Picnic - a platform for sharing pictures and socialize people

Daniel Putz
Abstract
The internet itself is continuously changing. Since it was widely introduced, the users’ main focus has shifted from a communication tool, to a huge file sharing pool, to a marketing tool, to an information gathering and information reproducing tool.
Nowadays the way of sharing files needs to be reconsidered, because more and more private information is shared over the internet. The common file sharing applications are no longer appropriate for today’s purposes. Also the common way of socializing people is not up to date anymore.
Sharing files already implies a communication between users. But why do we not use the given information of shared data in order to socialize them? This thesis discusses the problematic domains of sharing images and socializing and introduces a tool named Picnic that has been conceived and implemented in order to overcome those problems.

Keywords
Personalization, Pictures, File Sharing, Socializing, Metadata.

Acknowledgement
Without the help of some persons it would not have been possible to finish this thesis with the result you see now. I want to thank the following persons for their help, commitment and constructive comments: Marcelo Milrad for supervising this thesis, Sinan Yurtcan for being involved in programming the Picnic-GUI, Christian Folie for being involved in programming the Picnic-network layer, Daniel Abouakil for his support in designing class-diagrams and program structure, my family for their support in any way and everyone who developed tools we use to make our ideas come true.
Table of content

Abstract............................................................................................................................................ 2
Keywords ......................................................................................................................................... 2
Acknowledgement ........................................................................................................................... 2
Table of content ............................................................................................................................... 3
Table of figures: pictures, models, screenshots, tables.................................................................... 5
1. Introduction.............................................................................................................................. 6
  1.1. Problem Description .......................................................................................... 6
  1.2. Disposition ................................................................................................. 7
2. Envisioning Picnic ..................................................................................................................... 9
  2.1. Key-Points of the vision................................................................................. 9
  2.2. The idea behind Picnic............................................................................... 9
  2.3. Picnic project group .................................................................................. 10
  2.4. Related Solutions ...................................................................................... 10
  2.5. Metadata state of the Art ........................................................................ 13
  2.6. Practical aspects related to this thesis ...................................................... 13
  2.7. Limitations ................................................................................................. 14
  2.8. Transition to the theoretical foundation .................................................. 14
3. Theoretical foundations ........................................................................................................... 15
  3.1. File Sharing................................................................................................. 15
  3.1.1. Users’ motivation on sharing Data/Personal Information .................. 15
  3.1.2. File sharing and Privacy issues ......................................................... 16
  3.1.3. File sharing tools and Picnic ............................................................. 16
  3.2. Metadata ..................................................................................................... 17
  3.2.1. What is Metadata and why is it useful? .......................................... 17
  3.2.2. Where does Metadata come from? ............................................... 19
  3.3. Socializing People....................................................................................... 19
  3.3.1. Socializing via Metadata, how can Metadata be used in the case of Picnic? .... 21
  3.3.2. Location / Geospatial Metadata ...................................................... 21
  3.4. Transition to the Conceptual Approach .................................................... 22
4. Conceptual Approach................................................................................................................. 23
  4.1. A general view on Picnic............................................................................ 23
  4.2. Scenario as a basis for design ................................................................. 23
  4.3. A view on the communication of Picnic ................................................ 25
  4.4. Privacy, respective Personalization of Files .......................................... 26
  4.5. Envisioning the way of socializing people with Picnic ......................... 27
  4.6. Socializing People with Picnic ................................................................. 27
  4.7. Design Assumptions ............................................................................... 29
  4.7.1. Location Metadata .......................................................................... 29
  4.7.2. Intelligent Matching System ........................................................... 29
  4.7.3. Marked picture parts ....................................................................... 31
  4.7.4. Extend pictures by adding new information ................................... 33
  4.8. Transition to the Architecture and Implementation .......................... 33
5. Architecture and Implementation ......................................................................................... 34
Table of figures: pictures, models, screenshots, tables

3.1 The Different levels of multimedia information by John R. Smith ................................................. 17
3.2 The Semantic Web stack, as presented by Tim Berners-Lee .............................................................. 18
4.1 Picnic Screenshot – First prototype ................................................................................................... 23
4.2 Socializing Message .......................................................................................................................... 24
4.3 Description of a possible system ....................................................................................................... 25
4.4 Order of events of file-sharing between two users ........................................................................... 26
4.5 Screenshot of the picture clearance ................................................................................................. 26
4.6 Description of the metadata-process ............................................................................................... 28
4.7 Visualization of a possible architecture that relates metadata information to external information sources to achieve the better result .................................................................................. 30
4.8 Marking objects on a picture and describing them gives different advantages ................................ 31
4.9 Two pictures connected through a profile – in relation to each other by the objects ...................... 32
5.1 Technical Description of events between client, server, and database ............................................. 34
5.2 Class Diagram for the whole Picnic project. The red-highlighted parts are important for the prototype that is programmed for this thesis .................................................................................. 35
5.3 The order of events by updating the users’ information lists ........................................................... 38
5.4 Screenshots of two different user interfaces ................................................................................... 39
5.5 Code that describes the container of a user ...................................................................................... 40
5.6 PHP-code for retrieving metadata .................................................................................................. 41
5.7 Order of events; starting with the picture’s metadata, ending with a message to the user’s client ........................................................................................................................................... 43
5.8 Matching two users by comparing first the tables, then the table content by using an algorithm on the content, checking if a match occurs and at last saving the match to a database ........................................................................................................................................... 44
5.9 Extracting matches out of the metadata-strings; relating them to an information source called places ............................................................................................................................................... 45
5.10 Extracting an event out of the information that has been extracted before ..................................... 46
5.11 Intersection on Places. 3 Matches, means 30% match for user 1, 50% match for user 2 ............... 46
5.12 Different levels of a match .............................................................................................................. 47
6.1 The whole process visualized as a circle of actions .......................................................................... 51
8.1 Use Cases Client and Server .............................................................................................................. 63
8.2 Screenshot Picnic – Picture preview windows .................................................................................. 78
8.3 Screenshot Picnic – Message window ............................................................................................... 78
8.4 Screenshot Picnic – Add a new group ................................................................................................. 79
8.5 Screenshot Picnic – User information ............................................................................................... 79
8.6 Screenshot Picnic – Different views. Incl. right-click-menu ............................................................. 80
8.7 Screenshot Picnic – Search and add a new user ................................................................................. 80
1. Introduction
The internet is permanently changing. Since it was invented the main users’ focus has shifted from a communication tool, to a huge file sharing pool, to a marketing tool, to an information gatherer and reproducer. The upcoming web 2.0 is transforming the use of the web with all its capabilities. It allows users to express themselves to the whole internet community, share files and information, communicate in different ways over the net and supports its users in their daily life.

Users are no longer simply users of the internet; but become a part of it. More and more material is provided by the users themselves. They amplify the web by adding personal files and share this data, either with certain groups or just with the whole internet community.

Communication by sharing, even texts, pictures or video, is state of the art and the beginning of the web 2.0. As O’Reilly, founder of the web 2.0, puts it: “You can visualize Web 2.0 as a set of principles and practices that tie together a veritable solar system of sites that demonstrate some or all of those principles, at a varying distance from that core” [28]. Wikipedia, the online encyclopaedia, defines the Web 2.0 as “Web 2.0 refers to a supposed second-generation of Internet-based services — such as social networking sites, wikis, communication tools, and folksonomies — that let people collaborate and share information online in previously unavailable ways” [29].

Metadata supports the semantic web by containing information about every Rich Multimedia Object. Machines can interpret such metadata and perform results or extract new information by passing such data and putting it in a relation to each other. Smart algorithms might become powerful information extractors with good results and become even more powerful by generating new information out of the combination of information and knowledge.

1.1. Problem Description
Sharing personal files, pictures, diaries and other content with friends via the World Wide Web became very popular over the last couple of years. Content systems for pictures and videos like flickr.com, youtube.com and friendster.com are quite powerful by gathering user-related information that is hidden in the files and becomes obvious by relating files with each other.

The World Wide Web users offer more and more content on the web to their friends, family or even to the whole internet-community. Tools and websites such as Flickr, Kazaa or MySpace offer services to publish pictures, videos, etc, on the web. Often users are forced to publish their personal information to the whole internet community, even if they do not want that.

On the other hand the internet is used to meet new people and get (anonymously) in contact with strangers without having to meet them in person and without the possibility to ambaries themselves. The most important fact nowadays for the users is personalization [1] and at the same time the possibility to meet new friends who share the same interests [2].

As Olsson and colleagues wrote in 2005: “We found that peoples’ willingness to share depends on who they are sharing the information with. For example, there is promise in creating designs for specifying preferences that reveal, in a selective manner, finer-grained
choices within a hierarchical scheme, allowing users to navigate to the level of precision they are most comfortable with.” [1]

Terveen and colleagues wrote in 2005: “People are social creatures—fundamentally so. We look for other people for a multitude of purposes: dating and eventually marriage, pursuing shared interests, addressing community issues, solving technical problems, or maybe just having a good conversation.” [2]

In this content, an important question to be asked is: “How do people actually communicate on the internet”? Nowadays the communication between young people via instant messengers like MSN-messenger, ICQ or Skype is common. Through these messengers and on other websites, people share pictures and get in contact this way. But it is often not that easy to exchange this data. Especially the exchange of rich multimedia, such as pictures, sound, etc., is more or less problematic. Pictures, for example, need to be uploaded one-by-one and this requires a fairly long period of time. The users of these services do not really have the possibility to personalize their pictures as well—mostly they have just a Boolean option to show them to either everyone or to no one at all. Putting up the possibility of a personalization is quite complicated. A solution is needed that offers the opportunity to show content in a personalized way. A user should have the choice to show content to friends, family or co-workers without any limitations and barriers, but should have the possibility to meet new people without the need of offering personal information at the same time.

In this thesis, I have identified the following questions in order to explore a possible approach of sharing content in a different way as well as to investigate what kinds of technical possibilities are given for developing a solution:

1. Which methods and techniques already exist, and which methods and techniques can be implemented to provide innovative ways for people to share rich multimedia material (images, sound, etc) based on personal and collective interest?

2. How can a system be designed to support the features described in the question above and which is the most adequate and useful technical approach for its implementation?

3. How can a system be designed to provide sharing (information on a personalized way) and socializing (users with the same interests) at the same time?

1.2. Disposition

In the first section of this thesis I discussed the given problem domains. The next section will introduce the vision of a program called Picnic. In this section, I will describe the key points of the vision and lead to other solutions that are related to the given one. In section 4, I will describe the practical approach that is done related to this thesis, before I finish by explaining its limitations. In the conclusion, I will summarize my major arguments.

The theoretical foundation, section 3, will discuss the topic on a scientific background. The researches presented in this section are related to the problem domain or to similar ones. They describe the main topics of the thesis, ‘File sharing’, ‘Metadata’ and ‘Socializing people’. 
Section 5 introduces the given solutions as an architectural approach of Picnic. I will give an introduction through screenshots and an idea about the information flow and communication of the program. I will also describe how people should be socialized with each other through Picnic and how an intelligent matching system could possibly look like.

Section 5 describes, related to section 4, the solutions from the technical point of view. It describes the architecture and introduces the tools that should be used for the practical approach.

In section 6, the conclusion will summarize the major focus of this thesis. Some advantages and drawbacks will be described. A reflection upon the work and some ideas for future research will be given as well.
2. Envisioning Picnic

This section will explain the given problem domain and introduce a solution that may offer good ways to overcome these problems. Some related approaches give an overview on the problem domain. The last section of this chapter introduces the limitations that will rule out everything that will not be implemented in this thesis.

2.1. Key-Points of the vision

This thesis aims at exploring the possibilities of how an image sharing program could look like that takes into account the privacy issues users have by sharing their content; in our case pictures. Furthermore it examines the possibilities of how a system could look like taking this privacy issues into account, but socializes at the same time by analyzing the metadata of the shared Rich Multimedia Objects. By that, users will have the possibility to share content in a personalized way but still get into contact with different unknown users by the profile that is built based on their metadata.

This solution of sharing and socializing has been chosen by considering different designs approaches in related areas. There are alternative ideas of sharing Rich Multimedia Content, especially pictures and movies. The given approaches question how information may be widely spread, how different information can be analyzed and stored to any object’s metadata to describe such objects as detailed as possible and how users might be socialized with each other. But the given approaches seem to miss a way that uses different rudiments and tries to get much better solutions in an easier way. In fact the domain of matching users by their content and socializing them on such a system as Picnic is an innovative approach.

Picnic is a program that aims to solve such a problem and wants to establish a platform for matching users by their content. These main features of Picnic are presented below:

- Rich Multimedia Content may be shared via Picnic in a personalized way. That means that a user can choose by him-/herself which other users shall have the possibility to see and/or download the content.
- The metadata of such objects will be analyzed and matched with the metadata of other user’s content.
- Users get socialized with each other.
- The Rich Multimedia Objects as well as the metadata is hidden to the other users. Nevertheless the system can inform the users about other users’ interests, anonymously without offering personal information.

2.2. The idea behind Picnic

The idea of Picnic was a spontaneous idea. I saw a lot of people trying to share their pictures with friends and family by using tools that had actually not been made for amateurs. Some tried to send pictures via email, but there occurred problems with large files of 1,5 MB. Other people built websites and reduced the pictures on the website. The result was that one needed to wait more than five minutes to load the page to see a couple of pictures. A third group tried to send the pictures via instant messenger.
The problem most users faced was how to reduce the picture size. None of the amateur-tool provides the possibility to reduce the size of a file because there is usually no need for it. Also, the users did not know how to reduce pictures, but every user knew how to work with an instant messenger. So, the idea to build a messenger that shows pictures in a small size was born.

After some brainstorming and discussions with friends and colleagues I came to realize that it was not only the fact of sharing pictures that was very important for the target group but rather the possibility to choose the visitors. Sharing pictures with friends was important to them, choosing with whom to share was even more important. Nevertheless, socializing these users with each other seemed to be important too and should not be foreclosed. So I searched for a way to give users the possibility to share privately but get into contact with strangers.

Picnic builds a platform to execute an algorithm that socializes different users. This algorithm receives information from picnic and returns matches. These matches are afterwards used to socialize users with each other.

2.3. Picnic project group

In the beginning the idea was to program Picnic as a small application by my own. But during the planning process it became clear that the project was not possible to be completed on my own. Therefore, I decided to involve some specialists from the University of Technology, Vienna; Sinan Yurtcan and Christian Folie. They got involved after the first iteration and planning process and their task was/is the implementation of the program.

They are involved into the (File sharing) Picnic project only and not in the prototype that visualizes the socializing algorithm. They act as independent employees towards the project and are only involved into the programming process, responsible for GUI-, network- and interface programming. The project is continually managed, planned and coordinated by my person in permanent consultation with the rest of the team.

As it will be shown in the following sections, the program is implemented in Java. Java was chosen because of its platform independence and its advantages in object-oriented programming. Due to the fact that different people worked simultaneously on the program, the object-orientation offered the advantage to connect the different parts through predefined interfaces. In the class diagram in section 5.2 one can have an insight into the structure of the program. The use cases and test cases in the appendix visualize even deeper how Picnic is structured and in which way it has been developed.

2.4. Related Solutions

There are different solutions that represent parts of Picnic. In this section I want to explain the ones that have the most significant connection to Picnic. I exclude instant messengers because first of all I see it as a prerequisite that the reader should know what an instant messenger looks like and how it works.

First of all there are a couple of commercial solutions that provide the possibility to share pictures. During the last years the program Flickr [25] became very popular. People have the possibility to upload pictures to their personal photo album and show pictures to a small group of people or to the whole internet community. The problem with Flickr is that
all pictures need to be uploaded on a server. There is no possibility to show pictures that are saved on the own computer, but they need to be uploaded. Flickr has some privacy issues as well. The users’ private pictures are saved on a server, somewhere and therefore the user needs to trust the company. Picnic does not load any picture on a server; the pictures stay on the user’s computer and so users do not to rely on a system.

Flickr takes as well the possibilities of tagging pictures with metadata into account. Users have the possibility to tag a picture with information. The tags are then used for the search on a website. Flickr uses it to get better search-results (which, in my opinion, are good ones). A different approach of how the Flickr’s pictures’ metadata could be used is to relate it to different areas at loc.alize.com. On this webpage the pictures from Flickr are related to the Google map and the user has the option to search for a location by the users’ pictures metadata. For example, if a user has related any picture to the city of New York, the user can search for a keyword. If this keyword is included in the metadata of the picture s/he retrieves not only the picture, but also the place that the picture is referring to. The combination of data and place gives therefore a much better result with its new information. Such relation of different databases and information pools extract new, better results.

A not web-based but client-based solution for sharing picture was developed by Skype [26] together with Kodak. A software application that provides the possibility to show pictures on each others computer without the need of uploading pictures somewhere. While two persons are talking via Skype they have the possibility to exchange pictures. The users can see each others’ pictures in real-time. But they need to talk with each other and they need to show ambitions for communication at the same time. The tool only provides to show pictures to one person though. Indeed a personalization is given to 100 percent but the user is restricted to sharing pictures with one user at a time.

A solution that is not provided for sharing pictures rather than for sharing content are distributed systems [20]. Distributed systems offer the possibility to show folders that are stored on a computer to other users. These systems mean as well the risk to manipulate the access-rights other users have. For example user1 has the access-right to read and write while user2 has only the access-right to read. In that way personalization is possible, furthermore personalization is the advantage such systems have. The idea of distributed systems is that a user can see other users’ content as it is his/her content. Even if the information is stored on the other one’s computer, this content is made available with a distributed system. There are different applications that are called distributed file systems [20]. Some experts define a distributed file system as “a client/server-based application that allows clients to access and process data stored on the server as if it were on their own computer. When a user accesses a file on the server, the server sends the user a copy of the file, which is cached on the user’s computer while the data is being processed and is then returned to the server” [19]. While it is depending on the tasks of a distributed system other experts define more general as “provide a common, consistent global view of a file system, name space, time, security, and access to resources. To provide this common view, there are numerous constraints and requirements on all participating systems; thus, distributed systems are generally referred to as a tightly coupled system.”[20, page 3].

I use Galli’s definition [20] because it might fit better. Even the definition of ‘server’ is diverse. Every computer can behave like a server, or be one. And every application that imparts between server and client may be a kind of distributed system. Using Galli’s definition as a basis, Picnic can be regarded as a kind of a distributed system. It gives the
possibility to share and show files to other users. The way of “how and where the files are used” differs, but the system itself may be called a distributed system. To clearly define if Picnic is a, or is not, a distributed system, does not seem possible; it is in that way up to everyone’s definition. The way I defined a distributed system, a perhaps more general definition, includes as well the Picnic program.

Even distributed systems make it possible to share files via the internet. They are not commonly used by non-professional users, often due to usability reasons. The end user prefers easy solutions (even if they are not good) to share files. Everyone knows Napster and how easy file sharing can be. Therefore a couple of file-sharing tools are still available such as dc++ or Kazaa. The difference between Picnic and these tools is the privacy issue; users do not have the possibility to personalize content. A user needs to share with the whole community. Traditional file sharing tools are not used to personalize information because the idea is to divide a file and download from different users separate parts of one and the same file. A user downloads one part from user A, another part from user B and so on, until the file is completed. This includes that different users have the same file. These file-sharing tools are interesting for downloading films, music and pictures the major internet community is interested in. Personal pictures are in general only saved on one user’s hard drive.

Shifting the focus from sharing files to socializing people, I will now describe how the personal files can be used to socialize users. From the social point of view there are a lot of solutions that try to get people together. Dating portals boom on the internet, in England “toothing” became a national sport and chat rooms define different themes to socialize people with the same interests or age.

To show the difference between Picnic and such solutions I picked a mobile phone solution that socializes strangers by their profile as well by their local position. The localized dating service, developed by the University of Birmingham [11], matches people by their profile (the profile is defined by the user him/herself). If two people fit together, by common interest for example, these profiles are saved into a database. If the situation occurs that these two people are nearby each other, their mobile phones vibrate and give information about the other one’s position, the interests, contact information and so on. These people than have the possibility to chat or directly meet, because they are geographically close to each other [11].

Why did I choose this example? Of course, Picnic will not locate the position of a user. But the fact that the local dating service uses not only the profile the user defined by his-/herself but the location as well, the tool shifts from the university of Birmingham close to the solution Picnic is to provide: Picnic analyzes the metadata of a user’s pictures and compares them to others. If some matches occur Picnic socializes these users. The information to the end-user will not be “the user is nearby”, but “the user has the same interests because of his/her pictures that match to yours”.

The advanced version of such search would be the search in different data sources and combine the extracted knowledge from the picture with the knowledge the system receives from the external information source. For example a date and place could not only describe place and date, but also an event that took place at exactly this time at place. This information is not stored in connection to the picture but can be researched by a system that has the possibility to infer into different information sources. The new information can then be added to the former source and in that way the information can be enlarged. A scenario about this topic will be described in section 4, the conceptual approach.
At this point the Picnic approach shifts into the field of artificial intelligence. An algorithm that relates different kinds of information to each other and thus it becomes a decision maker. This already means a kind of intelligence. As Shannon [18] describes “Artificial Intelligence systems are computer programs that simulate some of the characteristics of human thought - the ability to learn, reason, solve problems and understand ordinary human language. With the advent of relational databases, computers changed from mere data processing machines into decision support systems. Since these relational systems facilitate finding and using the relationships between the various types of data stored, they can be used to develop applications that will aid the analysis and decision making process.” The field of the AI will only be a small part of this thesis but is important to be taken into account.

2.5. Metadata state of the Art

Metadata is one of the key elements of the new generation of Rich Multimedia Files and will soon be included into nearly every file. Even video film-, picture-, music-, text- or other computer readable files will contain metadata. By that the semantic content can be described to a computer by a human. Metadata describes an object itself and its attributes.

Metadata can be updated by a human and/or a computer. The computer may add ‘static’ information like date or size of a file, the human may add ‘various’ information such as ‘what is in the picture’ and ‘where are the most important elements in the picture’. In reality, a user takes a photo and the camera adds date and size to the metadata file while the user adds place and element to it. A combination of these two knowledge sources - human and computer - create the metadata file and add even more information to it.

Metadata has different tasks: It may provide content that explains the object, provide mutability to make sure certain information gets or gets not changed, provide logical functions that allow logical reasoning. There are three layers of logical function lying on top of each other: the bottom is the sub symbolic layer that contains the raw data. Metadata is on the symbolic layer and describes the content of the raw data and the topmost logical layer contains metadata that allows logical reasoning using the symbolic layer. [15][16]

More detailed information about metadata related to this thesis is given in section 3.

2.6. Practical aspects related to this thesis

Beside the theoretical research, this thesis contains as well a practical part that shows example wise how described approach might look like. The approach will be implemented as a prototype. The parts the prototype will contain are described below.

First of all the program will be a prototype. The idea is not to build a program that may be used commercially, but to show what the functionality of such a tool may look like. It shows a scenario and leaves different parts open that might be completed by further research and work.

The main focus of the practical part related to this thesis is about how to socialize users by their metadata. This will be realized by different PHP-files which read metadata and visualize the matching order of events. The script will then send a message to the Picnic tool.
The prototype is to show how it is possible to read and match metadata. The prototype will offer possibilities to add, alter or create further ideas and solutions.

Picnic is to build a framework for socializing users and is therefore an important tool for the matching prototype. You can evaluate the complete planning process by reading this thesis. Most diagrams, use cases, scenarios and brainstorming will be added as well to the appendix.

2.7. Limitations
In order to delimit the scope and efforts of this thesis the following parts will be, or will be not, considered:

- Metadata Creation: The prototype that will be programmed related to this thesis should not create metadata than only use it. It might have the possibility to add metadata information to an object. This aspect will not be part of the thesis.
- Algorithm to match Metadata: Matching users by their Metadata can be a small but as well an enormous task. This thesis will not be about “how to build a perfect algorithm that gives perfect results”, but about the possibility to build an algorithm. Thesis and prototype give incitations to implement an algorithm but will not take the main focus on building a perfect working algorithm.
- Picnic implementation: The Picnic program is described in this thesis in a theoretical way. Picnic is a theoretical application and a prototype might be implemented but is not directly related to this thesis.
- The thesis describes a conceptual design and takes the field of human computer interaction only partly into account.

2.8. Transition to the theoretical foundation
During the last section, I offered the reader an insight into the idea of Picnic and the way it is to solve the given problem domains of socializing and personal sharing by describing the key-points of the vision and some related solutions.

The following section, theoretical foundations, will introduce some related problems and solutions, and by that shape this thesis from the theoretical point of view.
3. Theoretical foundations

In order to describe a solution for any problem domain it is important to read about similar solutions that has been developed. A good solution should first discuss the advantages and disadvantages of the solution itself, as suggested by other researches. This section explores the problem domains by discussing related efforts undertaken by other researches.

In the last section “related solutions” I have given an insight into existing systems and the difference between these systems and my solution. The theoretical foundation will now explain my approach in detail and founds the ideas and solutions by former researches. I will also compare these solutions to existing ones to show the important differences.

3.1. File Sharing

Sharing files has become one of the most important reasons for popularity of the internet in the middle of the 1990’s. Everybody had the interest to share files. Due to Shawn Fanning’s Napster internet users worldwide knew how file sharing worked. The whole world shares files up to today though it is against federal law. But which files are users willing to share?

Nowadays the information we store on a computer are different from the information we stored on a computer in the mid 90’s. More and more personal content is produced. More and more personal data is saved on a computer.

3.1.1. Users’ motivation on sharing Data/Personal Information

There is a big difference in the kind of files and the kind of information these files contain, a user is willing to share. Often this fact does not seem to be taken into account by developing software applications.

Judith S. Olson and colleagues described in the paper “A Study of Preferences for Sharing and Privacy” [1] from the CHI in 2005 what kind of information users are really willing to share. They studied a number of questions and the most interesting for this paper are the following ones: “What are people’s concerns with sharing information? How do people differ, where are they in agreement, what kinds of people and kinds of information do they treat similarly and differently? Could we provide people with shrewd guesses as to their access choices, which they could then modify with ease?” [1].

As an expected result the test-users said that it depends on the person in combination with the information, if they were willing to share or not. We know that kind of behaviour from ourselves, often from well-known stereotypes; For example, a man would not show pictures of his childhood to his new girlfriend as well as a teenager would not show party pictures to the parents.

But often the given solutions do not offer the possibility to choose whom to share with but with everyone. This might lead Olson to the concluding argument: “We believe that such findings can provide guidance to the design of access controls and interfaces, which could make specification easier for the end user. We foresee designs that allow for control the grain size of definitions of groups of people and types of items [1].”
3.1.2. File sharing and Privacy issues

The previous description shows that sharing files is a common act in the internet. The internet is about sharing information and files. People like to exchange their files - as long as they can choose which files are shared.

But the usability of some tools is sometimes very bad in quality, so that users think they are not sharing any files while they are actually sharing all files on their hard disk – like Nathaniel S. Good and Aaron Krekelberg researched in 2003 [4]. They analyzed the file sharing tool Kazaa, one of the most powerful and common file sharing programs on the internet nowadays and found out that the major security lack is the user himself. Sharing information that s/he do not want to share is his/her own fault. But the users trust the software and his/her understanding of the software interface. If the software leaves space for interpretation the user will interpret it the way s/he is able to. Because of that the users will still be the victim by own misinterpretation, though the program itself becomes a complice. “While Kazaa is not a security application like PGP or personal firewall software, it nonetheless has privacy implications for its users. It must help them ensure that data is not accidentally shared with others [4]”.

Based on a list of security guidelines that are provided by Whitten and Tygar, Good and Krekelberg [4] created a modified list of guidelines adapted to Peer-to-Peer file sharing applications. These guidelines take the unique demands of continuously connected systems into account which distribute personal files. They are meant for the use of the picnic file sharing. “Peer-to-Peer file sharing software is safe and usable if users are:

1. clearly made aware of what files are being offered for others to download
2. are able to determine how to share and stop sharing files successfully
3. do not make dangerous errors that can lead to unintentionally sharing private files,
   and
4. are comfortable with what is being shared with others and confident that the system is handling correctly” [4].

Flickr offers a better approach for personalizing files, compared with Kazaa: The online picture service, owned by the yahoo!-group [25], offers the possibility to create user groups. The user has the possibility to invite other users to his/her group. At the same time the user can personalize pictures to such groups. In that case the different pictures are only shown to the group of people the user want to show his/her pictures to. The solution is quite smart. Flickr takes into account that users might prefer to personalize pictures – and keep their privacy. A problem that might occur is that choosing a group is not very efficient. A user can overlap in groups. It would be easier to have the option to choose the user directly from a group.

3.1.3. File sharing tools and Picnic

Right now there are two different popular ways of sharing files. On the one hand there are such traditional nets like Gnutella and Kazaa. On the other hand, there is a new solution that is called BitTorrent. Even systems such as Kazaa are still quite frequently used; a lot of the users already start sharing files via BitTorrent, mostly driven by law and performance. Sharing files is certainly part of the Picnic idea but has nearly nothing in
common with the given file sharing programs. These programs are aligned to share files that are public to everyone. The idea is to connect users to a net and download a file from any of these users. While these files are redundant the advantage is that a user might go offline but the file is still available on another one’s computer and the download can be completed by downloading the file from the other one’s computer. However, my approach takes personal files such as pictures into account. Private pictures are not spread over the whole internet community, but rather saved on only one hard disk. Pictures are only shared with friends. So, instead of a one to many (one receiver, many senders) the connection is one to one (one receiver, one sender) [9] [10].

Therefore the idea common file sharing systems are based on, that to make as many files as possible public and available for download from anyone out of the network, does not apply to Picnic.

3.2. Metadata
To get a better understanding about what metadata really is and what kinds of possibilities it offers, the following sections will explain metadata in greater detail. Metadata is the basis for the Web 2.0 and will become probably common in the next couple of years. It is important to mention that the prototype, which is built on the idea that metadata is contained in every picture; this is as well the idea of the semantic web.

3.2.1. What is Metadata and why is it useful?
The web 2.0 includes (and is mostly built on) aspects of the semantic web. It provides information that is readable for machines and can be ordered, related, researched and so on by computers. Intelligent computer systems will have the possibility to reproduce true, better and more (detailed) information than any information gatherer is possible to provide nowadays. Tim Berners-Lee, one of the ‘fathers’ of the semantic web, originally expressed the vision of the semantic web as follows:

“I have a dream for the Web [in which computers] become capable of analysing all the data on the Web—the content, links, and transactions between people and computers. A ‘Semantic Web’, which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of
trade, bureaucracy and our daily lives will be handled by machines talking to machines. The ‘intelligent agents’ people have touted for ages will finally materialize.” [16]

One of the keys to the web 2.0 is metadata. Metadata is machine-readable information for the web. The W3C Metadata Activity addressed the combined needs of several groups for a common framework to express assertions about information on the Web, and was superseded by the W3C Semantic Web Activity [6]. Every image, movie, program or other Rich Multimedia Object has the possibility to save metadata. Such data describes the document itself, its properties and content and any more information that is worth to know about such a file.

Take a picture for example. A picture has a semantic content; an object that is viewed on the picture, a situation, a person, a building etc. The computer does not know anything about this content; the computer only knows the file-format and some general information like size and dimensions. Here metadata becomes a bridge: the user can add the knowledge s/he has about the object that is shown in the picture into the pictures metadata. For example “university of Wuppertal, Germany, main entrance”. In that way the computer understands the content and can use it to get better results when it is, for example, searching for a picture of this category. Sarvas and colleagues, 2003: “Human memory and browsing can manage the content of ten or even a hundred pictures; when one has thousands of pictures, the ten pictures one is looking for are effectively lost. Management of personal digital media must be automated. However, effective organizing, searching, and browsing of digital media based on their content is currently not possible in most consumer media management tools [5].”

The general notation language used for describing metadata is xml. Depending on the type of file the order might be slightly different but the structure is still the same. That makes it easy for every web-application to read and work with such information.

Taking a look at the structural design of metadata, we need to divide it into three different layers: The raw multimedia data, the structural description and the semantic description. See figure „Different levels of multimedia information” by John R. Smith [7].

The subsymbolic abstraction level covers the raw multimedia information represented in well known formats for video, image, audio, text, metadata, and so forth. These are typically binary formats, optimized for compression and streaming delivery. They are not necessarily well-suited for further processing that uses, for example, the internal structure or other specific features of the media stream. To address this issue, we can introduce a symbolic abstraction level which provides this information. Information on this level is
typically serialized in XML. The standards that have been proposed and partly used in the literature for the representation of multimedia document descriptions mainly operate in this middle layer.

The third layer (the logical abstraction level) provides the semantics for the middle one, actually defining mappings between the structured information sources and the domain’s formal knowledge representation. An example of this is the Web Ontology Language (OWL). In this layer we can make the implicit knowledge of the multimedia document description explicit and reason with it—for example, to derive new knowledge not explicitly present in the middle layer. In that way all information can be structurally and semantically added to any rich media document [7].

To summarize the current semantic web the figure by Tim Berners-Lee was implemented. It describes a walk-through of the structure, or even more the architecture, of the semantic web with the main goal of providing trust. Machines shall have the possibility to find trustable solutions to human problems. Anyhow, the figure describes quite well the different layers starting with the Unicode, followed by xml through rules to the trusted software.

3.2.2. Where does Metadata come from?

I will only describe shortly how metadata is produced. This is a topic for further studies and is not a part of the actual problem.

The question where metadata comes from might be dispensable according to Adrienne Tannenbaum (2005); “Whether we realize it or not, metadata is already everywhere. What we are missing is an organized view of today’s metadata, despite its origin [8].” That may be true, but nevertheless metadata needs to be created somehow.

Practically, metadata describes a picture’s attributes and content. Metadata could be created a long time after a picture has been taken while most information is available at the moment the picture is taken. Sarvas and colleagues (2005): “Both the information in the minds of people at the time of capture, and the information in the technical devices the people are using at the time of capture (e.g., camera, camcorder, calendar, phone, PDA, game deck, etc.). [5]”

The problem is that the user often does not have the time to write a description of the picture’s content. This is why such a system was created by the research group from Sarvas to do this job. By comparing given information such as time, date and location the system can research the picture and is then able to analyze the object that was photographed. The user only needs to verify this information. Such a short process can add much information to a picture [5]. Afterwards this metadata can be used from such application tools like the one I am proposing.

3.3. Socializing People

“People are social creatures – fundamentally so. We look for other people for a multitude of purposes: dating and eventually marriage, pursuing shared interests, addressing community issues, solving technical problems, or maybe just having a good conversation [2]” (Terveen et al; 2005).
People use different tools on the internet to get in contact with others. The choice of the tool depends often on the age, the social surrounding and the interests. The internet is spanning these different groups of people. Young people, middle aged people and even seniors use the internet to meet new people to chat, share interests or meet someone to get to know better.

Different companies meet these needs and provide services that socialize people with each other. Users have the possibility to choose from a huge pool of people and can easily find a person s/he would like to contact. Some examples for these websites are match.com or liebe.de. Social matching systems attempt to make decision-making easier by identifying specific target groups.

But these websites have one weakness: The user himself has to fill in all the information about him-/herself. This procedure takes time and is a tiresome job. Also, the interests for examples do not get ranked. People may also add false information to get better rankings. This procedure definitely influences the results of these match-machines.

In 2005 Loren Terveen and David W. McDonald researched the way of social matching. The traditional way is a recommender system that helps users to pick from a large set of items. There are two different kinds of recommender systems: The collaborative and the content-based system. Collaborative recommenders combine many users’ preferences to recommend terms to a target user. The automated collaborative filtering technique matches a target user with other users who have similar preferences. It then recommends items these matched users rated as highly interesting and the target user has not rated at all [2, page 403]. These systems work quite well. But in order to get a match, these users need to show an important part of their privacy. Users need to inform the algorithm about the needs, wishes, wants and maybe painful secret information – to get a good match.

Therefore the solution that is presented in this paper is smarter as it matches people with each other on the basis of the information their pictures contain. So, it might be more trustable because the users do not have a chance to manipulate their profile in any way.

A quite similar solution is described by Stephan Bauman of the German Research Center for Artificial Intelligence [3]. He describes a theory to ad-hoc socialize people with each other by the music these people are listening to. “A bag full of MP3s should be sufficient to compute a musical profile. Either by using the audio content and apply the standard techniques found within Music Information Retrieval (MIR) or by relying on the ID3 tags which offer metadata about the artist, album, song and genre” [3]. And based on his researches Bauman is convinced that this kind of socializing could work; “The simplicity of such an approach will be surely a further argument in favour of using it. As far as I know there is no evidence about the joy-of-use of such applications”. This declaration might be adopted for the Picnic solution due of the similar concept of pictures and music integration

Picnic users do not need to disclose their private life; they do not even need to show their pictures to a stranger. They only get in contact with each other by the fact that the system tells them that they have something in common by the content of their pictures. As well Picnic provides the combination of “Implicit Social Matching”, “Opportunistic Social Matching” and “Social Navigation” [2, page 407-409]. The matches may become better considering this fact. The matching process will be more reliable by knowing the user’s activities or what s/he has been doing over the past weeks combined with static data and the common information considering the interests.
3.3.1. Socializing via Metadata, how can Metadata be used in the case of Picnic?

Picnic tries to socialize people and at the same time respects people’s privacy. This concept might sound like two opposing things. How can someone be socialized with someone else, without offering private information about him-/herself?

The idea is to compare metadata with each other. Picnic is a tool to share pictures. These pictures contain a lot of metadata. Even the users do not add metadata to pictures by themselves, the pictures contain, for example, information when and where the picture was taken and which camera model was used. This information might even give us an idea about the interests of these peoples. Picnic reads such information and sends it to a database. On this database an algorithm compares the metadata of different users with each other. If there are some similarities the users might have the same interests.

More details about this approach will be given in section 4.3 at the coming chapter.

3.3.2. Location / Geospatial Metadata

Even though it should not be part of this thesis, will I take location metadata (called as well geospatial metadata) into account. This sort of metadata is quite powerful, because “the location where a photo was shot is important because it says much about its semantic content, while being relatively easy to acquire, index, and search” [17].

Today’s portable electronic devices often merge a number of different functional components together, and cameras are one component. A number of high-end digital cameras allow connection to a handheld GPS device, and it is anticipated that more and more cameras will include GPS chips in the camera itself. In Japan, over six million cell phones with both camera and location-awareness (from either GPS or cell-based location) are already in active use [6]. The precision of this equipment depends on location-awareness technology. The possibilities location metadata possesses are described in section 4, the conceptual approach.

PDAs, laptops, digital cameras and other portable devices are increasingly beginning to have location awareness via GPS or wireless LAN. The GPS device keeps a time-stamped location history whenever it is successfully tracking GPS satellites. By connecting the timestamp to the GPS device a picture can easily be related to the GPS coordinate.

A device that takes pictures or videos and can already be located out of his nature is a mobile phone. The location can be saved onto the picture as a lat/long coordinate. The EXIF header in JPEG photographs supports the inclusion of lat/long coordinates. “More specifically, given a location history, we construct a function Loc(t) and Prec(t) which return a lat/long estimate and a precision, indicating the expected location of the user based on his or her location history. At present, Loc(t) simply looks up the temporally nearest location-history entry prior to t, and returns the recorded lat/long coordinates; for GPS, Prec(t) is fixed to 10 meters”[17].

The question about how to retrieve GPS-location-related-information, and as well related pictures, was researched by David Martinez who researched in the field of data retrieval. His thesis “Introducing location related aspects to mobile multimedia environments” [21] discusses as well some requirements such a system needs to have;

- The system needs to receive location information from the user periodically.
- The system has to authenticate the users.
• The user has to be able to manage his groups, contacts and preferences in order to define the information s/he wants to receive.
• The system has to combine user preferences, contents location and user’s location in order to send the right information.
• The user has to be able to send messages to other users.

3.4. Transition to the Conceptual Approach
The theoretical foundation as it is described above will now be connected to the conceptual approach. The different theoretical parts will be included as followed.

Researches Ohlson and colleagues [1] did about users’ motivation of sharing information will be related to section 4.1 and 4.5 where privacy issues of users are taken into account and the realisation in Picnic is described. The major arguments of Ohlson and colleagues, the influences that their research caused, as well as the architecture and implementation are taken into account in section 5. Connected to the research of Olson and colleagues [1] is the paper by Good and Krekelberg [4]. The security guidelines by Whitten and Tygar help to form a useful approach for the practical use. The final discussion about sharing files over the net that is based on researches by Lan and colleagues [9] and Liang and colleagues [10] find their adoption in section 4.1 and 4.3 were the use of a peer to peer system with a privacy issue is taken into account.

The following section “what is metadata” and “why it is useful”, mostly based on the W3C and Tim Berners-Lee [6] [16], frameworks the sections 4.6 and 4.7 where the use of metadata is described in greater detail. These references are as well the basis for using metadata through the whole practical part of this thesis. The way how to interact with metadata is described in section 4.6. In the following section the architecture and implementation is supported by Sarvas and colleagues [5]. It is shown how their research on a metadata creation system, connected to John R. Smith and his structural construction of metadata [7], influences the Picnic approach. Adrienne Tannenbaum’s question “where does metadata come from” [5] is more an informing section that finds its connection in section 4.1 where speculations were made about the possibilities of interfering into pictures and their metadata.

The need of socialization with strangers and its explanation by Loren Terveen and David W. Mc Donald [2] and Stephan Baumann [3] is described in sections 4.2 and 4.6. Also, the socializing aspect of Picnic and the “how to” is described and announced in a more detailed way. The section about a more specific sort of metadata, location metadata [17], is described in section 4.7. The relation between location metadata and the possibilities by connecting it with different information sources is taken into account.
4. Conceptual Approach

To connect the theoretical foundation and the technical approach to each other, this section will give an introduction to the implementation and references as well as to the theoretical foundation. The reader will get a comprehensible introduction to the implementation by connecting theoretical approaches to conceptual ones.

The personalization of the users’ content and the question of how to socialize these users to each other without disclosing any personal information become two systems that run site by site but interact with each other.

4.1. A general view on Picnic

Picnic is based on an instant messenger framework. The tool looks like a common instant messenger or the Skype-messenger, but Picnic is not meant for calling someone (like Skype) or sending messages (like MSN). Its idea is to share pictures.

Like with an instant messenger, users have the opportunity to click/contact (on) a user who is not on the contact list. Then s/he can look at the pictures (or general data) the other user has saved on his/her computer. These pictures were well chosen by the other user and they are only shown to a selected group of users or a single person; not to all users of the contact list.

The difference to other solutions is that users can easily structure their pictures and they can easily allocate their pictures to the users on the contact list. The pictures are also kept privately and the pictures are kept on the users’ computer and not on any server somewhere in the web. This approach takes the privacy issues a user has into account by showing pictures as well as the question how picnic can solve this problem.

4.2. Scenario as a basis for design

There is always a difference between a theoretical explanation and a real-life example. Therefore I will introduce the given problem domain and my approach in the following scenario.
Character 1: Bob is a 49-year-old engineer from Canada. Because of his job he needs to travel to Europe and Asia frequently. His addiction is Ice Hockey. He tries to see as many games as possible and takes pictures quite often which he saves on his computer and shows to his friends via Picnic.

Character 2: Martin is a 28-year-old Ph.D-student from Switzerland. He studies Design at University of Bern and takes some pictures in his free time and uses these for his studies as well. He offers the pictures he takes to the whole internet community via Picnic. His interests are sports in general, but Ice Hockey in particular.

Martin and Bob are two users of Picnic because they want to share pictures with friends and family, or even as Martin does with the whole internet community. Both of them tag their images with metadata to make sure that they remember what happened during the event when they took the pictures. Even though they are both very communicative, they do not want to give any personal information to strangers. They do not describe their hobbies or addictions to someone they do not know because of security and privacy reasons. But they both agreed that their pictures’ information might be used to get them into contact with someone of same interests.

Therefore Picnic’s metadata matcher is allowed to read the metadata the pictures offer. While both users are online the metadata matcher reads the metadata and saves them to a database. Then it starts to match the pictures’ information to each other as well as to different external information sources.

By matching the metadata information from both users the program gets a match on some pictures. For example, all pictures were taken on September, 25th at 19:00h, and all of them include the tags “Bern”, “Ice Hockey” and “National Team”. The metadata matcher saves this information to the database as a relation between the users Bob and Martin.

Now the system starts to search for information that is related to the metadata of both users. In the example, a match occurs on the term “Bern”. It is recognized as a city in Switzerland with some additional information. The next match occurs on a combination of information. Related to the place Bern and the date/time, September 25th /19:00h, different events are recognized that took place that day. One match is the ice hockey match between Canada and Switzerland. Because “National Team” is as well one of the metadata tags here and because the game took place between 18:00h and 20:00h, the system concludes this event to be a match.

Due to this information the system sends a message to Bob and Martin. Of course this scenario is a best case and because the external information retrieval is not that easy this case is a more advanced one for the future research.
4.3. **A view on the communication of Picnic**

Picnic is based on the concept utilized by an instant messenger application. The communication of this type of messenger could be managed by a server- or a peer-to-peer application between two users. The same technology applies to file sharing tools. Their communication runs as well managed by a server- or a peer to peer application.

However, it is important to choose one solution that fits best for Picnic. Because of the amount of traffic I will exemplify my solution managed by the peer-to-peer approach. The advantage is that the traffic will only take place between the two clients; the server will not be influenced by the communication between the two users and can concentrate on the authorization of the users.

The sequence of actions is as follows: The user logs on and his password and UserName are sent to the server. The server compares this information to the database and in case of a match the server sends a list of the user’s buddies to the user. At the same time the application compares the date of the local list of pictures with the date of the other user’s lists of pictures. If there are differences the list will be exchanged and all preview pictures that have not been sent yet will be sent. Both users are now up to date again.

If a user wants to see the preview-pictures of another user s/he opens the user-window and the shown pictures are only loaded from the computers cache. If the user wants to download a certain picture a request is sent. If the other user allows access, the picture will be sent.

---

**Figure 4.3: Description of a possible system**

Figure 4.3 explains the basic communication on Picnic while sharing pictures. The socializing is separated because it will take place on its own.
The distributor is placed between the two users. When a user tries to connect to the “Picnic net” the user sends a request to the Distributor that checks the user’s information. If the user’s attempt to connect is accepted, the Distributor sends permission to the user.

After the user is then logged on to “Picnic net” s/he is connected to other users and can exchange information with them.

4.4. Privacy, respective Personalization of Files

Keeping files private is the idea of the implementations in the Picnic framework. Users have the possibility to relate a picture to a user to make it easy to share it with a chosen person. In the description above, the research results from Olson and colleagues [1] and Good and colleagues [4] describe very well the user’s needs of that position.

Picnic lays its focus on pictures. Users have the possibility to choose a folder or even one picture and relate it to anyone in the user’s list. It is important to say that this security issue is given. At this point it is no longer a sociological or technical, but even more a usability problem. On the screenshot in figure 4.5 one can see the approach of Picnic that makes it possible to add a new folder and personalize it. At the same time, users have access to folders and pictures that have been shared before.

The information about the pictures will be saved on a kind of local database of every client. This database generates a special ID for every picture, as well as every user has an ID. The system checks which picture and which user are related to each other and shows the picture to a user that stands in a positive
relation to the picture-owner.

To make sure that a situation such as an unknowing sharing of files on Kazaa, like Nathaniel S. Good and Aaron Krekelberg [4] described, can not occur, the interface views all shared folders. Beside every folder the users’ buddies, that have the possibilities to see these folders, are shown.

4.5. Envisioning the way of socializing people with Picnic

As described above, Picnic offers the possibility to share pictures. By personalizing the pictures to just one group of selected people, the possibility that a stranger might see them by accident or leaves a message, is no longer given. But people like to communicate with others and people like to socialize with each other. As Terveen (2005) says: “People are social creatures—fundamentally so. We look for other people for a multitude of purposes: dating and eventually marriage, pursuing shared interests, addressing community issues, solving technical problems, or maybe just having a good conversation.” [2]

The web is often used to get to know new people anonymously and, even if people often do not behave that way, they want to talk to new people that have the same interests, needs or even problems. As I have described in section 3.

Therefore the question is how it could be possible to socialize people with each other, without showing pictures to anyone else. An approach was found in the semantic web. Pictures contain metadata. Metadata is text-information that describes media files such as pictures or movies. It is saved on such files on a special layer. Metadata can include any information but it describes in general what the picture is about, at which place the picture was taken or which camera was used to take the picture.

Using this metadata people can be socialized by analyzing their pictures’ metadata and looking for social matches. This process can be done by a machine that contains an algorithm to search for social matches. In that way the privacy of the users is unharmed but the possibility to meet people with the same interests is still given.

4.6. Socializing People with Picnic

"Meet Marc, he is interested in skiing, climbing and running and lives in your area. Like you, he took some Pictures in Verbier last year in August".

Howdy, nice to meet you, Marc. So, you went skiing in Verbier?

"Meet Natalie, she is interested in skiing and running and lives in your area. Like you, she took some Pictures in Verbier last year in August".

Howdy Natalie, well, I prefer snowboarding, but – yes - I went to Verbier for powder-sports.
Picnic offers the possibility to exchange pictures privately with your own chosen group of people. But Picnic should as well help to socialize people. During the theoretical approach I have already introduced alternative ways how this could work.

Every user has a client application running on his/her computer. This client application is used to send pictures, retrieve and show them. Any interaction the user wants to do takes place on this application. The use-cases in section 8.4 show how the user has the possibility to interact with Picnic. Additionally to the tool, a Picnic client has a metadata application running in the background. This application communicates with an application on the Picnic server. The server is passing the information from the metadata application through a MySQL database.

This process looks like the following scenario: the user informs the Picnic client about some pictures s/he wants to share. The metadata application reads the metadata from these pictures and sends it to the server application which passes it to the MySQL database. During the passing process the application compares if the metadata from the user may have matches with the metadata from other users that are already saved on the database. If there are matches the application will save the match-information on the database and send a package with all matching information about the two users to both of them.

The raw data information is saved on the picture. It then is passed on by the metadata application on the client-side and the filtered information is then sent to a database. Here the information is filtered by an algorithm and rules. The information can not be proved yet due to the lack on background information.
4.7. Design Assumptions
This paragraph depicts assumptions about what could be possible if, for example, location metadata was provided. Different suggestions have been made and will be explained in the following. The given assumptions do only partly involve the prototype but they offer a deeper insight into the field of metadata and its possibilities.

4.7.1. Location Metadata
A system with a good working search algorithm can be a powerful information extractor. The power is based on its different attributes. One attribute is the date, another one the description a user added. A really powerful/ good attribute is the location the picture was taken. The location explains a lot about the picture’s semantic content. If a GPS location is given and this information is connected with other data sources, the picture’s semantic content could be described without having seen the picture itself [5] [17].

The possibilities of location data can be shown by an example: A user from South Africa and one from Sweden have pictures of the city of Dortmund, Germany, saved on their computers. Both pictures were taken on June 19th, 2006 around 17:00 o’clock. The computer now analyzes that both users have been to Dortmund on the same day. In combination with other information the system extracts that the Soccer World Cup game Trinidad/Tobago vs. Sweden took place that day. A high-quality system could also extract that both pictures were taken outside the stadium in a hall nearby. Therefore, a mapping system could analyze that the hall was called ‘Westfalenhalle’ and the system could report about the Westfalenhalle, for example that there were 15.000 fans in that hall that day.

So, the system informs both users that they have been in Dortmund, have watched the World Cup game and have even been in the hall.

4.7.2. Intelligent Matching System
As described in section 3, theoretical foundation, the closeness to the field of artificial intelligence provides the possibility to include different intelligent parts. The last section explained already quite well what could be realizable by having location metadata and what information could be extracted by combining all the information a picture provides. Different information sources that are combined with the data a picture provides can extract new information by searching for similarities. Figure 4.6 shows what an information retriever might look like.

Let us go through the drawing, figure 4.6, by using the former example of the Soccer World Cup in Germany. The picture with the raw data is our starting point. The information will then be passed through a similarity source (the term ‘similarity source’ will be explained in the following section). In a following step, the information passes different information sources. These sources have saved different information. One source, for example, saves grammatical signs while the next one saves events with information such as event, place, beginning and end while a third information source saves names of buildings.

The information is “Dortmund, June 15th 2006, an amazing moment for everyone” containing as well a GPS location information.
This information is now passed through the different sources. The first source may extract the string patterns “on” and “an amazing moment for everyone” because they were recognized as useless information. The string “Dortmund, June 15th 2006” incl. GPS is now passed through the next information source that extracts “Dortmund” and “June 2006” while it recognizes that the Soccer World Cup game Trinidad/Tobago vs. Sweden took place at that time at that place. The next source will recognize, on the position the GPS location data provides, the “Westfalenstadium” as a soccer arena with the capability of 80,000 people.

All these sources reconstruct information and can provide a result like: “The picture was taken during the Soccer World Cup in Dortmund, Germany, 2006. In the picture one can see the inside of the Westfalenstadium while it was filled with fans from Sweden and Trinidad/Tobago”. Having a fourth and fifth information source that contain information about soccer games and soccer arenas some more information could be added. For example: “The result of the game was 0:0 while Sweden was considered to be the better team. The picture was taken from the top of the south part of the arena while the arena was filled with 80,000 peoples, 55,000 fans from Sweden”. etc. It could also be possible to extract “daily news” that describe the content of the picture by relating the information to a newspaper achieve.

As you can see, the possibilities of extracting information are infinite. The question is what a user wants to know and how detailed the information should be. The technique of putting a picture in relation with external information to extract more information is quite powerful. Location Metadata in connection with a translator which connects the semantic location to it, may be the most powerful metadata attribute that is available.

![Diagram of a possible architecture that relates metadata information to external information sources to achieve the better result](image-url)
The similarity source has not yet been explained. This part saves similarities on data strings that have been recognized by the system. If, for example, the word “on” was extracted from 80% of all pictures, the possibility that it will occur often is quite high compared to the word “huge” that was only extracted out of 5% of all pictures. The similarity source works like a cache for words that occur more often than other words. We need to differ between general similarities and user specific ones: One user might have the word “Växjö” saved on 99% of all pictures while it takes a long time for the system to extract the information about Växjö because no other user has saved this word on his metadata. This is a user specific similarity and therefore needs to be saved separately.

4.7.3. Marked picture parts

To extend the possibility of matching data and socializing users to each other is described in the following paragraph. Nowadays, different websites on the internet [23] use a new technique to connect and recognize people through pictures: A user has the possibility to upload a picture on a website and mark an area on the picture with a layer that is put on the top of the picture. This layer may then act as a link to the profile, etc., of the user from the picture. That way a connection between the different persons on the picture is made. The web portals use this technique to visualize that a user is shown on different pictures and that this user may have a connection to the picture owner. This is done to show the visitor the people that are in the picture. Mostly these links can be found in pictures taken at parties or events when people get together to socialize. Therefore it is interesting to explore who a person a user has seen and talked to during this specific event.

In this approach the descriptive metadata is combined with administrative metadata and can build structural metadata. Anyhow, the location of the object/person in the picture needs to be described and saved as metadata information. Attached to this information an information needs to be saved that describes the marked object.

This approach offers different advantages: From the user’s point of view, s/he may explore the objects in the picture in a different way. A common (metadata) description may just explain what is in the picture, but not structure the information described. Let us take figure 4.8 as an example. Without any tags it would just explain that there are the soccer players Ibrahimovic, Cannavaro, Buffon and Larsson in the picture. With the tags the players are easily identified. A visitor may recognize the players quite well by him-/herself, but in a more difficult picture everyone would have difficulties to make out details. This is where the approach shows more advantages: Just take the visitors in the
background as an example. It would be hard to recognize a single person by description, but this person could be easily recognized if s/he was tagged.

Another advantage is that the person in the picture can be related to a profile of the given user. Using Picnic as the framework tool the pictures’ characters can be referenced to the profiles of such users. The tag on Ibrahimovic may be linked to his profile and such references could offer a good source of information. A picture may link to a user in one picture to another user in another picture. A connection through pictures and a connection to a social network can be created. A red thread of social networks, of objects or places can build a map of social connections.

From the social point of view the connection of different users is of most interest to a user. This component that does not only connect users, but it also offers the possibility to add (beside a semantic) a social connection component.

Especially the last approach (building a map out of different pictures) may be an interesting one taking the Web 2.0 into account. If you consider that a picture could not only save a 3D-information but as well location-based information. In that case a map could be easily drawn by triangulating a location to any object. Computable created 3D-maps that extend themselves by linking picture to picture like pieces of a puzzle. The more pictures are related to each other, the bigger the drawn map gets. Related to this a 3-dimensional search could be implemented. In that way it is easy to find a picture of an object from another angle. For example, if any user has taken a picture from the north side of a building s/he might want as well a picture from the south side.

A technical approach related to this topic could look like in figure 4.9. Every object of a picture is related to a profile which describes every object. This means that the relation is one-to-one and that every object is clearly defined, even if it is described differently. It is therefore important that the profile is unique. A name of an object might be different in different languages, but describes the same object. A profile makes sure that the object is unique. Only a direct link from the picture to the profile ensures that the relation is still one-to-one. In that way the relation of different pictures to each other is easy to reconstruct.

By description some websites already use this technique to provide social environments. Facebook.com and Studivz.net put different layers onto a picture. These layers may then be related to a profile of any user. By taking a look at a picture it is easy to get more information about the user. It is as well easy to look at a user’s profile and see the pictures the user is in.
4.7.4. **Extend pictures by adding new information**

The idea of the approach described in 4.7.3 is to relate metadata from different users to different sources to get more information about the picture and its story. The best way is to save new information onto the metadata of the picture. The information puts the picture into a context and it becomes more valuable.

A more philosophical question is if the picture is still the same one or if the picture is a new one. Is a picture the same picture without metadata than it is with metadata? On the one hand the picture is what it has been before with an altered metadata description, on the other hand we have a complete new dimension to the picture. This problem is discussed by Luo and colleagues (2006) [22].

4.8. **Transition to the Architecture and Implementation**

The following section “architecture and implementation” will now describe the technical part that has been described in the architectural approach. While the section on the architectural approach focuses on the parts a program such as Picnic should include, the technical approach describes how these parts can be realized. In addition to the class diagram, different other drawings and technical explanations, the technical approach gives an idea on which framework and programming language the program could be build. At the end some source code and screenshots are shown from the actual (prototype) program example wise that was programmed in relation to this thesis.
5. Architecture and Implementation

So far, I have introduced the program Picnic, the way it should be used and its theoretical background. In the following chapter I will explain the architecture and implementation of the program. Starting with the technical approach and describing the use cases and class-diagrams, this section will get deeper into the different parts the system is built on. The socializing mechanism will be explained in detail and a number of speculations about further possibilities will end the section.

5.1. Technical Approach

The previous sections have described a need for a program for the given problem domains. This part will describe what such an approach could look like. Picnic offers the possibilities to share pictures in a personalized way. A class diagram and additional drafts will explain the technical background of this program. In addition to that a prototype is programmed that should relate the metadata of different users’ pictures to each other and generate by that user profiles that can be used to generate matches. These matches then get as well related to external information sources.

Figure 5.1 explains the former introduced match-scheme (figure 4.6) from the technical point of view. I use this draft as an introduction to show the parts from a different point of view to give the reader an idea about this section.
5.2. Class Diagrams

The class diagram in figure 5.2 illustrates the whole Picnic project. As every aspect of the program is included into the class diagram, it is very detailed. The picture is logically divided into the server and the client part, while the server parts are more placed at the bottom and the client parts are more placed at the top. Between these two parts the connection listener is logically placed; this application runs at the server. It checks the connection and builds an entity with the HandlerClass and the PipeManager. These classes are responsible for the communication between server and client and set the important steps to send and receive data.

The client-software of any user contains all information about a user; shares, status, preferences, buddies and the buddies’ shares. The attribute LastSharesChangesOn in the class contact is important: It contains information when a status update between two users has taken place last time they talked and compares this information which is saved on ShareStatus to the locally saved list in MyShareStatus. If these lists differ from each other,
an update takes place and the missing pictures are downloaded from the receiver. The class MyShareStatus contains the information about these updates and has an overview on the share status of the other users. The class Shares, saved in relation to every user, describes the location and attributes of all pictures, thumbnails and preview pictures to the other users.

The classes CheckMatch, SaveMetadata, MatchReceiver and MatchListener are all highlighted in red and describe the socializing part of Picnic. This part has been programmed for this thesis and will not be a possible commercial version of the Picnic program. The class CheckMatch includes the operations GetNewPictures and CheckNew. These operations receive the new pictures and check if they are new ones available or if the metadata has already been checked. If there are new pictures available the class SaveMetadata checks the picture’s attributes. If the picture contains metadata in the format the class saves the picture’s metadata to a database. The operation MatchMetadata compares metadata to the metadata of other users’ pictures. The matches are saved on a database.

In a next step, the MatchReceiver takes the metadata of different users and compares it to each other by an algorithm that has been described. The operation CreateMessage creates a message to the users that have been matched to each other. The operations AddMatchToUser saves this message including the match related to the user. These users can now be socialized with each other. The operation AddMatchToDone saves the match to a database to ensure that a match will not take place twice.

The last step is the transport of the information to the user. For this the MatchListener is included to the message “surrounding” of the program. The ListenMatches operation listens to the file that contains the matching messages. If a new message, related to a user, comes up the listener receives that information and informs the user about it via a message.

The class diagram may change during the developing process; therefore I want to mention that this class diagram describes the process in general but might differ from the program that results at the end.

5.3. Technical components for programming Picnic
In the context of this thesis, it is not of high importance to describe the programming language the program is written in. But as it might be important for the reader in regard to further researches, I will shortly introduce the programming language and the EclipseClass SWT. Java is the programming language Picnic is built in, the interface was built with the support of SWT.

5.3.1. Java
To give a short definition of Java, I will just use the definition of Webopedia [27] that describes Java as followed:

“A high-level programming language developed by Sun Microsystems. Java was originally called OAK, and was designed for handheld devices and set-top boxes. Oak was unsuccessful so in 1995 Sun changed the name to Java and modified the language to take advantage of the burgeoning World Wide Web.
Java is an object-oriented language similar to C++, but simplified to eliminate language features that cause common programming errors. Java source code files (files with a .java extension) are compiled into a format called bytecode (files with a .class extension), which can then be executed by a Java interpreter. Compiled Java code can run on most computers because Java interpreters and runtime environments, known as Java Virtual Machines (VMs), exist for most operating systems, including UNIX, the Macintosh OS, and Windows. Bytecode can also be converted directly into machine language instructions by a just-in-time compiler (JIT).

Java is a general purpose programming language with a number of features that make the language well suited for use on the World Wide Web. Small Java applications are called Java applets and can be downloaded from a Web server and run on your computer by a Java-compatible Web browser, such as Netscape Navigator or Microsoft Internet Explorer.” [27].

For implementing Picnic, Java was used because it is really flexible and offers good possibilities to program object-oriented. Java is also platform-independent. While some users do not have java-runtime installed on their computer the program will probably be translated to machine code for the most common operating systems.

5.3.2. Graphical User Interface; the ClasEclipses SWT
Because Picnic is programmed in Java I will shortly consider SWT, a class of Eclipse to program interfaces in a better way.

SWT is the software component that delivers native widget functionality for the Eclipse platform in an operating system independent manner. It is analogous to AWT/Swing in Java with a difference - SWT uses a rich set of native widgets. Even in an ideal situation, industrial strength cross platform widget libraries are very difficult to write and maintain. This is due to the inherent complexity of widget systems and the many subtle differences between platforms. There are several basic approaches that have helped significantly to reduce the complexity of the problem and deliver high quality libraries [12].

5.4. Jabber - The open IM protocol
The server-client-interaction needs to be handled and some problems occur at this point. Because different users are connected differently to the internet, firewalls, etc, it is not easy to connect users in a proper way to each other. It would be possible for an easy application to use, for example, IP addresses. But some problems may occur due to the usage of firewalls, hubs, routers, etc.

A solution for these problems was provided by a protocol called Jabber. Jabber is best known as "the Linux of instant messaging" - an open, secure, ad-free alternative to consumer IM services like AIM, ICQ, MSN, and Yahoo. It provides open protocols that are free for use. These protocols use standards in addition to provide good alignment to other applications. Jabber is a set of streaming XML protocols and technologies that enable any two entities on the Internet to exchange messages, presence, and other structured information in close to real time. Jabber as well supports to run a server for the applications that should use the protocol. The protocols that are used for transactions are Extensible Messaging and Presence Protocol (XMPP) that formalize the core protocols developed
within the Jabber open-source community in 1999. They were produced by the IETF’s XMPP Working Group and published as RFCs in October, 2004 [30].

The Jabber protocol fits best to our solution because of different facts: As explained before different problems might occur that will not make it possible to connect several applications to the built network. Jabber is an improved protocol that handles such problems. Another issue is the security of the communication. Developing an own technique that is definitely protected requires a huge amount of time. Using Jabber one can make sure that the communicational point of view will be safe and lessen the probability that someone interferes in the communication.

But most of all: Why would one program an own protocol if there is another one available for free? Jabber provides a perfect solution and there is no need to program another protocol.

5.5. Sharing Pictures from the technical point of view

Figure 5.3 shows the communication between two users while they are in the process of sharing pictures. One requirement is that both users have confirmed the connection.

The order of events is the following: every user has a list where s/he saves an ID for every picture s/he offers to others to look at. This list includes as well the rights another user has to see pictures and the date the user updated his/her list the last time.

<table>
<thead>
<tr>
<th>List</th>
<th>List</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1 sends listupdate</td>
<td>User 2 sends request to receive</td>
</tr>
<tr>
<td>Tries to open a slot and sends key</td>
<td>Accepts key to send information</td>
</tr>
<tr>
<td>User 1 sends thumbnails + previews</td>
<td></td>
</tr>
</tbody>
</table>

5.3 The order of events by updating the users’ information lists

Every user has the own list and as well a list of every user that is in his/her contact list (buddy-list). The information that personalizes user1 is saved in his/her own list as well as in another user’s computer. The idea is now to keep both lists updated. If user1 changes anything in his/her list (by uploading new pictures, deleting old pictures, etc) s/he sends a list update to all the other users. The clients of those users receive this list and compare it with the list they have. If there is any dissimilarity between the lists, the information will be updated.
Every element gets updated when it is received by user2. In order to update the elements user2 sends a request to receive the messages. User1 then tries to open a slot to send information via TCP/IP and sends a key to user2 that needs to be accepted before any object is sent.

User2 then accepts the key and waits for user1 to deliver information. Next, user1 now sends the received pictures. Which picture user1 sends is up to the preferences of user2; see description in conceptual approach.

By using lists a couple of advantages are given: If user1 is offline while user2 updates his/her list, user1 can simply check the latest updates. The transfer of data can occur simultaneously. But as the list works as a reminder as well, a transfer is allowed to occur asynchronously, too. For example, one can write a meeting into a calendar and inform the other person about it. If the other person is available s/he can be informed directly, otherwise s/he needs to take a look into the calendar.

5.6. Showing pictures in a personalized way

A user might think: “Different users see different pictures. I am the picture owner and want to choose who can see my pictures”. That’s the idea Picnic is built on. But what does that look like in reality?

As one can see on the screenshot in figure 5.4 the user chooses a folder and relates some users out of his/her contact list to the chosen folder. S/he can relate a whole group to a folder or just a couple of people out of that group. If the user is related to the folder s/he can see the picture. If s/he is not, the user will not be able to see the picture and will not be informed about it (See screenshot 1, figure 5.4). Other users that are related to the picture will get a notice about it and get related to the picture. They receive the picture and can look at it (See screenshot 2 in figure 5.4). If the user deletes a picture, it disappears from

the other users’ window and all thumbnails and preview pictures other users have on their cache get deleted.

Every user has a small database saved on his/her computer. Every picture a user offers to his/her friends gets an ID and as well the place and name of the picture are saved in the
database. The database as well contains information which user is related to that picture. This procedure regulates who is allowed to see a picture and who is not.

The conversation between the sender, the person who shares a picture and the receiver, the person that sees the picture, works as followed: The sender sends a list to every user. The IDs of all pictures are saved on that list that the user is allowed to see. This list is then saved on the receiver’s hard disk. Now s/he starts downloading the thumbnails for these pictures into the cache. When the thumbnails are downloaded the user has the possibility to see them. The client ends downloading when the user stops the process or when all pictures of the list were downloaded.

If the user offers new pictures the other user’s list needs to be updated. The user sends a request to update the list to every client. Because every list gets a timestamp when it becomes updated only the timestamp will be compared with the timestamp of the other users’ lists.

If (CSW.getSharesViewOption() == 2 && state == 2) {
    Image img = thumbsContainer.paint(getClientArea().width);
    if (getSize().y < img.getBounds().height) {
        setSize(getSize().x, 25 + img.getBounds().height);
    }
    gc.drawImage(img, 25, 25);
}

5.5 Code that describes the container of a user

In table 5.5 you can see client code that represents the process of viewing pictures and how they get updated: The class BuddyFolder presents the window the picture is shown in while class ThumbsContainer represents those pictures that are presented. BuddyFolder updates the container when events occur, such as new pictures. The pictures get also framed by the class ThumbsContainer.

5.7. How to retrieve Metadata from a picture

How the metadata got into the picture is unknown, but how do we retrieve it from the picture? As previously described, metadata should be saved with the EXIF-standard to
make it useful. Actually this standard is used by most devices that save metadata to pictures (digital cameras, mobile phones, etc.) as well as most of the software programs such as the Windows explorer.

The “how to retrieve” depends on the case how one would like to use the metadata. Some programming languages provide classes that read metadata. There is as well a lot of picture software that shows the metadata and offers the possibility to change the metadata. I use a PHP file that reads, analyzes and saves the metadata. The advantage is that PHP provides the possibility to read metadata and as well the data retrieval from a database works smoothly. Also, the presentation of the different steps and the result is easy to show in a browser window.

As one can see in the appendix, section 8, metadata from just one picture contains quite a lot of information. It is possible to locate the owner, the place where the picture was taken, the content of the picture, the camera the picture was taken with, etc..

So, the metadata was retrieved by a PHP script. The main code that retrieves the data is looking as followed:

```php
//reads the files of a folder
if ($handle = opendir($folder_pfad)) {
    while ($x <= $dateien) { //anzahl der bilder in einem ordner
        $file = readdir($handle);
        if ($file != "." && $file != "..") {
            $x++;
        [...]
    }

    //looks if an exif-header is found
    $exif = exif_read_data($filename, 'IFD0');
    echo $exif===false ? "No header data found.<br/>
" : "Image contains headers<br/>
";

    [...]
    $exif = exif_read_data($filename, 0, true);
    foreach ($exif as $key => $section) {
        foreach ($section as $row_name => $val) {
            [...]
        }
    }
}
```

5.6 PHP-code for retrieving metadata

The code reads all metadata that is saved within the EXIF-standard ($exif = exif_read_data($filename, 'IFD0'));. If this is not the case it throws an exception because the header is not an EXIF-format. These code-patterns extract useful data to match data-patterns with each other.
A problem with retrieving data is the performance: As every picture contains a big amount of information it takes time to retrieve the data. The problem is also that every picture needs to be related with each picture, this process takes time as well. The process will take place on a server and the users will not recognize it, therefore the user will not be restricted in his/her work, but it still takes time to retrieve the data.

Related to this section see reference [13], the EXIF Specification by JEITA.

5.7.1. A short introduction to PHP

PHP, "Hypertext Preprocessor", is a widely-used Open Source general-purpose scripting language that is especially suitable for Web development and can be embedded into HTML. Its syntax draws upon C, Java, and Perl, and is easy to learn. The main goal of the language is to allow web developers to write dynamically generated webpages quickly [14].

In my prototype the advantages of PHP are used to match different attributes of users. As PHP is built to read from databases and compare different attributes it is optimal in view to the possibilities of socializing people.

For a client solution PHP might not be adequate because of the fact that it is a script language and it is not easy to call it out of a client. Nevertheless, it visualizes the possibilities and may then be realized for a client version.

5.8. Socializing Users from technical point of view

After the metadata has been received from different pictures and useful data is available at the database the matching can take place. The metadata from the pictures will be compared to each other.

5.8.1. Architecture of the prototype

The architecture that socializes users with each other has different tasks and responsibilities. First of all the program has to read a picture’s metadata and save it in xml-format. This process could be done by a Java application that is included in the Picnic application. The pictures are placed in a folder on the user’s computer that is defined by Picnic. The application writes the information into a database.

The prototype that is programmed in connection to this thesis saves the metadata information locally and is not determined by a Java-application but by a PHP-script that reads the information and saves it onto the database related to the user.
The data is still formatted in XML code that is then read by a PHP-script. It extracts the information from the XML-formatted code and writes it into a Database. The code is then read by another script that extracts the information out of the XML format and writes this useful information into the database. It is important to note that this information has been saved before in the EXIF 2.10 standard format for metadata allocated to .jpeg, .tif and .gif files. To read more about the EXIF standard see section 8.2.

Why it is important to take this into account? It is important because XML is a very flexible script-language. The data that is saved in xml can be called differently by every application/person that changes metadata. As well different camera-models may save information in different ways. This possibly creates problems when assigning metadata the right way.

Nevertheless, using the EXIF-standard, the information is saved in a standard format and can retrieve information the same way for every picture. As the EXIF-standard is used it becomes now easy to assign the information to nomenclatures. Thus, compare retrieved information with each other. By using PHP an algorithm can be defined that is used on our database; that contains all the information. What the algorithm might look like and what kind of comparison should be used is variable. A very easy solution could be “SELECT camera FROM user1, user2 where ‘user1. camera’ = ‘user2. camera’;”. In that case we would compare the camera models the pictures of the different users were taken with and then generate a notification that pops up if two users have taken some pictures with the same camera model.

A more advanced solution is to compare the place the users took the picture. Two possible solutions are available: The first one is the information the users attached to the
picture. For example the user attached as a place “Vienna”. The second possibility is to take a look at the GPS coordinates that are saved on a picture. Some cameras already give the possibility to save the GPS-coordinates. Now it is possible to match them by place_gps and place_name. But to match place_gps and place_name with each other it becomes a bit complicated. At this point, a gateway is needed. A possible approach is to use GPS coordinates to generate a map. This map could show where people were placed while taking pictures.

A bit more complicated is the matching of information that is not constant data like a date or a place, but irregular data such as a description. To compare a date with another date is simple; to compare different descriptions of a picture with each other is not that easy. The text of the description needs to be divided into different small strings. These strings may contain key-words and an algorithm then compares these key-words with each other.

An algorithm that compares all information of different pictures is therefore quite complicated and takes an enormous amount of time to achieve really good results.

Some problems are not possible to solve, even if a lot of problems might be solved by a good algorithm. Different users use dissimilar description about something. For example, a co-worker calls his colleague Miss Nilsson, while her husband calls her Irene, and her children say Mama. These context-related descriptions are personalized information and to explore the connections is impossible. Relations to objects and persons are nearly not describable in rational ways without the user’s description about the relation to the object. Related to this problem see reference [15].

Nevertheless, getting back to the architecture of my prototype related to this thesis the model 5.8 shows the way it will operate.

![Diagram](image.png)

5.8 Matching two users by comparing first the table names, then the table content by using an algorithm on the content, checking if a match occurs and at last saving the match to a database.
The following part will explain, based on some source-code-examples, how a matching process works. It is restricted to the parts that actually compare and match the information.

The table in figure 5.9 explains how a usual match out of any text-string may be generated. The array $ex owns the text patterns of the text-strings a picture has saved in the metadata. The different patterns are now related to the different text-patterns from another picture another user owns. If there are any similarities they were saved on an array that is called $match[].

The process consists of different steps: First of all the useless text-patterns are deleted by relating them to another information source that sorts out every part that has been recognized as useless. The text-patterns that are left are then related to the different places another information source presents. The code extracts the places, respective cities that are described in the metadata text.

The extracted places are then saved onto the array called $places[]. As an element may occur more than one time the redundant elements are deleted.

5.9 Extracting matches out of the metadata-strings; relating them to an information source called places

At the code pattern in figure 5.10 the events (that are saved to places that are extracted from the general description) are shown. I explained before how places are retrieved; now it should be researched if any events have occurred at the date and time before the picture was taken.
Here the database called “events” is consulted if any event has occurred during a timespan at the place that has already been extracted before. It is important that the information source that owns the places and the one that owns the events are attuned to each other.

At the end of this code pattern the information is saved to a table called “relation” where all matches between different users are saved.

```php
foreach ($place as $eventplace) //array with places {
    $eventdate = "SELECT * FROM events WHERE start <= '$picturedatum' AND end >= 'Spicturedatum' AND place = '$eventplace'";
    $eventdate_bes = mysql_query($eventdate);
    while ($event_aus = mysql_fetch_row($eventdate_bes)) {
        //insert into the relation database
        $relation = "INSERT INTO relation VALUES ('$eventbild','','$event_aus[1]','$event_aus[0]')";
        $relation_bes = mysql_query($relation);
    }
}
```

5.10 Extracting an event out of the information that has been extracted before

5.8.2. Preconditions for a match

The definition of a match between two users is not as easy as it might seem. Getting a match might be easy as the system just needs to find two similarities that fit together. But do both of these users really have something in common when both of them have taken pictures in Vienna, for example? Not often. Considering that millions of tourists take pictures of Vienna’s famous sights every year, it seems unlikely that most of them have a lot of similarities despite a trip to Vienna.

Depending on the information that can be extracted from pictures it is important to build up a small algorithm that explains when there is enough matching information and when not. For example; 30% of user1’s information matches 60% of user2’s information. For user1 the match is not that good while it is quite good for user2. User2 might have more ambitions to meet user1 while user1 might have more ambitions to meet someone with a higher percent of match. Nevertheless it is up to these two users to get together.

But how to get 30%, respectively 60%? The algorithm needs to take several domains into account: First of all the intersection between two users. The total amount of information is divided by the intersection of matches. The result gives the percent of the
match. Thereby it is not possible to choose how, by which elements, the users are matched together. Therefore the user can choose how differentiated information should be rated. For example s/he could put 70% percent effort on meeting peoples that have the same interests and only 30% effort on people that have been at the same place. By that the algorithm takes not only into account the intersections but as well the rates of the user. This is done by a point system.

I take the former numbers as an example: a match of 50% (in places) for user2 and we imagine a 20% match for interests. By taking 30% effort for places and 70% effort for interests into account we would not have a 35% percent match (20% interests, 50% places => intersection 35%), but a 41% match (20% interests, 50% places => intersection 70%-interests and 30%-places makes 41%). Because of the users’ preferences the match is rated over 40% and is considered to be a better match than it would have been rated without the information.

At this point it is important how valuable a match is. By deciding how to rate a match the algorithm needs to be taken into account. As described beforehand, the algorithm considers the matches in relation to all files. This means that a match of more than 70% is already a quite a good match. Because the user can rate interests the rate might be higher, therefore a good match is approximately 80%. Lower percentages can be taken into account the same way while a general connection is easily found (compare to example of pictures from Vienna). In this light, matches under 20% might not be relevant. The system by that should blind out every match that is lower than 40%. All matches over 40% can be rated as “interesting matches”, higher than 65% can be rated as “good matches” and all matches over 80% as “very good matches”.

5.9. Speculations about possibilities for the future of Picnic

During the last sections the reader got an insight in how powerful the combination of information can be. With just a small amount of additional information a tool like Picnic could be even more powerful. During this section I will speculate with different possibilities and ideas that are related to the former section.

As described before, the use of location metadata would introduce to a wide range of possibilities such as evaluating possible monuments that are in a specific place. To substantiate the possibility of recognizing elements like “buildings” it would be very useful to save the angle the picture was taken from. In that way the recognition of an object would be even better and the result much more advanced. Let us take the example of the soccer
game in the Westfalenstadium of Dortmund. We have the information of the photographer’s location which is inside the arena. If we knew the angle the picture was taken from, we would be able to evaluate the object that is in the picture. If the person is placed on the south part of the arena and the picture was taken in north-direction we can imagine that the field or the tribunes on the opposite are in the picture. So, we can foreclose that a person standing behind the photographer or persons next to him are in the picture.

The next step could be to take into account the inclination of the camera and as well the zoom-factor. While the inclination has not been taken into account yet, the zoom-factor of a camera is saved to a picture’s metadata. The inclination can give us an advanced idea about the object in the picture. In our example, we would know that the picture would show the field or the tribunes. With the inclination information we can encircle the object. Combined with the information about the zoom we can even imagine how close the object in the picture might be.

Another advantage could be the combination of the location the picture was taken and the GPS-coordinates that mobile phones of different users provide. In that way it would be possible to recognize people that are in a picture. Nowadays a mobile phone is unique to its owner and a mobile phone is nearly a unique recognition factor. So, a user’s GPS-position combined with a picture’s GPS-position could match a user in a picture and, in a further step, give information to the photographer himself. The pictures of different people could in that context include the names of the different users and could connect this information to other sources. The German website Studivz.net provides a similar service while the users need to be tagged by the photographer. The system I proposed would be able do this automatically.

This hypothesis would possibly arise the question of secrecy but I would like to play with possibilities without questioning them at this point.

As one can see there are possibilities to get to know what is in a picture without having any content-related-information. With a strong information performer results become sophisticated and good. But then the question is what to do with the information. The use of it for former processes is certainly a good way of information processing and not only for exploring the information. Gathering them is a hard process; therefore it would be good to save such information. And the best place to save new information about a picture is to save it on the picture itself. Interfering into the picture and adding information would be a good solution. As gathering information in that way is quite hard, it would be much easier to have people describe the pictures themselves. If a user tells a machine that a picture was taken in Vienna, it does not need to extract this information.

Some programs and solutions such as Flickr already offer the possibility to tag metadata to pictures. Users depict their pictures and this information is just added to the information-slot on the picture. These kinds of tools offer a better way of adding information to a picture’s metadata. Here again, I would like to refer to the thesis of David Martinez [21] who researches in this area.

Interesting at this point is a recursion step by passing information further. The information that has been extracted before could be copied directly to other pictures if some objects matched. The process of extracting information could be accelerated in that way and a relation of different pictures could become more valuable. This would as well mean that the information related to a picture might become much more detailed because of a series of reactions by handing information over from one picture to another. The information would be passed on to the pictures as soon as it is extracted somewhere.
5.10. How can the approaches be founded and proved?

After the approach for the given problem domains has been presented the question is: how it is possible to prove the results? Indeed, the approach might present a solution but it is important to prove the approach and look if it really meets the expectations. Especially from a socializing perspective, it is important to prove if the found matches are really appropriate and true.

5.10.1. Is file sharing really personalized?

File sharing in a personalized way should be provided by Picnic. The previous sections have introduced the approach and in this section I will found a solution and show what the personalization looks like. Figure 4.5 shows the way a user can choose the viewers of some pictures. S/he has the possibility to decide who may and who may not see pictures. Figure 5.4 shows the results as they affect the user while different users do not see that there is a difference between users. Here the personalization is already provided: While the file-owner can see the different view-offers to all buddies, the buddies do not have the possibility to see how many pictures have actually been shared. Due to the fact that the file-owner can choose whom to share with and the fact that the file-visitor can only see files that are related to him/her a personalization is given.

5.10.2. Is socializing via files possible/true?

Besides proving the possibility to share files personalized, it is important to show that socialization through the users’ content is not only a possibility but produces as well good and true results. Still, some problems might occur.

As Michael Hardey in “The Editorial Board of The Sociological Review 2002” [24] writes users often use the internet to build a different personality than they show in real life. He differentiates between a real- and a cyber-personality and describes the action of users as followed: „From the posting of a self description on a dating site to the exchange of email with others, users are concerned about the possible discrepancy between ‘cyberselves’ and ‘real selves’. In contrast to what is usually written about the nature of internet interactions, users feel obliged anchor their on-line identity in their off-line embodied self.” [24]

The problem that occurs is obvious: Does a user actually want to show information or a ‘true’ profile about him- or herself? If the profile that is built by the program visualizes the truth, does the user really want that to happen? Or does s/he prefer the cyber-personality that has nothing to do with the files and the profile that is built on them?

The answer to such question is actually up to the user. And by that the control about the profile and its truth needs to be appointed by the user him-/herself. Whereby another problem occurs: If the user can influence the profile it might not be correct anymore. Because s/he might change the true information to get another profile about her-/himself.

Therefore the approach is to return information to the user; and only the user is able to confirm the information. If the information is not correct or the way the user wants it to be presented s/he can add information to a profile. The system uses this information to match users with each other, but does not rate the information as high as the information is retrieved from the personal files. Thereby, if a match occurs, matches through file-
information are more valuable than matches through user-added information. How does the information become better by using such an approach? One needs to take different facts into account: The idea of the system is to build a true and honest profile of a user to socialize two users as valuable as possible. The users might want to alter their profile the way they want to. Probably they do not add pictures about their hobbies but they want have their hobbies added to their profile. The most important part is to get a true result while it is still important to cover the users’ needs. Consequently, if some needs correlate solutions need to be found.

Because information that is added by the user is not such valuable/true it is lower-rated. Information that was retrieved out of files is rated higher. By that a control mechanism is added to the tool by letting the user decide whether the profile is correct or not. In fact this way of altering a profile by hand is the traditional way of dating portals that exist on the internet. Users describe themselves by attributes. The prototype combines the given solution with the traditional one to offer the possibility to alter information.

5.11. Transition to the conclusion

The last section, architecture and implementation, described from the technical point of view how such tools should look like that solve the given problem domains. Beside diagrams and technical information it introduced some tools that are appropriate for an implementation. As well the algorithms were described and founded. The section ends with some speculations and a foundation of the work.

The following section, conclusion, will conclude the results of my thesis and give an overview on it as well as motivations for further researches in this area.
6. Conclusion

To take a short look back, this thesis dealt with the problem of sharing personalized content on the one side, and socializing users anonymously at the same time on the other side. In the introduction the problem domain was described. Afterwards the theoretical approach described researches that founded the problem and discussed a solution while the conceptual approach introduced the software that is to solve such problems. The technical approach then explained such software from the technical point of view.

This section will now conclude and will review this thesis. It will reflect upon the work and give some indications about what future research could look like.

6.1. Overview / related questions

The goals of the thesis were to build a tool that provides innovative ways for people to share Rich Multimedia Material (images, sound, etc) based on personal and collective interest on the one hand and to socialize people by their interests without offering any personal information on the other hand.

As a result, the tool Picnic was built which offers its users the possibility to share pictures personalized. In order to socialize Picnic users, the metadata of the shared Rich Multimedia Objects was used to create profiles. Such profiles were then used to match users to each other. The matching results were then extended by different external information sources that provide additional information. The confirmation of the information was done by the owner of the Rich Multimedia Objects himself; the person a profile was built for. Figure
6.1 describes the circle of actions: The pictures’ data is extended by relating the pictures’ data to other data and by that extracting new information. Returning this information to the picture extends the metadata and by that the value of a picture. The new information is not only an add-on, but modifies the picture more or less. The life-cycle of a picture and its information ends up with extracted information that might be used for different purposes, in the case of Picnic it was used for matching different users to each other. In the best case the new information might be returned to the picture and by that elevate the value of the picture.

Different questions related to this result arise that might offer substance for further researches: Is the original picture still the same one or not (due to its new information one may not talk about the original picture, but about a new one)? Will the socializing information that the tool provides about its users be true or will it provide information that skews the user? If not, is it possible to build an algorithm that makes sure information about the owner of Rich Multimedia Objects becomes correct?

6.2. Advantages of the picnic approach
The advantage of Picnic is the fact that it keeps a users’ privacy in the process of sharing files, but socializes users at the same time. Picnic is a tool that provides to share pictures with selected people and not with random users. But it is also a tool that offers the possibility to extend a social network through it. This is surely a new approach of sharing and socializing. Matching metadata is as well an advantage taking into account how such an approach could be used in different ways. By matching information a net of information can be produced that can be powerful.

Picnic has its strength in dividing and connecting: Users might be different in character but they get the possibility to build a network of related strangers.

However, especially the use of Picnic will show the advantages for the future. By using the system the (dis-)advantages will become obvious pretty soon.

6.3. Drawbacks of the picnic approach
Picnic is built based on the idea that users build their own personal communities. If they do not do that the whole system is more or less useless.

For matching users to each other through Picnic it is important that users tag their pictures with metadata. This is mandatory and therefore the solution shows a field for future research. Nowadays most of the existing digital pictures have not been tagged yet. If users do not tag their pictures, the process that has been described is not possible. Even Picnic might have the possibility to support tagging in future.

Another shortcoming is the matching algorithm that has not yet been implemented into the system but is an important part of it. The way how users are matched to each other is really important to extract true information as it is the goal of the application. The prototype showed how such a system could look like, but what if the system needs to compare 100, 1000 or even more pictures with much more information than just selected keywords? Is is certainly a problem that the user him/herself needs to confirm his/her information that is extracted. Even if the system works well, the reliability of the
information is still up to the user. Eventually, people need to want to communicate. If they
do not want to communicate with strangers, the whole system is useless.

6.4. Future research
As many research projects open new problem domains, this does, too. My research
motivates further research in many areas. I will shortly explain the most interesting
problem domains and connect them to some researches that could be done related to such
domains.

As I explained before, my research describes the advantages and disadvantages of
socializing people by their metadata and shows by example what a socializing algorithm
could look like. A research about such an algorithm would be interesting, even from a
technical or from the behavioural point of view. A technical research in that area could
describe the best way of combining and relating many pictures to each other while a
behavioural research would explore the way people add information to pictures, what kind
of information they add and how the information retrieval in that way could become more
sufficient. In any case, a research on this topic would be very interesting.

A more user-oriented research would be a combination of this research with the research
David Martinez [21] undertook. An approach could look like the following: the information
that is retrieved out of a picture could be adapted to a map. This map could then show how
the users moved and what kind of similarities are given due to the way they moved, the
pictures they have taken and the information they have saved onto the metadata. This
would not only extract the research, but would add valuable information like route and
order of events.

A more general research would be to explore how information, and in specific words,
could be combined to each other. Problems as described in [15] about different notations of
one and the same object are given in the whole area of metadata and therefore a research in
this topic would be important. A researched solution on such a problem could as well solve
different related problems that are described by combining and relating information to each
other. As well the Picnic tool addresses a couple of research domains: From the HCI point
of view the question could be how such tool could be designed more sufficiently for the
user to understand. From the technical point of view the way how the protocol works and
how a better protocol could be implemented related to the domain of pictures would offer
an interesting starting point for further research. Also the question of how it could be
extracted to the domain of video and sound would be of interest.

An interaction between Picnic and the generation of metadata could be researched and
implemented as well by giving users the possibility to add information to the picture by
writing a diary, describing the picture with a headline, etc. This research should be
combined with a practical approach due to the fact that the task is more a practical one. As
you see, there is a wide range of possibilities for researches. Most of them do not only
touch Picnic but spread wider problems and are therefore much more valuable than it looks
on the first impression.
6.5. Motivation and hints related to this project

At this part of the thesis I would like to introduce interesting ideas (some of them are not very serious but they are all interesting in connection to the topic).

Google Epic. This is a project that was built by the museum of media history, Tampa Bay Federal District, Florida. Google Epic describes how the internet could look like in the year 2015 and how the use of the internet will change (There are two different versions available; epic 2014 and 2015. Version 2015 is the renewed version of 2014). It is really interesting how Epic describes the internet as an information gatherer and how Google, Amazon and MSN might use the information every element of the internet offers to perform results. You can find Google Epic on:  
http://media.aperto.de/google_epic2015_de.html

I would as well like to introduce a system called Parakey that is in development by the founders of the company Mozilla and is to be published in summer 2007. This system should show every computer as a server and the content of every user’s computer is shown to everybody else. The difference to Picnic is that every computer works as a file server at the same time. The idea is that this system should as well include a search for metadata. A matching-system like it is introduced in this thesis could, theoretically, be added as well to this system. In that way the information gathering could be widely extracted to every user’s files. You can find Parakey on: http://www.parakey.com/

6.6. Reflections and conclusions

First of all it is important to say that it was really interesting to get an insight into the different topics my thesis has been connected with. Especially the questions how metadata may be researched and how more information can be extracted, as well as the work with Picnic tool in general were exciting areas to work in. I came to realize that topics like information retrieval for metadata are huge. As well the description how Picnic, or IM’s in general, work is a huge topic. The technical point of view as well as the usability of such tools is certainly interesting for further research.

Reflecting upon the work I can say that it was not easy to find a proper solution for the problem domains. I am really happy with the result but it was a long and hard way to get to the result and took many hours/weeks of research. During the research I explored the countless possibilities these topics include. Especially metadata offer an enormous amount of possibilities and they surely will play a key component in the future.

Metadata will be a key component of the web 2.0 and by that the future of the internet, and I think that the ways of retrieving metadata nowadays are not satisfying yet. Users, as far as I can see, will not tag all their pictures with metadata by themselves. As they produce tons of pictures a year, they will not have the time and will not take the effort to tag them. Therefore I think the approach of extracting metadata information from different data sources, by combining information to each other, is a good and acceptable way for retrieving data.

Finally I think that it would be an interesting approach to let pictures pass their metadata information through each other. As people produce a lot of pictures that were taken in the same setting or stand in other connections it would be nice if an interaction between them was possible. This is probably a topic for another research.
6.7. Summary

In this last section of my thesis I would like to elaborate on the connection between the initial research questions and the results I have achieved. The first research question presented in chapter 1 aimed at investigating which existing methods and techniques can be implemented to prove innovative ways of sharing rich multimedia content based on personal and collective interest. The ideas and concepts presented in chapter 3, Theoretical Foundations, were an attempt to provide some answers to this first question. I provided some arguments about users’ motivation for sharing content in a personalized way, as well as I presented different techniques for sharing content, including a discussion about the advantages and drawbacks of the different techniques. Different file sharing applications and different ways of sharing content were discussed as well as the question of how and if personalization is possible while using these different tools. In chapter 2, Envisioning Picnic, I introduced an innovative approach for sharing images in a personalized way, thus paving the road for those ideas related to the Picnic application described later on in chapter 4, Conceptual Approach. Chapter 5, Architecture and Implementation, give a detailed explanation of those technical aspects related to the implementation of Picnic.

The main ideas behind the design of Picnic were set in relation to the already existing programs and systems, as those described in early chapters of this thesis. In this way, I could elaborate on the advantages of Picnic in relation to those systems. In chapter 4, Conceptual Approach, I described the advantages and drawbacks of the design of Picnic. Those aspects related to the design approach and implementation of Picnic provide an answer to the second research question raised in this thesis: how can a system be designed, in order to support the features of the first research question, and which one is the most adequate way to do that?

In chapter 5, I demonstrated how the different Picnic components and features may help people to share personalized information and at the same time to socializing with others. By taking these ideas into account, I make a relation to my third research question: how a system can be designed to provide sharing and socializing at the same time. Socializing people through users’ content was a major concern while designing the different Picnic components and functionalities. Section 4.5 in Chapter 4 describes in detail how the process of socializing people through content looks like, while sections 5.6, 5.7 and 5.8 in chapter 5 describe the technical aspects of how the way of socializing has been implemented.

The main focus of this thesis can be divided into two different fields of research that complement each other; first, the field of sharing content in a personalized way and secondly, the way of socializing users through the content of files. Having these ideas in mind, chapter 4 and 5 were presented in a way that allowed these two different fields to be discussed separately in the first place, and later on in relation to each other. Hopefully, this logical structure paved the way for a cross reference between the different sections and themes discussed.

Chapter 6, Conclusion, ended this thesis with an overall view on the worked I conducted, thus trying to summarize the results of these efforts. Answers to the research questions were thoroughly explored following a scientific approach. I expect that after going through this work, the reader gets a proper understanding of the ideas and research presented in this thesis. Hopefully, this content will spark a deeper interest in the different topics I discussed. Thank you for reading.
7. References


8. Appendix

8.1. Glossary of Terms

GUI: Graphical User Interface

IM: Instant Messenger; a chat application

Rich Multimedia Object: A multimedia file that contains information used for information retrieval; such as Metadata

Tagging (Metadata): Process of adding Metadata information to a file

P2P: Peer-to-Peer; a direct internet connection between two computers

GPS: Global positioning System; a system from the American government to localize objects all over the world via satellite

PHP: PHP: Hypertext Preprocessor (recursive acronym); a reflective programming language, originally designed for producing dynamic Web pages

Jabber: an open system, built to provide instant messaging service

SWT: Standard Widget Toolkit; a graphical widget toolkit for the Java platform. Mainly used to program GUI

XMPP: Extensible Messaging and Presence Protocol; an open, XML-based protocol for near-real-time, extensible instant messaging and presence information. It is the core protocol of the Jabber Instant Messaging and Presence technology

EXIF: Exchangeable image file format; a specification for the image file format that contains, among others, Metadata information

JEITA: Japan Electronics and Information Technology Industries Association; an electronics and IT industry trade organization

LAN: Local Area Network; a computer network covering a small geographic area, like a home, office, or group of buildings

8.2. Actors

Actors are users or systems outside the system to be specified that interact with it. Picnic’s conceptual design takes the needs and functional requirements of actors into account.
### Actor Description

<table>
<thead>
<tr>
<th>Actor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>A client is the program basis on a computer. Every client needs to have one user. The user is the identification to the server and other users.</td>
</tr>
<tr>
<td>User</td>
<td>Every client includes one user. The user has an ID that corresponds to an email address. Such address is an identity.</td>
</tr>
<tr>
<td>Sharing user</td>
<td>A sharing user offers material to different users and encapsulates Metadata that influence the comparison.</td>
</tr>
<tr>
<td>Buddy</td>
<td>A buddy is a user that stands in relation to another user. Every user can have different buddies, which are users that have again different buddies.</td>
</tr>
<tr>
<td>Group</td>
<td>Groups are different buddies related to one unique group. Every user may have different groups which do not need to be unique.</td>
</tr>
<tr>
<td>Searcher</td>
<td>A searcher calls different process at server and database.</td>
</tr>
<tr>
<td>Metadata Reader</td>
<td>A metadata analyzer interferes into the pictures metadata and copies such metadata to a database. This database is placed on a server that saves the metadata related to the user and the different pictures.</td>
</tr>
<tr>
<td>Metadata Analyzer</td>
<td>The analyzer uses the metadata, which is saved by the metadata reader onto the database, of different users to extract information and relate such information to each other. The information that is extracted by the relation is then saved onto the database.</td>
</tr>
<tr>
<td>Match Viewer</td>
<td>The matches that are won by the analyzer are explained to every user by the match viewer. The viewer is responsible for a proper representation of the information to the user.</td>
</tr>
</tbody>
</table>

### 8.3. An example of Metadata standardized by Exif

So, I talked a lot about the EXIF-standard. But how is it looking like? The following table contains the EXIF-content from a test picture. We have the different attributes in the first row. Depending on the information origin the second row explains the location of the origin. The third row explains the attributes of the first row and contains the information that is interesting for us.

<table>
<thead>
<tr>
<th>FileName</th>
<th>FileDateTime</th>
<th>FileSize</th>
<th>FileType</th>
<th>MimeType</th>
<th>SectionsFound</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILE</td>
<td>1.JPG</td>
<td>1146255490</td>
<td>image/jpeg</td>
<td>ANY_TAG, IFD0,…</td>
<td></td>
</tr>
<tr>
<td>FILE</td>
<td>1146255490</td>
<td>1218351</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FILE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FILE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FILE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FILE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 60 -
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>html</td>
<td>computed</td>
</tr>
<tr>
<td>Height</td>
<td>computed</td>
</tr>
<tr>
<td>Width</td>
<td>computed</td>
</tr>
<tr>
<td>IsColor</td>
<td>computed</td>
</tr>
<tr>
<td>ByteOrderMotorola</td>
<td>computed</td>
</tr>
<tr>
<td>CCDWidth</td>
<td>computed</td>
</tr>
<tr>
<td>ApertureFNumber</td>
<td>computed</td>
</tr>
<tr>
<td>UserComment</td>
<td>computed</td>
</tr>
<tr>
<td>UserCommentEncoding</td>
<td>computed</td>
</tr>
<tr>
<td>Thumbnail.FileName</td>
<td>computed</td>
</tr>
<tr>
<td>Thumbnail.MimeType</td>
<td>computed</td>
</tr>
<tr>
<td>Make</td>
<td>IFD0</td>
</tr>
<tr>
<td>Model</td>
<td>IFD0</td>
</tr>
<tr>
<td>Orientation</td>
<td>IFD0</td>
</tr>
<tr>
<td>XResolution</td>
<td>IFD0</td>
</tr>
<tr>
<td>YResolution</td>
<td>IFD0</td>
</tr>
<tr>
<td>ResolutionUnit</td>
<td>IFD0</td>
</tr>
<tr>
<td>DateTime</td>
<td>IFD0</td>
</tr>
<tr>
<td>YCbCrPositioning</td>
<td>IFD0</td>
</tr>
<tr>
<td>UndefinedTag:0x1001</td>
<td>IFD0</td>
</tr>
<tr>
<td>UndefinedTag:0x1002</td>
<td>IFD0</td>
</tr>
<tr>
<td>Exif_IFD_Pointer</td>
<td>IFD0</td>
</tr>
<tr>
<td>Title</td>
<td>IFD0</td>
</tr>
<tr>
<td>Comments</td>
<td>IFD0</td>
</tr>
<tr>
<td>Author</td>
<td>IFD0</td>
</tr>
<tr>
<td>Keywords</td>
<td>IFD0</td>
</tr>
<tr>
<td>Subject</td>
<td>IFD0</td>
</tr>
<tr>
<td>CustomRendered</td>
<td>IFD0</td>
</tr>
<tr>
<td>ExposureMode</td>
<td>IFD0</td>
</tr>
<tr>
<td>WhiteBalance</td>
<td>IFD0</td>
</tr>
<tr>
<td>DigitalZoomRatio</td>
<td>IFD0</td>
</tr>
<tr>
<td>SceneCaptureType</td>
<td>IFD0</td>
</tr>
<tr>
<td>Compression</td>
<td>THUMBNAIL</td>
</tr>
<tr>
<td>XResolution</td>
<td>THUMBNAIL</td>
</tr>
<tr>
<td>YResolution</td>
<td>THUMBNAIL</td>
</tr>
<tr>
<td>ResolutionUnit</td>
<td>THUMBNAIL</td>
</tr>
<tr>
<td>JPEGInterchangeFormat</td>
<td>THUMBNAIL</td>
</tr>
<tr>
<td>JPEGInterchangeFormatLength</td>
<td>THUMBNAIL</td>
</tr>
<tr>
<td>ExposureTime</td>
<td>EXIF</td>
</tr>
<tr>
<td>FNumber</td>
<td>EXIF</td>
</tr>
<tr>
<td>ExifVersion</td>
<td>EXIF</td>
</tr>
<tr>
<td>DateTimeOriginal</td>
<td>EXIF</td>
</tr>
<tr>
<td>DateTimeDigitized</td>
<td>EXIF</td>
</tr>
<tr>
<td>CompressedBitsPerPixel</td>
<td>EXIF</td>
</tr>
<tr>
<td>ShutterSpeedValue</td>
<td>EXIF</td>
</tr>
<tr>
<td>Tag Name</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>ApertureValue</td>
<td>EXIF</td>
</tr>
<tr>
<td>ExposureBiasValue</td>
<td>EXIF</td>
</tr>
<tr>
<td>MaxApertureValue</td>
<td>EXIF</td>
</tr>
<tr>
<td>MeteringMode</td>
<td>EXIF</td>
</tr>
<tr>
<td>Flash</td>
<td>EXIF</td>
</tr>
<tr>
<td>FocalLength</td>
<td>EXIF</td>
</tr>
<tr>
<td>MakerNote</td>
<td>EXIF</td>
</tr>
<tr>
<td>UserComment</td>
<td>EXIF</td>
</tr>
<tr>
<td>FlashPixVersion</td>
<td>EXIF</td>
</tr>
<tr>
<td>ColorSpace</td>
<td>EXIF</td>
</tr>
<tr>
<td>ExifImageWidth</td>
<td>EXIF</td>
</tr>
<tr>
<td>ExifImageLength</td>
<td>EXIF</td>
</tr>
<tr>
<td>FocalPlaneXResolution</td>
<td>EXIF</td>
</tr>
<tr>
<td>FocalPlaneYResolution</td>
<td>EXIF</td>
</tr>
<tr>
<td>FocalPlaneResolutionUnit</td>
<td>EXIF</td>
</tr>
<tr>
<td>SensingMethod</td>
<td>EXIF</td>
</tr>
<tr>
<td>FileSource</td>
<td>WINXP</td>
</tr>
<tr>
<td>Title</td>
<td>WINXP</td>
</tr>
<tr>
<td>Comments</td>
<td>WINXP</td>
</tr>
<tr>
<td>Author</td>
<td>WINXP</td>
</tr>
<tr>
<td>Keywords</td>
<td>WINXP</td>
</tr>
<tr>
<td>Subject</td>
<td>WINXP</td>
</tr>
<tr>
<td>ModeArray</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x0002</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x0003</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>ImageInfo</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x0000</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>ImageType</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>FirmwareVersion</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>ImageNumber</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>OwnerName</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x000D</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x0010</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x0012</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x0013</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x0018</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x0019</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x001C</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x001D</td>
<td>MAKERNOTE</td>
</tr>
<tr>
<td>UndefinedTag:0x001E</td>
<td>MAKERNOTE</td>
</tr>
</tbody>
</table>

- 62 -
8.4. Use Cases

Because we have a Client-Server application the use-cases are divided into two different cases. The server has more an administrative part; most of the action will take place on the client site. The communication between these parties is described in the model above (figure 25). The use cases are described in a written way below. Due to the fact that the program was provided in Austria the use cases are originally written in German. To give you an idea what they are about and how they are written I translated them to English.

Client: Programm welches bei einem User auf dessen Computer läuft.
User: Person deren Aktion beschrieben wird
Buddy: Kontakte welche der User in seinem Client gespeichert hat

1. Authentication

Use Case UC1.1: Login to the Program

<table>
<thead>
<tr>
<th>Actors</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>The program is running</td>
</tr>
</tbody>
</table>
Postconditions: The user has access to his/her files

<table>
<thead>
<tr>
<th>Process description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1. The user gets to the login-page.</td>
</tr>
<tr>
<td></td>
<td>2. He needs to subscribe email address and password</td>
</tr>
<tr>
<td></td>
<td>3. The server authenticates the user and loads his/her data.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Exceptions, error situations:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If the user is not registered the System throws an exception and asks the user to register him-/herself</td>
</tr>
</tbody>
</table>

Technical realisation:

**Use Case UC1.2: Sign up a new user**

See before Use-Case exception

Siehe vorherigen Use-Case exception.

<table>
<thead>
<tr>
<th>Actors</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td></td>
</tr>
<tr>
<td>Postconditions</td>
<td>The user has an account s/he needs to login to the system.</td>
</tr>
<tr>
<td>Process description</td>
<td>User clicks on the link „register new account“</td>
</tr>
<tr>
<td>Exceptions, error situations:</td>
<td></td>
</tr>
<tr>
<td>Technical realisation:</td>
<td></td>
</tr>
</tbody>
</table>

**Use Case UC1.3: Show the data**

<table>
<thead>
<tr>
<th>Actors</th>
<th>Program, Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>The User-information during the Login need to be correct</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The program virtualizes all information that shall be viewed. Therefore: the user is logged on.</td>
</tr>
</tbody>
</table>
Process description: After the user has subscribed h/er login information s/he. The client sends this information to the server. If the server responds in a positive way it sends the user-information to the client. As well it sends the buddies and groups. The client responds to the server with the actual list of shared pictures. The server compares the list with the list that is saved on the server. If some information is new the server sends a reminder to every client for updating pictures. See more in the technical documentation.

Exceptions, error situations:
Technical realisation:

<table>
<thead>
<tr>
<th>Use Case UC1.4: Inform clients about on- and –offline-status</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actors</strong></td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
</tr>
<tr>
<td><strong>Postconditions</strong></td>
</tr>
</tbody>
</table>
| **Process description** | 1. Server and client communicate with each other permanently. They exchange information as soon as some user wants to offer new pictures, changes his status etc.  
2. If a user gets online the status of the user gets changed on the server. This information is then passed to the user’s buddies that are online at that time.  
3. Now the server sends all status information about the buddies to the user. The status is then shown in the user’s window. |
| **Exceptions, error situations:** | If a user went online the same chain of command occurs in the opposite way. |
| **Technical realisation:** | |

### 2. Personal Data

**Use Case UC2.1: Change personal information**

<table>
<thead>
<tr>
<th><strong>Actors</strong></th>
<th>User, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Preconditions</strong></td>
<td>User is already redigstered at the system and has a profile.</td>
</tr>
<tr>
<td><strong>Postconditions</strong></td>
<td>The profil eis actual.</td>
</tr>
</tbody>
</table>
Process description: After a click at Datei a window with the actual information about the user opens.

Exceptions, error situations: It would be possible to link directly to the website.

Technical realisation:

Use Case UC2.2: Offer folder

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client, Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td></td>
</tr>
<tr>
<td>Postconditions</td>
<td>Folders and Pictures are shown as wanted</td>
</tr>
</tbody>
</table>
| Process description | 1. Through click on Button „Ordner verwalten“ or click through User -> Order verwalten a window opens.  
2. If no folders have been offered yet this window is empty.  
3. If some folders are already offered are these folders and the usergroups/users shown in relation to the folder.  
4. To add a new Ordner one clicks on the button „Ordner anlegen“. Now a new window opens in which one can choose local folders. Afterwards a new window opens where a user can choose whom to share with.  
5. To delete a folder one marks the folder and licks on the button called „Ordner löschen“. |
| Exceptions, error situations: | Description from groups need to be added from use cases 3 |
| Technical realisation: | |

Use Case UC2.3: MyPictures

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client, Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>The user offered already one or more pictures.</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The pictures are shown as the user wants them to.</td>
</tr>
</tbody>
</table>
### Process description

1. Beside the buddys the user is shown himself on the buddy-list. In that way s/he can see easily which pictures are offered to the community.

2. Beside his pictures two buttons are shown: add Folder and delete Folder. The description in different functions are described in other use cases.

### Exceptions, error situations:

### Technical realisation:

### Use Case UC2.3: Pictures in folder

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>A folder is already there and recognized by the client.</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The added pictures are shown in the client.</td>
</tr>
</tbody>
</table>

| Process description | 1. The user shall have the possibility to copy easily Pictures into a Folder. As well s/he shall delete pictures easily. This is not possible to disallow.  
|                     | 2. The client adds automatically these pictures and shows them in the preview window. |

| Exceptions, error situations: | Therefore: permanent check from pictures in Client. |
| Technical realisation: | |

### 3. Buddy and Groups

#### Use Case UC3.1: Add a buddy to the client

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client, Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td></td>
</tr>
<tr>
<td>Postconditions</td>
<td>A new buddy is shown in the client</td>
</tr>
</tbody>
</table>
**Process description**
The user clicks on the Button add-User. A new window opens where one can search for a buddy through different criteria. If the user finds the buddy s/he can add it and it is shown in the client window.

**Exceptions, error situations:**
Is no buddy found it can't be added.
Adding a buddy needs to be saved on the server.

**Technical realisation:**

<table>
<thead>
<tr>
<th>Use Case UC3.2: Offers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actors</strong></td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
</tr>
<tr>
<td><strong>Postconditions</strong></td>
</tr>
<tr>
<td><strong>Process description</strong></td>
</tr>
<tr>
<td><strong>Exceptions, error situations:</strong></td>
</tr>
<tr>
<td><strong>Technical realisation:</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use Case UC3.3: Create a new group</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actors</strong></td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
</tr>
<tr>
<td><strong>Postconditions</strong></td>
</tr>
<tr>
<td><strong>Exceptions, error situations:</strong></td>
</tr>
<tr>
<td><strong>Technical realisation:</strong></td>
</tr>
</tbody>
</table>
### Use Case UC3.4: Group

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>A group exists</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The group is changed as wanted</td>
</tr>
</tbody>
</table>
| Process description | 1. Using drag and drop a buddy can be put into another group. Afterwards the buddy is shown in that group.  
2. To alter a name of a group on shall right-click on the group and the click on alter name.  
3. To delete a group on can right-Click on a group and then on „Gruppe löschen“. |
| Exceptions, error situations: | All actions need to be send to the server. |
| Technical realisation: | |

### 4. User – Buddy interaction

### Use Case UC4.1: Send short-message

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Buddy, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>Buddy was added before</td>
</tr>
<tr>
<td>Postconditions</td>
<td>Buddy gets a short-message</td>
</tr>
</tbody>
</table>
| Process description | 1. With right click on User on can choose the action send message.  
2. A new Message Window is opened in which a message can be written and send.  
3. The buddy receives a message in a similar window and can answer directly. |
| Exceptions, error situations: | |
| Technical realisation: | |
### Use Case UC4.2: Watch Thumbnails

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Buddy, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>Buddy has offered a folder with pictures</td>
</tr>
<tr>
<td>Postconditions</td>
<td>User can watch Thumbnails from buddy</td>
</tr>
</tbody>
</table>
| Process description | User click on the buddy.  
A window opens where the Thumbnails are shown. |
| Exceptions, error situations: | See screenshots |
| Technical realisation: | |

### Use Case UC4.3: Watch Bilder

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Buddy, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>See use-case „Watch Thumbnails“</td>
</tr>
<tr>
<td>Postconditions</td>
<td>User can watch pictures from buddy</td>
</tr>
</tbody>
</table>
| Process description | User clicks on the buddy.  
If a user clicks on a thumbnail a new window opens where the picture is shown widely (175/150 px). |
| Exceptions, error situations: | See screenshots. |
| Technical realisation: | |

### Use Case UC4.4: Download of different pictures

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Buddy, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>See use-case „Watch pictures“</td>
</tr>
<tr>
<td>Postconditions</td>
<td>User saved a picture locally</td>
</tr>
</tbody>
</table>
Process description
Beside the picture there is a button called „Download Bild“. A click on the button opens a window that shows the status of the download. The location of the picture can be chosen before in options.

Exceptions, error situations:
If no folders is offered where the picture shall be saved it will automatically saved in Eigene Dateien – Eigene Bilder.
Is there already a picture that has the same name an Exception is shown.

Technical realisation:

Use Case UC4.5: Download Folder

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Buddy, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>Buddy has choosen a folder where pictures shall be saved in.</td>
</tr>
<tr>
<td>Postconditions</td>
<td>Ordner with all pictures is locally available.</td>
</tr>
</tbody>
</table>
| Process description | There are different possibilities to download a folder.  
2. Click in Button „Download Ordner“ |

Exceptions, error situations:

Technical realisation:

5. Alter Buddy

Use Case UC5.1: Show Buddy info

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client, Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>Buddy is available in list</td>
</tr>
<tr>
<td>Postconditions</td>
<td>User can see information about a buddy</td>
</tr>
</tbody>
</table>
| Process description | 1. With a right-click in the user one can choose the option „userinfo“.  
2. Now an Info-Window opens where information is shown, which is saved on the server. As well a picture is shown which is locally saved. |
### Use Case UC5.2: Change Buddy name

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client, Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>Buddy is added on list</td>
</tr>
<tr>
<td>Postconditions</td>
<td>A new name is shown</td>
</tr>
</tbody>
</table>

**Process description**

1. Every user information is shown with name and emailaddress BSP: Daniel Putz <dp@d-ino.com>
2. With a right-click on the user one can choose „Name ändern“.
3. A window opens where the name is shown. The name can be changed directly.

### Use Case UC5.3: Delete a buddy

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client, Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>Buddy is added to contact list</td>
</tr>
<tr>
<td>Postconditions</td>
<td>Buddy is now longer added to contact list</td>
</tr>
</tbody>
</table>

**Process description**

Right-click on buddy -&gt; „User löschen“. Click on acknowledgement. Buddy is no longer available in contact list and is as well deleted from the server.

**Exceptions, error situations:**

Technical realisation:
6. User settings

**Use Case UC6.1: Change status**

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td></td>
</tr>
<tr>
<td>Postconditions</td>
<td>The user has changed ist status</td>
</tr>
</tbody>
</table>
| Process description | The user has the possibility to change his/her status:  
  1. Available, Download of pictures possible  
  2. Abwesend, Download of pictures possible  
  3. Lange Abwesend, Download of pictures possible  
  4. Nicht verfügbar, Download of pictures not possible  
The change is possible towards clicking on the status-button in main window down-left |
| Exceptions, error situations: |               |
| Technical realisation: |               |

**Use Case UC6.2: Automatical Picture-Download**

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client, Buddy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>A buddy is available in client</td>
</tr>
<tr>
<td>Postconditions</td>
<td>Depending on the settings the pictures were automatically saved in the cache.</td>
</tr>
<tr>
<td>Process description</td>
<td>At Optionen there ist he possibility to choose how and when pictures shall be downloaded automatically. The program may download only Thumbnails or all pictures- depending on the setting.</td>
</tr>
<tr>
<td>Exceptions, error situations:</td>
<td>If the buddy is in the status „nicht erreichbar“ an exchange is not possible.</td>
</tr>
<tr>
<td>Technical realisation:</td>
<td></td>
</tr>
</tbody>
</table>
7. Actions protocol

**Use Case UC7.1: Protocol for chat**

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>A chat has taken place</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The user can read the former message</td>
</tr>
</tbody>
</table>
| Process description | 1. Right-click on the user -> „Protokoll anzeigen“.
2. Locally a .tct-file is saved for every buddy. Here every chat is saved. |
| Exceptions, error situations: | |
| Technical realisation: | |

**Use Case UC7.2: Protocol for download**

<table>
<thead>
<tr>
<th>Actors</th>
<th>User, Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td></td>
</tr>
<tr>
<td>Postconditions</td>
<td>The user can read which folder from whom is download at which time.</td>
</tr>
<tr>
<td>Process description</td>
<td>Right-click on the folder or „Datei -&gt; Downloadprotokoll anzeigen“. Now a .txt-file is shown where the folder and the user is shown, as well the time.</td>
</tr>
<tr>
<td>Exceptions, error situations:</td>
<td></td>
</tr>
<tr>
<td>Technical realisation:</td>
<td></td>
</tr>
</tbody>
</table>

8.5. Black Box Testfälle

Version 0.1
Autor Daniel Putz
Datum 13.11.2006

1.0 Test cases „User registration“
<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>NF</td>
<td>Correct registration of a user</td>
<td>User registered, Login possible</td>
<td>Userdaten</td>
</tr>
<tr>
<td>1.2</td>
<td>FF</td>
<td>Leaving different information that are mendatory</td>
<td>Message to fill in information</td>
<td>Leaving mendatory information</td>
</tr>
<tr>
<td>1.3</td>
<td>NF</td>
<td>Adding Ä, Ü or Ö</td>
<td>Accepted</td>
<td>Ü, Ä, Ö etc</td>
</tr>
<tr>
<td>1.4</td>
<td>FF</td>
<td>Insert of wrong email address</td>
<td>No email to User</td>
<td>Wrong email address</td>
</tr>
</tbody>
</table>

Tab. 1: Testfälle „Registrieren“

2.0 Test cases „Login“

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>NF</td>
<td>Insert of right Information</td>
<td>User is logged on</td>
<td>Existing user name, correct password</td>
</tr>
<tr>
<td>2.2</td>
<td>FF</td>
<td>Login through unknown user</td>
<td>Message that login-information is not correct</td>
<td>Non existing Username, Passwort</td>
</tr>
<tr>
<td>2.3</td>
<td>FF</td>
<td>Login with wrong password</td>
<td>Message that login-information is not correct</td>
<td>Existing Username, wrong Password</td>
</tr>
<tr>
<td>2.4</td>
<td>FF</td>
<td>Login with Password from another user</td>
<td>Message that login-information is not correct</td>
<td>Existing Username, Password of another user</td>
</tr>
<tr>
<td>2.5</td>
<td>NF</td>
<td>Login as user</td>
<td>Login into program</td>
<td>Data of user</td>
</tr>
</tbody>
</table>

Tab. 2: Testfälle „Login“

3.0 Test cases „add User“

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>NF</td>
<td>Search a user</td>
<td>User is shown</td>
<td>Parameter (ex.: City, Country)</td>
</tr>
<tr>
<td>3.2</td>
<td>NF</td>
<td>Adding a new buddy</td>
<td>New buddy is shown in list</td>
<td>Parameters that are searched for</td>
</tr>
</tbody>
</table>

Tab. 3: hinzufügen

4.0 Test cases „Add folder“
<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>NF</td>
<td>Choose Folder, choose viewer</td>
<td>Pictures of a folder were shown. Folder is added to list „Freigaben“</td>
<td>Link to the folder</td>
</tr>
<tr>
<td>4.2</td>
<td>SF</td>
<td>Adding an empty folder / a folder without pictures</td>
<td>Ordner in Freigaben, shown.</td>
<td>Link to the folder</td>
</tr>
<tr>
<td>4.3</td>
<td>SF</td>
<td>Adding without user notice</td>
<td>Into the list „Freigaben“ are no users listed</td>
<td>Only folders</td>
</tr>
<tr>
<td>4.4</td>
<td>FF</td>
<td>Cancel during choosing users</td>
<td>Folder is not shown, Process is canceled</td>
<td>Click on cancel</td>
</tr>
</tbody>
</table>

Tab. 4: Ordner hinzufügen

5.0 Test case „Add group“

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>NF</td>
<td>Adding oof a new group</td>
<td>Group is shown in the main window</td>
<td>Name of the group</td>
</tr>
<tr>
<td>5.2</td>
<td>FF</td>
<td>Adding without entering a name</td>
<td>Exception -&gt; enter name</td>
<td>Leaving information</td>
</tr>
<tr>
<td>5.3</td>
<td>NF</td>
<td>Insert of Ä Ü Ö</td>
<td>No problem</td>
<td>Ü, Ä, Ö etc</td>
</tr>
</tbody>
</table>

Tab. 5: Gruppe hinzufügen

6.0 Test case „View Pictures“

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>NF</td>
<td>Watching pictures of a user</td>
<td>Folder with pictures is shown</td>
<td>Double click on user</td>
</tr>
<tr>
<td>6.2</td>
<td>SF</td>
<td>User does not offer any pictures</td>
<td>No Folder is shown &amp;&amp; Info that user did not offer any pictures</td>
<td>Double click on user</td>
</tr>
</tbody>
</table>

Tab. 6: Bilder betrachten

7.0 Test case „View preview pictures“

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>NF</td>
<td>Watch preview-pictures of a user</td>
<td>Bild is shown. By further click all pictures from the folder are shown.</td>
<td>Double click on picture</td>
</tr>
</tbody>
</table>
7.2 SF User does not offer any pictures  
No pictures is shown  
Not possible (double click on picture)

Tab. 7: Vorschaubilder betrachten

8.0 Test cases „Download (all) pictures“

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>NF</td>
<td>Download picture</td>
<td>Chosen pictures is downloaded into folder</td>
<td>Choose picture / Click „Download picture“</td>
</tr>
<tr>
<td>8.2</td>
<td>NF</td>
<td>Download all pictures</td>
<td>All pictures from target folder are downloaded</td>
<td>Choose picture / Click „Download all“</td>
</tr>
<tr>
<td>8.3</td>
<td>FF</td>
<td>No pictures available</td>
<td>Buttons inaktiv</td>
<td></td>
</tr>
</tbody>
</table>

Tab. 8: Bilder downloaden

9.0 Test cases „Action of user“

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Kl.</th>
<th>Description</th>
<th>Result</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1</td>
<td>NF</td>
<td>Block user</td>
<td>Confirm – User is shown offline. Watching pictures not possible anymore</td>
<td>Right-click on user</td>
</tr>
<tr>
<td>9.2</td>
<td>SF</td>
<td>Unblock user</td>
<td>User is shown as usual</td>
<td>Right-click on user</td>
</tr>
<tr>
<td>9.3</td>
<td>NF</td>
<td>Delete group</td>
<td>Confirm – Group is deleted. All contacts in main group</td>
<td>Right-click on user</td>
</tr>
<tr>
<td>9.4</td>
<td>NF</td>
<td>Delete contact</td>
<td>Confirm – Deleted</td>
<td>Right-click on user</td>
</tr>
</tbody>
</table>

Tab. 9: Useraktionen
8.6. Screenshots

8.2 Screenshot Picnic – Picture preview windows

8.3 Screenshot Picnic – Message window
8.4 Screenshot Picnic – Add a new group

8.5 Screenshot Picnic – User information
8.6 Screenshot Picnic – Different views. Incl. right-click-menu

8.7 Screenshot Picnic – Search and add a new user