusion**

To review, the as-is analysis has shown that there is a strong need for an interactive communication between the two investigated hospitals (see section 6.4.1). Further interviews and conversations with the stakeholders revealed that video conferences with a first-person view is a promising approach.

The detected use case, with its requirements, will be validated using a prototype (see chapter 7, 8 and 9). Particularly interesting are some of the quality constraints such as video resolution, or whether a consultation is actually possible per video. As hardware basis Glass will be used, but the functionality can also be extrapolated and transferred to other HMD devices.

### **6.5 Subproject: Doctor's Visit**

Patients with a disease which requires an admission to the hospital are in a so called inpatient stay. The normal procedure foresees that if a new patient is admitted, she or he gets a hospital room allocated depending to the patient's disease. The term Doctor's Visit describes the procedure where medical staff (e.g. doctors and nurses) are visiting patients to get an impression of the person's recovery. During these visits the doctors discuss medical details such as pending medical examinations or results of examinations, or changes medications for patients. Normally such visits are performed at least once a day, typically in the morning. Nowadays, most hospitals are using IT systems to manage medical records including information about duration of the stay or what medical decisions were made etc. In short, all data needed for organizational and medical aspects of a hospital (see chapter 4). In general, these systems are subsumed with the term HIS and are heavily used during Doctor's Visits.

#### **6.5.1 As-Is Analysis and Identification**

This section describes the initial situation as well as the problems which were revealed during the as-is analysis process. As a technique to gather the initial situation, process observations and informal conversational interviews were chosen. The interviewed and observed stakeholders were mainly medical staff.

The field research took place during a Doctor's Visit in the wards. The HIS used by this hospital enables users to create electronic medical records or to manage appointments etc. Additionally the application shows for each patient four parameters which are relevant for the medical staff, called *special warnings*.

- The first parameter shows if the patient is a risk for the medical staff.
- A second value shows if the patient has any allergies.
- The third parameter displays if the patient has any implants.
- Finally, the fourth parameter shows the medical staff if the patient has any infections, which must not be automatically contagious.

The observed Doctor's Visit was done by two doctors and two nurses. One doctor mainly interacted with the patients by asking or answering questions. The second doctor assisted the first one and primarily interacted with the HIS application by entering or querying data. The nurses focused on different activities such as administration of the medications. The process behind the Doctor's Visit is done in a similar way for each patient, so there can be a pattern extracted which will be explained here in brief (see Figure 18).

Commonly the doctor first asks the patient about his or her subjective condition to get a quick overview. During this conversation the doctor asks some medical questions which are relevant in the current situation, in the meantime the assisting doctor selects the patient in the HIS application to get an overview of the patient's medical records. Next, both doctors are examining this data, e.g. reports of a medical examination which was done by the patient. Sometimes reports are missing, because they are currently not added to the HIS or by another reason. In such a case the assistant calls a secretary, which has the control over these reports, to inquire about the missing document. In some cases the document gets added right after the call, so the doctors can use this information. Afterwards, most likely the doctor informs the patient about the results of medical examinations - provided there was one. Then the next medical steps will be explained to the patient to ensure that she or he is sufficiently informed. If additional medical information is needed (e.g. the assistance from a specialist or a medical test) a request is created directly in the HIS program by the assistant. An interesting fact is that not all medical requests can be

handled promptly. For instance, the observed hospital doesn't employ its own dermatologist, therefore an external expert is visiting the hospital weekly. Consequently, the result of a request can take up to one week. Subsequently, all patient related information e.g., medication treatment or condition, will be written in the medical discourse to document the procedure, for internal medical, as well as, for legal reasons.

During the as-is analysis the doctors mentioned problems in the physical distribution of the medical information. During the visit the HIS application is only running on one notebook and if the doctors want to examine data, both have to gather in front of the screen, which is annoying particularly in small rooms. As solution a doctor proposed a table in addition to the notebook where it is possible to read the same medical report as the notebook currently shows.

A further problem was that the HIS application doesn't allow to view multiple medical reports at the same time. I.e., each document has to be closed before another can be opened. This restriction makes it hard to compare two different results with each other, or to conclude something through multiple documents. To get a better overview, the current process of the Doctor's Visit is outlined in Figure 18 with the use of a BPMN diagram (see section 2.3.3).

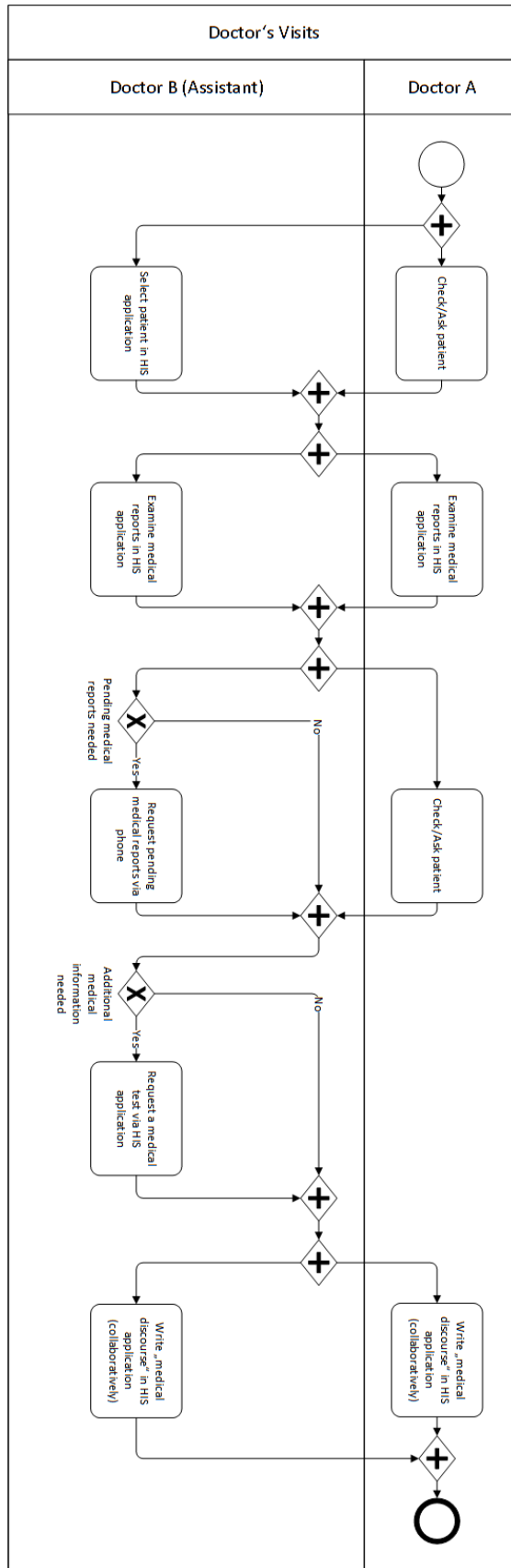


Figure 18: Overview of the current process of the Doctor's Visit in the investigated hospital, shown as BPMN diagram.



### **6.5.2 Potential for Improvement**

During the as-is analysis the medical staff already made several suggestions for improvements (see 6.5.1).

The two most relevant cases are listed below:

- The medical staff has seen a problem in the distribution of the information, provided by a notebook, between the two doctors who perform the visit. Both doctors have to gather behind the monitor to examine the medical records. In this case a near-eye display device could be used to stream the information from the notebook to the FOV of the second doctor. The problem with this application is that medical reports or images require a high resolution. Since currently available HMDs (e.g. Glass) have only limited resolution capabilities, it is difficult to display the information appropriately. In addition, doctors identified areas of improvement based on the fact that only one medical report can be shown at the same time in the HIS program. To accomplish displaying multiple documents on a HMD is even harder due to resolution restrictions.
- Another use case which can be detected, has to do with the four parameters called special warnings. As already explained in the as-is analysis these parameters indicate important information for the medical staff, e.g. if the patient has an implant. A HMD could be used to display these parameters to the doctor wearing the HMD before entering the room. But interviews with medical staff revealed that the benefit of this feature would be very small, because the doctors know these facts about their patients by heart.

To conclude, a near-eye display device would not provide any benefits for the use cases above. A reason therefore is the fact that both doctors have their hands free and can use it to interact with the environment. Consequently, devices such as a tablet or a notebook would be a better choice, because for information processing input techniques with a higher usability level such as keyboard and mouse can be used.

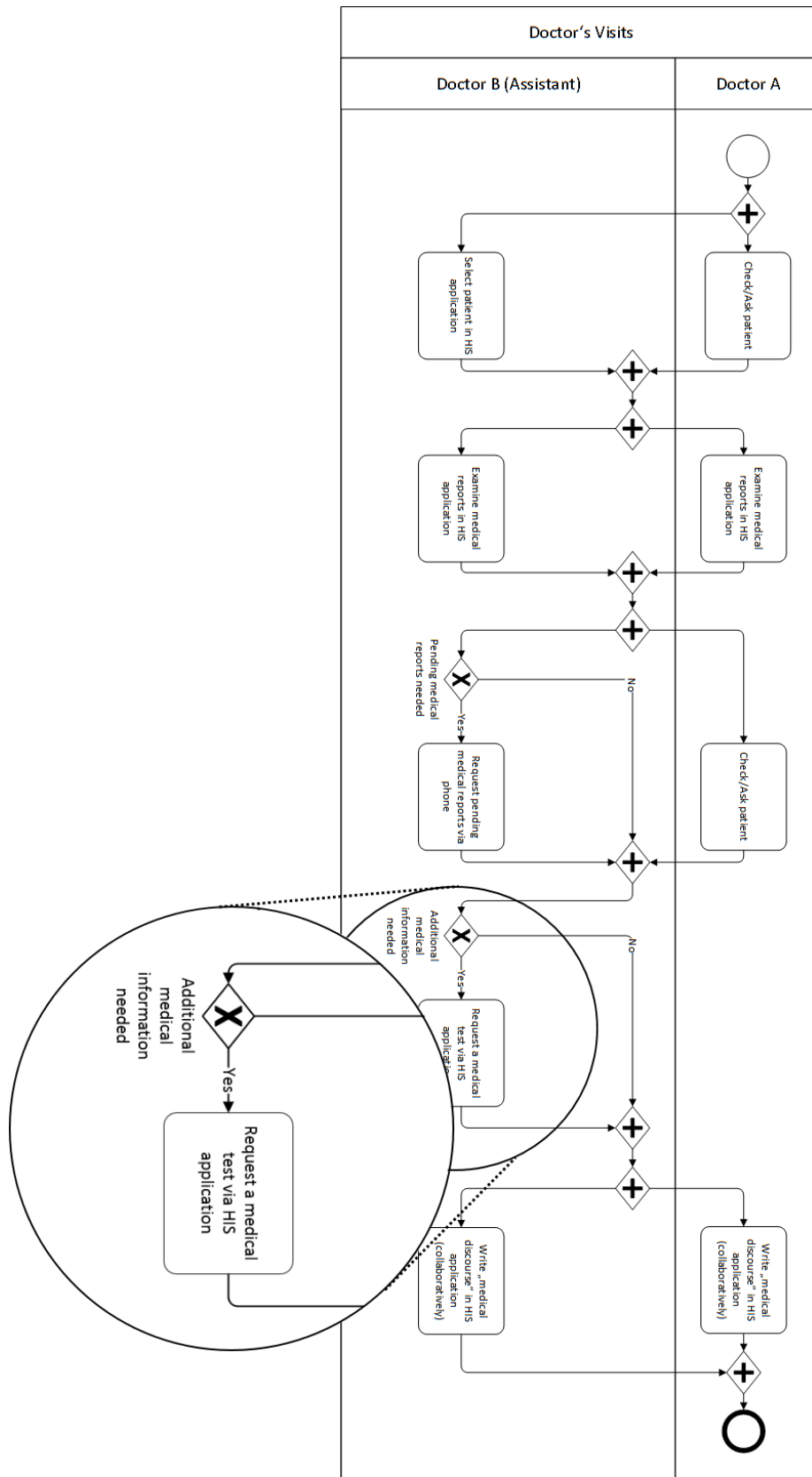


Figure 19: If during the visit additional medical information is needed, doctors are able to create a medical request in the HIS. But often just a quick look of a specialist would be sufficient to make a decision.

An interesting point during the Doctor's Visit is the situation where doctors request additional medical information from a specialist, see Figure 19. In the observed hospital a dermatologist is only available once a week. But the doctors often just need a brief response for a medical case, which can be done just by a quick look from an expert. The possibility of live video and audio conferences as well as simple picture transmissions to the medical specialist would provide this functionality. I.e. if during the Doctor's Visit a medical request from an external specialist is necessary, the doctors would first decide if this medical case should be inspected by an expert or not. In the former case, the doctors would arrange an appointment with the expert for this remote consultation.

This use case is equivalent to the Virtual Consultation discussed in section 6.4, so using a near-eye display device, with an included head mounted camera, for the interactive communication, could produce benefits for the involved parties. The advantages of this method are not described here in detail, as it is already explained in section 6.4.2. The same applies for the BPMN diagram which would only be a combination of Figure 17, the Virtual Consultation, in the yes-branch of Figure 19.

### **6.5.3 Doctor's Visit – Conclusion**

The as-is analysis in section 6.5.1 above revealed that the process of Doctor's Visit is fairly optimized and there is not much room for improvement. The use of a near-eye display device during the standard procedure of the Doctor's Visit would not provide any benefits for the medical staff. A reason therefore is the fact that both doctors have their hands free and can use it to interact with the environment.

Nonetheless, sometimes the help of external specialists is necessary which revealed the use case of the Virtual Consultation during the Doctor's Visit. This use case is obviously not strictly related to the process of the Doctor's Visit, but may also improve the efficiency and effectiveness of the procedure.

This subproject called Doctor's Visit does not provide any documented requirements because the only use case which was found during the as-is analysis was the Virtual Consultation which is already discussed in section 6.4 in detail.

In brief, the Doctor's Visit is a fairly optimized process, nevertheless an improvement, using a near-eye display device, could be achieved by the use of the Virtual Consultation to contact remote specialists.

## 7 Architecture and Design: Virtual Consultation Proof-of-Concept

---

This part of the thesis discusses architectural details as well as decisions which were made during the design of the proof-of-concept. The proof-of-concept of the Virtual Consultation is neither intended nor suited to be a finished product that can or should be used on a daily basis. As the name suggests it is an incomplete version of the final program to evaluate and to test the Virtual Consultation use case. Consequently, it will not implement all requirements listed in section 6.4.4.2. The focus for the proof-of-concept is to show and validate the advantages and disadvantages of using the Virtual Consultation in conjunction with a near-eye display device such as Glass, as well as to evaluate technical details which would be required if a marketable product should be developed.



*Figure 20: The Google Glass near-eye display device, which serves as development basis for this master's thesis (Torborg & Simpson, 2014)*

The hardware platform which serves as a development basis for this master's thesis is the Google Glass Explorer Edition (see Figure 20). This hardware is suitable for implementing a proof-of-concept, because it build on Android and offers a Java API. This includes debugging functionality, object-oriented code and an **Integrated Development Environment (IDE)**. For this reason, testing promising features or verifying requirements is easy and fast. A second design choice for applications which are not directly developed on Glass is the programming language C# with .Net

Framework 4.5. C# has major advantages compared to other languages, which will be discussed in detail in section 7.5.

## 7.1 System, Context and Environment of the Proof-of-Concept

An important step during the software engineering is the demarcation of the application with the rest of the environment. In other words to determine the scope of the system. At this phase of the project, the requirements are already available in structured form which makes it possible to determine the scope of the project quite precisely. The classification of system, context and environment is based upon (Rupp, 2009, pp. 72-76).

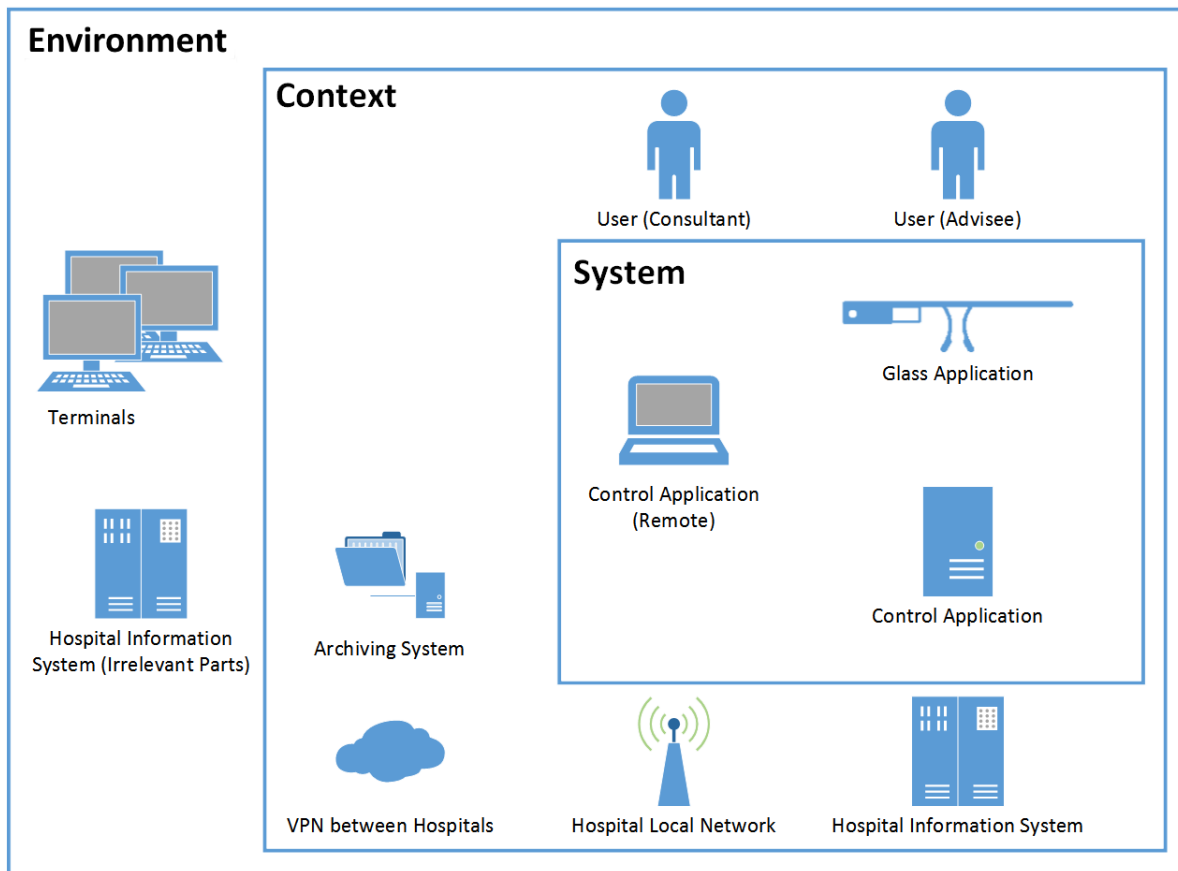


Figure 21: Environment including context and system in which the Virtual Consultation takes place, using a schematic representation proposed by (Rupp, 2009)

The requirements of section 6.4.4.2 show that the context in which the system takes place includes more than one hospital. This fact has of course a major impact on the software

architecture and thus on the scope of the system. In Figure 21 a schematic overview of the environment, the context and the system for the Virtual Consultation can be found, which will be briefly explained below:

### **System**

The system includes the application running on Glass and two instances of a Windows control application which are responsible for the communication between the hospitals. Furthermore, the functionality which allows users to store patient related pictures in the HIS, which are taken during the consultation. The system includes:

- Streaming video and audio signal from Glass or a stationary camera to the consultant in the remote hospital.
- Streaming audio signal from the consultant to Glass.
- Taking pictures with Glass by the user of the near-eye display device or the consultant.
- Saving pictures in the HIS (using the archiving system).

### **Context**

The context includes everything which is in direct contact with the system. This includes the network in which all applications are connected with each other. This contains the local network of the hospitals as well as the part of the internet that is used to establish and perform the communication. As requirement VC-N-NR-34 shows, for the communication over the internet a VPN connection can be used. Furthermore, the context includes the HIS to store patient related pictures, as well as the direct involved medical staff (primarily the consultant and the doctor who needs support). The context includes:

- The local network of the hospital including the VPN which connects both hospitals.
- The part of the HIS which is needed to store patient related pictures.
- The staff who is involved in the Virtual Consultation. Mainly the consultant and the advisee.

## **Environment**

The environment includes all entities which do not directly influence the system. In the case of the proof-of-concept this contains all computer terminals which are not directly prepared for the Virtual Consultation as well as users outside the scope of the use case. The environment includes:

- Other computer terminals which are not directly included in the process.
- Parts of the HIS which are not needed for the functionality of the system.

## **7.2 Communication flow**

The Virtual Consultation is a streaming application which connects two doctors from different hospitals with each other (see Figure 22). The system streams audio and video from Glass to a server application (Control Application). The server application streams the incoming data to a client program (Control Application – Remote) in the remote hospital where the video and audio signal is used from the consultant. The client from the consultant on the other hand streams the audio signal, captured by the microphone, back to the server application, which in turn streams the data to Glass. Furthermore, the consultant is able to take pictures from the view point of the doctor who is wearing Glass. In addition these pictures can be saved in the HIS.



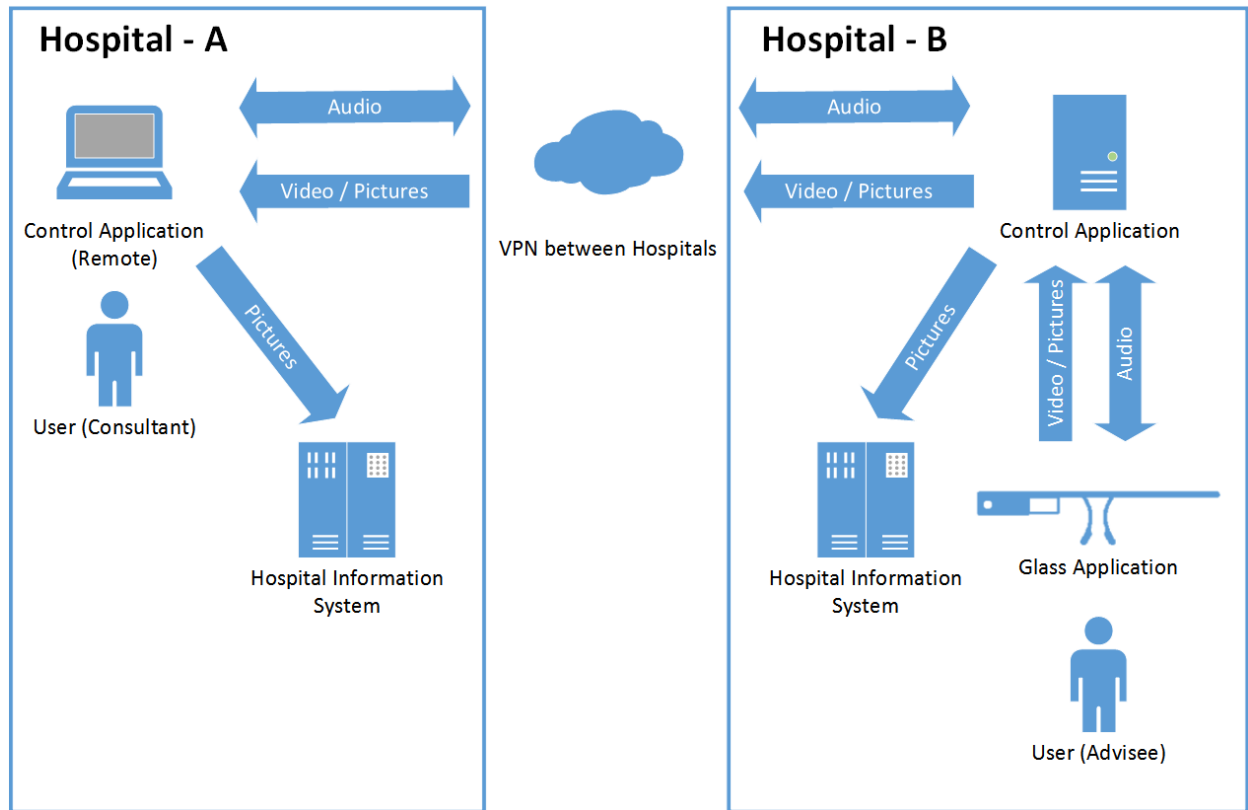


Figure 22: The communication flow between two hospitals.

### 7.3 Related Work

As already mentioned in the chapters above, this master's thesis is part of a research project and hence there was already work done. A proof-of-concept with basic functionality such as sending image data to Glass or receiving image data from the device as well as sending and receiving audio was already implemented by Sachs M. (Sachs, 2014, p. 1 ff.). The application itself was separated into two fundamental parts, a server and a thin client, called *Glass Server* and *Glass Client*. This client-server architecture with the thin client was chosen because the hardware basis of Glass doesn't have the same computing power than a traditional personal computer. Therefore, the device is not suitable for computationally intensive tasks. Furthermore, the Google Glass Explorer Edition is a preliminary version of the finished product, so changes of software and hardware can and will happen by Google. For these reasons, a thin client architecture for the Google device, written in Java, was a reasonable choice. The Glass Server is a .Net Windows application which is written in C#. It communicates with the Glass Client over the Wi-Fi network via TCP and UDP.

Additionally, it is an application with a user interface programmed in **Windows Presentation Foundation (WPF)** and thus it is developed with the **Model View ViewModel (MVVM)** software design pattern, which is the natural way to use WPF. Moreover, the application serves as a control center for the near-eye display device. For instance, it is possible to start and stop video capturing of Glass, remote. The communication between the applications was designed with custom data types which are shared between the two applications. To illustrate:

- Control message data types for Glass. For instance, a start-video-capturing message or a message to shut down the device.
- Payload message data types which are used to transfer data between the applications. For instance, image data or audio data.

Despite that, the main purpose of the application was to stream vital parameters of patients to Glass. (Sachs, 2014, pp. 73-92)

## **7.4 Requirements for the Proof-of-Concept**

This section will summarize the requirements which are important for the proof-of-concept from section 6.4.4 above. The overarching objective is to evaluate and ensure the benefit of the Virtual Consultation. So the main question is, if this use case could improve and support the medical processes. To evaluate this, the proof-of-concept must implement the main requirements which were gathered during the requirements elicitation phase.

### **Mandatory functionality**

There are essential functionalities which must be implemented by the proof-of-concept to accomplish the Virtual Consultation use case. The most important task is that the consultant sees the first-person perspective of the doctor who receives support (VC-B-FR-02). Similar important is that the doctor who gets support hears the audio signal from the consultant (VC-B-FR-01). These requirements lead to the basic behavior of the system which is responsible for the transport of image and audio signals between two endpoints (VC-S-FR-13, VC-S-FR-14, VC-S-FR-15 and VC-S-FR-16). Furthermore, the proof-of-concept shall enable both parties to take pictures from the

first-person perspective of the doctor who gets support (VC-S-FR-20) and subsequently it should be possible to store those pictures related to the patient in the HIS (VC-S-FR-25).

### **Exploratory features**

Certain functionalities must be evaluated during the field tests. Such functionalities are the minimal video and picture resolution (VC-B-NR-05, VC-B-NR-07) as well as the minimal frame rate of the video (VC-B-NR-06). Those parameters must be validated to ensure that the quality demands of the medical staff are satisfied. A further important point is the duration of the use of the system, because this influences the possible hardware choices, regarding to e.g. battery life or overheating problems (VC-B-NR-08). Additionally, there are some technical questions which affect the possible transmission techniques with regard to their use of network bandwidth (VC-S-NR-26, VC-S-NR-27).

### **Usability**

Some usability requirements are important for the proof-of-concept, because if they will not be implemented they would interfere with the field tests. For instance, if Glass loses its Wi-Fi connection during a test and the device can't reestablish it by its own, the test has to be interrupted which can have a direct influence on the test result (VC-S-NR-28). The same applies for the requirements VC-S-NR-29 and VC-S-NR-30 which should guarantee that the medical process is interrupted as less as possible by technical issues. Obviously, the proof-of-concept must not implement every possible technical detail which could lead to a problem in the real world, but at least it must ensure that the field tests can be performed under normal conditions. Furthermore, the system shall enable the user to view image data, such as pictures or videos, in full screen (VC-S-NR-32) and (VC-S-NR-33), which is important for the verification of the image quality. Another requirement which will be met by the proof-of-concept is the condition that the system shall work on Windows 8 systems (VC-S-NR-31). Since the application serves only as proof-of-concept it is implemented on a single platform and not on additional operating systems such as iOS or Android.

### **Optional features**

A feature which would be nice to have is to transfer an additional video signal from a stationary camera to the consultant (VC-S-FR-19). This functionality could be used to compare the pros and

cons from a first-person camera to a stationary camera. A similar experiment was already conducted by Fussell et al. (Fussell, et al., 2003, p. 513 ff.).

## **7.5 Choice of Technology**

This section discusses technical choices behind the programming language and the framework for the **Graphical User Interfaces (GUI)**. The software for the proof-of-concept has to support two platforms, first Glass / Android platform and second the Microsoft Windows environment.

### **7.5.1 Technology Choices for Google Glass**

Glass is used as hardware platform for the proof-of-concept, which has some impact on the possible technology choices such as the programming languages. Glass works with a modified Android operating system and thus it supports similar IDEs or programming languages. There are multiple ways to build programs for Glass:

#### **The Glass Development Kit (GDK)**

The GDK enables developers to build applications, called Glassware, that run directly on the device. This is necessary if the application should support real-time interaction, offline functionalities or hardware access. (Google (c), 2014)

#### **The Mirror API**

The Mirror API enables developers to asynchronously pull and push data from and to Glass. Furthermore, it is web-based and the developer can choose his or her favorite programming language. This technique does most of the development work on its own and it should be used if platform independence, a common infrastructure or build-in functionalities are key factors for the application. (Google (c), 2014)

#### **Hybrid Glassware**

The Hybrid Glassware allows the developer to invoke GDK functionalities through the Mirror API. (Google (c), 2014)

For the proof-of-concept, access to the Glass hardware as well as real-time functionalities are needed. This limits the technology choices to the GDK approach. The main possible programming languages for the GDK option are Java or, for native applications, Object Pascal and C++. Java has some advantages to the native language C++ or Object Pascal. For instance, there is an automatic memory management, build-in security e.g. boundaries for accessing arrays and it is easier and faster to develop applications. Further, the existing application already uses Java as development basis. Therefore the Glass Development Kit in conjunction with Java was the selected choice for the proof-of-concept.

### **7.5.2 Technology Choices for Windows Application**

There are numerous possibilities to develop applications on Microsoft Windows, only to enumerate all of those options is already beyond the scope of this master's thesis. Therefore three commonly used methods will be considered:

- C# in conjunction with WPF for the user interface
- Java in conjunction with Swing for the user interface
- C++ in conjunction with Qt for the user interface

Those three choices will be compared with each other with respect to the following requirements:

1. Easy to use / develop: The technology shall provide a fast and easy way to build the user interfaces.
2. Performance: The technology shall be able to provide the user a fluent interaction with the user interface.
3. Availability of multimedia frameworks: Multimedia frameworks, for instance, to compress and decompress video and audio signals, shall be available for the technology.

#### **C#**

C-Sharp is a programming language which has significantly gained importance in the last decade. First, the language supports garbage collection, so the developers don't need to put much

attention on problems such as memory leaks. Second, it is pure object oriented code, not like C++ which mixes object orientation with procedural code. Third, it is type-safe which prevents the developers from problems during the conversion between data types. Finally, one of the biggest advantages is the .Net framework library behind. It allows fast and easy development for things such as user interfaces, network communication, working with data structures etc. This is only a short overview of some of the advantages, of course there are many more.

An important fact about C-Sharp is that it is mostly used within runtime-aware compilers. Those compilers compile an **Intermediate Language (IL)**, which is generated out of the C-Sharp code, in native code within a managed execution environment, which prevents the system from damage caused by the code. (Microsoft (c), 2015)

This overhead has some impact on the performance, but for the use case of this master's thesis the performance loss of a managed language should not have any drawbacks.

On the other hand, the available multimedia frameworks for C-Sharp are fewer than in C++. In order to compensate this disadvantage, in C-Sharp it is possible to wrap native e.g. C++ code to access functionalities which are otherwise not available in C#. But this causes additional development work that should not be underestimated.

WPF is a graphic framework which works perfectly together with C# and allows separating the user interface from the business logic. The language which is used to write WPF interfaces is called **Extensible Application Markup Language (XAML)**. In brief, WPF is a very elegant way to develop modern looking user interfaces for windows systems. One of the few disadvantages of WPF are the hardware requirements. User interfaces build in WPF consume in the most cases more memory than e.g. Qt interfaces.

## **Java**

Java is a well-known programming language which exists since the mid-1990s. It is very similar to C#. For instance, it supports garbage collections, has pure object oriented code, is type-safe and has a big class library behind, which allows fast and easy development of applications. Java applications are typically compiled to an architecture independent bytecode which can be

executed by the Java virtual machine. This means the same code can be executed on different systems without recompiling.

However, Java shares the same weakness, with regard to the availability of multimedia frameworks, as C-Sharp. Consequently, Java doesn't have any significant benefits concerning multimedia frameworks.

Swing is a graphic framework for Java to develop graphical user interfaces. It exists since the 1990s and allows similar functionalities such as WPF. A big advantage of Swing is, it can be used on multiple platforms without changing or recompiling the code. In comparison to WPF, it is more difficult to get a Windows like look and feel in a Swing application.

## **C++**

C-plus-plus allows the programmer to develop an object oriented code with low level memory access for high performance applications. Therefore C++ should be used in scenarios where real-time response is needed for computationally expensive tasks. For instance, tasks such as multimedia processing. For that reason most multimedia frameworks and software libraries are written in C++ or C. The problem with C++ is that the developers have a lot of responsibilities regarding to the usage of the programming language. C++ forces the programmers to know about underlying structures and don't hide much of the system's complexity. For example, there is no garbage collection or automatic array boundary checking. This means the programmers have to take care of things such as memory management. As a result developing C++ applications is more time consuming and error prone than e.g. in Java. The advantage is of course a better performance of the application, provided that everything is used in the right way.

As already mentioned most multimedia frameworks are developed in C++ or C. Consequently, C++ provides a wide choice between different libraries and frameworks, which is a major benefit.

Qt is a graphic framework which is available for a lot of languages, such as C++, Java or C#. Very popular is the library for user interfaces developed in C++. The library provides inter alia cross platform support, but unlike Java and Swing the code has to be recompiled for each platform.

| Technology \ Criterion                | C#/WPF | Java/Swing | C++/Qt |
|---------------------------------------|--------|------------|--------|
| Easy to use / develop                 | ●      | ●          | ○      |
| Performance                           | ●      | ●          | ●      |
| Availability of multimedia frameworks | ◐      | ◐          | ●      |

Table 9: Comparison of different technologies for the proof-of-concept

Table 9 compares the three discussed solutions. The symbols indicate the degree of satisfaction. The symbol ● represents that the corresponding criteria is fully satisfied. The symbol ◐ represents that the corresponding criteria is partly satisfied and the symbol ○ indicates that the criteria is not satisfied. Table 9 shows, C# with WPF and Java with Swing end up tied. Since the proof-of-concept is only used in Windows the choice for the technology was C# compared with WPF. This allows the usage of MVVM to separate the logic from the view. Furthermore, wrappers enable the use of C++ multimedia libraries in C#, which guarantee a better performance than a pure C# framework. For a marketable product the use of C++ would also be a good choice in case the development of wrappers should be avoided.

## 7.6 Software Design and Architecture

This section describes the design and architecture choices behind the proof-of-concept and why these choices were made. In section 6.4.4.2 a list of requirements for the Virtual Consultation can be found, these requirements as well as advantages and disadvantages should be examined by the proof-of-concept. As a result of these requirements and the use case itself, the software architecture of the proof-of-concept must fulfill some design constraints:

1. Extensibility and flexibility: The architecture shall be extensible and flexible to accommodate changes for testing and validating requirements. For instance, it shall be possible to compare different technologies with each other.
2. Reuse existing software: The architecture shall reuse existing software as good as possible to keep the development costs within a limit.



3. Usable for final product: The architecture shall be usable for a final product not just for a prototype or proof-of-concept.
4. Independence of technology: The architecture shall be independent from the technology in which it is implemented.

Furthermore, the system is distributed across different hospitals, this implies a communication over the internet and of course this influences the software architecture.

First, if the application has to communicate over the internet, this could be a problem if a large amount of data has to be transferred or if a low latency is needed. Especially in the case of the Virtual Consultation, large video and audio data has to be transferred in real-time. This implies the usage of video and audio compression techniques to reduce the amount of data which has to be transferred.

Second, there are security constraints for transferring medical data over the internet. But this issue gets solved by using a secure VPN connection between the hospitals. Hospital intern, there is secure network available.

Third, there is more than one application instance involved in the communication. For example, the Glass client already needs a server application in order to work properly. The communication between the applications could be realized via **Windows Communication Foundation (WCF)** or pure socket communication. As already mentioned in the introduction of this chapter, this application is a proof-of-concept which is included in a research project and therefore there are some other requirements besides those from the hospitals. Specifically, it is required that the proof-of-concept consists of as less as possible different applications, for the reason that the configuration and installation is as easy as possible. This adds a further design constraint:

5. Compactness of the application: The proof-of-concept shall consist of as less as possible different applications.

Nevertheless, the program should be well structured and reusable for further projects.

### **7.6.1 Core Component Architectural Software Design: Sinks and Sources**

As already mentioned in section 7.3 the research project in which this master's thesis is placed, has already working software which should be reused as good as possible. The existing software uses a client-server architecture with a thin client on Glass. This makes sense because of the reasons which are already formulated in section 7.3. The client application located on Glass supports custom types for communication. These types are related to the task which they should accomplish. For example, there is a data type for transferring images, or a type to tell the device to take a picture. This is a reusable and flexible design and it makes it easy to integrate the client application to a new server program. Considering these advantages it can be concluded that reusing most parts of the thin client for the Virtual Consultation should have a positive impact on the project.

The existing server application was implemented with a MVVM software design pattern using WPF for the user interface. The software design was strictly adjusted to the functionality which the system should accomplish. Extensions which are needed for the Virtual Consultation could be added to the existing Code, but this would decrease the maintainability of the code and thus cause problems. For that reason the software architecture for the server should be carefully chosen, with respect to the design constraints from the introduction of 7.6.

For the possible software architecture there are four designs considered in the architecture phase of the proof-of-concept:

#### **Add the Virtual Consultation to the current design**

The main advantages of this concept are that it is possible to reuse almost 100% of the existing code, which matches design constraint 2 from the introduction of 7.6. Furthermore, it is not necessary to develop or seek for a design pattern, which saves time in the design phase. Moreover, it is possible to immediately show running functionality without writing one line of code.

There are numerous disadvantages in this concept. For instance, using MVVM for the whole functionality of the application isn't a real software architecture. MVVM is mainly used in conjunction with a GUI. Moreover, in the current application the majority of the code is located

in one ViewModel. Extending this ViewModel with the functions needed for the Virtual Consultation, will not have a positive influence to the transparency of the code. Next, if a further feature should be implemented the main part of the application, the ViewModel, has to be touched. Which influences the existing code, even if it is not directly related to the new feature. Also, the MVVM pattern is not completely independent of the technology. For example, WPF is designed to work together with MVVM, but if e.g. Windows Forms is the chosen technology for the GUI, the usage of this design pattern isn't that intuitive, simply because Windows Forms isn't designed for MVVM. In view of the major disadvantages, this concept will not be taken into account.

### **Using the Broker Pattern**

The broker pattern is designed for distributed systems with decoupled heterogeneous components which communicate via remote services with each other, using a broker to coordinate the communication (Buschmann, et al., 2007, pp. 99-122). For the Virtual Consultation uses case, this would mean that servers and clients have to register itself in a central application called broker. Hence the individual applications don't have to know where its communication partner is located. Furthermore, this would allow heterogeneous applications to communicate with each other, e.g. an application which supports HL7 functionalities with applications which don't. This software architecture would be a promising approach for a marketable product, but would lead in multiple programs which violate the fifth design constrained. Furthermore, this architecture is complex to implement and the benefits, for its purpose as a proof-of-concept, are not sufficient.

### **Using the Pipes and Filters Pattern**

The pipes and filters architectural pattern is designed to process or modify data streams. The pattern can also be seen as a pipeline which consists of four types of components, the filter, the pipe, a source and a sink. The filter is the processing unit, it manipulates the data of the stream, whereas a pipe connects filters with each other. Furthermore, the source represents the input and the sink represents the output of the pipeline. (Buschmann, et al., 1996, pp. 35-45)

With that in mind, the functionalities of the Virtual Consultations could be encapsulated in filters and pipes. For example, a camera would be a source which generates image data. The first pipe

transfers the data to a filter which compresses / encodes the video stream. The second pipe transfers the stream over the network to a filter which decompresses / decodes the data, before the last pipe delivers it to a sink which renders the video. This design is very extensible and flexible. Because, if for example, different video encodings should be tested, each of those would be represented by its own filter. This makes it possible to compare different technologies with each other. Additionally, parts of the code which are used for the communication with Glass could be reused by outsourcing it to a sink / source.

On the contrary, a separation in filters and pipes is not always possible, because the transfer and encoding of video frames is often inseparable connected in one framework component. For the most part such frameworks can easily be represented as sinks and sources. This means the components are black boxes which consume or provide data, but a later modification of the framework to separate the transfer from the processing (encoding or decoding) is not always possible. Hence, if a clear division in filters and pipes should be accomplished a reimplementa-tion of large parts of a multimedia framework may be necessary, which in turn restricts the usability of the architecture for a final product.

### **Using data sinks and data sources**

Data sinks and sources is a common software architecture which is part of the UNIX interface design patterns (Raymond, 2003, pp. 305-306) (Buschmann, et al., 2007, pp. 236-238). For example, it is often applied in video frameworks (FFmpeg (a), 2014), (Microsoft (b), 2014) or in conjunction with buffers and streams. For the use case of the Virtual Consultation the video and audio signal has to be streamed from Glass or a stationary camera to a server application and from the server application to a client which views the stream. In the other direction the audio signal has to be streamed from the microphone of the client to the server application and from the server application to Glass. Each of these steps can be represented by a data source or sink. For instance, a streaming endpoint which publishes a video stream would be a video-sink, because it consumes a video signal which will be published afterwards. A webcam which captures a video signal to provide it for the streaming server can be represented by a video source. The same, of course, works for audio data and pictures. Another advantage is that different streaming clients and streaming servers can be used in the same application without destroying the software

architecture. Because every streaming client and streaming server, no matter what technology it uses, implements the same (sink or source) interface. This encapsulates the code and increases the reusability of it in other projects. Additionally, parts of the code which are used for the communication with Glass could be reused by wrapping it in a source and sink interface. Furthermore, this design is independent of the technology, as long as the used programming language supports interfaces or a similar concept.

| Architecture \ Criterion       | Broker | Pipes and filters | Data sinks and data sources |
|--------------------------------|--------|-------------------|-----------------------------|
| Extensibility and flexibility  | ●      | ●                 | ●                           |
| Reuse existing software        | ◐      | ◐                 | ◐                           |
| Usable for final product       | ●      | ○                 | ◐                           |
| Independence of technology     | ●      | ●                 | ●                           |
| Compactness of the application | ○      | ●                 | ●                           |

Table 10: Comparison of different software architectures for the proof-of-concept.

Table 10 compares the three essential solutions. The symbols indicate the degree of satisfaction. The symbol ● represents that the corresponding criteria is fully satisfied. The symbol ◐ represents that the corresponding criteria is partly satisfied and the symbol ○ indicates that the criteria is not satisfied. Table 10 shows, that the solution using data sinks and data sources is the best choice for the proof-of-concept.

### 7.6.2 Model View ViewModel Architectural Software Pattern

The MVVM pattern was published from Microsoft by (Gossman, 2005). It is almost identical to the **Presentation Model (PM)** pattern from Martin Fowler which was introduced 2004 (Fowler, 2004). The difference between PM and MVVM is that PM has a platform independent design, while MVVM is designed to be a standardized way to use WPF (Smith, 2009).

The MVVM pattern separates the GUI development from the business logic or back end. There are three components: the Model, the View and the ViewModel.

- The *Model* is the data or business logic which is completely independent from the View.
- The *ViewModel* on the other hand is the “Model of the View”. It contains the display logic and represents a converter between the View and the Model.
- The term *View* describes all elements of the GUI; for instance, buttons, windows and textboxes etc.

The communication between the View and the ViewModel is achieved by *data bindings* in both directions and by *commands* from the View to the ViewModel. An example for a data binding is the selection of a list control that is bound to a property in the ViewModel. Every time the user changes the selection in the control, the corresponding property is also updated with the new item. A command on the other hand is bound to a View event. For instance, a click event, which calls the ViewModel after the event is fired. The ViewModel is the layer between the Model and the View and, as a result, it interacts with the Model by transforming data between these two. Furthermore, it is possible that the Model notifies the ViewModel which notifies the View, for example, to update a data field. The relations between the View, the Model and the ViewModel can be seen in Figure 23.

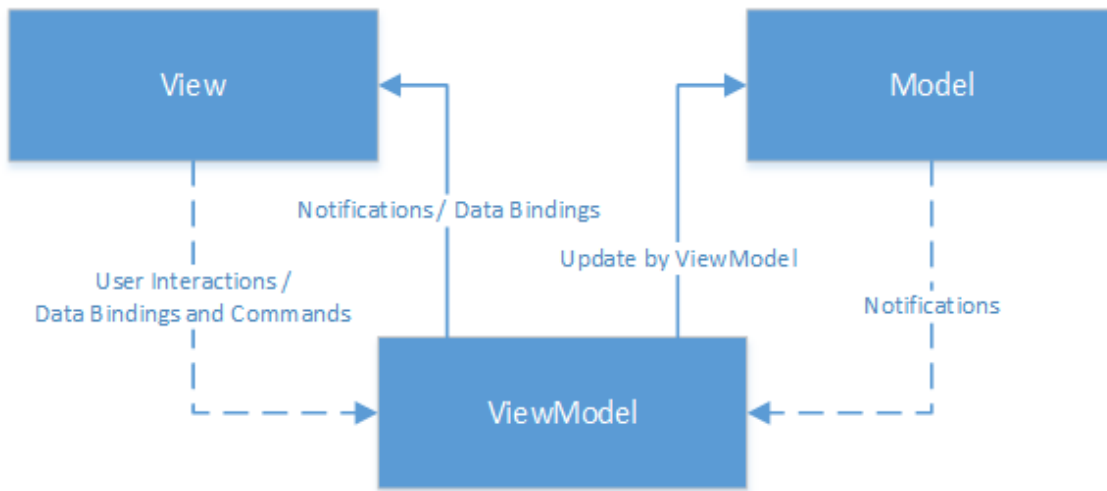


Figure 23: The MVVM pattern introduced by (Gossman, 2005), showing the relations between Model, View and ViewModel.

### 7.6.3 The Final Software Design

To conclude, the software architecture which is chosen for the proof-of-concept is based on a generalization of the different streaming functionalities in data sinks and data sources. Key features are:

- **Extensibility and flexibility:** Functionalities can be implemented without touching existing code. For such a functionality a new class must be created which implements the interface of a source or a sink. Thereafter the new code has to be implemented in the new class which is independent of the other sinks and sources. The interfaces guarantee seamless communication with the rest of the application. As a result of this architecture, there is no difference between streaming a video from a webcam or from Glass, because they share the same interface. The part where the video signal is captured is independent of the part where the video signal is streamed / consumed. The same applies for the audio signal as well as for pictures.
- **Reuse existing software:** The important communication part with Glass can simply be integrated in this concept.
- **Usable for final product:** In a final product the same architecture can be used.
- **Independence of technology:** The concept is independent of a special technology, this implies the freedom of choice for the programming language as well as for other technologies, such as GUI frameworks.
- **Compactness of the application:** There is no extra implementation of a client application or a server application needed. Both share the same code but use different sinks and sources. For instance, the application which streams video and audio to the consultant in the remote hospital, uses as video and audio source the Glass and as sink some kind of streaming protocol. The consulting doctor on the other hand consumes the data with a source and streams the audio signal in the same way back. It can be seen from this that both applications can use the same code basis, respectively the same application if the user interface can handle the described configuration (see Figure 22).

Behind the implementations of the sources and sinks are further concepts and architectures which are hidden by the interfaces. For example, to transfer an audio signal from one application to another a TCP socket connection can be used. This requires a server component which accepts incoming connections and a client component which tries to connect. Moreover, there may be some higher level protocols involved other than just a pure TCP connection, which adds additional complexity to the code. Furthermore, to connect a source with one or more sinks another interface is needed, which is called Link. This interface allows the communication between sources and sinks and serves as mediator for data between those. The full UML Class Diagram of the sources and sinks architecture can be seen in Figure 24. Additionally, for the Windows WPF application the MVVM pattern is the best choice to guarantee a clean and serviceable code.

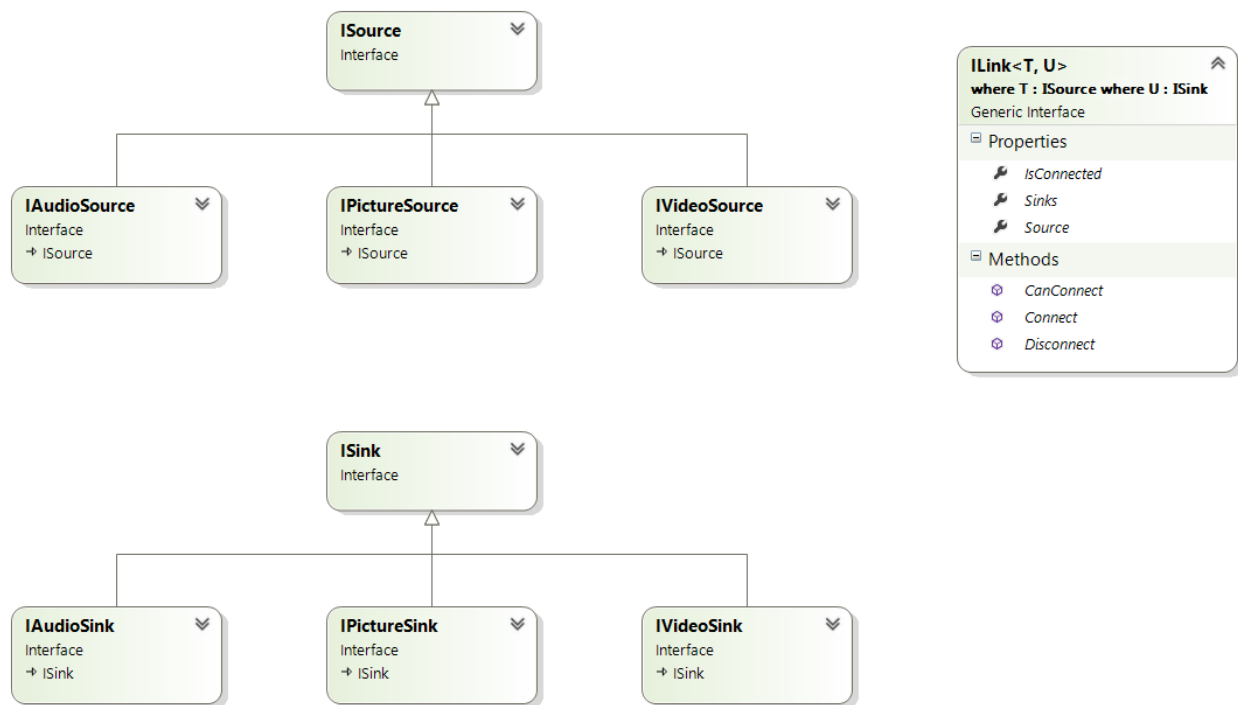


Figure 24: UML Class Diagram of the sources and sinks architecture.



## 8 Implementation: Virtual Consultation Proof-of-Concept

---

This chapter discusses implementation details which are not directly related to the architecture of the system. This includes a description of the used frameworks and an overview of the functionalities of the proof-of-concept and its components.

### 8.1 Excursus: Video and Audio Compression

In 1948 Shannon E. Claude introduced the fundamentals of information theory in his paper “A Mathematical Theory of Communication”, which is the basis of modern compression techniques (Shannon, 1948, p. 379 ff.). A particularly interesting fact is that the more redundancy data has, the more it can be compressed. This raises the question of whether there is a data compression limit. Shannon proved that there exists a limit for the maximum / best lossless data compression, called *Shannon limit*. As a consequence, the target of a lossless compression algorithm is to get as close as possible to that limit. Complementary to this, lossy compression algorithm, or in other words lossy codecs can discard information until a specified compression ratio is achieved. A very common technique for lossy image and audio compression is the **Discrete Cosine Transformation (DCT)** which is similar to the discrete Fourier transform. It transforms a group of pixels into several coefficients which are compressed in further steps. DCT is used e.g. in the JPEG image compression algorithm. A very simple approach for a video compression algorithm is **Motion JPEG (MJPEG)**. It simply uses the JPEG image compression algorithm for each video frame independently. But between video frames there is much redundancy. I.e. the most parts of a video frame look similar to the frame before. So, to get *interframe* compression this redundancy has to be taken into account. (Waggoner, 2010, pp. 35-51)

Well known terms of video compression are I-frames, P-frames and B-frames which will be outlined here in brief. An intra-coded frame or I-frame is not based on other frames, this means the compression of this frame is similar to the above mentioned JPEG algorithm. A P-frame on the other hand is predicted from a previous frame. It contains the difference between a current block of image data and the previous one. For instance, the pixels which have changed since the

previous frame. But videos often contain motion where e.g. every pixel in the image moves a view millimeters to the right. In such a case, P-frames also include motion vectors. Finally, a B-frame can be based on the next frame and the previous one, which yields in better efficiency. These basics are used for instance in MPEG-1. Modern codes such as H.264 are much more complicated and have many more features but are based on these principles. (Waggoner, 2010, pp. 51-58)

Audio compression works similar to video compression, it removes redundant information and additionally, information which cannot be heard by a human. Further compressions can be achieved with stereo audio. The difference between the two audio channels is in most cases small and thus it would be inefficient to encode those independently. A further interesting aspect of audio compression is, most audio codecs, unlike video codecs, use a constant bitrate. (Waggoner, 2010, pp. 58-60)

Particularly interesting for this thesis are codecs which are used to compress voice signals, for example:

G.722 is a broadband voice codec with a bandwidth up to 7 kHz. Depending on the used mode the codec provides a constant compression of 64, 56 or 48 kb/s. (Karapantazis & Pavlidou, 2009, p. 2057)

Speex is an open source and patent free broadband audio codec. It supports three different bandwidths 8, 16 and 32 kHz and provides a dynamic compression from 2.2 to 44 kb/s. (Karapantazis & Pavlidou, 2009, p. 2057)

iLBC is a royalty-free not open source narrowband codec which is used in applications such as Skype or Google Talk. It provides a compression of 15.2 kb/s for 20 ms blocks and 13.33 kb/s for 30 ms. Furthermore, the encoded data block is independent of the previous one which makes the algorithm very robust against lost packets. (Karapantazis & Pavlidou, 2009, pp. 2056, 2057)

## **8.2 General Overview**

Real-time video and audio streaming through the internet is common these days and yet it is not an easy task to implement. There are numerous ways to realize video and audio streaming. For

this thesis multiple approaches are implemented to check and compare their suitability for the Virtual Consultation use case. There are basically two challenging parts for the streaming task:

- The compression of the data
- The transfer of the data

For the video and audio compression there are several techniques also called *codec's*, which are in simple terms programs for encoding and decoding video or audio data (see section 8.1). For this master's thesis two different methods for the video compression are used. The first is one of the easiest approaches, called MJPEG. MJPEG is a video compression technique which is unaffected of the motion of the video. The second approach uses the H.264 (International Telecommunication Union, 2014) video compression format that is one of the most used formats these days. This method is relatively complex and an implementation, specifically for this proof-of-concept would be far beyond the scope of this thesis. For these reasons a framework with this capability is included, called FFmpeg. Equally important, for the audio compression the G.722 (International Telecommunication Union, 2012, p. 1 ff.) codec is used, which is designed for voice over IP applications and is contained in the NAudio framework.

To transfer the data over the network several protocols are involved, which are located at different levels of the network layers. For instance, low level UDP or higher level HTTP or a protocol especially designed for real-time streaming data, called RTP/RTSP (see section 3.2). To avoid re-inventing the wheel, frameworks are used to accomplish the data transfer over the network. For the widely-used protocols, such as HTTP or TCP the .Net framework provides the required functions, but for the less common protocols such as RTP/RTSP special software libraries are needed. Depending on the video compression technique, FFmpeg or Managed Media Aggregation is the used framework.

### **8.2.1 Frameworks used in the Proof-of-Concept**

For the proof-of-concept different frameworks are used to simplify the task of video and audio processing as well as data transfer. Since the .Net framework is part of the C# development with Visual Studio under Windows, this framework will not be explicitly listed.

### ***8.2.1.1 FFmpeg***

FFmpeg is a framework as well as an application to record, convert and stream audio and video data. It is published under the **GNU Lesser General Public License (LGPL)**, version 2.1 (FFmpeg, 2014). The framework is used to stream H.264 encoded video data between the application instances using the build-in RTSP protocol. Particularly interesting are the license terms of H.264. Most of the techniques which are used in the H.264 standard are patented. This means that a manufacturer who wants to use the standard has to pay a fee to e.g. MPEG-LA. Consequently, H.264 is an open standard but not free (MPEGLA, 2014, p. 1 ff.). Nevertheless, FFmpeg uses a free software library to realize H.264 compression called x264, which is also published under the **GNU General Public License (GPL)**, Version 2.0.

### ***8.2.1.2 AForge.Net***

AForge.Net is an open source software library for .Net applications in the fields of computer vision, artificial intelligence – image processing etc. It is published under the LGPL, version 3 (AForge.NET, 2012). The framework is used to capture the video signal of the webcam from the consultant as well as from a stationary camera.

### ***8.2.1.3 NAudio***

NAudio is an open source software library for .Net applications to work with audio and **Musical Instrument Digital Interface (MIDI)** data. It is published under the Microsoft Public License (Ms-PL) (Heath, 2014). The framework is used to capture the audio signal from the microphone as well as playing the audio data. Furthermore, the G.722 codec which is implemented in the library is used to compress the audio data for the transfer between the application instances.

### ***8.2.1.4 Managed Media Aggregation***

Managed Media Aggregation is a C# software library for .Net applications to work with RTSP, RTP, RTCP and further protocols. It is published under the Apache License, Version 2.0 (Friedman,

2014). The framework is used to transfer MJPEG via the RTSP protocol between application instances.

### 8.3 The Applications

As already outlined in the previous chapter 7, the proof-of-concept consists of two parts, an application that runs directly on Glass and a Windows control application named “Control Kit” which handles the data transfer between the involved hospitals and Glass (see Figure 25).

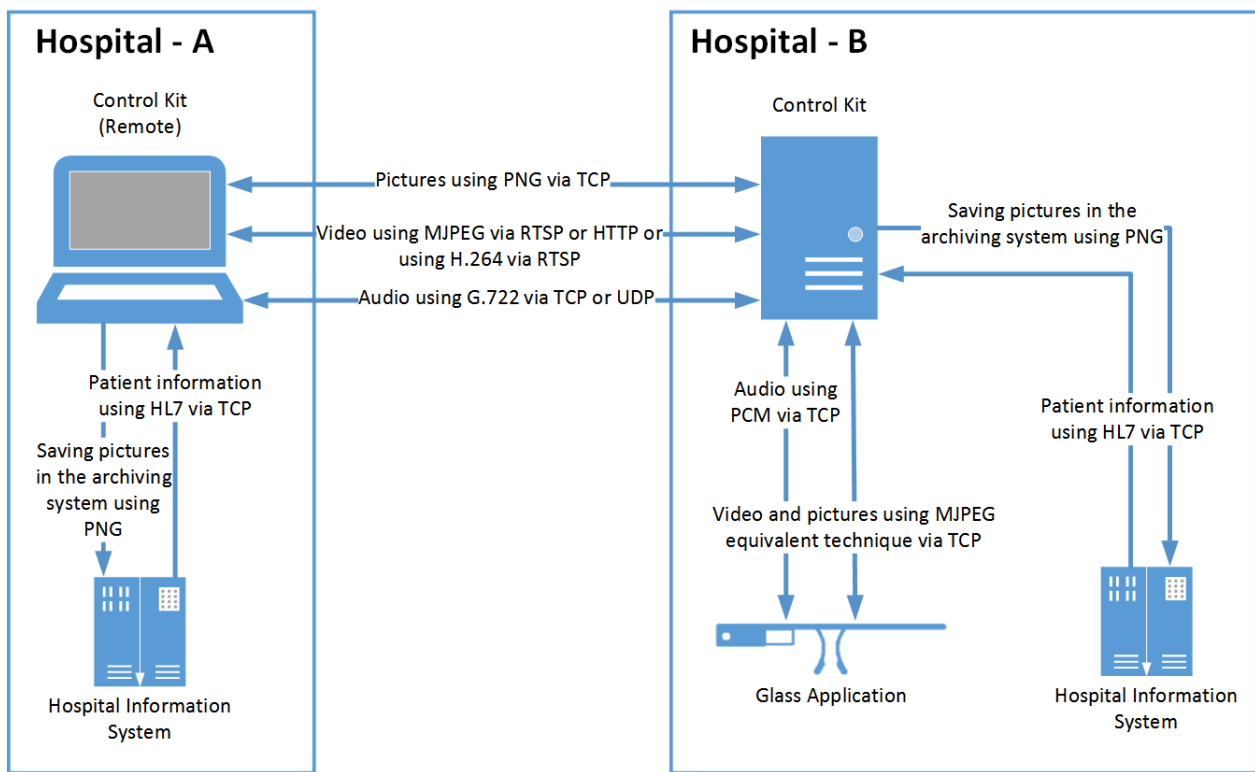


Figure 25: Communication flow between the involved applications.

#### 8.3.1 The Google Glass Application

The client application on Glass allows the user to stream video and audio signals to the Windows application and vice versa. The whole communication is running over one TCP socket where the distinction of the data packages is done by different data types. Hence, there is no standardized protocol above the TCP protocol stack involved in the communication. For the audio communication between Glass and the Windows application uncompressed **Pulse-Code**

**Modulation (PCM)** data is used. For the video communication an individual compression algorithm based on MJPEG is utilized, so there is at least a simple data compression. The advantage of this approach is that the near-eye display device is doing as less as possible computation to support low-power hardware. Additionally, it is also possible to take pictures direct by the camera button on the device or force this function via a remote message.

### **8.3.2 The Windows Control Application**

Control Kit takes care of the communication with the Glass app as well as for the video and audio conferences between multiple instances of this program. Furthermore, it also provides an interface to the HIS, the main view can be seen in Figure 26. It allows an individual configuration of its functionalities depending on the desired purpose of use. For instance, it is possible to configure multiple video streams running on different protocols. However, the main reason for this flexibility is that it is possible to configure the application for each user individually in the field tests. After choosing the type of the stream, the user has to select the source that should be used, e.g. the webcam, as well as the sink which is for instance, a MJPEG streaming server. It can be seen that e.g. a sink may start a server socket where clients can connect to. This implies that the application communicates with multiple sockets with the environment. Additionally, the software architecture supports multiple sinks per source, but for the GUI only one sink is available. The reason is, that this functionality is not needed for the field tests and moreover, it lets the user interface look simpler. To make the configuration of e.g. the needed ports, as simple as possible the application loads default values from an attached configuration file.

The initial configuration of the program is relatively complex and time consuming, but as already mentioned the application is a proof-of-concept and it isn't designed as a marketable product. Hence, to guarantee the most possible flexibility the concept of the sinks and sources is mapped nearly one to one to the GUI.

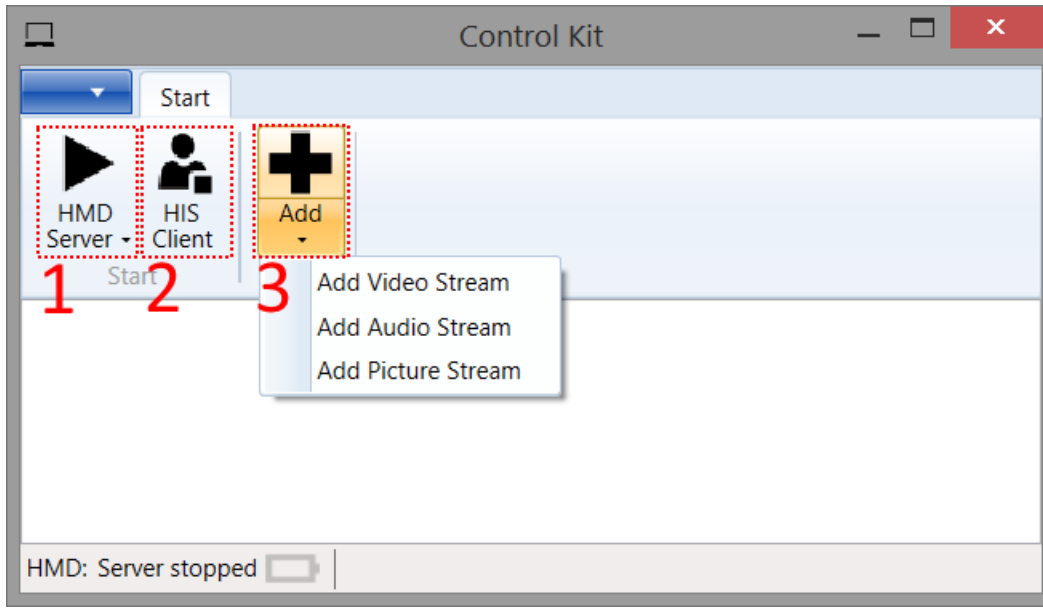


Figure 26: The Windows control application called “Control Kit”. 1: Starts or stops the server socket for the communication with the Google Glass. 2: Starts the HIS client to save pictures in the hospital information system. 3: Adds a new functionality to the application which allows the user to stream audio, video or picture data between multiple application instances and Glass.

## 8.4 Capabilities

This section illustrates the concrete functionalities which are implemented in the proof-of-concept. It lists and explains the different capabilities without going into too much technical detail, but rather explains the components of a user’s point of view.

### 8.4.1 Video Transmission

To send and receive video streams, the user has to click *Add - Video Stream* in the ribbon bar of the application. This adds a new element to the main view of the program. In this element it is possible to choose a suitable video source from a list of sources as well as a proper video sink (see Figure 27). Depending on the type of the source the application receives the video signal from:

- Another instance of the program or another compatible application.
- The Google Glass.
- A camera connected to the computer.

After selecting the source, the user has to select a suitable resolution. If the configuration is correct, the *Start* button can be clicked. When everything is working, a preview window should appear which displays the current video input signal. With a click in the video area a separate video window opens, which can be enlarged to full screen.

In case that the video signal should be published via a stream, a suitable sink must be selected and launched with the *Start* button. If both, the source and the sink, are running the *Link* button can be pressed. After this click the source will start feeding the sink with image data.

Obviously, it is also possible to end the video transfer with the *Unlink* button or terminate the sink and source with its *Stop* buttons.

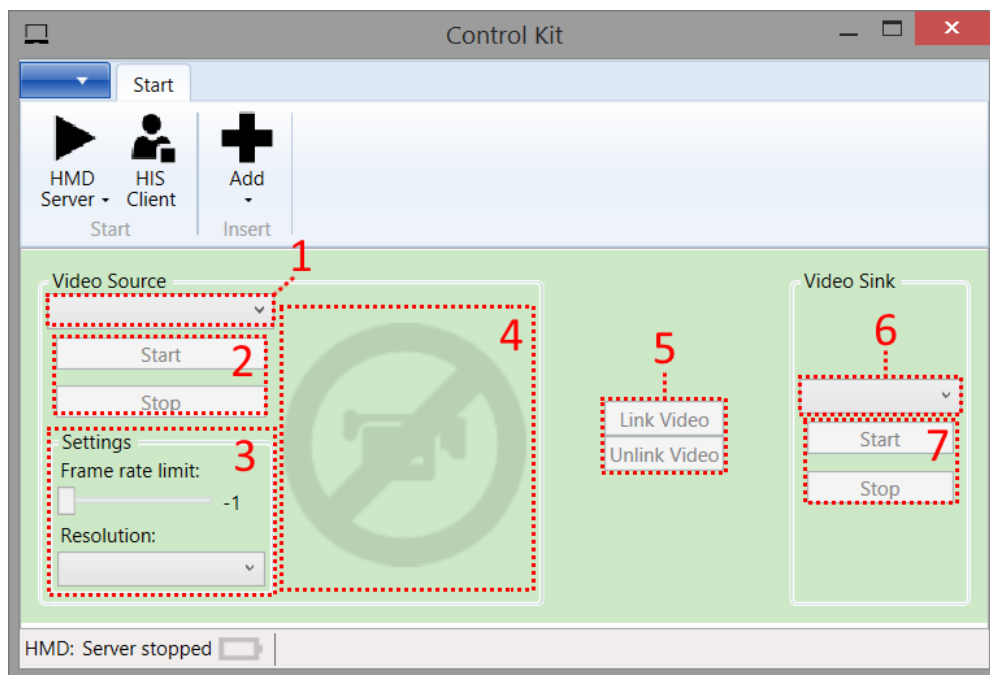


Figure 27: The Control Kit with an added Video Stream item. 1: Enables the user to choose a video source e.g. the webcam. 2: Starts or stops the video source. 3: Allows the user to change the frame rate and image resolution. 4: The preview of the video signal; with a click in the image a separate video window opens, which can be enlarged to full screen. 5: Connects or disconnects the source with the sink. 6: Enables the user to choose a video sink e.g. a MPEG file. 7: Starts or stops the video sink.

#### **8.4.1.1 Possible Video Sources**

The list below shows possible video sources which can be chosen from the dropdown menu of an added Video Stream item in the Control Kit.



### **[Webcam-Name]**

The menu item “Webcam-Name” is only available if a camera is connected to the computer. Before the video capturing can be started the user has to select a suitable resolution of all resolutions supported by the camera.

### **Google Glass**

The menu item *Google Glass* can only be used if the HMD Server is started (see Figure 26) and Glass is connected to the application. Furthermore, the user has to select a suitable resolution of all resolutions supported by the Glass camera, to receive a live video stream from the near-eye display device.

### **MJPEG HTTP Client**

The menu item *MJPEG HTTP Client* will start an MJPEG streaming client which connects via HTTP to an MJPEG streaming server. This server may be hosted from a different instance of this application, but because the communication works over a standardized interface, it can also be hosted from another program which supports MJPEG streaming via HTTP. After the start of the source the application will ask the user for the address of the server and will only continue if a correct URL is entered. Additionally, the MJPEG HTTP Client does not permit to change the resolution of the video, i.e. only the origin resolution can be used.

### **MJPEG RTSP Client**

The menu item *MJPEG RTSP Client* will start an MJPEG streaming client which connects via RTSP to an MJPEG streaming server. This server may be hosted from a different instance of this application, but because the communication works over a standardized interface, it can also be hosted from another program which supports MJPEG streaming via RTSP. After the start of the source the application will ask the user for the address of the server and will only continue if a correct URL is entered. Additionally, the MJPEG RTSP Client does not permit to change the resolution of the video, i.e. only the origin resolution can be used.

### **H.264 RTSP Server**

The menu item *H.264 RTSP Server* will start an H.264 streaming server which accepts connections via RTSP from an H.264 streaming client. This client may be started from a different instance of

this application, but because the communication works over a standardized interface, it can also be started from another program which supports H.264 streaming via RTSP. After the start of the source a console window will pop up, this window is responsible for the processing and transmission of the data. Additionally, the H.264 RTSP Server does not permit to change the resolution of the video, i.e. only the origin resolution can be used.

#### **8.4.1.2 Possible Video Sinks**

The list below shows possible video sinks which can be chosen from the dropdown menu of an added Video Stream item in the Control Kit.

##### **MPEG File**

The menu item *MPEG File* enables the user to save a video stream on the hard drive. After the start of the sink the application will ask the user for the path and file name and will only continue if it is correctly entered.

##### **Google Glass**

The menu item *Google Glass* can only be used if the HMD Server is started (see Figure 26) and Glass is connected to the application. If this constraint is satisfied a video can be streamed to the device.

##### **MJPEG HTTP Server**

The menu item *MJPEG HTTP Server* will start a MJPEG streaming server which accepts connections via HTTP from a MJPEG streaming client. This client may be started from a different instance of this application, but because the communication works over a standardized interface, it can also be started from another program which supports MJPEG streaming via HTTP.

##### **MJPEG RTSP Server**

The menu item *MJPEG RTSP Server* will start a MJPEG streaming server which accepts connections via RTSP from a MJPEG streaming client. This client may be started from a different instance of this application, but because the communication works over a standardized interface, it can also be started from another program which supports MJPEG streaming via RTSP.

## H.264 RTSP Client

The menu item H.264 RTSP Client will start a H.264 streaming client which connects via RTSP to a H.264 streaming server. This server may be hosted from a different instance of this application, but because the communication works over a standardized interface, it can also be hosted from another program which supports H.264 streaming via RTSP. After the start of the sink the application will ask the user for the address of the server and will only continue if a correct URL is entered.

### 8.4.2 Audio Transmission

To send and receive audio streams, the user has to click *Add - Audio Stream* in the ribbon bar of the application. This adds a new element to the main view of the program. In this element it is possible to choose a suitable audio source from a list of sources as well as a proper audio sink (see Figure 28). Depending on the type of the source the application receives the audio signal from:

- Another instance of the program.
- The Google Glass.
- A microphone connected to the computer.

Next, the source and the sink has to be launched with the *Start* button. If both are running the *Link* button can be pressed. After this click the source will start feeding the sink with audio data.

Obviously, it is also possible to end the audio transfer by pressing the *Unlink* button or terminate the sink and source using its *Stop* buttons.

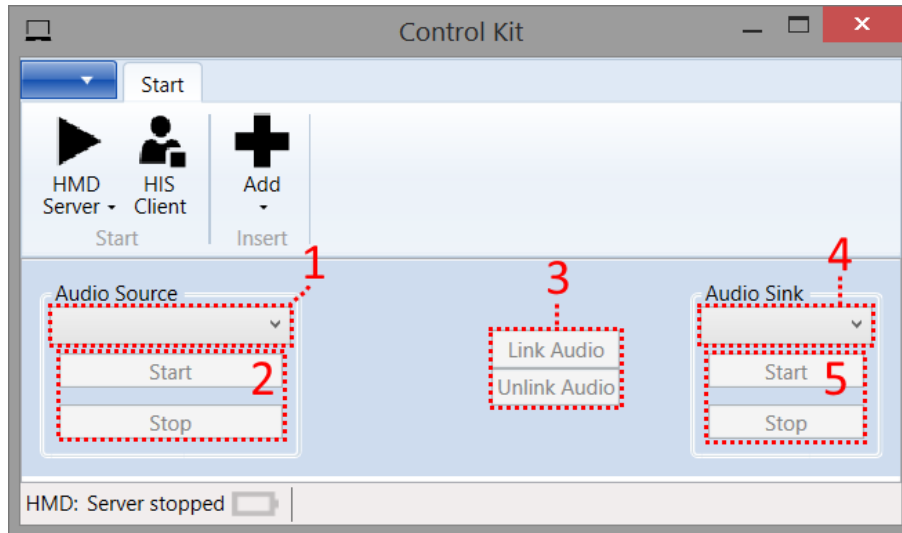


Figure 28: The Control Kit with an added Audio Stream item. 1: Enables the user to choose an audio source e.g. the microphone. 2: Starts or stops the audio source. 3: Connects or disconnects the source with the sink. 4: Enables the user to choose an audio sink e.g. the G.722 TCP Server. 5: Starts or stops the audio sink.

#### **8.4.2.1 Possible Audio Sources**

The list below shows possible audio sources which can be chosen from the dropdown menu of an added Audio Stream item in the Control Kit.

##### **WaveIn Source [Microphone-Device Number]**

The menu item “WaveIn Source Microphone-Device Number” allows the user to access the microphone of the computer and thus to record his or her voice.

##### **Google Glass**

The menu item *Google Glass* can only be used if the HMD Server is started (see Figure 26) and Glass is connected to the application. If this constraint is satisfied the audio data or voice can be received from the near-eye display device.

##### **G.722 UDP Client**

The menu item *G.722 UDP Client* will start an audio streaming client which is connected via UDP to an audio streaming server using the G.722 codec. After the start of the source the application will ask the user for the address of the server and will only continue if a correct URL is entered.

### **G.722 TCP Client**

The menu item *G.722 TCP Client* will start an audio streaming client which is connected via TCP to an audio streaming server using the G.722 codec. After the start of the source the application will ask the user for the address of the server and will only continue if a correct URL is entered.

### **8.4.2.2 Possible Audio Sinks**

The list below shows possible audio sinks which can be chosen from the dropdown menu of an added Audio Stream item in the Control Kit.

#### **Play**

The menu item *Play* will just simply play the audio data.

#### **Google Glass**

The menu item *Google Glass* can only be used if the HMD Server is started (see Figure 26) and Glass is connected to the application. If this constraint is satisfied the audio data or voice can be transferred to the near-eye display device.

### **G.722 UDP Server**

The menu item *G.722 UDP Server* will start an audio streaming server which accepts connections via UDP from an audio streaming client using the G.722 codec.

### **G.722 TCP Server**

The menu item *G.722 TCP Server* will start an audio streaming server which accepts connections via TCP from an audio streaming client using the G.722 codec.

### **8.4.3 Picture Transmission**

To send and receive picture streams, the user has to click *Add - Picture Stream* in the ribbon bar of the application. This adds a new element to the main view of the program. In this element it is possible to choose a suitable picture source from a list of sources as well as a proper picture sink (see Figure 29). Depending on the type of the source the application receives the picture signal from:

- Another instance of the program.
- The Google Glass.
- A file from the computer.

Next, the source and the sink has to be launched with the *Start* button. If both are running the *Link* button can be pressed. After this click the source is able to feeding the sink with pictures. Additionally, the *Take Picture* button enables the user to force an image acquisition from the source. When everything is working correct, a preview window should appear which displays the current picture. With a click in the image area, a separate window with the picture in it opens. This window enables the user to view the image in full screen.

Obviously, it is also possible to end the picture transfer with the *Unlink* button or terminate the sink and source with its *Stop* buttons.

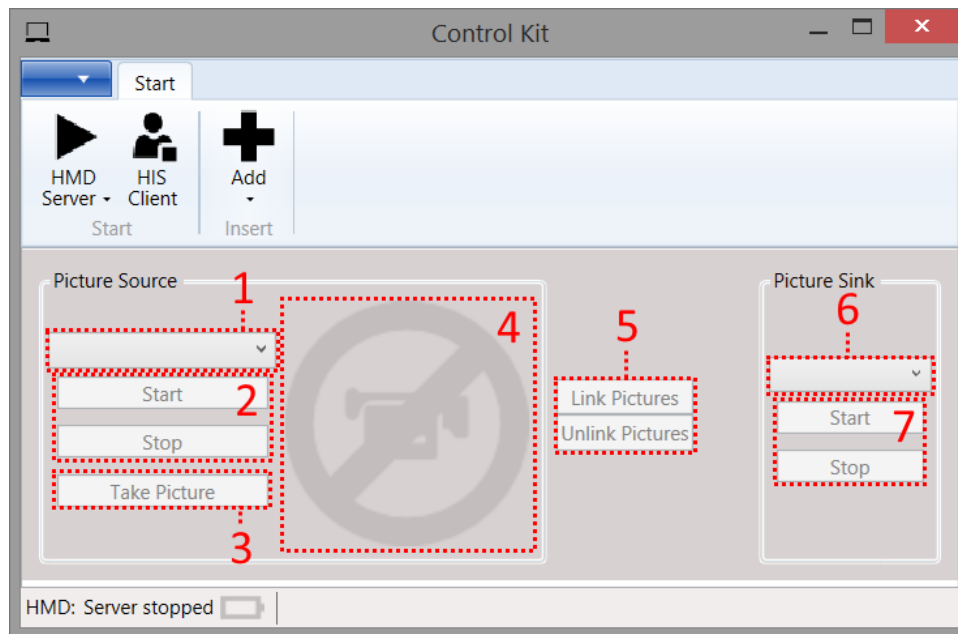


Figure 29: The Control Kit with an added Picture Stream item. 1: Enables the user to choose a picture source e.g. the Google Glass. 2: Starts or stops the picture source. 3: Forces an image acquisition from the source. 4: The preview of the picture; with a click in the image a separate window with the picture opens, which can be enlarged to full screen. 5: Connects or disconnects the source with the sink. 6: Enables the user to choose a picture sink e.g. the PNG TCP Server. 7: Starts or stops the picture sink.

#### **8.4.3.1 Possible Picture Sources**

The list below shows possible picture sources which can be chosen from the dropdown menu of an added Picture Stream item in the Control Kit.

##### **Google Glass**

The menu item *Google Glass* can only be used if the HMD Server is started (see Figure 26) and Glass is connected to the application. If this constraint is satisfied the pictures taken by the camera button of the device as well as by the *Take Picture* button, can be received.

##### **PNG TCP Client**

The menu item *PNG TCP Client* will start a picture streaming client which connects via TCP to a picture streaming server using the **Portable Network Graphics (PNG)** format. After the start of the source the application will ask the user for the address of the server and will only continue if a correct URL is entered.

##### **File**

The menu item *File* lets the user stream images in PNG format from the hard drive. Obviously, this source will not automatically publish pictures, instead, the user has to use the *Take Picture* button and select an image manually.

#### **8.4.3.2 Possible Picture Sinks**

The item below shows a possible picture sink which can be chosen from the dropdown menu of an added Picture Stream item in the Control Kit.

##### **PNG TCP Server**

The menu item *PNG TCP Server* will start a picture streaming server which accepts connections via TCP from a picture streaming client using the PNG format.

#### **8.4.4 HIS Related Functionality**

The proof-of-concept also has a simple connection to the hospital information system implemented, which allows the user to receive patient information and store pictures related to the patient in the HIS. In the HIS Client the user can take pictures remote or receive these images

which are directly taken with the camera button on the Glass, regardless if the application is directly connected to the near-eye display device or if it is launched from the remote consultant. This allows the users of both applications to simultaneously manage the captured images.

To use this functionality the user has to click the *HIS* button in the ribbon bar of the application which opens another window: the *HIS Client*. The *HIS Client* distinguishes whether the current application instance is connected directly to Glass or if the client is used from the remote consultant. In case of the first option, the application will directly display the *HIS* view. In case of the second option, the application will ask the user for the address of a server and will only continue if a correct URL is entered. The server would be an instance of this application configured with a picture stream, started by the doctor who gets support, where the source of the picture stream is Glass and the sink is a PNG TCP Server. Additionally, the *HIS Client* has to connect with the hospital information system via HL7. The button *Start HL7* will launch a TCP socket where the *HIS* will connect to and send medical records to the Control Kit. To store patient related pictures in the *HIS*, first of all the patient data must be provided by the HL7 interface. To receive this data a function in the *HIS* application of the hospital has to be triggered. After the data is available in the *HIS Client* the pictures can be saved in the archiving system. From a technical point of view, saving a picture in the archiving system is the same as saving a picture on the hard drive, expecting that the directory path points to a network drive instead to the local hard drive.



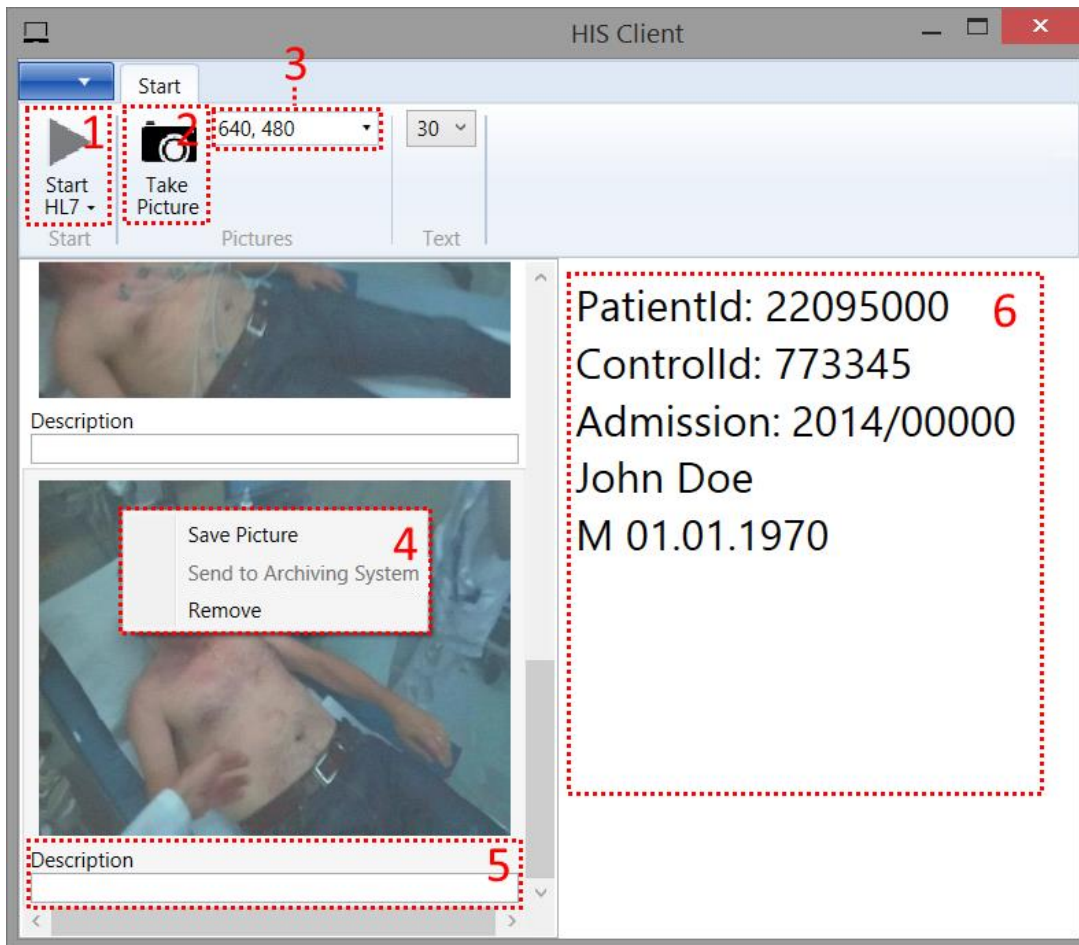


Figure 30: The HIS Client window of the Control Kit. 1: Starts or stops the HL7 server socket for the communication with the HIS. 2: Takes a new picture. 3: Changes the resolution in which the pictures should be taken. 4: Saves the picture to the hard disk or to the HIS via the archiving system or removes the picture from the list. 5: A description for the picture which gets stored in the HIS. 6: Medical records of a patient received via the HL7 interface.

## 8.5 Technical Limitations of the Proof-of-Concept

The proof-of-concept has some problems and limitations, which need to be considered. The restrictions are mainly related to the implementation as well as to Glass itself, which is the hardware platform that serves as development basis for the proof-of-concept.

One of the biggest limitations of the proof-of-concept is the restriction of the maximum video resolution of 720p and the limitations regarding to the hardware performance. For instance, streaming a video in the maximum resolution will bring the device to its performance limits. As a result the device will become very hot, which ultimately yields in a forced shutdown. (Vorraber, et al., 2014, p. 1271).

Additional potential for improvement exists in the current implementation on Glass which supports only video streams compressed with MJPEG and uncompressed audio streams, transmitted without any standardized protocol. As already mentioned, the advantage of this approach is that the near-eye display device is doing as less as possible computation to support low-power hardware.

Another problem is the battery capacity of the device, which is under extreme conditions, such as the Virtual Consultation, less than an hour. But this issue can be solved with an additional battery pack which extends the battery life of the near-eye display device (Vorraber, et al., 2014, p. 1271). Thereby the device gets of course heavy and clumsy which may have negative influence to the usability.

Finally it should be mentioned that the HL7 functionality is implemented in the proof-of-concept, but has never been used during field tests or otherwise.

## **8.6 Possible Extensions of the Proof-of-Concept**

During the development and the design of the proof-of-concept experience has been gained which can be used for an extension of the proof-of-concept or a marketable product. This section will outline possible design and implementation choices to suggest improvements for the proof-of-concept.

Discussions with the medical staff and the IT department have shown that the handling with the system which implements the Virtual Consultation use case should be similar to applications such as Skype. So it should be easy to establish a video conference with different doctors or specialists, without any preparation or configuration of a program. For instance, the user can start a conference by just clicking at the name of the person.

To provide such a functionality the most appropriate way is to use a middleware application that simplifies the communication in a disturbed system. The advantage of this is that a central point knows exactly which clients are online and hence who is currently available for a video conference. Furthermore, the communication with the HIS should also be managed from the

middleware application which has the advantage that the HIS only has one endpoint to communicate with and not an endpoint for each application instance.

During the implementation of the proof-of-concept the FFmpeg framework has proven to be very advantageous. For instance, it allows live streaming between computers out of the box. FFmpeg includes a streaming server application where video streams can be hosted on and multiple clients can receive the stream without knowing the network endpoint of the computer which provides the video. Additionally, FFmpeg supports streaming protocols, such as RTSP to transfer video and audio signals over the network. Hence, FFmpeg would provide a fast way to implement the desired functionalities without putting too much effort in technical details such as video compression or transmission protocols.

Also, from a legal perspective electronic medical records are subject to certain conditions depending on the country and thus hospitals are often obligated to guarantee an appropriate level of data security. The proof-of-concept does not explicitly implement a secure communication, instead it uses a VPN connection to provide data security. But in case of a real product it cannot be presumed that a VPN connection always exists between two parties. As a result, the system must also be capable to manage the security requirements by its own and shouldn't rely completely on the network structure.

Moreover, the implementation of the proof-of-concept on Glass is very straightforward. For a marketable product the applications should support better video and audio encoding algorithms as well as standardized transmission protocols (Sachs, 2014, p. 103). Furthermore, an optimization of the software is necessary to run properly on Glass. For instance, video and audio processing which is currently done in the managed Java environment could be implemented in native C++.

## **9 Results**

---

This chapter discusses the results as well as feedback from medical staff for the Virtual Consultation and the Doctor's Visit.

### **9.1 Feedback Methods**

Feedback for the Virtual Consultation was gathered through a qualitative approach using informal conversational interviews, which take place after the field tests.

### **9.2 Feedback Results**

The conversational interviews with the medical staff revealed that the use of a near-eye display device for the Virtual Consultation is a promising topic. The doctors see in the first-person perspective of the video significant advantages compared with a stationary camera or an audio only communication. It has been argued that in case of an unexperienced trainee or a doctor which is unacquainted with the situation, this perspective provides the consultant some additional information. For instance, on which areas of the patient does the doctor set the focus, or is the doctor hectic moving the head, which indicates nervousness. Additionally, the way how a patient looks at the doctor is also interesting, because it reveals some details about the condition of a patient. To conclude, the feedbacks of the doctors after the tests were consistently positive, especially the video from first-person perspective was assessed to be valuable for such video conferences.

### **9.3 Conclusion**

The Virtual Consultation and the Doctor's Visit were investigated systematically based on the RE framework of the SOPHIST-approach (Rupp, 2009, p. 1 ff.). For the Doctor's Visit the as-is analysis already revealed that the process is fairly optimized and there is not much room for improvement. The use of a near-eye display devices during the standard procedure of the Doctor's Visit would not provide any substantial benefits for the medical staff. This is the case because the doctors

who perform the visits have their hands free and can use it to interact with the environment. This circumstance restricts the use of a near-eye display device and hence makes the use case uninteresting for the thesis. Complementary to this, the Virtual Consultation has proven to be a very promising area for the usage of a near-eye display device, to share competencies between hospitals. For this use case a prototype / proof-of-concept has been developed which implements certain requirements to evaluate and test the Virtual Consultation.

## **10 Conclusion and Future Work**

---

The aim of this work is to reveal, evaluate and discuss new technologies and methods to improve the efficiency and effectiveness of the clinical care, using near-eye display devices such as Glass. The importance of the discussed subjects is resulting from the current situation in the hospitals. The today's healthcare sector is highly specialized. For each area there are some medical experts who work on particular fields. Those experts are rare and thus expensive. Additionally, the optimization and improvement for medical processes is also an important topic, because it allows a better use of the available resources. For this master's thesis two use cases, the Virtual Consultation and the Doctor's Visit are analyzed in detail. Those use cases were revealed in previous research projects from Vorraber et al. and are the basis for this thesis. The Virtual Consultation serves as foundation to analyze and evaluate the use of near-eye display devices for video conferences to share medical competencies between hospitals. Similarly, the Doctor's Visit use case analyzes the utilization of near-eye display devices for the existing medical processes during the daily visits of the patients by the doctors.

The analyzed and discovered use cases, as well as the implemented proof-of-concept shows that the acceptance of new technologies and processes is present in hospital environments. As a result, the medical staff sees the Virtual Consultation as improvement compared to the current process, which makes this use case a very promising area.

### **10.1 Limitations**

The main limitation of the findings is the small sample size. This applies for the number of field tests as well as for the number of participants. The field tests were only conducted with two hospitals and four participants. Nonetheless, further quantitative and qualitative studies are planned to get results which have more statistical relevance.

### **10.2 Future Work**

This master's thesis focuses on two use cases, the Virtual Consultation and the Doctor's Visit, which is only a small subset of the possible use cases of a near-eye display device in hospital

environments. This aside, the current implemented proof-of-concept of the Virtual Consultation gives some further use cases which are not evaluated during this thesis.

Firstly, the proof-of-concept can be used during a surgery to support a surgeon or to show the video stream to trainees (Mackle, 2013). Secondly, the use case can be used to supervise surgeons in training e.g. on training dummies. Thirdly, in general there are several people involved in a surgery which do not all have a perfect view at the part of the patient at which the operation takes place. This opens the possibility to display the first-person perspective of the surgeon to the rest of the staff e.g. to a monitor or to other near-eye display devices (Vorraber, et al., 2014, pp. 1270, 1271). Lastly, the possibility to display information in the FOV of the user could be an advantage during video conferences. An example would be the projection of vital parameters of a patient in the FOV of the doctor. Also medical information of a patient such as allergies or medical results could be displayed. Similar experiments were already conducted or proposed by: (Vorraber, et al., 2014, p. 1266 ff.), (Sachs, 2014, pp. 102-106) and (Mackle, 2013).

Furthermore, the Doctor's Visit includes the Virtual Consultation use case between external specialists and hospitals. Hence, an interesting area would be to study how to strength the networking between the hospitals and the external specialists to support and improve medical processes.

Further research in this area may also include the usage of a telepointer during the Virtual Consultation. The paper (Bauer, et al., 1999, p. 151 ff.) already evaluated the utilization of a reality-augmenting telepointer during videoconferences and revealed that the guidance of a remote specialist is more efficient with such a device.

# 11 References

---

- AForge.NET, 2012. *AForge.NET*. [Online]  
Available at: <http://www.aforgenet.com>  
[Accessed 27 10 2014].
- Aurrecochea, C., Campbell, A. T. & Hauw, L., 1998. A Survey of QoS Architectures. *Multimedia Systems*, 6(3), pp. 138-151.
- Bangay, S. & Preston, L., 1998. An investigation into factors influencing immersion in interactive virtual reality environments. In: *Virtual Environments in Clinical Psychology and Neuroscience: Methods and Techniques in Advanced Patient-therapist Interaction*. Grahamstown, South Africa: IOS Press, pp. 43-51.
- Bauer, M., Kortuem, G. & Segall, Z., 1999. "Where are you pointing at?" A study of remote collaboration in a wearable videoconference system. In: *Wearable Computers, 1999. Digest of Papers. The Third International Symposium on*. Washington, DC, USA: IEEE Computer Society, pp. 151-158.
- Beeler, G. W., 1998. HL7 Version 3 - An object-oriented methodology for collaborative standards development. *International Journal of Medical Informatics*, 48(1-3), pp. 151-161.
- Birnholtz, J., Ranjan, A. & Balakrishnan, R., 2010. Providing Dynamic Visual Information for Collaborative Tasks: Experiments With Automatic Camera Control. *Human-Computer Interaction*, Volume 25, pp. 261-287.
- Blobel, B., Engel, K. & Pharow, P., 2006. *Semantic interoperability--HL7 Version 3 compared to advanced architecture standards*, 93042 Regensburg, Germany: Methods Inf Med..
- Boehm, B. W., 1976. Software Engineering. *Computers, IEEE Transactions on*, C-25(12), pp. 1226-1241.
- Bolas, M. T., 1994. Human factors in the design of an immersive display. *IEEE Comput. Graph. Appl.*, Jan., Volume vol.14, no.1, pp. 55-59.
- Buschmann, F., Henney, K. & Schmidt, D. C., 2007. *PATTERN-ORIENTED SOFTWARE ARCHITECTURE*. 5 ed. Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England: John Wiley & Sons Ltd.
- Buschmann, F. et al., 1996. *Pattern-Oriented Software Architecture, A System of Patterns*. 1 ed. s.l.:Wiley.
- Campbell, A. T., 1996. *A Quality of Service Architecture - A thesis submitted for the degree of Doctor of Philosophy*. s.l.:Computing Department Lancaster University.
- Citrix Online, 2014. *GoToMeeting*. [Online]  
Available at: <http://www.gotomeeting.de/meeting>  
[Accessed 15 11 2014].
- Cockburn, A., 2008. *Use Cases effektiv erstellen*. s.l.:mitp Verlags GmbH & Co. KG.
- Fakespace Lab, 2004. *Fakespace Lab*. [Online]  
Available at: <http://www.fakespacelabs.com/tools.html>  
[Accessed 24 09 2014].



- FFmpeg (a), 2014. *Buffer sink API*. [Online]  
Available at: [https://www.ffmpeg.org/doxygen/2.4/group\\_lavfi\\_buffersink.html](https://www.ffmpeg.org/doxygen/2.4/group_lavfi_buffersink.html)  
[Accessed 08 10 2014].
- FFmpeg, 2014. *FFmpeg (b)*. [Online]  
Available at: <https://www.ffmpeg.org>  
[Accessed 27 10 2014].
- Fowler, M., 2004. *Presentation Model*. [Online]  
Available at: <http://martinfowler.com/eaDev/PresentationModel.html>  
[Accessed 09 12 2014].
- Fraunhofer Research Institution for Organics, Materials and Electronic Devices COMEDD, 2012. *EVALUATION KIT OF OLED BASED BINOCULAR INTERACTIVE SEE-THROUGH HMD*. 01109 Dresden: Fraunhofer.
- Fraunhofer Research Institution for Organics, M. a. E. D. C., 2014. *interactive see-through hmd*. [Online]  
Available at: <http://www.interactive-see-through-hmd.de/evaluation-kit/>  
[Accessed 24 09 2014].
- Freeman, E. & Reed, D., 1983. *Stockholders and Stakeholders: A New Perspective on Corporate Governance*. California, Spring, pp. 88-106.
- Friedman, J., 2014. *Managed Media Aggregation*. [Online]  
Available at: <https://net7mma.codeplex.com>  
[Accessed 27 10 2014].
- Furness, T. & Kollin, J., 1992. *VIRTUAL RETINAL DISPLAY*. USA, Patent No. US5467104 A.
- Fussell, S. R., Setlock, L. D. & Kraut, R. E., 2003. Effects of Head-mounted and Scene-oriented Video Systems on Remote Collaboration on Physical Tasks. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Ft. Lauderdale, Florida, USA: ACM, pp. 513-520.
- Glinz, M., 2007. *On Non-Functional Requirements*. Delhi, IEEE Computer Society, pp. 21-26.
- Google (a), 2014. *Google*. [Online]  
Available at: <https://www.google.com/glass/start/>  
[Accessed 24 09 2014].
- Google (b), 2014. *Hangouts*. [Online]  
Available at: <http://www.google.at/hangouts/>  
[Accessed 25 09 2014].
- Google (c), 2014. *Platform Overview*. [Online]  
Available at: <https://developers.google.com/glass/develop/overview>  
[Accessed 12 10 2014].
- Gossman, J., 2005. *Introduction to Model/View/ViewModel pattern for building WPF apps*. [Online]  
Available at: <http://blogs.msdn.com/b/johngossman/archive/2005/10/08/478683.aspx>  
[Accessed 24 10 2014].

Health Level Seven, Inc, 2003. *HL7 MESSAGING STANDARD VERSION 2.5 An Application Protocol for Electronic Data Exchange in Healthcare Environments*. 3300 Washtenaw, Suite 227 Ann Arbor, MI USA: s.n.

Heath, M., 2014. *NAudio*. [Online]  
Available at: <http://naudio.codeplex.com>  
[Accessed 2014 10 27].

IEEE Standards Board, 1990. *IEEE Standard Glossary of Software Engineering Terminology*. New York, NY 10017, USA: IEEE.

International Telecommunication Union, 2008. *E.800 - Definitions of terms related to quality of service - SERIES E: OVERALL NETWORK OPERATION, TELEPHONE SERVICE, SERVICE OPERATION AND HUMAN FACTORS*. s.l.:International Telecommunication Union.

International Telecommunication Union, 2012. *G.722 - 7 kHz audio-coding within 64 kbit/s - SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS - Digital terminal equipments - Coding of voice and audio signals*. s.l.:International Telecommunication Union.

International Telecommunication Union, 2014. *H.264 - Advanced video coding for generic audiovisual services - SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS - Infrastructure of audiovisual services - Coding of moving video*. s.l.:International Telecommunication Union.

kapsch, 2013. *kapsch*. [Online]  
Available at: [http://www.kapsch.net/KapschGroup/press/kbc/kbc\\_131105\\_pr?lang=de-AT](http://www.kapsch.net/KapschGroup/press/kbc/kbc_131105_pr?lang=de-AT)  
[Accessed 25 09 2014].

Karapantazis, S. & Pavlidou, F.-N., 2009. VoIP: A comprehensive survey on a promising technology. *Computer Networks*, Volume 53, pp. 2050 - 2090.

Kress, B. & Starner, T., 2013. *A review of head-mounted displays (HMD) technologies and applications for consumer electronics*. 1600 Amphitheatre Parkway Mountain View, CA 94043 USA, Google Inc..

Lanir, J., Stone, R., Cohen, B. & Gurevich, P., 2013. Ownership and Control of Point of View in Remote Assistance. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, pp. 2243-2252.

Leffingwell, D. & Widrig, D., 2000. *Managing Software Requirements: A Unified Approach*. s.l.:Addison-Wesley.

Lewis, M. & Slack, N., 2003. *Operations Management*. s.l.:Routledge.

Li, J. & Alem, L., 2013. Supporting Distributed Collaborations between Mobile Health Workers and Expert Clinicians in Home Care. pp. 493-498.

Mackle, B., 2013. *Ohio State Doctor Shows Promise of Google Glass in Live Surgery*. [Online]  
Available at: <http://sportsmedicine.osu.edu/article.cfm?ID=8013>  
[Accessed 28 11 2014].

Mann, S., 1997. *Wearable Computing: A First Step Toward Personal Imaging*. Los Alamitos, CA, USA, IEEE Computer Society Press, pp. 25 - 32.

- Mann, S. & Picard, R., 1995. On Being 'undigital' With Digital Cameras: Extending Dynamic Range By Combining Differently Exposed Pictures. *IS&T, 48th annual conference*, pp. 422-428.
- Mccollum, T., 1943. *Stereoscopic television apparatus*. USA, Patent No. US2388170 A.
- McConnell, S., 2004. *Code Complete*. 2nd ed. s.l.:Microsoft Press.
- Mellouk, A., 2009. *End-to-End Quality of Service Engineering in Next Generation Heterogenous Networks*. s.l.:Wiley.
- Melzer, J. E., 2011. *Head-Mounted Displays: Designing for the User*. s.l.:CreateSpace.
- Microsoft (a), 2014. *skype*. [Online]  
Available at: <http://www.skype.com>  
[Accessed 25 09 2014].
- Microsoft (b), 2014. *Stream Buffer Engine Filters*. [Online]  
Available at: [http://msdn.microsoft.com/en-us/library/windows/desktop/dd695303\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/dd695303(v=vs.85).aspx)  
[Accessed 08 10 2014].
- Microsoft (c), 2015. *What Is Managed Code?*. [Online]  
Available at: <https://msdn.microsoft.com/en-us/library/windows/desktop/bb318664%28v=vs.85%29.aspx>  
[Accessed 04 04 2015].
- Mildenberger, P., Eichelberg, M. & Martin, E., 2001. Introduction to the DICOM standard. *European Radiology*, 12(4), pp. 920-927.
- Mindflux, 2006. *Mindflux*. [Online]  
Available at: <http://www.mindflux.com.au/products/iis/vfx1.html>  
[Accessed 06 02 2015].
- MPEGLA, 2014. *SUMMARY OF AVC/H.264 LICENSE TERMS*. s.l.:MPEGLA The Standard for Standards.
- Narciso, C. & Verner, J., 2009. *Why did your project fail?*. s.l., ACM, pp. 130-134.
- National Electrical Manufacturers Association, 2011. *Digital Imaging and Communications in Medicine (DICOM)*, Rosslyn, Virginia 22209 USA: National Electrical Manufacturers Association.
- Object Management Group®, 2011. *Business Process Model and Notation*. 2 ed. s.l.:Object Management Group.
- Oculus VR, Inc., 2014. *Oculus VR*. [Online]  
Available at: <http://www.oculusvr.com/rift/>
- Pamela, Z. & AT&T Labs, 1997. Classification of research efforts in requirements engineering. *ACM Computing Surveys*, 29(4), pp. 315-321.
- Partsch, H., 2010. *Requirements-Engineering Systematisch: Modellbildung für softwaregestützte Systeme*. 2 ed. s.l.:Springer.

Pohl, K. & Rupp, C., 2009. *Basiswissen Requirements Engineering: Aus- und Weiterbildung nach IREB-Standard zum Certified Professional for Requirements Engineering Foundation Level*. s.l.:dpunkt Verlag; Auflage: 1 (12. März 2009).

Prokosch, H. U., 2001. *KAS, KIS, EKA, EPA, EGA, E-Health: Ein Plädoyer gegen die babylonische Begriffsverwirrung in der Medizinischen Informatik*, Westfälische-Wilhelms-Universität Münster: Institut für Medizinische Informatik und Biomathematik.

Rash, C. E., 1999. *Helmet Mounted Displays: Design Issues for Rotary-wing Aircraft*. Press Monographs ed. s.l.:Society of Photo Optical.

Raymond, E. S., 2003. *The Art of Unix Programming*. s.l.:Pearson Education.

Rohrbach, B., 1969. Kreativ nach Regeln – Methode 635, eine neue Technik zum Lösen von Problemen. *Absatzwirtschaft*, pp. 73-75.

Rupp, C., 2009. *Requirements-Engineering und -Management: Professionelle, iterative Anforderungsanalyse für die Praxis*. s.l.:Carl Hanser Verlag GmbH & Co. KG; Auflage: 5., aktualisierte und erweiterte Auflage (2. Juli 2009).

Sachs, M., 2014. *Master's Thesis: Requirements engineering and design of human-centered information services utilizing near-eye display devices*. Graz: Institute of Engineering and Business Informatics Graz University of Technology.

SAI Servizi Avanzati per le Imprese srl & SAP, 2013. *HIS - Hospital Information System*. [Online] Available at: <https://store.sap.com/sap/cpa/ui/resources/store/html/SolutionDetails.html?pid=0000003744&pcnty=US> [Accessed 2014 12 03].

Schmalstieg, D., 2013/14. *Lecture Virtual Reality*. Graz: TU Graz.

Schulzrinne, H. et al., 1998. *Real Time Streaming Protocol (RTSP)*. [Online] Available at: <http://www.ietf.org/rfc/rfc2326.txt> [Accessed 28 12 2014].

Shannon, C. E., 1948. A Mathematical Theory of Communication. *The Bell System Technical Journal*, Volume 27, pp. 379-423.

Shenker, S. & Wroclawski, J., 1997. *Network Element Service Specification Template*. [Online] Available at: <http://www.rfc-editor.org/rfc/rfc2216.txt> [Accessed 01 02 2015].

Siemens, 2014. *medico*. [Online] Available at: <http://www.healthcare.siemens.de/hospital-it/krankenhausinformationssysteme/medico> [Accessed 05 12 2014].

Smith, J., 2009. *WPF Apps With The Model-View-ViewModel Design Pattern*. [Online] Available at: <http://msdn.microsoft.com/en-us/magazine/dd419663.aspx> [Accessed 24 10 2014].

- Srinivasan, B., 2008. *Leadership & Project Management Champions*. [Online]  
Available at: <http://leadershipchamps.wordpress.com/2008/03/24/what-is-stakeholder-analysis-part-3/>  
[Accessed 24 07 2014].
- STANDARDIZATION, E. C. F., 2005. *DIN EN ISO 9000:2005*. s.l.:EUROPEAN COMMITTEE FOR STANDARDIZATION.
- Sutherland, I., 1968. *A head-mounted three dimensional display*. Salt Lake City, Utah, ACM, pp. 757-764.
- Szigeti, T., Hattingh, C., Barton, R. & Briley, K., 2013. *End-to-End QoS Network Design: Quality of Service for Rich-Media & Cloud Networks (Cisco Press Networking Technology)*. 2nd ed. s.l.:Cisco Press.
- Tanenbaum, A. S. & Wetherall, D. J., 2011. *Computer Networks*. 5th ed. s.l.:Pearson Prentice Hall.
- Torborg, S. & Simpson, S., 2014. *What's Inside Google Glass?*. [Online]  
Available at: <http://www.catwig.com/google-glass-teardown/>  
[Accessed 24 10 2014].
- Vallurupalli, S. et al., 2013. Wearable technology to improve education and patient outcomes in a cardiology fellowship program - a feasibility study. *Health and Technology*, 3(4), pp. 267-270.
- Velde, R. v. d., 1992. *Hospital Information Systems - The Next Generation*. 1st ed. s.l.:Springer-Verlag Berlin Heidelberg GmbH.
- Vidyo, 2014. *Personalized Healthcare WITHOUT BORDERS*. [Online]  
Available at: <http://www.vidyo.com/solutions/healthcare/>  
[Accessed 25 09 2014].
- Vorraber, W. et al., 2014. Medical applications of near-eye display devices: An exploratory study. *International Journal of Surgery*, 12(12), pp. 1266 - 1272.
- Waggoner, B., 2010. *Compression for Great Video and Audio*. 2 ed. s.l.:Focal Press.
- Wheatstone, C., 2014. *Stereoscopy*. [Online]  
Available at: <http://www.stereoscopy.com/library/wheatstone-paper1838.html>
- Whiteman, H., 2014. *Google Glass 'could transform the way surgery is performed'*. [Online]  
Available at: <http://www.medicalnewstoday.com/articles/271182.php>  
[Accessed 04 10 2014].
- Yoshinaka, K., 1985. *DISPLAY DEVICE*. Japan, Patent No. JP61198892.

## 12 List of Figures

---

|   |    |
|---|----|
| Figure 1: Cost-to-fix-an-error curve (modified from (Boehm, 1976, p. 1228)) .....   | 5  |
| Figure 2: The iterative RE process (based on (Rupp, 2009, p. 1ff)) .....  | 6  |
| Figure 3: RE process: elicitation .....   | 11 |
| Figure 4: RE process: documentation.....  | 14 |
| Figure 5: Requirements template from (Rupp, 2009, p. 177) .....   | 16 |
| Figure 6: RE process: validation.....   | 18 |
| Figure 7: RE process: management .....  | 21 |
| Figure 8: OSI model versus TCP / IP and related protocols (modified from (Tanenbaum & Wetherall, 2011, p. 42)). .....   | 30 |
| Figure 9: Typical view of HIS for SAP Business One (SAI Servizi Avanzati per le Imprese srl & SAP, 2013). .....   | 35 |
| Figure 10: Typical view of Medico (Siemens, 2014). .....  | 36 |
| Figure 11: An incomplete HL7 2.5 message .....  | 39 |
| Figure 12: First real HMD, called "The Sword of Damocles" (Sutherland, 1968, p. 298).....   | 41 |
| Figure 13: Binocular omni orientation monitor from Fakespace Labs (Fakespace Lab, 2004) .....   | 44 |
| Figure 14: The OLED based binocular interactive see-through HMD Eval-Kit (Fraunhofer Research Institution for Organics, 2014) .....   | 44 |
| Figure 15: The Google Glass (Google (a), 2014) .....  | 45 |
| Figure 16: Overview of the current Virtual Consultation process between the two investigated hospitals, shown as BPMN diagram.....  | 52 |
| Figure 17: Overview of the updated Virtual Consultation process between the two investigated hospitals, shown as BPMN diagram.....  | 60 |
| Figure 18: Overview of the current process of the Doctor's Visit in the investigated hospital, shown as BPMN diagram. ....  | 64 |
| Figure 19: If during the visit additional medical information is needed, doctors are able to create a medical request in the HIS. But often just a quick look of a specialist would be sufficient to make a decision.....   | 66 |
| Figure 20: The Google Glass near-eye display device, which serves as development basis for this master's thesis (Torborg & Simpson, 2014) .....   | 69 |
| Figure 21: Environment including context and system in which the Virtual Consultation takes place, using a schematic representation proposed by (Rupp, 2009).....   | 70 |
| Figure 22: The communication flow between two hospitals.....  | 73 |
| Figure 23: The MVVM pattern introduced by (Gossman, 2005), showing the relations between Model, View and ViewModel. ....  | 86 |
| Figure 24: UML Class Diagram of the sources and sinks architecture. ....  | 88 |
| Figure 25: Communication flow between the involved applications.....  | 93 |
| Figure 26: The Windows control application called "Control Kit". 1: Starts or stops the server socket for the communication with the Google Glass. 2: Starts the HIS client to save pictures in the hospital information system. 3: Adds a new functionality to the application which allows the user to stream audio, video or picture data between multiple application instances and Glass. .... | 95 |
| Figure 27: The Control Kit with an added Video Stream item. 1: Enables the user to choose a video source e.g. the webcam. 2: Starts or stops the video source. 3: Allows the user to change the frame rate and image resolution. 4: The preview of the video signal; with a click in the image a separate video window  |    |

opens, which can be enlarged to full screen. 5: Connects or disconnects the source with the sink. 6: Enables the user to choose a video sink e.g. a MPEG file. 7: Starts or stops the video sink..... 96

Figure 28: The Control Kit with an added Audio Stream item. 1: Enables the user to choose an audio source e.g. the microphone. 2: Starts or stops the audio source. 3: Connects or disconnects the source with the sink. 4: Enables the user to choose an audio sink e.g. the G.722 TCP Server. 5: Starts or stops the audio sink. .... 100

Figure 29: The Control Kit with an added Picture Stream item. 1: Enables the user to choose a picture source e.g. the Google Glass. 2: Starts or stops the picture source. 3: Forces an image acquisition from the source. 4: The preview of the picture; with a click in the image a separate window with the picture opens, which can be enlarged to full screen. 5: Connects or disconnects the source with the sink. 6: Enables the user to choose a picture sink e.g. the PNG TCP Server. 7: Starts or stops the picture sink. . 102

Figure 30: The HIS Client window of the Control Kit. 1: Starts or stops the HL7 server socket for the communication with the HIS. 2: Takes a new picture. 3: Changes the resolution in which the pictures should be taken. 4: Saves the picture to the hard disk or to the HIS via the archiving system or removes the picture from the list. 5: A description for the picture which gets stored in the HIS. 6: Medical records of a patient received via the HL7 interface. .... 105

## 13 List of Tables

---

|   |    |
|---|----|
| Table 1: Basic elements of BPMN (Object Management Group®, 2011, p. 29).....  | 17 |
| Table 2: Example of a Use Case Specification using the scheme proposed by Rupp et al. (2009). ....  | 18 |
| Table 3: Typical QoS parameter values for generally known applications (modified from (Mellouk, 2009, p. 31)). ....                             | 26 |
| Table 4: Categorization scheme example to distinguish use cases, proposed by Vorraber et al. (2014)...  | 48 |
| Table 5: Categorization scheme Virtual Consultation proposed by Vorraber et al. (2014) .....  | 53 |
| Table 6: Definition of the terminology which is used for the SOPHIST Requirements Template approach. ....                                       | 55 |
| Table 7: Use Case Specification of the Virtual Consultation using the scheme proposed by Rupp et al. (2009). ....                               | 56 |
| Table 8: Requirements for the Virtual Consultation using the SOPHIST Requirements Template and Object Ids proposed from Rupp et al. (2009)..... | 59 |
| Table 9: Comparison of different technologies for the proof-of-concept .....  | 80 |
| Table 10: Comparison of different software architectures for the proof-of-concept. ....   | 85 |