VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SOCIAL NETWORK INTEGRATION INTO AN INFORMATION PORTAL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE DRAHOMÍRA HERRMANNOVÁ
AUTHOR

BRNO 2010

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INTEGRACE SOCIÁLNÍCH SÍTÍ DO INFORMAČNÍHO PORTÁLU

SOCIAL NETWORK INTEGRATION INTO AN INFORMATION PORTAL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — DRAHOMÍRA HERRMANNOVÁ
AUTHOR

VEDOUCÍ PRÁCE — Ing. RADEK BURGET, Ph.D.
SUPERVISOR

BRNO 2010

Abstrakt

Cílem bakalářské práce „Integrace sociálních sítí do informačního portálu“ bylo vytvoření portálu, který by předvedl způsoby integrace současných sociálních sítí na internetu do jednoho portálu. Práce obsahuje seznámení se základními pojmy v této oblasti, jako je Social Web, Web 2.0 a sociální sítě. Dále představuje jednotlivé sociální sítě, které byly intergovány do portálu, a technologie a standardy k tomu použité - OpenID, OAuth a OpenSocial. Dále práce popisuje také jednotlivé etapy vývoje zadaného webového portálu. V závěru práce je také shrnutí a zhodnocení výsledků a zamyšlení nad možnými rozšířeními výsledného portálu.

Abstract

The task of the Bachelor's Thesis „Social Network Integration into an Information Portal“ was creating a web portal, which would demonstrate technical possibilities of integration of current social networks on the internet into one portal. The thesis contains introduction of basic concepts in this field - Social Web, Web 2.0 and Social Networking. Furthermore it introduces the individual social networks which were integrated in the portal, and the technologies and standards used for the integration - OpenID, OAuth and OpenSocial. It also describes all stages of development of the portal. In the end of the work, there is a summary and evaluation of the results of the work and discussion of the possible future extensions of the portal.

Klíčová slova

Socisl Web, Web 2.0, Sociální sítě, Facebook, MySpace, Twitter, Google Buzz, OpenID, OAuth, OpenSocial

Keywords

Social Web, Web 2.0, Social Networking, Facebook, MySpace, Twitter, Google Buzz, OpenID, OAuth, OpenSocial

Citace

Drahomíra Herrmannová: Social Network Integration into an Information Portal, bakalářská práce, Brno, FIT VUT v Brně, 2010

Social Network Integration into an Information Portal

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana Ing. Radka Burgeta, PhD., a že jsem uvedla všechny literární prameny a publikace, ze kterých jsem čerpala.

....
Drahomíra Herrmannová
July 29, 2010

Poděkování

I would like to thank my supervisor from BUT FIT, Ing. Radek Burget, PhD., for his pedagogical and professional guidance, which helped me to successfully finish my project. I would also like to thank my supervisors from Universidade de Trás-os-Montes e Alto Douro, Benjamim Fonseca, PhD. and Hugo Paredes, PhD., for their constant comments of my work and for their professional advice, which apart from finishing my work also helped me with developing my skills and learning lots of new things. Their help was very important during all the stages of development of my project. I cannot also forget my family, who was supporting me a lot during my work at UTAD, for that I would like to thank them a lot.

Contents

Chapter 1

Introduction

In last years a new trend has appeared on the Internet - something what we call *social networking*. People are creating online groups and communities, form online friendships and willingly share personal information about themselves. Web is becoming more and more social as number of social networking websites' users constantly grows. Every day there are lots of new standards and technologies used for creating social networking applications appearing.

This project's aim was to create a web portal, which would demonstrate integration of some current social networking services and also some most important standards and technologies which are related to social networking applications and social networking in general. It was also required to integrate these applications in a form of a common interface, which would be then usable in any web project.

This project can be divided into two main parts. First part focuses on theoretical background of the topic and describes the concept of social networking, it's impact on the rest of the web content and standards which are used in this area, while the second part of the project describes the different phases of the implementation of the portal.

At the beginning of the theoretical part of the project, the term social web[1] is being introduced. The reader can learn something about the significance of social networking for the Web 2.0, about different kinds of socialisation, which exist and are used nowadays, but also about possible future of social web.

After this brief introduction to the topic, there is a part devoted to introduction of social networking, description of the social networking applications and their APIs, which were selected for the integration, and description of some standards used in social networking applications.

Second part of the project is more practical. It is describing the analysis of the problem, design of the application and the following implementation. In this part the most attention is paid the the design and implementation of the common interface for the social networking applications.

The final two chapters describe the application testing and the project conclusion. In the final chapter some possible extensions of the portal are discussed.

[1]Some sources write this term with capital letters at the beginning of both words, some sources write it with lower letters. I don't see any special reason why it should be written with capital letters, so in the whole document I will use lower letters.

Chapter 2

Project aims

The aim of this project was, most importantly, to gain knowledge of current popular social networking services and of the happening in the social web in general. The task was to create an information portal, which would integrate some most popular social networking applications and through that demonstrate how can these applications be integrated.

2.1 Common Interface

It was also required to integrate these applications in a way which would provide some universality and reusability. Nowadays, the social networking websites are probably one of the simplest ways of presenting and sharing information and so it would be useful to have a simple interface, which would allow any website to present it's data on the social networking sites.

Therefore I was asked to integrate these applications in a form of a simple and extensible API, which could be later used in other projects. The requirements for this API were:

- The code used for integration of social networking applications in the portal should be easily reusable. It can be simply added and used in any web project.
- It should be possible to simply add any other social networking applications, after the portal is finished. Basic implementation will integrate only the most popular sites but it should be possible to (more or less) simply add any other site later.
- The interface should provide a set of methods to connect with selected applications and then to post data (simple text, links, pictures and video) to these applications without always asking user for permission.

Therefore during the analysis, design and implementation of the project I focused mostly on the part of the integration of the social networking applications. The final interface was later really used in a different web project, this usage is described in chapter 10.

The following chapter discusses the first of the project aims - introduction to the social web.

Chapter 3

Social Web

The contents of this chapter is an explanation of the very basics of Internet socialisation concepts. The term social web is introduced in the chapter, as well as term Web 2.0 and the significance of social web for Web 2.0. This chapter also contains a brief description of the different kinds of Internet socialisation and different social applications. In the end of the chapter the reader can read a short discussion about the future of the social web.

3.1 Definition of the term Social Web

Social web is a term describing one important aspect of today's Internet - the electronic socialisation of the users of the web. Unlike in earlier stages of the Internet, the content of the Internet is now made by it's users and also thanks to different social applications which bring these users online and connect them together.

People don't come online just to search for and view content, one of the biggest trends of the current Internet is the fact that users can interact between themselves, share their interests, present themselves to wide public or just to a few close friends. There are really no limits when it comes to the different kinds of Internet socialisation.

The common key features of all social applications is that they focus on allowing users to create user profiles and to create connections with other users of the application. The nature of these connections may vary a lot [1].

Probably the most common kind of social applications is the one which will probably come in most people's minds if they hear the term social web. It's the kind of socialisation where all focus of the communication are the people who communicate. In this kind of social applications user profiles are very important aspect and in general these applications serve just for fun. A representation of this kind of socialisation are websites like Facebook, MySpace or Orkut.

Apart from this kind there are websites where users can make connections according to their shared interests and activities, like hobbies or work field. A good example of this kind of socialisation are sites like Last.fm or IMDB (connections according to the musical/movie preferences) or a popular site LikedIn, which helps people to create their professional network online.

One very special characteristic of social applications is that they may also connect people who never met in real life and who are thus complete strangers. This may result to connections between people who would otherwise never even have a chance to meet - like people from different parts of the world or from completely different social background.

Quite often people who met online can also later meet in real life (like for instance users of dating sites).

3.2 Web 2.0

As mentioned before, the biggest trends in today's Internet are concepts like information and data sharing, collaboration and focus on user. The users of the Internet are now allowed to share any data online. Nowadays this is something that is matter of fact for us. However Internet was not always so „user friendly“.

The term Web 2.0 was firstly used in the O'Reilly Media Web 2.0 conference in year 2004. The name of the conference was chosen to present the concept of new Internet, which allows user communication in contrast to the past static web concepts [19].

When the Internet was presented to wide public in 1991, the main feature of it was static presentation of data. Internet contained lots of useful information and downlowdable applications, but all the content was static, it was not possible to participate on it's improvements or see how the applications work [9].

A good example of such presentation might be a personal portfolio. Before the Web 2.0 came, such portfolio would be a static presentation with static information about it's creator. User can read the information, but that's all. In Web 2.0, user can leave comments while the owner of the portfolio is able to update the content of his portfolio easily and often. It's obvious, that the Web 2.0 is dynamic, interactive and always focusing on it's users [9].

The situation on the web began to change around the beginning of the 21st century, after the famous fall of the Internet companies at the end of so-called „dot-com bubble“. Before that it was common that the Internet companies focused mostly on fast growth of the company through increasing the number of the users which was then supposed to create higher profit. As opposite to this, at the begging of the 21st century, big companies like Google or Microsoft slowly started to share their data with users, started to focus on creating content and on involving the user in the content creation [13].

In the Web 2.0 the view of the user completely changed. User was not anymore just a reader of the page. In fact, most of the content of today's Internet is created by it's users. Internet visitors are constantly encouraged to connect to social networking sites and share their interests, to comment on everything they see, to collaborate together and through that improve the content of the Internet. Of course there are not only bright sides of this, one of very discussed topics today is Internet privacy. However, the Web 2.0 brought us a completely new and incredibly rich information source.

3.2.1 Web 2.0 and socialisation

A massive boom of Internet socialisation was a logical outcome of moving to Web 2.0. Socialisation is one of the most important features of Web 2.0. In general, the social applications are built to connect users and allow them to share any kind of data - through that these applications take part in improving the content of the web.

Some current sites are even driven only by their user community, a good example of this is a video sharing service and social network in one - YouTube.

The process of web socialisation also brings together consumers and businesses and greatly helps Internet commerce and brand enhancement [14]. Social Web is simply most powerful way to share your content and to present yourself.

3.3 Different kinds of Internet socialisation

A social software or social applications are software systems which allow users to present themselves and share data and ideas. There are new applications and concepts being birth every day and the list of all kinds of social software would be incredibly long. However, I will try to shortly describe here the most common services [8]:

- **Blogs**
  The name blog is a short for words web log. Blogs are kind of online diaries, journals or personal websites. They store entries which are generally sorted according the date. The focus of this kind of social software is either the user (the creator of the blog) or some interest of the blog's creator/s. Blogging services allow their users to post articles about any topic. These articles may generally contain pictures, links or any other Internet media and their combination. Readers of the blog may leave a feedback in form of comments of the articles. One very popular form of blogging is so-called microblogging (the user is posting only short messages), represented with very well known application Twitter. Popular blogging sites are for instance Blogger (powered by Google) or WordPress. Also some social networking websites allow users to create blogs as a part of their profiles, that is, for instance, MySpace or Windows Live Spaces.

- **Wikis**
  Wikis allow their visitors to create pages with formatted text and links to other pages or sites. Other visitors can then change the content of the page (generally all changes are automatically documented). This kind of social applications can be used to create online encyclopedias, to create collaboration pages for different projects or to simply post any kind of notes online. Generally, the point of wikis is that that they allow their readers to correct the mistakes in the text and to improve it. The most popular wiki is the Internet encyclopedia Wikipedia.

- **Social bookmarking**
  Social bookmarking websites are sites where users can create a profile and then save links for articles, sites or any other web content which they liked. The bookmarks of all users can then be put together, sorted in categories and ordered by popularity and/or tags, so they create a massive database of popular web resources. Most popular bookmarking sites are Digg, Delicious or StumpleUpon.

- **Social networking websites**
  The social networking websites are nowadays probably the most common way of Internet socialisation. Focus of these social applications is aimed to the user. This kind of web sites serves it's users to create online profiles, create connections with other people and through this to present themselves. And the presentation is not limited only to sharing text information. In general, almost every social networking site allows it's users to share also images, video and other web content. Apart from the most popular site Facebook, other examples of popular social networking sites are MySpace, Orkut, Hi5 or Windows Live Spaces.

- **Massively multiplayer online games**
  Massively multiplayer online games (MMOGs) are kind of games, which are played online and where the users play together. Generally, these games create online worlds,

where each user can create it's own avatar and then explore the game world and interact with other users. The most popular multiplayer online game is World Of Warcraft. Also a game called Second Life is a good example of massively multiplayer online game.

- **Internet forums**
  Internet forums serve it's users to create online discussions. Usually, each Internet forum is related to one field or one topic and then the users can create individual threads where any other user can add a comment or his opinion of the topic. Usually every forum uses a specialised software. Most Internet forums are public, but there are also forums which require registration in order to view the content and post new comments. One successful service that supports Internet forums is Google Groups.

- **Instant messaging**
  Instant messaging is a service which allows the users to communicate online in real time. Most often the communication is done between two people, but lots of messaging software allows discussion involving more people. Instant messaging can be done directly inside a web browser or by using a specialised software. This kind of communication offers it's users a relative privacy. Recently, instead of just text messaging, it's also possible to create online calls and video calls. Very often all this functionality is provided by the same service. For instance Google Talk and MSN Messenger both offer text and video chat. Other examples of popular instant messaging services are ICQ and Skype.

- **Other kinds of Internet socialisation**
  There are many other kinds of Internet socialisation and as mentioned already, there are new kinds being born almost every day. An example can be so-called e-learning systems. Very often social networking sites also combine more items from this list. For instance Facebook is a social networking site which also offers a place for blog, instant messaging between friends or creation of Internet forums (Facebook Groups).

3.4 Future of the Social Web

Current situation in Social Web is that there are many separate applications. It's common that users have accounts on more than one social networking application. For instance, somebody might use Facebook for connecting with friends and family, Twitter for reading short actualities from some field of interest of the person, and LinkedIn for professional connections. But generally users have more accounts than just three. Every new account and profile they create makes it more and more difficult to keep everything updated and also to remember which data has the user shared and where.

However in last years there has been some progress in changing this situation. Social networking applications share their data through APIs and through this they allow other sites to download data about the user. Some sites also allow a third-party login - a mechanism when user can log into one site using his login credentials from another site and also share some basic profile data between the sites. List of third party login providers contains Facebook (according to [7] Facebook login is currently used by almost half of the people using third party account to sign in), Google, Twitter or Yahoo!.

There are always some attempts to make the data of the Social Web more connected, but the current state is not optimal. However the future development of Internet socialisation will probably lead to completely shared data. In the future, user could have only one profile which he could connect with any application available on the web. This profile would contain all information about the user, complete list of his friend connections and list of all data user has shared online. Then the user could define which parts of his profile would be accessible from which application. For instance with some professional network like LinkedIn user could share his education history and employment history, but other parts of the profile, like favourite music or movies, would remain inaccessible for the site.

Through this, web could become even more social and it's possible that the future development of social applications will go this way. For instance Google is already creating a set of APIs which could help this task (Google OpenSocial, Social Graph or Portable Contacts). There is also a W3C Social Web Incubator Group belonging to World Wide Web Consortium which tries to suggest possible ways of social Internet development, and a small group of developers called OneSocialWeb which is trying to define a language for communication between all social networking applications. The standards affecting Internet socialisation will be introduced in chapter chapter 5.

Chapter 4

Social Networking

This chapter contains an introduction to the Internet social networking. Apart of presentation of most commonly used social networking sites, there is also a section about general social networking concepts and about the application domains of social networking. This chapter also shortly introduces the common known issues of this form of socialisation.

4.1 Social Networking concepts

Social network is composed from individual people and organisations and from connections between these actors. Every actor symbolises a node in the network [10]. Social networking sites allow their users to create online profiles and connections with other people. Through social networking sites, social networks are built. These online social networks may or may not reflect the users real life connections.

In general there are several types of social networking sites. They can either connect people of any age and with any interests or they can help people build online communities according to their shared interests. An example of general social network is Facebook. On the other hand there is Last.fm, which is a social network connecting people according to their music preferences.

Each social networking site allows it's users to create profiles and connections, but usually every site adds some extended functionality. It's common that the social networking site allows the users to send private or public messages, enhance the users profile with uploading photos or videos, with adding applications to the profile, with allowing users to create interest groups and so on.

In general we can see that social networking applications integrate the following attributes [14]:

- **Identity:** who are you?
- **Reputation:** what do people think you stand for?
- **Presence:** where are you?
- **Relationships:** who are you connected with? who do you trust?
- **Groups:** how do you organise your connections?
- **Conversations:** what do you discuss with others?

- **Sharing:** what content do you make available for others to interact with?

Social networking application generally implement these features, but not necessarily all of them.

Online social networking is a concept which is older than it would seem. The first social networking sites were created already in the second half of the 1990s. Probably the first real social networking site was called SixDegrees. However a real boom of social networking came few years later, at the begging of the 21st century with social networking applications like Friendster (2002), MySpace (2003), LinkedIn (2003) and Facebook (2004) [2].

Nowadays the most popular social networking applications which are used all over the world are Facebook and Twitter (launched 2006). Other applications are usually used only in some parts of the world. For instance MySpace and LinkedIn are very popular in North America, while Orkut is the most common social network in Central and South America [2].

4.2 Application domains

Social networking is recently being used for many different goals. One of them (probably the most often) is usage of social networking sites for business, to promote a product or a brand. Of course it doesn't have to be a company or a brand, nowadays it's quite common that for instance politicians create public profiles at social networking sites and through that they try to increase their popularity.

One way of promotion are the various adds presented to the social networking sites' users. These sites connect people from all over the world and have access to different kinds of information about it's users, like topics they read, content they like and similar things. Using this gathered data they are able to offer the user appropriate content. This method is used by Google to display adds to it's users in all Google's services (even inside your GMail you can see adds which are selected according to the content of your mail).

The other way of promoting a product or a brand is called brand networking. It's a way how a brand or a company can interact with it's customers using Internet socialisation. A good example of social network used for business purposes is LinkedIn where users can create connections according to their professional interests.

Another use of social networking can be for online dating. There are several sites which were created specifically for these purposes. This kind of social networking sites often connects users according to their shared interests.

4.3 Issues of social networking

As mentioned before, social networking is not always only positive. It brings several more or less important issues.

4.3.1 Privacy and information security

Although it represents very rich information medium, social networking also brings a big loss of privacy. Of course no data put online will ever be 100% safe, there is always a risk that someone will manage to breach the security of the system and steal a potentially sensitive data. Major or minor attacks on the big social networking sites are in the Internet world very common.

However, people often don't realize this risks and are willing to share lots of very personal information about themselves on different places. Of course, the more data user shares online the more he is in danger. People also feel more secure when they know that their profile is accessible only for their friends, but if such person has a connection with, let's say, 200 people, for an attacker it would be enough to get a password of one of them and he would already be able to access all his friends profiles.

There is also always a risk of misusing the data about the users. As mentioned before, Google is displaying adverts inside GMail according to the content of the mail. We can really never know who has or has not access to our private information and therefore the best solution is to simply not share too many information online.

4.3.2 Spamming

Very annoying issue of social networking is also spamming. It's very common that social networking applications send notifications to it's users. These notifications can be about anything - that someone added the user as a friend, that the user received a personal message and similar. If the user has lots of connections on the network, number of these notifications per day can grow really rapidly. Fortunately nowadays all social networking sites allow their users to switch off all notifications.

4.3.3 Time consumption

Time consumption is a serious issue of all kinds of online social networking. Even the users themselves admit, that they spend too much time at social networking sites, by looking on pictures of friends, chatting or simply playing games. This issue has been discussed publicly many times, but it changed nothing. It even happens that people use social networking applications at work. Among teenagers it's common that they communicate together more often online, although they could go out and meet. However this situation in not likely to change very much in the future.

4.3.4 Other issues

There are lots of other issues related with social networking. It can be creating fake identities and misusing them for any purposes[1]. Social networking sites also allow any kind of gossip or fake information to be spread world wide. The problems really are various and therefore all users of these sites should be careful about everything they do online.

4.4 Popular social networking sites

Nowadays there are many different kinds of social networking applications. Some focus completely on the user. On such sites, the user profiles are the main building blocks. Examples of these applications are Facebook, MySpace, Orkut, Hi5, Windows Live Spaces and similar.

There are also websites which connect users according to their interests and activities. People who like photography can connect via Picasa (powered by Google) or via Flickr

[1]For instance recently a hacker created a fake military intelligence profile under name Robin Sage and by using this profile he manages to acquire sensitive military information. Source: http://www.darkreading.com/insiderthreat/security/privacy/showArticle.jhtml?articleID=225702468

(powered by Yahoo!). There are lots of sites which connect users for instance by preferred music or movies. Examples of such websites are Last.fm or International Movie Database (IMDB).

As already mentioned there are also social networking sites for business purposes, like LinkedIn. Another kind are sites for uploading, sharing and viewing videos and other media, such networks are for instance YouTube and Wikimedia Commons.

Also online auctions behave a little like social networking sites. Examples of popular online auctions are Ebay and Amazon.

The above mentioned sites can be found at following addresses:

- **Facebook**
  `http://www.facebook.com`
- **MySpace**
  `http://www.myspace.com`
- **Orkut**
  `http://www.orkut.com`
- **Hi5**
  `http://www.hi5.com`
- **Windows Live Spaces**
  `http://home.spaces.live.com`
- **Picasa**
  `http://picasaweb.google.com/`
- **Flickr**
  `http://www.flickr.com`
- **Last.fm**
  `http://www.last.fm`
- **IMDB**
  `http://www.imdb.com`
- **LinkedIn**
  `http://www.linkedin.com`
- **YouTube**
  `http://www.youtube.com`
- **Wikimedia Commons**
  `commons.wikimedia.org`
- **EBay**
  `http://www.ebay.com`
- **Amazon**
  `http://www.amazon.com`

Chapter 5

Standards in Social Web

This chapter contains an introduction to the some common standards and technologies which are related to online social networking. Most importantly this chapter introduces OpenID and OAuth, two open technologies which are used practically in every social networking site. However also some other technologies are shortly described in this chapter.

As already mentioned in section 3.4, the current situation situation in Social Web is that there are many distinct applications which by default do not share data with other applications, generally the user has to create a new profile for each application. This approach is not very user-friendly, because the user has to remember many passwords and keep a trace of which applications is he already using and which data has he posted there. However there are already some efforts towards changing the current state of the online social networking.

5.1 OpenID

One option of moving towards standardisation on the web is using a web ID to identificate people. One such web ID is OpenID. The following section about OpenID is based on [17].

5.1.1 Introduction to the standard

OpenID is an open standard which allows decentralised authentication of users. Simply put, OpenID allows users to easily sign in all over the web with just one ID. In case of OpenID open standard, this ID takes form of URI (Uniform Resource Identifier), because URI is in web architecture used to identify resources. So, that means that every person on the Internet can be identified by one URI (e.g. `bob.openid.example.com`).

To prove that the user is really owner of the identifier there is OpenID Authentication protocol. One key feature of the protocol is that the user doesn't need to provide his login credentials to the third party application. Instead he is redirected directly to the OpenID provider and the provider manages authentication of the user.

If the user signs in successfully, the OpenID provider then generates a token and returns this token to the third party, which uses the token to identify the user.

Another important feature of OpenID protocol is that this form of authentication is completely decentralised. That means that there doesn't have to be a central authority to approve the providers.

OpenID Authentication is completely HTTP-based, so it doesn't require any special features to be installed on the user's computer or browser.

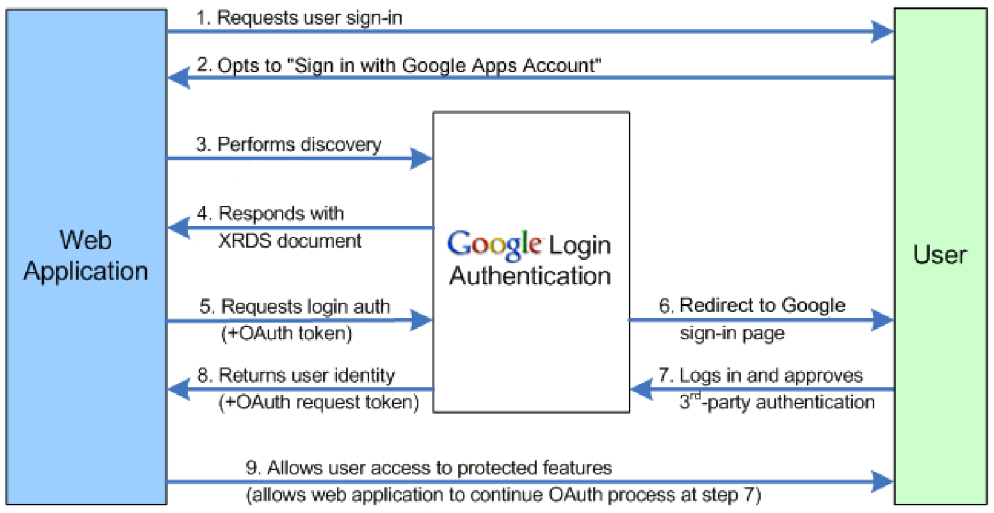
Web Application
Google Login Authentication
User
1. Requests user sign-in
2. Opts to "Sign in with Google Apps Account"
3. Performs discovery
4. Responds with XRDS document
5. Requests login auth (+OAuth token)
6. Redirect to Google sign-in page
7. Logs in and approves 3rd-party authentication
8. Returns user identity (+OAuth request token)
9. Allows user access to protected features (allows web application to continue OAuth process at step 7)

Figure 5.1: OpenID Authentication flow as used for Google Accounts (source: Google - http://code.google.com/apis/accounts/docs/OpenID.html)

5.1.2 Protocol overview

OpenID Authentication protocol contains several steps. The outcome is then a unique token which can be used to identify the user.

1. The third party application should first (before or after the user requests sign in with OpenID) do a discovery on the OpenID provider. Outcome of this step is address of OpenID endpoint which should be used to request the user sign in.
2. After the endpoint address is obtained, the user should be redirected to this address. There he will be asked to sign in with his OpenID login credentials and to allow the third party application to use this sign in.
3. If the user allows the application to use his OpenID for sign in, he is redirected back to the client application, together with OpenID token.
4. The client application should then validate the response from the server by sending a validation request to the server.

5.1.3 OpenID extensions

There are currently several extensions to the OpenID protocol which may nicely improve the user's impression of the sign in process. Some of the extensions are:

- **Attribute Exchange (AX)**
  This extension allows exchange of detailed information about the user between the provider and the relying party [16].
- **Simple Registration (SREG)**
  SREG allows simple registration process. When user wants to register a new account, with this extension he can share some basic profile information between the site and the provider [15].

- **Provider Authentication Policy (PAPE)**
  An extension to make the OpenID Authentication flow more secure. For instance it's possible to set the maximum time for which the user authentication is valid. After this time, the provider should again prompt the user for password [18].

5.2 OAuth

Another standard used to unify the communication in the Social Web is OAuth Open Authorisation. The following section about OAuth is based on [12].

5.2.1 Introduction to OAuth

OAuth is an open protocol which allows a secure way of accessing protected resources on the web by generating access tokens and signing the requests.

If a third party wants to access protected resource or resources of a user which are stored on some web server, it doesn't have to work directly with the user's login credentials to access the resource. Instead the user authenticates directly on the server and grants the third party an access token which can be then used to access the protected data.

When the user grants access to the third party, he can specify which resources can the third party access. It is also possible to generate a token which is valid only for limited period of time.

In general, OAuth protocol defines a secure way of API authorisation. Like in OpenID, the OAuth Authorisation is entirely HTTP-based.

5.2.2 Protocol overview

OAuth Authorisation contains four steps, the outcome is a pair of tokens - OAuth access token and OAuth access token secret. The token secret is used for signing the API requests and should never be sent through Internet in open form.

1. If a client application wants to get access to protected resource of a user, first of all it asks the server to provide a request token.

2. After the client application gets the request token, it redirects user to the server (with the request token as a parameter of the request) and the user is then asked to grant access for the resource to the client application.

3. If the user grants the access, he is redirected back to the client application, together with authorised request token.

4. The client application exchanges the authorised request token for an access token. This token can now be used to access the protected resource of the user.

5.2.3 Signing the requests

Once the client application receives the OAuth access token, it can start making requests to the server API to access the user's resources. To validate the requests, every request must be signed according to the OAuth specification.

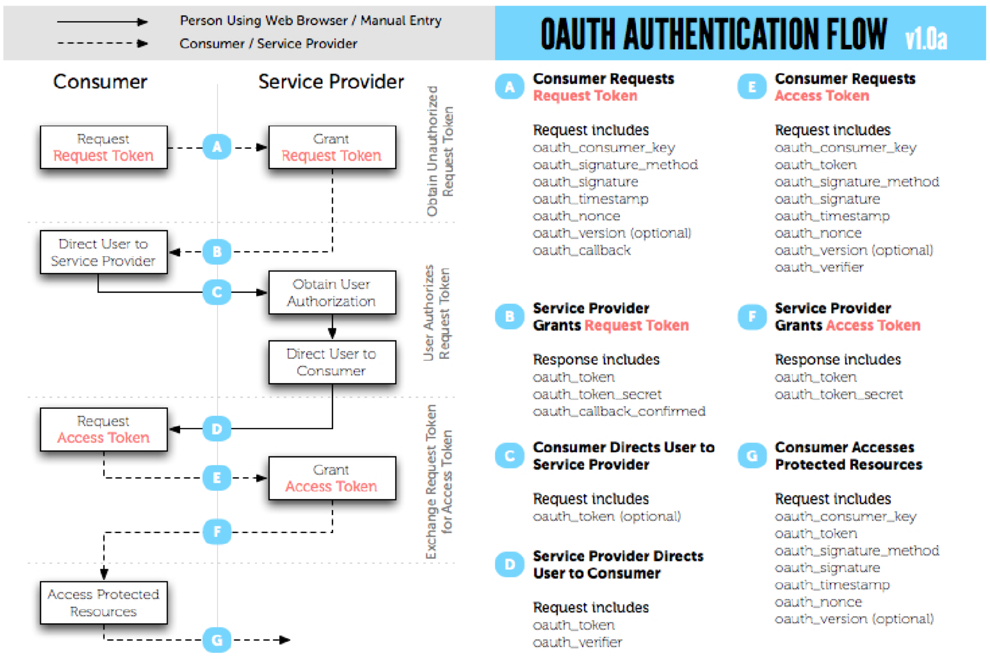

Figure 5.2: OAuth Authentication Flow (source: OAuth Core Workgroup - http://oauth.net/core/1.0/)

The request must contain OAuth specific parameters - most importantly the OAuth access token and OAuth signature. To generate a signature of a request, all OAuth parameters as well as any other parameters and the post body must be concatenated into one string which is called *base string*. The base string also contains method of the request, address of the request and path to the resource.

The signature is then created from this base string using algorithm selected by the server and added to the OAuth parameters of the request.

5.3 OpenSocial

OpenSocial is a common API developed by Google along with some other companies. This API is being developed to allow uniform communication with different social services on the web.

It can be used for both - embedded applications which are run directly inside the social networking site, or for external applications which want to access the social networking application's data from the web. It allows developers to create social applications which can communicate with any social networking site that supports OpenSocial.

The API is based on HTML and JavaScript, but to ease the creating of the embedded applications the developers can also use OpenSocial Templates [6].

5.4 W3C Social Web Incubator Group

W3C Social Web Incubator Group is part of World Wide Web Consortium and was created to keep track of the happening in the Social Web, to collaborate in creating new standards and through this to improve the Social Web. It's task is also to promote existing standards on the web. Additional information about this group: http://www.w3.org/2005/Incubator/socialweb/.

5.5 Other standards

There are several other standards and technologies which are related with Social Web. Just to mention some of the for an example:

- **OneSocialWeb**
  http://onesocialweb.org/
  OpenSocialWeb is an initiative of Vodafone Group Research and Development and it's task is to create one universal language to bridge existing social networking applications. For this task, the developers of OneSocialWeb use XMPP protocol.
- **Portable Contacts**
  http://portablecontacts.net/
  Portable Contacts is a group which is currently trying to unify the way how developers can access the contacts data from different social networking sites. Big aim is put in making the communication secure in a way which would not require that the user provides his login credentials to the third party site.
- **Social Graph API**
  http://code.google.com/apis/socialgraph/
  Social Graph API (developed by Google) provides developers a way how to maintain user's connections from several social networking sites.

Chapter 6

Integration of the social networking sites into a portal

The following chapter is introducing the social networking sites which were selected for integration into the portal and also the reasons which led to choosing these networks.

As explained in the chapter 1, the topic of the project was to integrate social networking sites available on the Internet into an information portal. First step of this integration was a selection of sites to integrate and then studying APIs of the selected sites and the technical possibilities of integration of these sites in the portal.

While selecting the sites I was paying attention mostly to the number of users of the network. For demonstration of how the social networking sites can be integrated, I selected four commonly used networks: Facebook, Twitter, MySpace and Google Buzz. First three are really used by millions of users world wide. Google Buzz is a new social networking application, which is not used so commonly, but I decided to integrate this application to get some basic knowledge of the technologies and data formats used by Google.

After I selected these four sites for integration, I created accounts at those sites where I didn't have an account yet and I started with studying their APIs.

6.1 Social networking sites APIs

An application programming interface (API) is a way how computer software requests other software for services. For the requests the asking program uses a set of standardised calls which are defined by the called program [3].

All of the four selected social networking sites have their own APIs which can be used to access their data. Since these programs are all web applications, the APIs are based on simple HTTP requests. Each of the networks provides a several libraries in different languages, which wrap these HTTP calls, however I decided to use directly the HTTP calls for creating my application, because this would be the simplest way how I could make calls in all the applications using the same code.

As I started to study the APIs, I learnt that all four APIs follow the principles of the REST architecture.

6.1.1 REST architecture

This section is based on [5].

Representational state transfer (REST) is a software architecture which is used in World Wide Web. It's a client-server architecture, clients are separated from servers to allow the components to be developed independently. For this there is also one another requirement - the interface between clients and servers must be uniform.

The communication between client and server is stateless, that means, session state is kept one the client and each request from the client must contain all the information necessary to understand the request.

In order to improve network performance and efficiency, the client must be cacheable. That requires that all requests explicitly say if they are cacheable or non-cacheable.

There are intermediary servers between the client and the server, so the client cannot tell whether it's connected to the end server or not.

REST architecture also allows code-on-demand. Functionality of the client can be extended though downloading simple scripts which will then be run directly on the client. This improves system extensibility and simplifies the client.

These basic principles define the REST architecture. All four APIs which I selected for integration in the portal follow these principles.

6.2 Facebook

Facebook is a social networking site launched on 2004. It is a social networking site with most users over all the world.

6.2.1 Description of the network

It allows it's users to add connections to their friends and build their user profile by adding information or uploading pictures and video. Users can also create interest groups, send personal or public messages to others and use a built-in real time chat.

6.2.2 Facebook Graph API

To communicate with the external applications, Facebook defines the Facebook Graph API (it is called Graph API, because it represents the Facebook social graph).

This API follows the principles of the REST architecture. The requests in the API are made in form of simple HTTP requests. For secure authentication and authorisation, Facebook uses OAuth 2.0 (section 5.2). OAuth is also used to allow users to sign in third party sites with their Facebook account [4].

Facebook Graph API allows to select and search for data, to input and delete data and to update data. The format of the responses is always JSON (JavaScript Object Notation - format for a text representation of arrays and simple data structures).

API introduction and documentation can be found on the following pages:

- **Graph API documentation**
  `http://developers.facebook.com/docs/api`
- **Graph APi reference**
  `http://developers.facebook.com/docs/reference/api/`

6.2.3 Social plugins

Facebook also offers external pages so-called social plugins. These plugins can be included in any site across the web by adding only simple short HTML code in the web site. The social plugins offer a way how to add a Facebook „like button“ or comments on the website.

A documentation of Facebook social plugins can be found at the following page:

- **Social plugins**
  http://developers.facebook.com/plugins

6.3 MySpace

MySpace is (after Facebook) the second biggest social networking site, used mainly in North America.

6.3.1 Description of the network

MySpace is a social networking site which focuses on music. Musicians and bands can create special profiles, upload online their music and have access to milions of users of MySpace.

Apart from this MySpace also offers all the basic functionality like any other social network. Users can connect together, send public or private messages, use chat and so on.

6.3.2 MySpace REST API

This section is base on [11].

To communicate with external applications, MySpace defines REST API, which follows the principles of the REST architecture and implements the OpenSocial common API (section 5.3). This API allows secure server-to-server communication using simple HTTP requests. The format of the response from the server can be XML or JSON.

For the secure authentication and authorisation, MySpace uses OAuth (section 5.2). It's possible to allow users to sign in to third party sites using their MySpace ID, which is also an OpenID identifier (OpenID was described in section 5.1).

MySpace REST API documentation can be found at the following pages:

- **RESTful API**
  http://developerwiki.myspace.com/index.php?title=RESTful_API
- **OpenSocial v0.9 REST Resources**
  http://developerwiki.myspace.com/index.php?title=Category:OpenSocial_v0.9_REST_Resources

6.4 Twitter

Twitter is a social networking and microblogging site launched at 2006. It is one of the sites with most users and it's used world wide.

6.4.1 Description of the network

Twitter allows it's users to post and read short messages called *tweets*. The length of every message is limited to 140 characters. Tweets are by default publicly visible, however it's possible to change this settings and hide the tweets from people who are not at the user's contact list.

The connections between users are called *following*, while the subscribers to the users news feed are *followers*. It's also possible to send personal messages to the users.

6.4.2 Twitter API

Twitter offers a RESTful API to access it's data from third party sites. The API is entirely HTTP-based, but Twitter also offers libraries for every commonly used language. The response format is always XML or JSON. For secure authentication Twitter uses OAuth. OAuth is also used to allow users to sign in third party sites with their Twitter account.

Twitter REST API documentation can be found online:

- **Twitter API Documentation**
  http://apiwiki.twitter.com/Twitter-API-Documentation

6.5 Google Buzz

Google Buzz is a social networking site powered by Google. It's the newest of the integrated social networking sites, Google Buzz was released at the beginning of year 2010.

6.5.1 Description of the network

Google Buzz is integrated into other service from Google - into Google Mail. User can share the same content like on any other site - text, links, photos and videos. User can also choose which external sites he wants to connect to Google Buzz. If the user connects the Buzz with for instance Twitter, all messages which user posts to Twitter are then automatically propagated to Google Buzz. The services that are currently integrated into Google Buzz include also Picasa, YouTube, Flickr, Google Reader and Blogger.

6.5.2 Google Buzz API

Just like other integrated sites, Google Buzz provides RESTful API which is entirely HTTP-based. For third party sign in, Google uses OpenID, HTTP requests are signed with OAuth. The response from the server can be in JSON or in Atom formats. If a user profile is returned from server, it is in Portable Contacts format.

The Google Buzz API documentation can be found at the following link:

- **Google Buzz API**
  http://code.google.com/apis/buzz/

Chapter 7

Analysis

This chapter describes the task of the project in detail and explains why I chose this project. There is also a section about the common API which introduces the requirements for the API and explains how I decided to solve them. In the end of the chapter there is a note about availability of resources related to this topic.

7.1 Project assignment

The task of my project was to create an information portal which would integrate some of the social networking applications that are currently available on the Internet. A specific functionality of the portal was not defined in the project submission. It was only required that the portal would demonstrate the different means of how social networking sites can be integrated into a web application. The language in which the project should be written was not specified.

I had several reasons for choosing this project. Already when I started to think about some possible topic, I knew I would like to create a project for the web. Last time, number of things we can do online is growing. When I use computer, generally I don't need to run any other program apart from web browser.

Of course, I am also user of several social networking sites. Personally I use Facebook, Twitter and LinkedIn, and just like many other users sometime I get a little frustrated when I want to check all the networks for updates or when I want to post some message everywhere. I have been using few methods how to simplify this process, more or less successfully (for instance adding a Twitter plugin into Facebook, to resend the Twitter messages, or adding plugins to my iGoogle page, to read the statuses from my networks).

However, even after using some gadgets and similar things, it was never optimal. I could read statuses at one page, but when I wanted to post something on my social networking sites, I had to go to other page. I was really never completely satisfied with the way how I accessed these sites and I never found a simple and compact solution (like for instance one iGoogle gadget which would cover all this basic functionality).

I think it's now quite obvious why I chose this topic. It was a good way to learn something more about the social networking services I use and about the Social Web in general. But also I was happy that I will have a chance to try to implement the functionality I wanted to have.

7.1.1 Common API for social networking applications

I was writing this project during my study period at Universidade de Trás-os-Montes e Alto Douro in Portugal and there I received a detailed specification of the project.

It was required that I integrate the social networking applications in a way which would provide some extensibility and reusability - that means in a common interface which would allow communication with several social networking sites on the Internet. This interface was meant to be used in another project. Developer of a web application, who would like to extend the application to communicate with the social networking sites would only have to add one small package in his project and could start communicating with the different sites.

The other project which was going to use my interface was going to be an application to upload and share different content online. This application was meant to be accessible also from mobile devices. The idea was that the user of the application would be able to upload content (photo, video, article) and to distribute this content to social networking sites on the web.

I think this part of the project is the most important one. As far as I know there is no similar API or library available on the Internet, at least not free to download. If there was an API which would allow communication with several social networking applications, it could provide simple and fast way of extending web applications. Therefore during all the development of my thesis, I was focusing on creation of this interface. However of course I also designed and implemented the required portal. The portal serves as an example of usage of the common API.

7.2 Availability of resources

Already from the very beginning of the development I was facing one unpleasant problem. Because this field of web programming is quite young, sometimes it was hard to find proper resources about the topic.

It was happening to me that I couldn't find an article about the topic I was looking for, or to find a solution for a problem which I was having. Fortunately I was always able to find a technical forum for web developers, run by the social network developers, where it was possible to discuss any kinds of problems.

Chapter 8

Design

The following chapter is describing the process of designing the final application. It talks about both - the design of the common API as well as of the design of the final application. There is an explanation of different layers in which the API and the application are divided.

8.1 Project assignment - Information portal with the API layer

As explained in the chapter 7.1.1, the task of the project was to create a common API to communicate with social networking sites. The functionality of the API then had to be demonstrated in an information portal.

The API would create a layer between social networking applications on the Internet and the web portal which would use it's functionality. The idea of the common API layer is show at picture 8.1.

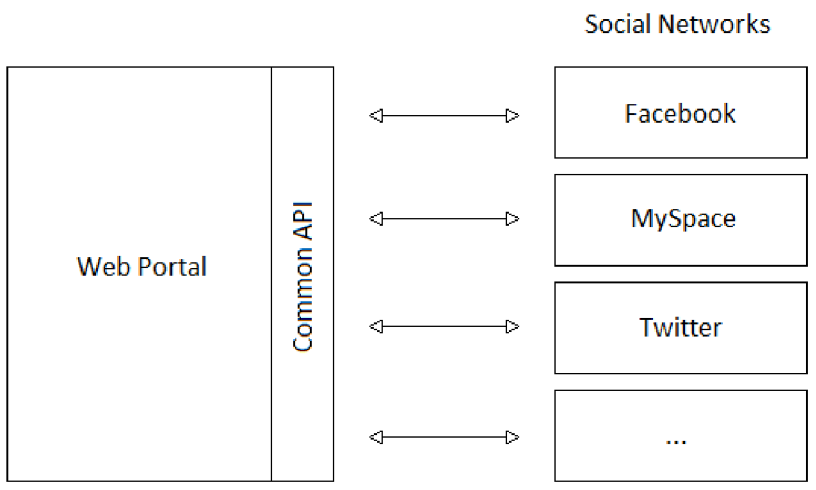

Figure 8.1: Overview of the project - web portal with common API

8.2 Functionality of the application

At the beginning of designing my application I had to specify for myself which functionality should the API and portal implement. The API was going to be used in an application created for sharing content, therefore it was logical that the API should provide some functionality to send simple public messages in the social networking sites.

I also had to create an information portal to demonstrate the API functionality. In this case it was not specified which functions should the API provide. I decided to create a simple portal to view/share status updates from different social networking applications on the web.

During the implementation of the portal it became necessary to implement a function which would allow to delete status messages of the currently authenticated user. I also wanted to allow the users of my portal to see list of all their friends profiles on different social networking sites.

That means the functionality of the API would have to include posting/viewing/deleting statuses and downloading of basic profile information.

8.3 Layers of the application

To make the implementation easy and well-arranged, I had to divide the application in several layers. One of the layers of the application is already known. It's the common API. This API would provide an interface between social networking sites on the Internet and between the rest of my application.

For the demonstration of the usage of the API I needed to allow users to log into my portal so the users could save their access tokens and later connect to the networks faster and more simply. Therefore other layer of the portal had to be a database layers - an interface which would unify and ease the communication with the project database.

Then there had to be a layer to connect these two mentioned layers. This layer would also provide some extended functionality - like a complete user management system and translation of the texts of the portal.

The final layer would be the user-interface layer. This layer would only take care of presenting the data obtained from the layers below it.

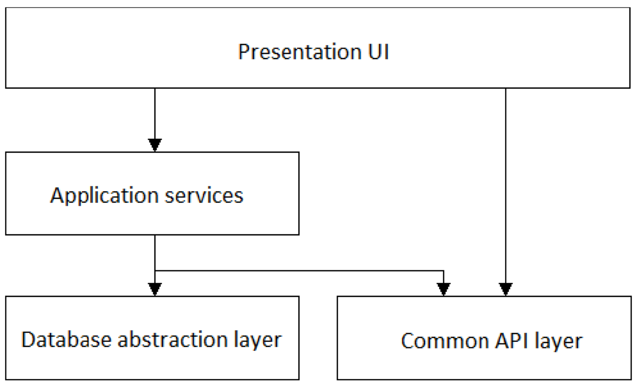

Figure 8.2: Layers of the portal

8.4 Layers of the API

Also the API itself had to be divided in several layers. All of the social networking applications integrated in the portal use OpenID for signing of users and OAuth for authentication of the requests. These two parts would therefore create two separate layers of the API.

However these both layers will need to use some basic services like HTTP requests class. These basic services would then create another lower-level layer of the API.

Above the two classes there should be another one, which would connect the functionality of both and add a specific parameters of the individual social networking sites. This layer would create an access points separately for each provider.

In the end there has to be a layer which would create the desired common API. This layer would provide the desired functionality. It would be the interface which a developer using the common API would access from his application. In other words, this layer would shadow all the layers below it.

The order of all layers of the API is visible at the picture 8.4.

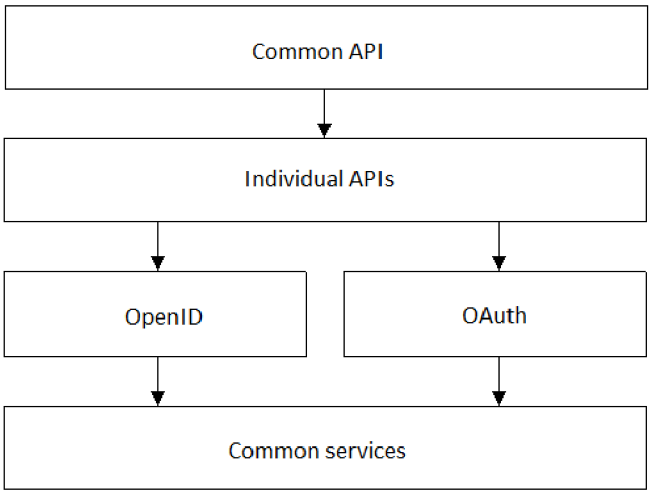

Figure 8.3: Layers of the API

8.5 Design of the database

On the web portal implementing the common API I wanted to allow users to sign up and store information about the connections with their social networking sites so the next time users come back they would not need to set up everything again. For this I needed to create a database.

To store the user data only three tables were necessary. One table is used to store the basic information about the users - their name, password, email and similar things. This table contains a unique ID identifying each user in the whole database.

Except the first table there are two more. One contains OpenID tokens for each user and one OAuth access tokens. Both tables are indexed with columns *userID* and *provider* - this provides a check if each user has only one active token for each provider, which should be the only possible situation.

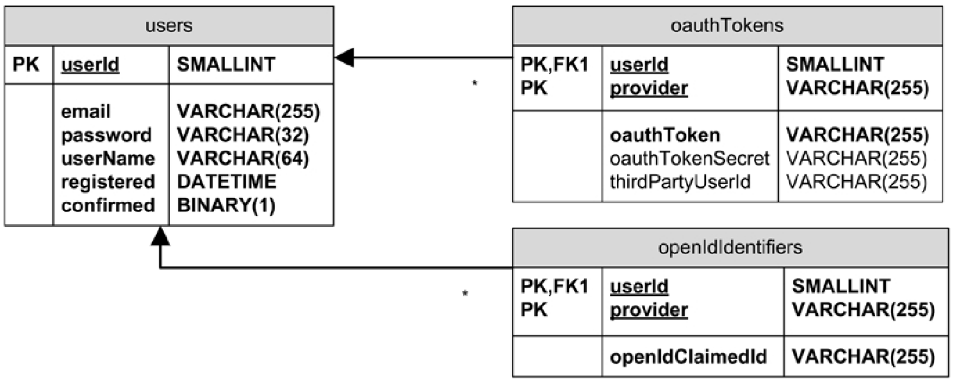

Figure 8.4: Model of the database

8.6 Connecting user accounts with third party login

To improve the users impression from the application I wanted to allow users to sign in the application by using their third party account. However it was also necessary to keep the standard accounts on the web, for the case someone would prefer to sign in with email and password. Therefore I had to design a way how to connect these two kinds of accounts.

It was required that the user can register using both ways - with creating the standard account or with third party account. After the user registers in the application, it should be also possible to create a new connection to any social networking application - that means the user would be able to sign in with two or more different accounts. These two requirements had to be taken care of during the design of the sign in process. The final authentication flow as used in the application can be seen at the image 8.6.

As seen at the picture, if the user requires sign in with a third party account, first of all the application will try to acquire a token for the user (OpenID or OAuth token). If this token already exists in the database, then it means that the user already did the registration and can be signed in.

If the token is not in the database, it can be because the old token has expired. Therefore the application checks not just tokens in the database, but also a value of user ID which users gets from the third party. If not even the ID is found it means that the user didn't create this connection yet or that he's not registered at the portal.

To check if the user is already registered at the application or not, the portal reads his email address. This email is again searched for in the database. If it exists, a new token is saved to the database for the current user and the user is signed in.

However if the email was not found in the DB, the user is registered to the portal and his identity is validated by sending a confirmation email to his address.

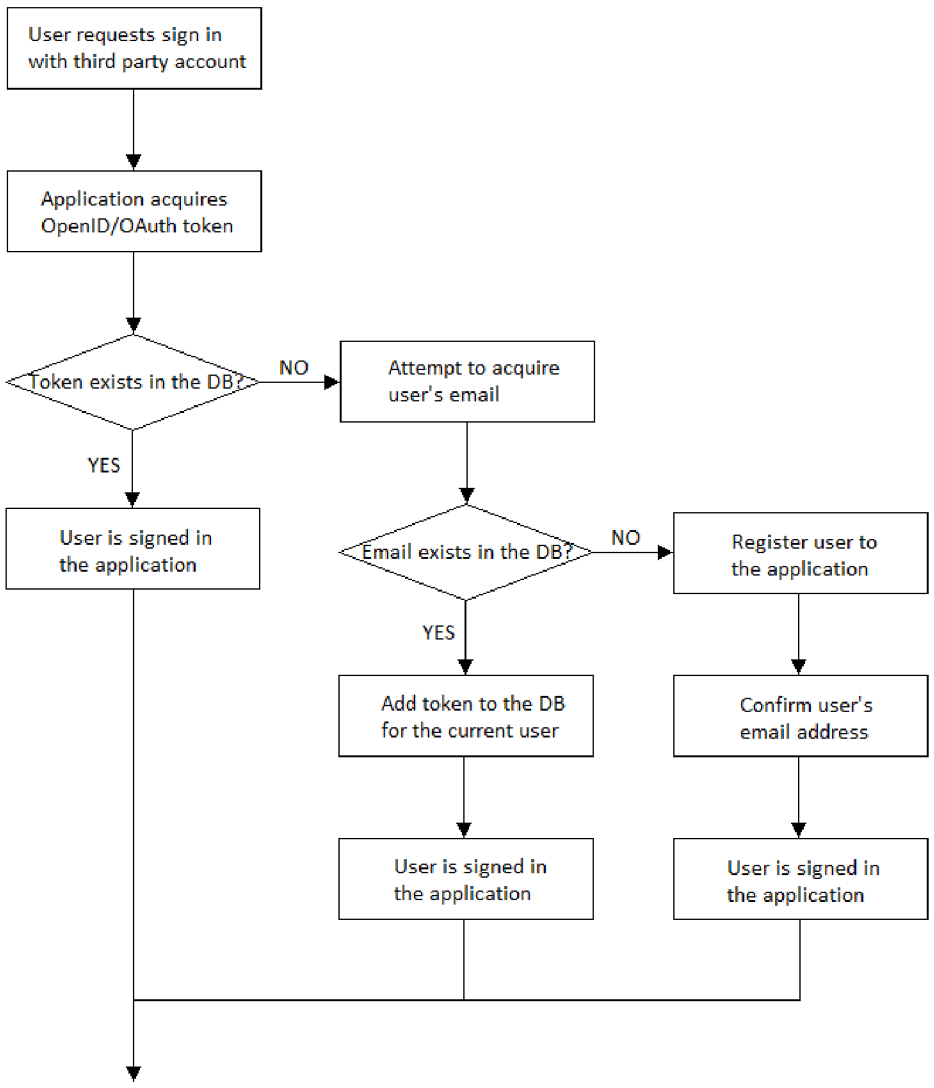
User requests sign in with third party account
Application acquires OpenID/OAuth token
Token exists in the DB?
NO
Attempt to acquire user's email
YES
User is signed in the application
Email exists in the DB?
NO
Register user to the application
YES
Add token to the DB for the current user
Confirm user's email address
User is signed in the application
User is signed in the application

Figure 8.5: User authentification flow

Chapter 9

Implementation

This chapter describes the most important part of this project - the implementation of the designed API and portal. It describes implementation of all layers of the portal.

9.1 Preparation

For this project I chose PHP (together with MySQL) as the implementation language, basically because this language is really simple and light and that is what I wanted my API to be. There is also incalculable number of resources for this language available on the Internet. At last, is it also much easier to find cheap hosting with this language than with other languages.

As a data format used for communication with social networking sites I chose JSON, again for it's simplicity. All the project was developed using NetBeans IDE 6.8 from SunMicrosystems.

9.2 Application layers

As already explained in section 8.3, the application consists of three layers and the fourth topmost layer which only displays data generated by the lower layers. The three layers are `DbAbstraction`, `CommonApi` and `AppServices`.

9.2.1 Database layer

The `DbAbstraction` is a layer which unifies the connection between the database of the application and the application itself. It consists of three classes - `DbConnection`, `DbDatabase` and `DbTable`.

The class `DbConnection` holds the open connection to the database. The connection can be created with method `dbConnect`, which throws an exception `DbException` in case that the connection fails. To close the connection it's possible to call method `dbDisconnect`.

However it is possible to do so, this class in never called directly in the rest of the application. Instead, the class `dbDatabase` is always used. This class is a singleton class (only one instance of the class can exists at one time) and is available always throughout all the rest of the application.

The `dbDatabase` class is saved inside a session using `serialize` function at the end of every page and is loaded again at the beginning of every page. That creates a situation when there is exactly one open connection to the database and this connection is valid during all life of the application.

Apart from calling the `dbConnector` class, the `dbDatabase` class also contains the database model stored in an array and set of methods used to create and drop database tables. This methods allowed me to create a simple installation script for the portal, which will be mentioned in appendix A.

To operate with the data stored inside the database there is third class, called `dbTable`. This class contains a set of methods to access directly the database data (`select`, `insert`, `delete`, `update` and `query` to execute any MySql query).

There are also classes `dbException` and `dbLogging` which are used for basic logging of the arose errors and file `dbConstants.php` where all database constants can be changed.

9.2.2 Common API

As explained in section 8.4, the API is divided in several layers or packages, just like the rest of the application.

The lowest layer contains general classes which are then used in all the rest of the API. Most important of them is a class `HttpRequest`, which is used to create custom HTTP requests - allowed HTTP methods are `GET`, `POST`, `PUT` and `DELETE`. The class also contains a method `forwardTo` which does a URL forwarding. The response of the request is stored inside an array with three fields - `code` (containing a status code of the response), `data` (with the data of the response) and `info` (with information about the request, like the request headers and time of the response).

In the lowest layer there are also classes `XrdsParser` to parse XRDS documents returned by OpenID discovery and `ApiException` and `ApiLogging` to manage basic logging of the errors inside the `CommonApi` layer. There is also a very important file `Constants.php`, which defines all global constants used in this layer. The contents of this file will be described in the appendix A.

After the lowest layer there are two layers which create most of the functionality of the `CommonApi` layer.

Open ID

The class (or package) `OpenID` performs login with any OpenID provider. It covers the functionality of the client side of the OpenID authentication flow as shown at the picture 5.1.2. The class is created according to OpenID standard as described in [17].

The class has to be used in specified order. To perform login with OpenID, firstly the method `performDiscovery` has to be called. This method asks the server for an XRDS document telling the client which services the server supports and giving him the address of the OpenID endpoint which can take care of the user login. The method `performDiscovery` parses this document and saves the OpenID endpoint inside the class for later use.

After the discovery, the method `requireLogin` can be called. This method redirects the user to the login page of the OpenID provider. The user is asked to sign in and to allow the application to connect with his OpenID provider. After the sign in process, the user is returned back to the application to a page which can be specified in one of the parameters of the login request. After the redirection back to the application (if the user didn't decline the application) the OpenID token can be read from HTTP GET response from the server.

However before using this token, the token must be validated. For that there is a method `validateResponse` which takes whole GET response from the server as it's only parameter. To use this method, the OpenID endpoint must be set again (by calling the `performDiscovery` method). If the token is validated, it can be finally used to identify the user of the portal.

OAuth

For signing the calls in the social networking sites APIs there is a class `OAuth`. This class can be used for both - acquiring the OAuth access tokens and signing the requests. The class covers functionality of the client side of the OAuth authentication flow as shown at picture 5.2.2. This class is created according to OAuth v1.0 standard as described in [12].

For acquiring the access token the class contains three methods which have to be called in specified order. First of all it's necessary to acquire a request token, which will be later authorized and exchanged for the access token. To acquire the request token, the method `getRequestToken` has to be called. This method returns an array with OAuth request token and secret. These two tokens then have to be saved inside the class for further usage using the method `setToken`.

After the request token is set, the method `authorizeToken` can be called. This method will redirect user to the OAuth provider page, where he will be asked to sign in and allow the application to access his data. After this process, the user is returned back to the application to a specified page, with data from the server saved inside an HTTP GET response. This response contains the authorised request token and a string called verifier (if the sign in process of the user was successful).

If the authorisation was successful and the authorised request token was received, the method `getAccessToken` can be called (however both tokens from the first step have to be set again). This last method will exchange the authorised request token for an access token and secret, which can be finally used to access the user's data on the server.

OAuth signature

As said in the previous section, the OAuth access token and secret received from server can be used to access the user's data on the server. However to verify that the access token was really used by the calling application and not by anyone else, every request has to be signed according to OAuth documentation.

For signing the requests there is a class `OAuthSignature` contained in the OAuth package. This class provides methods to generate the OAuth signature parameters (OAuth nonce and timestamp) and to concatenate all OAuth parameters into the signature base string (method `generateSignatureBaseString` inside each of the `OAuthSignature` classes). This class supports four methods of signing - HMAC SHA1 (class `OAuthSignature_HMAC_SHA1`), MD5 (class `OAuthSignature_MD5`) and PLAINTEXT (class `OAuthSignature_PLAINTEXT`). The method to create the signature is `generateSignature`.

After all the OAuth parameters including the OAuth signature are created, these parameters can be appended to the OAuth request. To perform an OAuth request, the class `OAuth` contains a method `fetchResource`.

Common API

The topmost layer of the `CommonApi` is a class `CommonApi`. This API covers all the functionality of the lower layers. It provides a set of methods which can be used to access the data of the social networking applications in a simple way. Available methods of this class are shown below.

```
class CommonApi {
        public function requireSignIn($provider, $scope = null) { }
        public function finishSignIn($provider, $get) { }
        public function requireAccess($provider, $scope = null) { }
        public function getAccess($provider, $get) { }
        public function makeApiCall($auth, $url = null, $path = null,
                $headers = array(), $parameters = array(), $post = null,
                $method = HttpRequest::METH_GET) { }
        public function postStatus($auth, $data) { }
        public function getStatuses($tokens) { }
        public function deleteStatus($auth, $data) { }
        public function getProfileInfo($auth, $userId = null,
                $provider = null) { }
}
```

Figure 9.1: Common API methods

The methods `requireSignIn` and `finishSignIn` are used to manage the sign in of the user. The method `requireSignIn` only requires the name of the provider as a parameter. It doesn't matter if the provider uses OpenID or OAuth to authenticate the users, the method will select the proper technology and redirect the user to the correct page. After the user is signed in and returned back to the application, the method `finishSignIn` can be used to process the parameters of the response. This method returns an array with OpenID or OAuth token (or both) according to which provider was used for sign in.

The methods `requireAccess` and `getAccess` are then used to acquire an OAuth access token for the specified scope. If the scope is not specified, a default scope is used. This pair of functions works same as the first one. The `getAccess` method can be called after the user is redirected back to the application to process the response from the server.

The other methods of the API are used to work with data stored at the network application server. Method `makeApiCall` can create any custom API request. For testing these custom calls in the social networking sites APIs I created a console which can be found at `http://thesis.memoo.cz/ApiCall.php`. This console was very useful during the development of `CommonApi`.

Last four methods of the API then provide a specific functionality. Method `postStatus` can be used to post a custom status (which may contain a link) to the selected social network. To retrieve the status updates from the networks, there is method `getStatuses`. To delete a status of the currently authenticated user, I created a method `deleteStatus`. Finally the last method `getProfileInfo` retrieves a basic profile information of a selected user.

9.2.3 Application Services

Apart of the `dbAbstraction` layer and the `CommonApi` layer, it was necessary to create a layer with specific functionality used for the final presentation of the data. Most importantly this application layer contains a class `User`, which handles the user management. There is also a `Dictionary` class which, just like the `dbDatabase` is a singleton class, and which contains a dictionary with english-czech translations of all texts and messages in the portal.

Another class from the layer is class `PageLayout` which was created to provide a simple templating system to make the final presentation of the data easier. Finally, just like in the other layers, there are classes `AppException` and `AppLogging` to create a basic logging of error messages.

9.2.4 User interface

The topmost layer of the portal is a layer which creates a graphical user interface presented to the user. This layer uses all the lower layers like show on the picture 8.3.

The design of the application was created using Adobe Photoshop. Later it was put together using HTML and CSS.

9.3 Other implementation details

For putting the application online, I used my own domain *memoo.cz*. The final version of the application can be found at page `http://thesis.memoo.cz/`. I put online also the generated software documentation, which can be found at `http://doc.memoo.cz/`.

Chapter 10

Testing

Following chapter is describing the process of testing the application. The application was tested in several ways, to prove the functionality of the portal, but most importantly of the API.

The portal integrating the social networking sites was tested using all most common web browsers and on operating systems Windows 7 and Kubuntu Linux. Following table is showing the which browser versions in which operating systems were used.

	Windows 7	Kubuntu Linux
Internet Explorer	8.0	-
Google Chrome	5.0	5.0
Mozilla Firefox	3.6	3.6
Opera	10.60	10.60
Safari	5.0	-

Table 10.1: Browsers and operating systems used for testing

10.1 Functionality of the application

After the application development finished, I thoroughly tested all functionality of the application. Specifically, I tested all following parts of the application:

- **Registration of users**
  I tested both options of registration - registration using third party login and registration by creating a new account. This test uncovered a problem with sign up using MySpace. One of the OAuth parameters of the request was not set to a proper value and so MySpace was not returning an OAuth token. This mistake probably happened while I was moving the application to a different domain. Problem was already removed and the registration is now working perfectly.

- **User sign in**
  Again I tested all options available - signing in with existing account (also with not existing account), sending a forgotten password and signing in with third party account. This test didn't uncover any problems.

- **Settings**
  I tried to change the password and to add and delete connections to all the social networking sites available. This test was without problems.

- **User's profiles**
  Displaying of the list of user's profiles was working for both - the currently authenticated user and for his friends.

- **Posting statuses**
  I discovered a problem with posting a new message to the networks. If I wanted to post only a link, it was never posted to Google Buzz. This was caused by incorrect setting of the parameters of the request, the problem was already corrected. In the same situation (posting only a link) also the string sent to MySpace and Twitter was not concatenated correctly. Again this was caused because of incorrect work with the empty message parameter. The problem was already removed. Apart of these things there were no other problems with posting statuses.

- **Reading statuses**
  Reading of the statuses on the dashboard was working perfectly. The reading of the statuses was a little slow, this was happening especially if the page was loading statuses from MySpace. However the functionality was without any implementation problems.

- **Deleting statuses**
  I tried to delete a status from all four supported networks and this test proved that deleting statuses works fine.

10.2 Google Analytics

When I finished with the development of the application I created an account at Google Analytics and I registered there the application. Google Analytics is a useful service provided by Google, which allows monitoring of the usage of a website and which is then generating useful statistics.

For instance it is possible to view which web browsers the users of the application use most, which pages of the application were visited (also how much time the users stayed at the page) or speed of the user's connection to the Internet.

I intend to keep this monitoring running during the whole life of the application, because the generated statistics can be very useful in improving the user's impression from the application. However currently there are not enough users of my portal to create and present statistics with realistic results.

10.3 Usage of the API in a different project

The common API was already used in another project. This project is called Adaptive Web Portal and is using my API to post status messages (with information about newly uploaded data) to the social networking sites. Link for the Adaptive web portal: `http://thesis.adampacha.com/`.

The API was used in this project for some time already and this usage didn't show any problems of the implementation. The posting of the status messages works as expected.

Chapter 11

Conclusion

The task of the project was integration of social networking applications into an information portal. The project had to demonstrate the ways how the social networking sites can be connected, to test the technical possibilities of the integration but also to learn something about social networking applications and social web in general.

I managed to create the desired portal and to integrate social networking applications in it. I was also able to integrate these applications into a common API which can be now used in any other web project (and it already was used).

The portal which I created now offers a simple way how to read and post statuses from one place. I managed to connect the social networking sites on the Internet with most users. However, my common API offers a simple way of adding any other social network which uses OAuth for authentication (generally nowadays that means any social networking applications on the Internet). Also the functionality of the API can be very easily extended.

At last I also learnt new things about the Internet socialisation. From a view of a developer, these things are very interesting. I summarized my learnings in first theoretical half of this thesis. I think this text now offers a nice overview of aspects of online social networking.

The application could be now extended in several ways. Firstly there are the two options which I already mentioned - other social networking sites could be added in the API and also the functionality could be extended. The API could be also used in different web projects. For instance it could be used to create a simple gadget for iGoogle page which would, just like my portal, provide a way to read and post statuses.

To summarise the fulfilling of my task: there were five points I had to complete. The first two of them (acquaint familiar with social networking applications available and study technical possibilities of integration of these applications into a web portal) I completed already during winter semester. During the summer semester I was studying as an exchange student at a university in Portugal, where I had to finish my project. I managed to finish it and with a guidance of teachers from this university I also managed to extend the project and to connect it with another project by providing the common API which I created.

Bibliography

[1] Danah M. Boyd and Nicole B. Ellison. Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1), 2007 [cit. 2010-07-25]. `http://jcmc.indiana.edu/vol13/issue1/boyd.ellison.html`.

[2] Christopher Nickson. The History of Social Networking [online]. `http://www.digitaltrends.com/features/the-history-of-social-networking/`, 2009-01-21 [cit. 2010-07-25].

[3] David Orenstein. QuickStudy: Application Programming Interface (API) [online]. `http://www.computerworld.com/s/article/43487/Application_Programming_Interface`, 2000-02-10 [cit. 2010-07-13].

[4] Facebook. Graph API [online]. `http://developers.facebook.com/docs/api`, 2010 [cit. 2010-07-14].

[5] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.

[6] OpenSocial Foundation. Articles & Tutorials [online]. `http://wiki.opensocial.org/index.php?title=Articles_&_Tutorials`, 2010-06-14 [cit. 2010-07-14].

[7] Gigya, Inc. Multiple identities [online]. `http://info.gigya.com/Identity.html`, 2010 [cit. 2010-07-13].

[8] Graham Vickery and Sacha Wunsch-Vincent. *Participative Web And User-Created Content: Web 2.0 Wikis and Social Networking*. OECD Publishing, 2007.

[9] Jonathan Strickland. Is there a Web 1.0? [online]. `http://computer.howstuffworks.com/web-10.htm`, 2008-02-28 [cit. 2010-07-13].

[10] Mike Gotta. Social Networks: Making Sense Out Of Terminology [online]. `http://mikeg.typepad.com/perceptions/2008/04/social-networks.html`, 2008-04-09 [cit. 2010-07-25].

[11] MySpace. RESTful API [online]. `http://developerwiki.myspace.com/index.php?title=RESTful_API`, last modified 2010-01-17 [cit. 2010-07-14].

[12] OAuth Core Workgroup. OAuth Core 1.0 [online]. `http://oauth.net/core/1.0/`, 2007-12-04 [cit. 2010-07-14].

[13] S. E. Smith. What was the Dot-com Bubble? [online]. `http://www.wisegeek.com/what-was-the-dot-com-bubble.htm`, last modified 2010-05-19 [cit. 2010-07-25].

[14] Shaun Connoly. 7 Key Attributes of Social Web Applications [online]. `http://connollyshaun.blogspot.com/2008/05/7-key-attributes-of-social-web.html`, 2008-05-24 [cit. 2010-07-13].

[15] The OpenID Foundation. OpenID Simple Registration Extension 1.0 [online]. `http://openid.net/specs/openid-simple-registration-extension-1_0.html`, 2006-06-30 [cit. 2010-07-14].

[16] The OpenID Foundation. OpenID Attribute Exchange 1.0 - Final [online]. `http://openid.net/specs/openid-attribute-exchange-1_0.html`, 2007-12-05 [cit. 2010-07-14].

[17] The OpenID Foundation. OpenID Authentication 2.0 - Final [online]. `http://openid.net/specs/openid-authentication-2_0.html`, 2007-12-05 [cit. 2010-07-14].

[18] The OpenID Foundation. OpenID Provider Authentication Policy Extension 1.0 [online]. `http://openid.net/specs/openid-provider-authentication-policy-extension-1_0.html`, 2008-12-30 [cit. 2010-07-14].

[19] Tim O'Reilly. What Is Web 2.0 [online]. `http://oreilly.com/web2/archive/what-is-web-20.html`, 2005-09-30 [cit. 2010-07-13].

Appendix A

Manual

To start using the common API and the portal it's necessary to do few simple steps first. Most importantly, the connection with the social networking sites won't work unless the application which wants to communicate with them is registered at the social networking sites.

This is necessary because after the registration the registered application receives a consumer key and secret which are then used to identify the application during the communication and also to sign the requests. To register the application at the social networking sites it's necessary to visit the following pages:

- **Facebook**
  `http://www.facebook.com/developers/`
- **Google**
  `https://www.google.com/accounts/ManageDomains`
- **MySpace**
  `http://developer.myspace.com/Apps.mvc`
- **Twitter**
  `http://dev.twitter.com/apps`

After the application is registered, it's necessary to put the received consumer key and secret in the source code of the API. For this, there is one file with constants in the API - *Common/Constants.php*. This file should be read carefully and the constants should be altered as necessary. After this, the API and the portal are ready to communicate with the social networking sites.

A.1 Portal installation

To use the portal, the database must be installed. For this task I created a simple installation file *Install.php* which is included to the application but after the installation of the database it should be removed from the portal so it's not misused. This file also offers a way how to easily backup the database data - the data can be saved in both XML and SQL format.

The database constants like host and user name can be changed inside a file with database constants - *DbAbstraction/DbConstants.php*. The constants of the `AppServices` layer are stored in file *AppServices/AppConstants.php*.

A.2 API usage

To start using the API it's just a matter of copying the folder with the source codes in the project. The API can be accessed through one simple interface - a `CommonApi` class. Documentation for this class can be found online at software documentation for the project at following page: `http://doc.memoo.cz/class_common_api.html`. The software documentation is accesible from `http://doc.memoo.cz/`.

Appendix B

CD content

This thesis comes with a DVD containing all data of the project. The final version of this text can be found in a root directory of this DVD in format PDF. There are also two subdirectories in this directory:

`doc/` directory

- `doc/html`
  Software documentation in HTML format.
- `doc/pdf`
  Software documentation in PDF format.
- `doc/src`
  Source codes of the software documentation, written in LaTEX language.
- `doc/thesis`
  Source codes of this thesis, written in LaTEX language.

`src/` directory

This directory contains all source codes of the final web portal.