Mathias Mayrhofer, BSc.

Clickdata acquisition for evaluating user navigation models

Master's Thesis

to achieve the university degree of
Diplom-Ingenieur
Master's degree programme: Telematics

submitted to
Graz University of Technology

Supervisor and Assessor:
Assoc.Prof. Dipl.-Ing. Dr.techn. Denis Helic
Institute for Knowledge Technologies

Graz, March 2015

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources or resources and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Date:

Signature:

Zusammenfassung

Diese Arbeit beschäftigt sich mit dem Verhalten von Benutzern beim Suchen von Informationen auf Internetseiten. Hierzu wurde eine Web-Anwendung programmiert, die auf beliebigen Internetseiten das Klickverhalten der Benutzer aufzeichnen kann. Eingebettet wurde das Programm in eine Art Spiel, wo mehrere Benutzer gegeneinander antreten und ausgehend von einem Startdokument nur mittels Klicken auf die dort vorhandenen Links ein vorgegebenes Zieldokument erreichen sollen. Ziel ist es, das gewünschte Dokument mit möglichst wenigen Klicks zu erreichen. Beim Beispiel von Wikipedia ist die Benutzung der Kategorien interessant. Die teilnehmenden Personen können einander zusehen um eine Art sportlichen Anreiz zu bieten, wer das Dokument am schnellsten erreicht hat. Die Auswertung der Klickpfade gibt dann Aufschluss über die Denkweise der Teilnehmer.

Es wird einerseits die Realisierung des Programms beschrieben, andererseits die verwendeten Technologien für die verschiedenen Verarbeitungsschritte erklärt und mit anderen verglichen. Des Weiteren wird die Navigation des Benutzers selbst untersucht, da es für Hersteller von Webseiten und Suchmaschinen ein Anliegen ist, dass der Benutzer die Information findet nach der er sucht. Die Navigation kann auch in speziellen Netzwerken stattfinden, wie Informationsnetzwerken, sozialen Netzwerken und mit Beschlagwortung unterstützt werden. Weiters werden verschiedene Arten von Netzwerken mit denen wir zu tun haben diskutiert und analysiert. Abschließend werden auch Sicherheitsaspekte der verwendeten Komponenten beschrieben und auch auf generelle Sicherheitsprobleme im Alltag beim Durchstöbern des Internets aufmerksam gemacht, weil Firmen sich ähnlicher wenn nicht sogar der selben Methoden bedienen wie sie in dieser Arbeit verwendet und diskutiert werden.

Abstract

This work deals with user behavior when searching for information on Internet pages. Therefore I programmed a web application, which can record the clicks done by a user on a specific (remote) web page. The program can be pulled over an existing web page to perform the recording task and the reporting of the clicked links. The program itself serves a game, where a number of users are encouraged to navigate from a start page to a target page only by clicking on the available hyper links. A useful example would be Wikipedia, where each page has many cross references and categories below. The participants can see the other players performing, so there is a competition who will find the target document first. The evaluation of the click paths contains information about the mindset of the participants.

The thesis explains how the program works, which technologies were used and how the modules work together. Moreover the user's navigation itself is described, as it is an important role for developers of web sites to know how the user can find the desired information he/she searches for. The navigation itself can take place in information networks as well as social networks, so the characteristics of various networks are investigated and analyzed. Furthermore security aspects of the used components are discussed as well as general security aspects when browsing through web sites in everyday life as companies might use similar or even the same technologies to trace their users for their own benefit.

Acknowledgments

To my mother and my deceased father, who always supported me throughout my life. TTD

Contents

1 Introduction

1.1 Introduction

We are surrounded by a broad variety of networks. Many different types of information are accessible through these networks, mostly (but not necessarily) for free. Providers of such information networks, as well as the users, are interested in making it available to as many people as possible and that the people can easily find the information they are looking for. Network structures can be very different to each other and are highly dependent on the type of the network. To navigate through a network can be a tough task and one probably does not know the complete structure of a network and on which criteria the items are connected. Who can be asked for the way? Which topics might be related but are still not connected to each other? How can one reach a destination point without knowing the shortest path but only by intuition?

In this work, I will give an overview of the topology of the networks we are using every day. We are surrounded by large networks and the difference between the topologies might not be as obvious as expected. Within the largest network - the Internet - information can be structured again according to information or knowledge like Wikipedia or other directories. On the other hand we need to know how our brain handles such information. How does the human brain find answers to arising questions. How can information be categorized? And how should it be categorized (or tagged) for humans to be optimally findable?

The Internet gives us a huge instrument for researching techniques in this field like the research to find out how the human brain works in certain fields and larger networks as well as improvement of guidance for finding desired information. We can track search processes and evaluate success in finding information to adapt results and hints for future questions. This will be the practical part of the work, which is to record the user's behavior within such an information network. As a starting point for an empirical study, I wrote a program that can be "pulled over" an existing web site to record the user's behavior. I will describe the techniques which I used for the software and which methods exist to track the user. The findings will help to improve the view on this topic in the future.

Using this intention as a starting point, many security questions arise because the information gathered could also be used to track users. Companies that make profit from profiling users take advantage of similar techniques and probably can extract much more information from the user by just watching him browsing through their web sites. The user is mostly not aware

of the fact that he leaves so many footprints. On the one hand, the techniques can be made very subtle, on the other hand there are many technologies involved and each of them can be used to exploit information. There are hardly any laws against using such methods, therefore companies cannot be held responsible for maximizing the amount of data they can extract to their advantage. Additionally, when companies can provide services, users tend to voluntarily give away their data without thinking of the consequences only because they think they cannot waive the benefit.

1.2 Motivation

Categorizing information so that it can be easily found for a human being may be a hard task if dealing with huge amount of data. We need reasonable methods to guide the user through different kinds of networks. Networks itself can have different topologies and computers help us to navigate through them. When searching for information it depends on which site one is searching. In Wikipedia for example, the information is highly categorized (hierarchical) and linked together with similar topics and articles. But finding these similarities is not a trivial task. We understand pretty good how the information can be stored and presented to the user, but do we understand what information is relevant for the user right now? Given only little information like a few keywords, do we know what a person is trying to find?

With this work, I want to present a software tool, that can be "pulled over" an existing web site, to view the links the users click. It will be asking the user to take part in a game to navigate from a random starting point within the web site to a random destination page. The user's actions, respectively the links or pictures the users click will be recorded by this software and stored into a database for statistical analysis. To give the users incentives, the game will be a competition with other participants, so the users feel encouraged to find short paths to the destination page. Users can be invited to perform the game voluntarily, they can be tested as part of studies or in the course of university lecture, or they can be asked on the web page itself to take part in the game. In the end, the data can be processed and distances between topics can be analyzed. Conclusions from that may be the adaption of topic names or better placing the link or adjusting results from search requests. There may be many even more benefits from correlating the collected data which would also give psychological insights of human thinking.

1.3 Thesis Outline

Finding connections within information networks is the first aspect of this work. The second is an excursion to the different types of networks we probably want to find information within. This includes not only physical networks but information networks in the broadest sense. Every piece of information can be linked multiple times. A single item (e.g. an Internet site or a content page) can appear on multiple aspects.

After that, I want to give a rough overview of the technologies that were used for the software project. Advantages and disadvantages of certain techniques are being discussed as well as I will shed a light on security aspects in nowadays daily Internet activities and methods that can be exploited and used even without the knowledge or consent of the users.

Finally I will describe the software project itself in detail. I will show the interfaces the software comes with and some screen shots from the operation. I will describe how the software can be deployed on the server and how it must be adjusted to function on a certain web page.

2 Network Navigation

2.1 General Definition of Networks

The Internet interconnects thousands of computers worldwide. It has already begun that not only computers are connected to the Internet, but also gadgets like mobile phones are using Internet connections too. The next step is not far away, where even smaller devices like wearable objects, wrist watches, power meters, light bulbs, cameras and even small sensors can directly access and be accessed within the web. So we are talking of a huge network of billion devices that are already connected or at least will be connected within the next years. We as humans tend to think of the Internet as the World Wide Web offering free pages like Wikipedia[1] for an encyclopedia, LastFM[2] for music and many more. There are also hidden pages which can not be found or accessed directly. These pages might be intentionally hidden pages or pages that only trusted authorities have access to. These pages are called "Deep Web".

The Deep Web[3] (which is also called the Deepnet, Invisible Web, or Hidden Web) is World Wide Web content that is not part of the Surface Web, which is indexed by standard search engines. The Deep Web can only be accessed by entering search queries[27] on HTML forms for example on their entry web sites. This circumstance aggravates the accessibility for other search engines and therefore much effort is made to automatically fill in search queries to determine the underlying documents. So these pages probably contain valuable information which are somehow hidden from the public and may not be found on conventional ways. Additionally there might be computers that contain information which should not be found occasionally and which are used for criminal and illegal activities. Only people who know how to access the documents within this kind of network are able to find them. Another part of the Internet are computers that can not be reached any time or went completely offline so that their content will not be accessible anymore.

Having this huge network of connected devices it is not easy for someone to find the information he is searching for. Search engine driven searches are a common way to find web sites within the Surface Web. Within a certain web site, we can also provide full text searches. Another idea is to traverse the network hierarchies, so called categories. An example for that is Wikipedia. Every article is an element of one or more categories. For example the mathemati-

[1] http://www.wikipedia.org/
[2] http://www.last.fm/
[3] http://en.wikipedia.org/wiki/Deep_Web

cian "Leonhard Euler" is featured in the following categories:

- Mathematics of infinitesimals
- Leonhard Euler
- 1707 births
- 1783 deaths
- 18th-century Latin-language writers
- 18th-century Russian mathematicians
- Swiss emigrants to the Russian Empire
- 18th-century Swiss mathematicians
- Swiss mathematicians
- Blind people from Switzerland
- Mental calculators
- Fluid dynamicists
- Latin squares
- Mathematical analysts
- Ballistics experts
- Members of the Prussian Academy of Sciences
- Full Members of the St Petersburg Academy of Sciences
- Honorary Members of the St Petersburg Academy of Sciences
- Members of the Royal Swedish Academy of Sciences
- Number theorists
- People celebrated in the Lutheran liturgical calendar
- Russian music theorists
- Russian people of Swiss descent
- Russian physicists
- Saint Petersburg State University faculty
- Swiss music theorists
- Swiss physicists
- Optical physicists
- Swiss expatriates in Russia
- Swiss Protestants
- University of Basel alumni
- Deaths from cerebral hemorrhage
- People with eidetic memory

It is quite interesting, how different the perspectives of these categories are. Some are based on the person and when he lived, some are based on the work he did, some are based on his education and his field of research. So in case of Wikipedia, we are dealing with a highly

interconnected graph, where the nodes are the articles and the categories and the edges are the connection between them represented by hyperlinks and therefore relations between them. This is also called taxonomy.

Taxonomy[4] is a uniform model in information science of defining groups or classification schemes. The resulting model is used to provide a conceptual framework for discussion and analysis. Most taxonomies may have a hierarchical structure, but this is not mandatory. Taxonomy is a major concept in natural sciences that help facilitating the handling of individual cases and provide summary statements that may lead to a declaration of contexts or categories. They force to clarify the differences between the categories and thus lead to a better understanding of the study area. Taxonomy uses taxonomic units, namely taxa. Originally taxonomy referred only to the classifying of biological organisms or a particular classification of organisms. In a wider sense, it can be used to classify knowledge, concepts and categories. Taxonomy is different from meronomy[5] or partonomy which deal with the classification of part-whole relationships which corresponds to object composition in object-oriented programming. This means, every constituent part is member of something else.

Navigating within such taxonomies requires special knowledge of the subject and the categories. Usually the human brain can deal very easily with these concepts and therefore can exploit such a network to find the desired information. So there are many ways one can navigate from a certain article to another depending on the type of category one is using. Selecting an efficient category that brings one close to the destination is crucial although rarely obvious. Knowing the entire graph would be required which is impossible for the human brain. So we approach the destination step by step with the limited knowledge we have. This operation is very subjective because it depends on the knowledge of a person and what the persons associates with the subject. So persons will choose different ways through the network, although some ways will turn out to be more popular (frequent, and therefore more convenient) than others.

Anthropologists found out, that human cultures also use taxonomies to allow people to identify objects. Cultures have their own naming system of plants and animals which were important for survival. Wikipedia categories illustrate a taxonomy and a full taxonomy of Wikipedia categories can be extracted for research by automatic means. An interesting work from Ponzetto and Navigli[33] tries to restructure the knowledge of intelligent systems like Wikipedia. They took a manually-constructed taxonomy of a computational lexicon named WordNet[6] and used it to improve and restructure the Wikipedia category taxonomy. Other types of relationships that can be found in network structures can also be described with taxonomies. It is also possible, that a node has multiple parent nodes, which means, that objects can be assigned to more than one category. This is important when searching for information, because a single category with the object as member would not be found that

[4] http://en.wikipedia.org/wiki/Taxonomy_(general)
[5] http://en.wikipedia.org/wiki/Meronomy
[6] http://wordnet.princeton.edu/

easily. A taxonomy can also be just a group of objects or a list. The definition of taxonomies is similar to ontologies.

"A network is a system that can be modeled with graphs. Graphs are mathematical structures consisting of vertices and edges connecting the vertices. When we observe large graphs that exist in nature, societies or systems we refer to them as networks" [20]. Examples of such networks are:

- Social networks: Nodes are people and links are acquaintances, friendships, family relationships
- Business networks: Nodes are companies and links are incorporations or business relations
- Biological networks: Metabolism, nodes are substances and links are metabolic reactions
- Geographical: Road networks, electricity grid, water distribution net
- Epidemiology: Spread of viruses
- Communication networks: Internet, nodes are computers and links are cables connecting computers
- Information networks: World Wide Web, nodes are web pages and links are hyperlinks connecting the pages

As an example for a social network graph, see Figure 2.1. It shows 390 nodes, where each node represents a person within the network. The links show the connection among themselves. All nodes are directly connected to the center node that is a little bigger than the others. For better visualization this connection is not drawn in the figure. The graph is partitioned into groups of people that know each other. It was created by a software named TouchGraph[7] which automatically reads the friend list of a Facebook[8] account and also gathers the friend lists from these accounts. The figure was created in summer of 2014 from my Facebook network. It shows only my personal Facebook friends at that time and denominates each person with their initials. The program also gathers the friend list of each friend to construct the partitions of the groups that have similar friends. What we can say about the topology of the graph is, that there are many interconnected friend-groups. These would represent the bigger circles of friends. But there can also be seen many smaller groups, that only are connected amongst them. There are also a large number of people that are not connected to others at all. This supports the theory that many nodes have low degree but a few have very high degree which will be described below.

Figure 2.2 is the network graph of the IPv6 Internet nodes created by Lumeta[9] in the year 2009. The IPv6 network in that days was just arising. Each end node can represent a handful of computers on a small network, or perhaps a large company with hundreds of thousands of hosts. We see, that the pattern of connections is not a regular one, neither is it completely

[7] http://www.touchgraph.com/facebook
[8] https://www.facebook.com/
[9] http://www.lumeta.com/

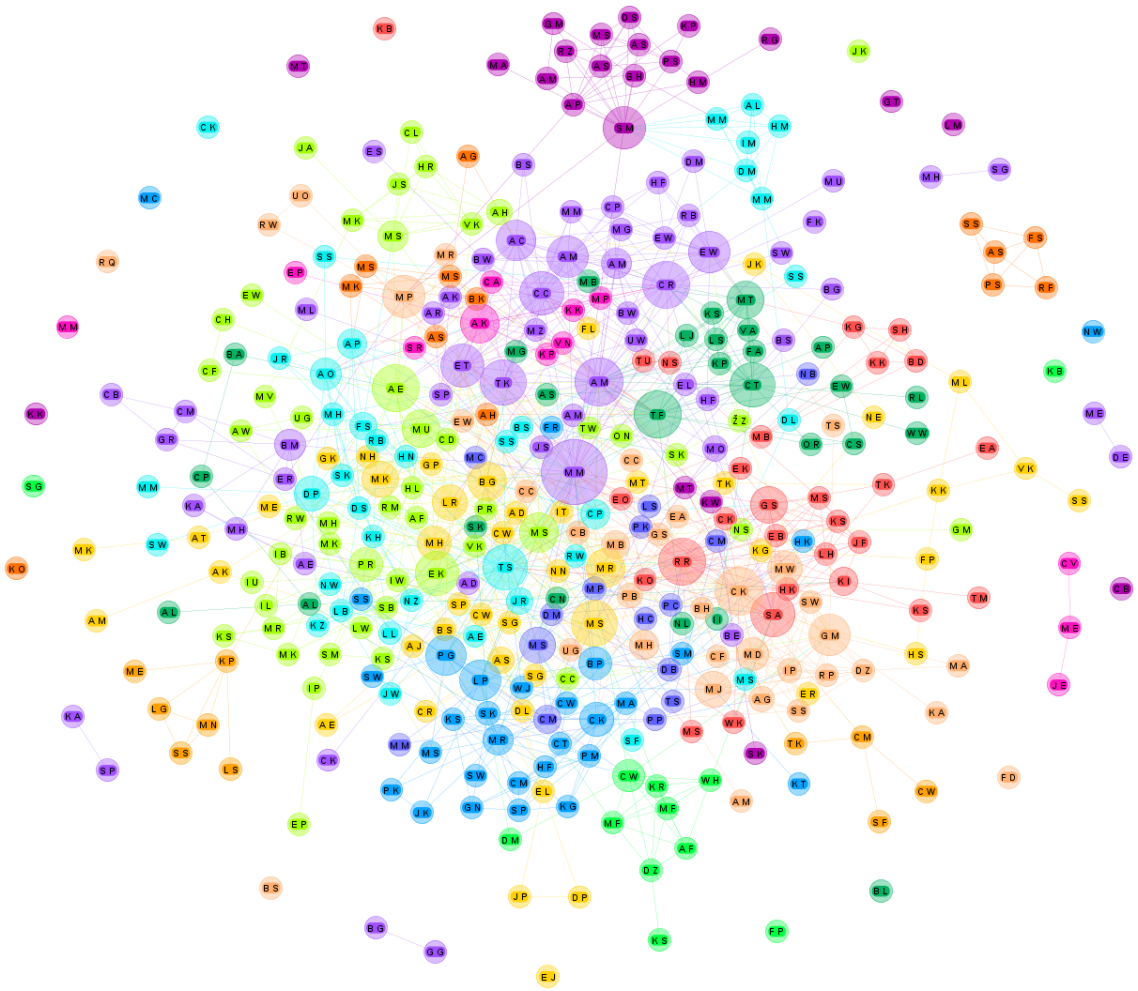

Figure 2.1: Network graph of my personal social network, showing 390 nodes, colored in 20 partitions. All nodes are directly connected to the center node (me) and are denominated by the person's initials. The partitions are found by sharing the same friends.

random [32]. This structure serves several aspects to the network. It certainly represents a type of geographical abstraction as physical distances between nodes need to be bypassed. Additionally it highly depends on the utilization of particular network nodes to probably design evasion routes to relieve highly utilized nodes. Last but not least, there is a commercial aspect to better connect certain nodes instead of using already established links. Company network arise to fulfill their own communication needs as well as their customer's. As this network is man made and costs a lot of money, it follows a specific intention or plan.

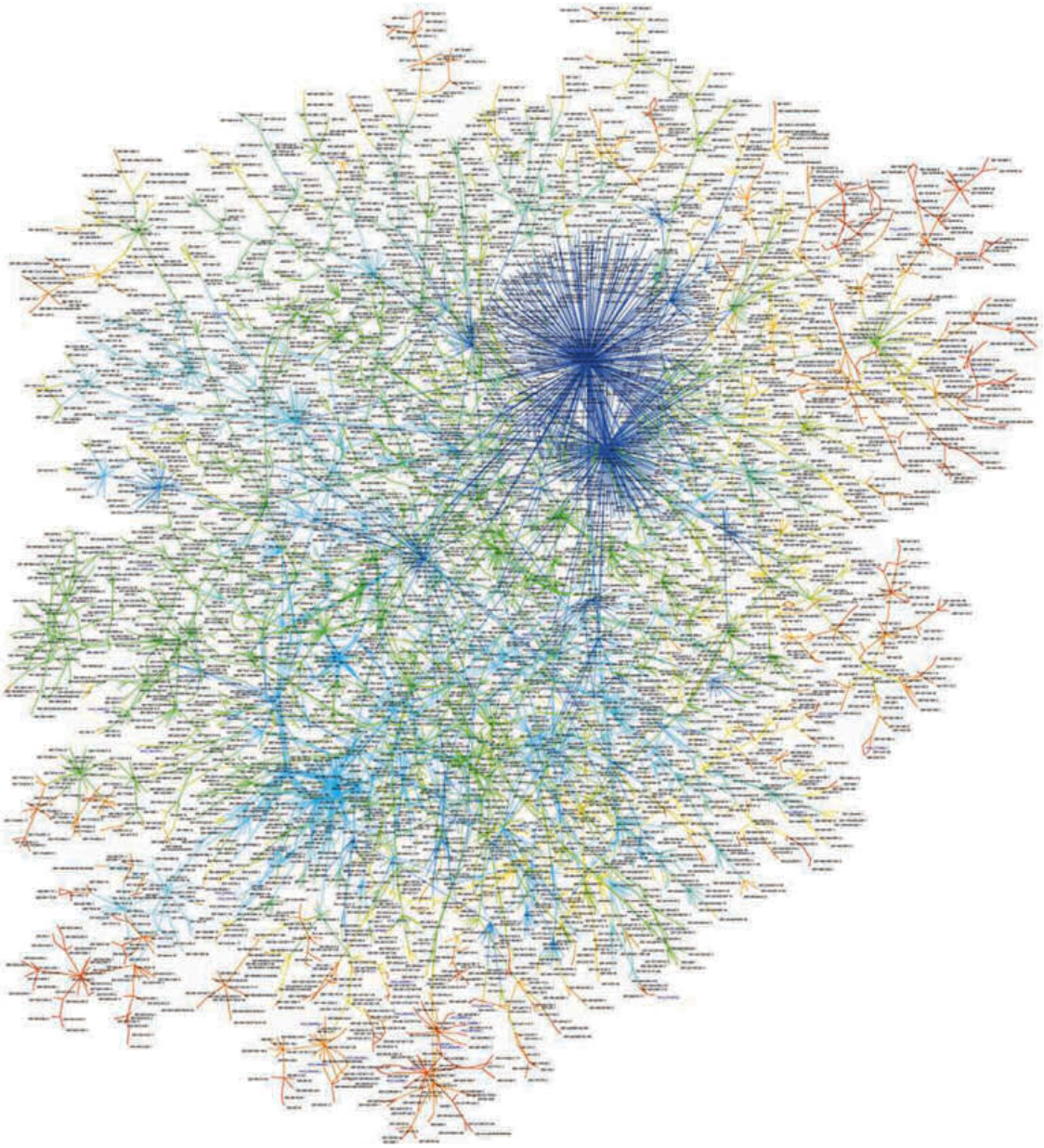

Figure 2.2: Network Graph of IPv6 Nodes from the year 2009 created by Lumeta. Each end node can represent a handful of computers on a small network, or perhaps a large company with hundreds of thousands of hosts. The interesting thing is the tree structure with many star-like formations.

2.2 Structure and Function of Networks

There is a relation between structure and function of networks. Let's take the Internet as an example: The link structure of the Internet should support efficient routing as well as it's hyperlinks should be efficiently navigable. The reliability of the network also highly dependent on it's structure and the number of redundant paths influences the reliability of the whole

network. There can also be characteristics defined as the resilience to the removal of nodes whether a network is robust or not. Other properties might be defined on certain nodes or on the whole graph itself. We can look for the nodes with the most connection or their distribution within the graph. Or define the average number of connections a node has on the complete graph. The pattern of connections can be regular or random, there can be clear structures like filamentary connections around edges or star-shaped formations. [32]

An interesting type of networks are mesh networks. A mesh network[10] is a special kind of network topology in which each node passes on data for other nodes inside the network. All nodes cooperate with each other in distributing the data inside the network. A mesh network can be designed using a flooding technique or a routing technique. When operating on the routing technique, the message is forwarded towards the target node along a calculated path, by hopping from one node to another until the destination is reached. To find an available path, the routing network uses self-healing algorithms to allow continuous connections and reconfiguration when paths break or nodes disappear (go down). This technique is described below. A mesh network in which all nodes are connected to each other is a called a fully connected network. Fully connected wired networks have security and reliability advantages, because problems in one cable only concern the two nodes attached to the cable. The disadvantage is the higher cost, because the number of connections increase dramatically with the number of nodes. It is an economic question and a question of how many data transfer is needed as a highly interconnected network can prevent transfer bottlenecks as the data may be transferred in parallel paths.

A mesh network can be established on top of a wired network like the Internet. This can be seen as computers linked together using communication protocols but still using the Internet as network layer. But there are also ways to communicate without using other underlying network technologies, especially avoiding commercial links. In Graz (Austria), there is a project called "Funkfeuer[11]". In Germany the corresponding name would be "Freifunk". This is a voluntary alternative network where people share their routers to propagate the network. As a benefit they can access the Internet through a single access point where the local network gets routed into the Internet by a local telecommunication provider. The interesting point is, that the local distribution is mostly ensured by the wireless routers. Longer distances between bigger nodes are provided via cable, fiber or directional radio. Every node gets a local IP address assigned and starts to communicating via ad hoc network wireless mode on an agreed frequency. A special designed routing protocol is employed that measures quality of links to other nodes and finds out an optimal routing for the IP packets. The protocol is called OLSR (Optimized Link State Routing Protocol[12] and is available for most Linux distributions and also for routers using the popular OpenWrt[13] software for their open source router. The current topology, that developed, enhanced and changed over years in Graz, can be seen in Figure 2.3. It shows the

[10] http://en.wikipedia.org/wiki/Mesh_networking
[11] http://graz.funkfeuer.at/
[12] http://en.wikipedia.org/wiki/Optimized_Link_State_Routing_Protocol
[13] https://openwrt.org/

client routers that the people installed on their own as well as the long range directional radio routes that connect distant networks or nodes.

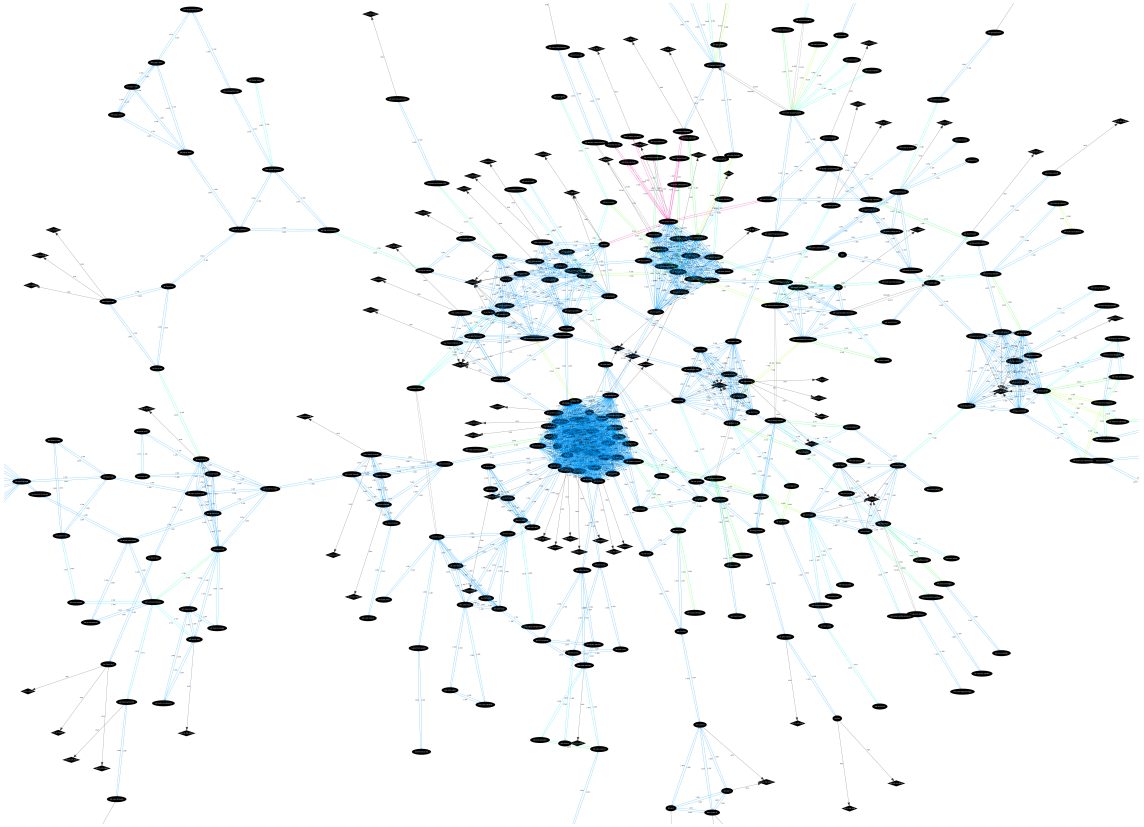

Figure 2.3: The topology of the Funkfeuer network in Graz. It shows, how the communication nodes are connected together by physical reception to form a mesh network. The routing is provided by the OLSR protocol. It does not show exact geographical relations between the nodes.

A wireless ad hoc network[14] (WANET) is a decentralized type of wireless network. In this mode, a network can be created, that does not build onto an existing infrastructure like routers or cable networks. Routes are set dynamically, so that each node knows where to pass the data. Every node receives data from other nodes and passes it on until the destination is reached. There is also a mode, where networks uses a flooding technique for forwarding. The network devices typically have equal priority and can associate with any other ad hoc network device that is in range. IEEE 802.11 describes that mode where two devices connect this way which is usually a network layer 2 activity implementing media access control, flow control and error checking. To make IP routing possible one additionally needs Layer 3 capabilities for packet forwarding and routing.

In contrast to the IPv6 network in Figure 2.2, the Funkfeuer network (Figure 2.3) does not follow commercial needs or political dictation. It developed from a couple of enthusiasts that

[14] http://en.wikipedia.org/wiki/Wireless_ad_hoc_network

brought the idea of having an alternative to former networks and to be independent of local (commercial) Internet or mobile network providers. To set up such a node and take part in this Funkfeuer network is not really much work. One only needs a router, that can be configured to use the ad hoc network mode with specific settings. It also needs to use the OLSR protocol described above so it is listed inside the network as available communication device with the connections it has to nearby friendly routers. The characteristics of the network itself shows again some highly connected areas as well as long range links to distant areas. So Funkfeuer is a geographical orientated network comparable to the Internet itself, road networks, water distribution networks or electricity grids. This is in contrast to the anatomy of social networks, where the connections are highly overlapping.

2.2.1 Small World Networks

The special structure of social networks leads us to to Stanley Milgram who first described these kind of networks as "Small World Networks". In the year 1967 he did an experiment in which he wanted to find out the distance distribution of an acquaintance graph. His experiment "The Small World Problem" [30] followed a stream of research started in sociology and psychology in the late 50s [3]. It is still the basis for current research, as it initially formulated the term of small networks in a time, where we could not analyze the network graphs of present social networks like Facebook. The question he answered was: "Given any two people in the world, person X and person Z, how many intermediate acquaintances links are needed before X and Z are connected?" In this experiment, he selected 296 volunteers as starting point within the USA to propagate a letter to one specific target person. The letter could only be mailed to personal acquaintances who is more likely to know the target person. So the people decided the letter to be passed on only by their local knowledge. See Section 2.3 for the context of local knowledge. This is important, as it likely hides the actual shortest path to target. So the most efficient way can rarely be exploited. Milgram also separated the starting population into three groups to see, how the region and the profession affects the path. He chose 100 people randomly living in Boston, 100 people living in Nebraska with the same profession as the target person and additionally 96 people living in Nebraska chosen randomly. The target person was working in Boston. Mentioning that Nebraska is quite away from Boston, this was to find out which impact the same profession had on knowing people around the target person. The findings of this experiment were quite surprising. The numbers of the table were taken out from the summary from Travers and Milgram [41]:

It is important to mention, that Milgram himself writes of "intermediaries" which means how many people are between two other people. In case we look at the whole social network as a graph, it seems more convenient for me to talk from the distance between two nodes, where each node is a person and the intermediaries are the distance between the nodes minus one. Milgram found out, that the social graph differs from a random graph quite a lot. He stated, that a person might know 500 friends. These friends might also know 500 people, but these people

	completed	average distance
all chains	29%	6.2
Nebraska group random	24%	6.7
Nebraska group same profession	31%	6.4
Boston group random	35%	4.4

Table 2.1: The distribution of chains from the three different groups. It shows the percentage of completed chains as well as the average distance of the completed chains.

may extensively overlap with the first group. So the network graph will be fragmented into social classes and cliques [30]. He also stated, that there might be unbridgeable gaps between various groups of people so that some groups may never link up because people have circles of acquaintances which do not necessarily intersect. Milgram also found out, that the linking between people is not only influenced by the geographical location and profession, but also on the gender of the participants. Nearly 79% of the participants passed the letter on to a person with the same sex. Sex roles and employment probably differed much back in the year 1967 which might not be comparable to today's society. Also there might be a difference between friends and relatives. Only 15% of the letters were sent to relatives. It is also remarkable, that 48% of the chains that reached the target came from 3 of the target's friends. These popular channels may be seen as node similarity described in Section 2.3. Evaluating the two different starting locations Boston and Nebraska, it can be said that the social distances matters more than the physical distance. It might also be true, that ethnic affiliation between the persons also would have a strong effect on the distance. His findings for sure can be seen as an upper bound for the distance function between people as there might be shorter paths the involved actors did not find out. But the actors living in the small world are able to exploit its smallness [3].

The term "small world network" finally got a face. It describes a network that consists of clusters (or circles) that are highly connected and connections between circles. The distances within the network can be smaller than within a random network. But there also might be groups that are not connected to each other. If a particular person a embedded in population A (which consists of his circle of acquaintances) cannot make contact with a particular person b, embedded in population B, then no other person in A can make contact with b and no other person in A can make contact with any other person in B [3]. Furthermore it can be said that the findings from the Milgram experiment first surprises our perception, as people tend to estimate the number of links it takes to reach the Canadian president would be much larger. Probably around ten or even more. But the experiment proves this feeling to be wrong. On the other hand a distance of six (=separation of five) should be thought of "five circles of acquaintances apart" or "five structures apart". This better describes the real situation, because only when a friend of mine knows a person (the distance is only two), this person probably has nothing to do with me in the first place. Another way of thinking about it is, that geometric progression is much more powerful than we feel.

2.2.2 Social Networks Today

The findings of the Milgram experiment described in Subsection 2.2.1 are still surprising today, but we can take this more up-to-date now. We have complex friendship structures in our modern communication and we can build representative graphs out of the data. There are social platforms like Twitter, Google+ or Facebook, and other chat systems like Skype or WhatsApp that can provide us with data about the connections between their users. Each of the mentioned platforms knows it's user's acquaintances (friend list stored on server) and probably also when they send messages to each other. This highly depends on the architecture of the platform. Very centralized applications like Facebook and WhatsApp do know exactly every friend-relation and even every message their users send to each other. In particular this is often criticized (quite rightly) for the purpose of privacy. But for finding the social network between the users, this is very useful and was investigated by researchers. In their paper "Four Degrees of Separation" [3], Lars Backstrom et. al. did a study on an actual Facebook relationship graph of many users. First of all, the "friendship-relation" in Facebook can be interpreted very broad. There might be some friends from older times, co-workers and probably relatives that one barely knows. Also some stars or celebrities might be defined as friends although there is a special function for that. This rises a general question of how good or intense a friendship is. To take these considerations into account, the researchers of this paper did not use all friends, but only the portion, that a person interacted with within the last month. This is in my opinion a very reasonable approach and would filter out the mentioned nominal members. This leads to a social graph with quite intense relationships between the persons.

The findings were similar to the experiment from Milgram, but even more meaningful as they had a much higher number of "participants". They investigated the entire Facebook network counting 721 million active users and about 69 billion friendship links. Of course, this huge amount of data needed special treatment. They found out, that the average distance of this graph was 4.74 which corresponds to 3.74 intermediaries or "degrees of separation". They also measured the dispersion of the distance distribution $spid$[15] which is defined by the variance–to–mean ratio $spid = \frac{\sigma^2}{\mu}$. They measured the dispersion of the distance distribution with 0.09 (underdispersion), which is typical for social networks whereas for web graphs a $spid > 1$ is typical (overdispersion). Still these results are a little different to the research from Milgram as in his experiment, the persons needed to be first-name acquaintances which is probably not true within the Facebook network. On the other hand many first-name acquaintances are missing inside this network because not everybody uses Facebook and therefore short paths will be missing [3]. These two effects might cancel themselves out, but we have no proof for this hypothesis. Look at Figure 2.1 to get a notion of how a typical Facebook graph for only one person can look like, is described in Section 2.3 as follows.

[15] spid = shortest-path index of dispersion

2.3 Navigation in Networks

As described above, the Internet is a huge network of interconnected nodes which initially consisted only of computers, servers and routers. In the last years a huge amount of smaller devices like mobile phones, tablets and handhelds joined the network. Including the Internet of things and the fact that people themselves are somehow connected to the Internet within our society, it makes it even more necessary to orient oneself within the network. Finding relevant information is the task of successfully navigating through the network.

"We need structural clues that make networks navigable and searchable. We want to measure navigability in order to make algorithms and generate tools" [20]. In their presentation about "Networks Navigability: Theory and Applications", the authors Denis Helic and Christoph Trattner approach the issue as follows:

Definition: A network is navigable if and only if there is a short path between all or almost all pairs of nodes in the network. So there has to exist a giant component and the effective diameter is bounded by $log(n)$, where n is the number of nodes in the network. To visualize the properties of various networks, there are some sketches that help pointing this out. The sketches and examples are taken from the slides from Denis Helic and Christoph Trattner. Figure 2.4 shows components of a network that is not navigable because there is no giant component, so the network is not connected. To continue with example networks, Figure 2.5 shows a network that has a giant component and the network is connected. However the network is not navigable because the effective diameter (longest shortest path) is too big: $7 > log_2(8)$. The network in the next example (Figure 2.6) is navigable because there is a giant component and the effective diameter is $2 < log_2(9)$. This properties can be seen as global network navigability, where we have a navigable network and we have global knowledge of the connection graph. In this case it is easy to design efficient procedures to find an arbitrary target node from an arbitrary start node. An applicable algorithm would be the breadth-first search which has a time complexity of $O(n + m)$, where n is the number of nodes an m is the number of links. [20]

Other problems arise when we only have local knowledge of the network. Local network navigability means that we only know the outgoing links from a certain node but nothing beyond that. Tasks that perform searches on such kind of network are called decentralized search. An example would be a search in a social network. As shown by an experiment from Stanley Milgram [30], people are extremely efficient in social searches. As shown in his experiment, people are able to find each other in less than seven hops (friends) even though none of the members have prior knowledge of the path. So we need to find the properties of social networks, that will help us to develop algorithms to efficiently search such structures.

The next example shows the ideas of these structural clues. In Figure 2.7, we have a network defined and the task is to navigate from node A to node D. There are two possible ways pointing out from A. See Figure 2.8 for the red connection. Obviously the optimal path leads to B. The structural property that guides us here is node degree. B is a node with many

outgoing connections. This makes it a good choice for the path. Nodes with a great number of outgoing connections may also be described as hubs or super nodes. To move on to node D, the possible paths are shown in Figure 2.9. As we can see, the optimal path leads to C. The structural property behind this is node clustering. A cluster is where nodes tend to create tightly knit groups characterized by a relatively high density of ties[16]. These are the requirements for navigability on local networks: The existence of network hubs that are connected with many nodes, and the existence of network clusters where nodes are highly interlinked with each other [20]. If we expand the network like shown in Figure 2.10, there are eight possible paths from B. The property that can guide us to select C over E is node similarity. Node similarity is external to the network. It can be derived from additional information that we have about network nodes. In the case of Milgram's experiment, the people selected the next person according to their occupation and their geographical location. Summing up the information that can be described as background knowledge about the network:

- Network hubs are nodes with many outgoing connections
- Network clusters are regions where nodes are highly interlinked
- Node similarity (e.g. Milgram: occupation and geography)

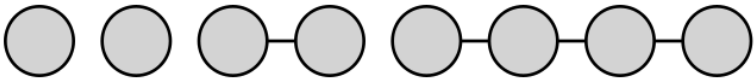

Figure 2.4: The network is not navigable because there is no giant component, i.e. the network is not connected.

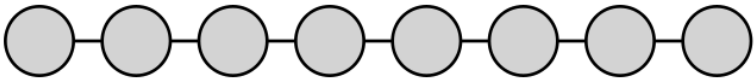

Figure 2.5: There is a giant component, i.e. the network is connected. However the network is not navigable because the effective diameter is 7, which is greater than $log_2(8)$

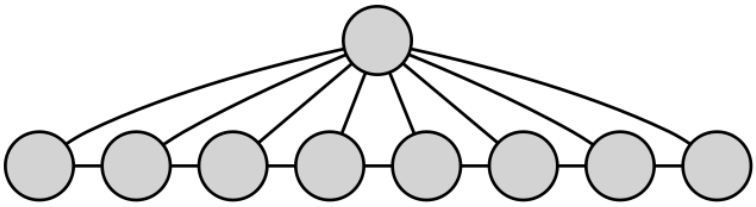

Figure 2.6: The network is navigable because there is a giant component and the effective diameter is 2, which is less than $log_2(8)$.

With these properties, we can build a metric space, where each node has unique coordinates and the distance between the nodes represents a notion of that background knowledge. Begin-

[16] http://en.wikipedia.org/wiki/Clustering_coefficient

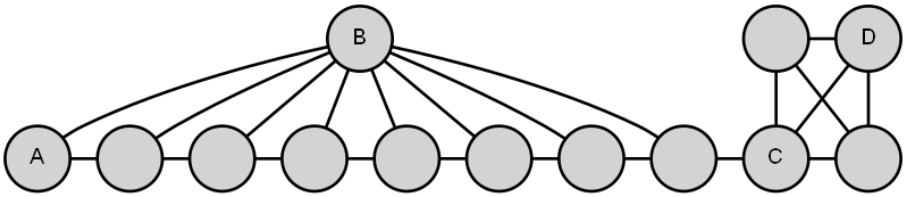

Figure 2.7: The task is to navigate from node A to node D within the network. There are two possible ways pointing out from A. Which one is to take?

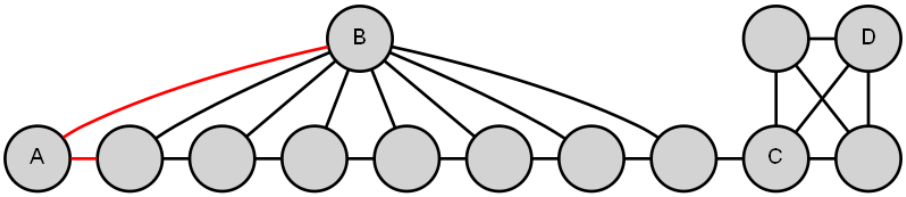

Figure 2.8: The task is to navigate from node A to node D. The structural property of "hubs or supernodes" with many outgoing connections guides us over B.

ning with the starting node, we need to maximize the probability of finding the target node. To do so, we always select the node with the smallest distance to the target. It has been already shown, that the greedy algorithm [17] is very efficient in this case and can reach the target in $log(n)$ number of steps. The algorithm is a problem solving heuristic that always selects the shortest path to the next node first. It generally cannot be said, that using these local optimal decisions will also yield a global optimum. But it can approximate a global optimal solution in reasonable time. Kleinberg [18] proved that theoretically, Watts [19] by simulation. The two also observed that hierarchical representation of the background knowledge is favorable and is also a good approximation of how people think.

Navigation in information networks is a kind of decentralized search, as users at each particular step of their navigation are only aware of links emanating from the current document [40]. Thus, this situation is intuitively very similar to decentralized search in social networks.

[17] http://en.wikipedia.org/wiki/Greedy_algorithm/
[18] http://www.cs.cornell.edu/home/kleinber/
[19] http://en.wikipedia.org/wiki/Duncan_J._Watts

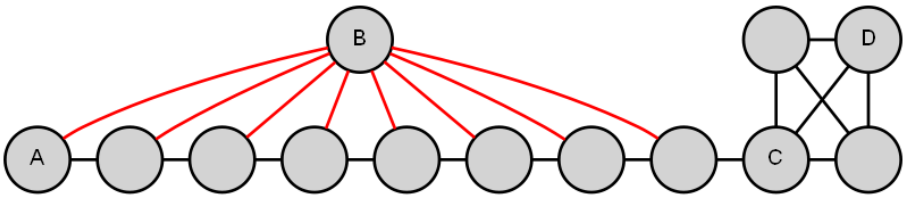

Figure 2.9: The task is to navigate from node A to node D. The structural property of "node clustering" guides us over C.

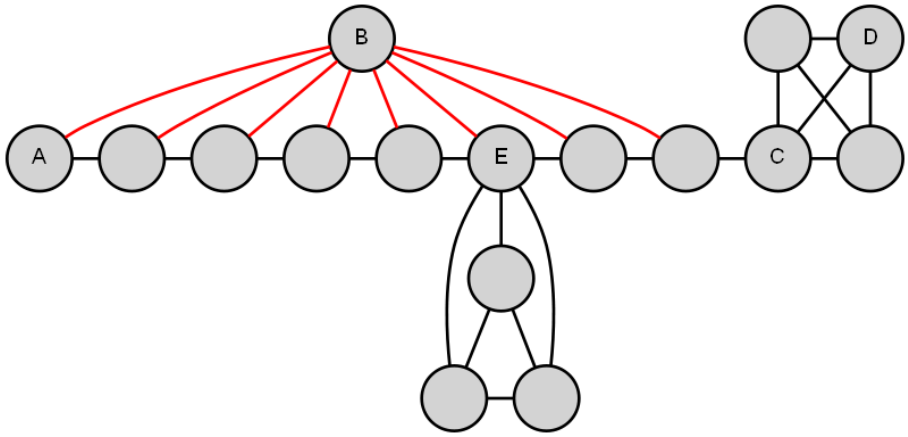

Figure 2.10: The task is to navigate from from node A to node D. The structural property of "node similarity" as background knowledge can guide us in selecting C over E.

So it would be an advantage of having the entire graph of all items. Knowing all nodes and their connections, one can tell exactly how direct relations with other items and one would probably find similar nodes to the requested one. This can of course be exploited when performing suggestions on search results, but for the user that kind of information is hidden. The user probably has a good sense for nodes to be similar or connected tighter. As mentioned in Section 2.3, people have very good intuition in connection with their social networks, but this can only be exploited in their local environment. It will be for the user's benefit to use this information we know about certain connections to get faster to the target and respectively to the desired information.

2.4 Information Retrieval

We usually search for a specific piece of information inside the network. When investigation any kind of network, the method we can use to find them is called information retrieval. But let us first define, what information is and what kind of information is valuable for us.

2.4.1 Definition of Information

Which kind of information are we talking about? First of all, the term information is used here for the broadest variety of possible items. It can be an article in Wikipedia, a research finding, an extract of a newspaper article, a recipe for cooking, an instruction leaflet for a medicine, the timetable of the local bus line or what the temperature is outside. It can be all items a human can ask for knowing. Probably also personal things like "where is my insurance folder" which possibly will not be available to everyone via Wikipedia. Any kind of information which can be stored somewhere needs to be indexed, so it can be found again later on. The number of captured information grew rapidly within the last decade and will grow further. Every object

can now generate an enormous amount of data and can nowadays be captured as all our stuff is instrumented and interconnected for having access to it in the future [21]. "The Internet of Things" describes the scenario where every small control system can be accessed and provides data and adjustment possibilities. Let's imagine a house, that plans automatically when the heating is switched on according to the outer temperature and the time schedule when the people will come home. Let's then imagine an intelligence, where all such houses take part in a big power grid and can negotiate how to distribute the power inside the grid. Trains and buses are transmitting their position in real time, even cars broadcast their speed for traffic jam evaluations, so that others can bypass. So there are many levels of different information that can (and will) be connected to each other. The result can be: increased efficiency, increased usability, better supporting services and in some cases a matter of survival. Our planet can be looked at as an information creation and transmission system we now can use for benefit of our society and our personal life. [21]

The "DIKW Pyramid" here comes in handy to unravel the different levels of information. It was well articulated by Russell Ackoff[1] in the Journal of Applied Systems Analysis in the year 1989 although the idea was invented many years ago but the originating source is not known. Later Jennifer Rowley reformulated his ideas as follows: "Typically information is defined in terms of data, knowledge in terms of information, and wisdom in terms of knowledge, but there is less consensus in the description of the processes that transform elements lower in the hierarchy into those above them, leading to a lack of definitional clarity. In addition, there is limited reference to wisdom in these texts." [35] "The presentation of the relationships among data, information, knowledge, and sometimes wisdom in a hierarchical arrangement has been part of the language of information science for many years. Although it is uncertain when and by whom those relationships were first presented, the ubiquity of the notion of a hierarchy is embedded in the use of the acronym DIKW as a shorthand representation for the data-to-information-to-knowledge-to-wisdom transformation." [42]

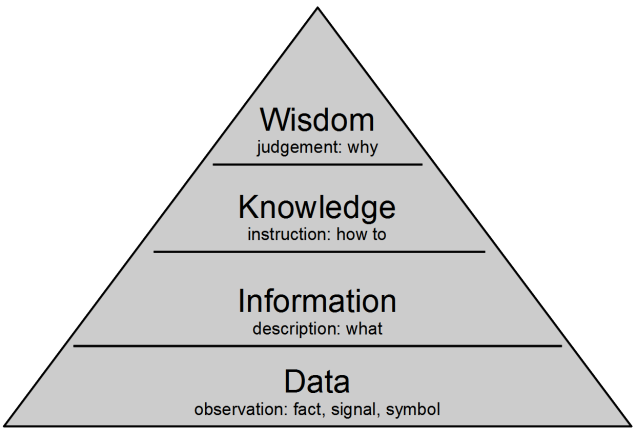

Figure 2.11: DIKW Pyramid consists of wisdom, knowledge, information and data. It shows what is meant by each layer and that each layer is dependent on the underlying layer(s).

The data-information-knowledge-wisdom hierarchy is used in information sciences to describe different levels of abstraction in human centered information processing. Have a look at Figure 2.11 for the abstraction and the basic meaning of the parts of the model. According to the paper Knowledge Retrieval from Yiyu Yao et al. computer systems can be designed for the management of each of them. Data Retrieval Systems (DRS), such as database management systems, are well suitable for the storage and retrieval of structured data. Information Retrieval Systems (IRS), such as web search engines, are very effective in finding the relevant documents or web pages that contain the information required by a user. What those systems lack is the management at the knowledge level. A user must read and analyze the relevant documents in order to extract the useful knowledge. Knowledge Retrieval Systems (KRS) are the next generation retrieval systems for supporting knowledge discovery, organization, storage, and retrieval. Such systems will be used by advanced and expert users to tackle the challenging problem of knowledge seeking. [46] The parts of the model can be described as follows.

Data

Data is the basis and can be seen as being discrete, objective facts or observations, which are unorganized and unprocessed and therefore have no meaning or value because of lack of context and interpretation [35]. Data may also be seen as signals any kind of sensor can perceive. In connection with human perception it may be light, vision, sound, smell, taste, touch which in some literature is called "subjective data". In more technical saying, these signals could be electromagnetic waves or particles. Another approach is that data are recorded symbols like numbers, characters, words, text, diagrams, images, videos. Together they form the basis of the "DIKW Pyramid". Data infers information.

Information

Information is about what the data is. Information is inferred from data in the process of answering interrogative questions like "who", "what", "where", "how many" and "when" [1]. It can be seen as structural or functional distinction to data, others define information only as "matter-of-fact". Information can exist universally as symbols and signs, can be subjective giving symbols meaning or both of it. Information makes data useful for further decisions or actions and therefore gives data meaning and purpose.

Knowledge

Knowledge is typically defined with reference to information [35]. It can be seen as processed, organized or structured information or information being applied or put into action. Knowledge might be defined as follows: "Knowledge is a fluid mix of framed experience, values, contextual information, expert insight and grounded intuition that provides an environment and framework

for evaluating and incorporating new experiences and information. It originates and is applied in the minds of knowers. In organizations it often becomes embedded not only in documents and repositories but also in organizational routines, processes, practices and norms."[42] In this context, knowledge is mainly seen as being subjective rather than universal. It may be synthesis of multiple sources of information over time or been seen as organization and processing to convey understanding, experience and accumulated learning. Knowledge may also be described as skill, expertise, capability or experience. The distinction here between subjective knowledge and subjective information is that subjective knowledge is characterized by justifiable belief, where subjective information is a type of knowledge concerning the meaning of data.

"Knowledge is a thought in the individual's mind, which is characterized by the individual's justifiable belief that it is true. It can be empirical and non-empirical, as in the case of logical and mathematical knowledge (e.g., "every triangle has three sides"), religious knowledge (e.g., "God exists"), philosophical knowledge (e.g., "Cogito ergo sum"), and the like. Note that knowledge is the content of a thought in the individual's mind, which is characterized by the individual's justifiable belief that it is true, while "knowing" is a state of mind which is characterized by the three conditions: (1) the individual believes that it is true, (2) she/he can justify it, and (3) it is true, or it appears to be true." [47]

Wisdom

Probably wisdom can be defined as intelligence, insight or science based on life experience, maturity and sufficient distance to the topic. According to Rowley, wisdom is the ability to increase effectiveness. Wisdom adds value, which requires the mental function that we call judgment. The ethical and aesthetic values that this implies are inherent to the actor and are unique and personal. [35]

Ackoff refers to understanding as an "appreciation of 'why'", and wisdom as "evaluated understanding", where understanding is posited as a discrete layer between knowledge and wisdom. [1]

2.4.2 Searching for Information

When searching for a specific piece of information on the Internet, we generally speak of information retrieval. Information retrieval deals with the representation, storage, organization of, and access to information items [4]. According to Baeza-Yates and Riberio-Neto, the user has to translate his information need into a query which can then be processed by a search engine. This set of keywords is handed to the search engine which hopefully gives back websites or documents which are relevant for the user. The description of the desired information is very fuzzy, as opposed to data retrieval, the conditions (e.g. regular expressions) are clearly defined. This means, that only strict matches appear in the results, as for information retrieval, the system has to interpret the contents of the information items and rank them according to the

degree of relevance to the query.

In Figure 2.12 taken from Baeza-Yates and Riberio-Neto [4], the process of retrieving information from a query given by a user is shown. Briefly summarized, they describe the process as follows: The right half of the figure shows the process of preparing the present data that is used for the search. The documents and the operations on the documents to extract the text are defined. The text model then generates the logical view of the documents. The index is generated and stored into the database. Different index structures might be used at the same time because of the huge size of the data. A commonly used method is the *inverted file* as indicated in Figure 2.12. It is to mention, that resources in time and storage space spent on defining text database and building index are amortized by querying the retrieval system many times. On the left half of the figure, the user formulates his needs into a text query using natural language. Query operations might be applied to combine certain conditions before the query is processed. The database index gets searched for the defined queries and the retrieved documents get sorted by relevance before they are printed to the output page. The sorting (often referred as ranking) should measure the likelihood of the relevance of the document. After presenting the result, the user might give feedback or even reformulate her/his query and the process can start over again.

The whole process is quite complex and the relevance ranking is not easy to perform, as the interpretation of the human intention is mostly ambiguous and unclear. Classifying information and putting them into categories helps users to find information. This is where also tags emerge. Tags are a kind of meta data that describes an item. In contrast to categories, tags are non-hierarchical, but in case of connecting pieces of information together fulfill the same idea as categories. The term browsing is used for finding information using these categories or tags for navigating though the pages.

As an example for a browsing site I used Wikipedia and the music portal LastFM. Browsing in Wikipedia can be very convenient as every article is grouped in different categories. For example, "Arnold Schwarzenegger" shows up in every of the following categories: "*1947 births, Actors from California, American actor–politicians, American athlete–politicians, American bodybuilders, American entertainment industry businesspeople, American film actors, American film directors, American film producers, American health activists, American philanthropists, American Roman Catholic politicians, American video game actors, Arnold Schwarzenegger, Austrian bodybuilders, Austrian film actors, Austrian immigrants to the United States, Austrian soldiers, American people of Austrian descent, California Republicans, Disability rights activists, Governors of California, Kennedy family, Laureus World Sports Awards winners, Living people, Naturalized citizens of the United States, People from Graz, University of Wisconsin–Superior alumni, Professional bodybuilders, Shriver family, Sports people from California, University of California regents, Writers from California*".

The music portal LastFM is structured very strict. Every music track is part of an album, and every album is made by an artist. Additionally, every item is tagged with keywords to

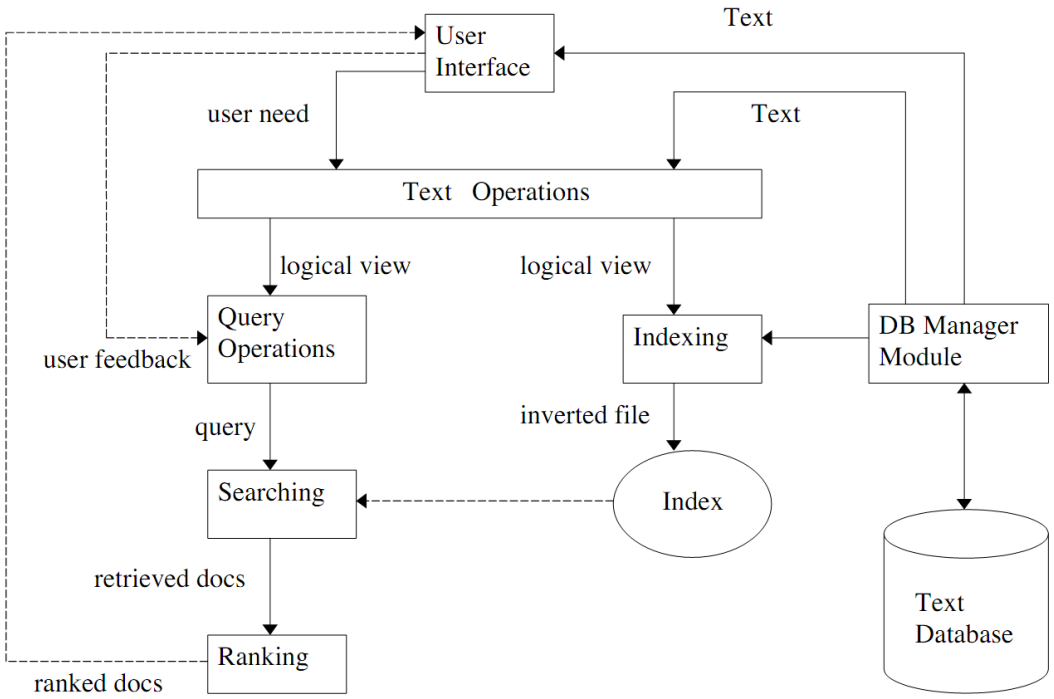

Figure 2.12: The process of retrieving information starts with a query from a user. Query operations might be applied to combine certain conditions. The query is processed by the database which matches the keywords against previously processed documents over a large index of documents. After finding documents, the relevance is measured and a sorted output is generated. After presenting the result, the user might give feedback if she/he is already satisfied and can start the process again.

connect them. For example, the artist "Falco" is in the following categories: "*80s, 80s pop, 90s, alternative, austria, austrian, austrian music, austropop, classic rock, cool, crossover, dance, dance pop, dead, deutsch, deutschpop, deutschrock, deutschsprachig, disco, eighties, electronic, electronica, electropop, european, europop, falco, funk, genius, german, german hip hop, german lyrics, hip hop, indie, legend, legendary, male, male vocalist, male vocalists, melancholy, multiple languages, ndw, neue deutsche welle, new wave, old school, oldies, pop, pop rap, pop rock, pop–rock, poprock, rap, rock, singer–songwriter, soft rock, synth pop, synthpop, vienna, wien, österreich*".

We now have a sense for how interconnected a single subject can be and therefore how many ways may lead to that certain subject. Depending on the kind of information that was searched for, it is hard to tell which type of category should be used.

2.4.3 Searching for Knowledge

According to Yao et al.[46], the DIKW hierarchy concisely summarizes different types of resources that we can use for problem solving. The hierarchy represents increasing levels of

complexity that require increasing levels of understanding. The generations of the retrieval systems may be studied based on this hierarchy. Yao suggests an evolution process of retrieval systems from data retrieval to knowledge retrieval, and from information retrieval to information retrieval support (a step towards knowledge retrieval). So the next step will be a change from data and information retrieval to knowledge retrieval, including retrieving and searching for knowledge objects in advanced knowledge management systems.

A comparison of data, information and knowledge retrieval as put together in the following table can emphasize the meaning of the step from data and information retrieval towards knowledge retrieval. The structures in data and information retrieval are well known today. Handling these categories is not a big challenge any more and is done by search engines every day.

	Data Retrieval	Information Retrieval	Knowledge Retrieval
Match	Boolean match	partial match best match	partial match best match
Inference	deductive inference	inductive inference	deductive inference inductive inference associative reasoning analogical reasoning
Model	deterministic model	statistical model probabilistic model	semantic model inference model
Query	artificial language	natural language	knowledge structure natural language
Organization	table index	table index	knowledge unit knowledge structure
Representation	number rule	natural language markup language	concept graph predicate logic production rule frame, ontology semantic network
Storage	database	document collections	knowledge base
Retrieved Results	data set	sections or documents	a set of knowledge unit

Table 2.2: A comparison of data, information and knowledge retrieval to emphasize the meaning of the step from data and information retrieval towards knowledge retrieval.

Knowledge Retrieval System

Yao et al. summarize the task of knowledge retrieval as follows and propose a knowledge retrieval system as outlined in Figure 2.13: Knowledge represented in a structured way is consistent with human thoughts and is easily understandable. Sometimes, users do not know exactly what they want or lack contextual awareness. If knowledge can be provided visually in a structured way, it will be very useful for users to explore and refine the query. The figure shows a conceptual framework of a typical knowledge retrieval system. The main process can

be described as follows: [46]

- (1) Knowledge Discovery: Discovering knowledge from sources by data mining, machine learning, knowledge acquisition and other methods.
- (2) Query Formulation: Formulating queries from user needs by user inputs. The inputs can be in natural languages and artificial languages.
- (3) Knowledge Selection: Selecting the range of possible related knowledge based on user query and knowledge discovered from data/information sources.
- (4) Knowledge Structure Construction: Reasoning according to different views of knowledge, domain knowledge, user background, etc. in order to form knowledge structures. Domain knowledge can be provided by expert systems. User background and preference can be provided by user logs.
- (5) Exploration and Search: Exploring the knowledge structure to get general awareness and refine the search. Through understanding the relevant knowledge structures, users can search into details on what they are interested in to get the required knowledge.
- (6) Knowledge Structure Reorganization: Reorganizing knowledge structures if users need to explore other views of selected knowledge.
- (7) Query Reformulation: Reformulating the query if the constructed structures cannot satisfy user needs.

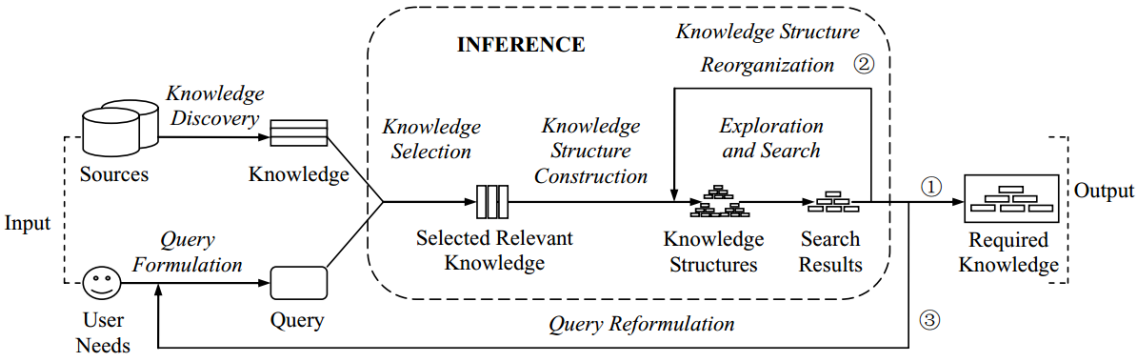

Figure 2.13: Outline of a Knowledge Retrieval System. The figure taken from [46] puts the described parts together in one picture. Read the points 1 to 7 above for explanation.

"One of the key features of knowledge retrieval systems is that knowledge is visualized in a structured way so that users could get contextual awareness of related knowledge and make further retrieval. Main operations for exploration are browsing, zooming-in, and zooming-out. Browsing helps users to navigate, zoom-out provides general understanding, while zoom-in presents detailed knowledge and its structure." [46]

Research and Supporting Theories

According to Yao et al. [46] knowledge retrieval can draw results from the following related theories and technologies. The list is not exhaustive and the topics listed under each entry only serve as examples:

- Theory of Knowledge: knowledge acquisition, knowledge organization, knowledge representation, knowledge validation, knowledge management.
- Machine Learning and Knowledge Discovery: preprocessing, classification, clustering, prediction, postprocessing, statistical learning theory.
- Psychology: cognitive psychology, cognitive informatics, concept formation and learning, decision making, human-computer interaction.
- Logic and Inference: propositional logic, predicate logic, attribute logic, universal logic, inductive inference, deductive inference, associative reasoning, analogical reasoning, approximate reasoning.
- Information Technology: information theory, information science, information retrieval, database systems, knowledge-based systems, rule-based systems, expert systems, decision support systems, intelligent agent technology.
- Linguistics: computational linguistics, natural language understanding, natural language processing.

The application of knowledge retrieval for example is when scientists explore literature and search for scientific facts and research results. Knowledge structures provide a high level understanding of scientific literature and hints regarding what has been done and what needs to be done. Different views provide various unique understanding of the literature. The knowledge structure is constructed based on proceedings indexes, document structures and domain knowledge. The knowledge structures not only provide a relation diagram of specified discipline, but also help researchers to find the contribution of each study and possible future research topics in rough sets. [46]

2.5 Category Hierarchies and Tags

Searching for specific information can be supported by a number of techniques. Searching for keywords within full text may be insufficient as there may be different meanings of the same word. In some cases the context of a word can completely change the meaning of it. Sometimes the lack of technical vocabulary also prevents a user to find certain articles. A commonly used solution for this is to tag the articles with words that are suitable for summarizing the article as well as with words that people may search for so they can find the article.

A tag[20] is a keyword that is added to an item like an article or more generally to a piece of

[20] http://en.wikipedia.org/wiki/Tag_(metadata)

information. It is like a label that describes the item which helps the item to be found again. It is metadata that is usually chosen by the author or creator of the item. Sometimes also viewers can choose tags. Every piece of information like documents, files and pictures can be tagged with one or more tags. Tags can be words, numbers, images or marks. Searching for a tag is like searching in an index of an encyclopedia with the difference, that combining tags can help enormously as the intersection of more tags is usually small which helps finding the desired information.

A tag cloud[21] is a visual image of a text that shows the amount of occurrences of words inside the text. Linking words are canceled out whereas other words remain and get counted. The more often a word occurs, the bigger gets the word in the visualization of the tag cloud. To see the most used words inside a text can be surprisingly helpful to get a notion what the text is about. At least the topic can be found out very easily. Combining tags with tag clouds and use them in a search function can be very helpful. See Figure 2.14 for an example for a tag cloud. It was created from the text of this thesis by an online web site named TagCrowd[22]. It illustrates the topic of the text quite well and has a good visualization effect. The main topics are easy to locate and surrounded by additional information. Usually irrelevant words (like conjunctions) can be suppressed and specific words can be filtered that they do not show up unintendedly.

Figure 2.14: Example for a tag cloud. I used this thesis to create this tag could. The document was only lexically analyzed and the conjunctions were suppressed. Although it is only a statistical analysis, it represents the content (and the meaning) of this text very well.

The use of tags misses integrity and uniqueness. It is a good tool to make information findable, but it has too little structure and it highly depends on the effort someone puts into tagging. There are also ideas of combining tag clouds with ordinary search queries like described by Christoph Trattner and Denis Helic [38]. The authors want to tackle the problem, that users usually come to a Wikipedia site by searching for some keywords within their preferred search engine. After reading the information from this site, the users leave if they found the information

[21] http://en.wikipedia.org/wiki/Tag_cloud
[22] http://tagcrowd.com/

they needed, or go back to the search engine and refine their query. The idea is to keep the user on the Wikipedia page and let him continue his search by offering him related content based on tag clouds or the original search query (which is often not known at this point). This will guide the user to related content and probably to the right information if his search has been unsuccessful so far.

More structure comes into the system when using categories. On Wikipedia, each page is categorized[23] sometimes by more than one category. "The central goal of the category system is to provide navigational links to all Wikipedia pages in a hierarchy[24] of categories which readers, knowing essential-defining-characteristics of a topic, can browse and quickly find sets of pages on topics that are defined by those characteristics". [6] Compared with a list, a category may have both advantages and disadvantages. Every page in the article namespace should have at least one category. Categories should be on major topics that are likely to be useful to someone reading the article:

- Article: Michael Jackson
- Useful category: *Category:American pop singers*
- Not useful: *Category:Musicians whose first name starts with M*

An article will often appear in several categories. Restraints should be used, however categories become less effective the more there are on a given article. An article should usually not be in both a category and its subcategory, e.g. Microsoft Office is in *Category:Microsoft software*, so should not also be in *Category:Software* except when the article defines a category as well as being in a higher category, e.g. Ohio is in both *Category:States of the United States* and *Category:Ohio*. A good way to understand this exception is that if an article exists, and then a category is created on the same subject as the article, it should not cause the article to be removed from any of its categories. A category might be appropriate when all of the following questions can be answered with yes:

- Is it possible to write a few paragraphs on the subject of a category, explaining it?
- If you go to the article from the category, will it be obvious why it's there?
- Is the category subject prominently discussed in the article?

According to Wikipedia[6], advantages of a category is, that it automatically creates a link to a category on an article page, and a corresponding link to that article. A category can contain multiple subcategories, and can also be part of several categories. Categories are organized within Wikipedia into a web of knowledge starting with *Category:Wikipedia* categories. So it is good for exploratory browsing of articles. They are relatively unobtrusive in that they generally don't distract from the flow of the article. Disadvantages of a category may be, that they are difficult to maintain. Sometimes articles need to be moved as categories arise or get merged. There is no guarantee, that a topic meets a category's criteria of inclusion.

[23] http://en.wikipedia.org/wiki/Wikipedia:Categorization
[24] http://en.wikipedia.org/wiki/Hierarchy

Additionally multiple categories on one article can lead to confusions. Sometimes it could be useful to know the reason why an article is within the category, but this information is hard to represent.

3 Click Data Acquisition

This chapter gives an overview over the available methods that can be used to retrieve and store the desired kind of data from the user and what the retrieved data can be used for. This task is quite related to the term web analytics[1] used for advertising. But advertising only focuses on a few aspects for using this methods which is for example suggesting products on sales pages. Typically recommender systems[2] exploit this data for their use.

"Recommender Systems are software tools and techniques providing suggestions for items to be of use to a user. The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read. 'Item' is the general term used to denote what the system recommends to users. A recommender system normally focuses on a specific type of item (e.g., CDs, or news) and accordingly its design, its graphical user interface, and the core recommendation technique used to generate the recommendations are all customized to provide useful and effective suggestions for that specific type of item. Recommender systems are primarily directed towards individuals who lack sufficient personal experience or competence to evaluate the potentially overwhelming number of alternative items that a web site, for example, may offer. The popular web site Amazon.com[3] employs a recommender system to personalize the online store for each customer." [34]

Recommender systems are information filtering systems, that try to quantify an affinity of a certain user to an item. The system measures previous inputs like search terms, views or purchased items for predicting interests and makes suggestions out of the available items. It consists of two parts: the data mining part and the prediction part that often falls into the area of machine learning. The application of such systems have a broad variety. The most familiar systems are in online shops like Amazon that suggests books, movies and products. Another application would be suggesting friends on a social platform, online dating or Twitter followers. Also search queries can be predicted. The prediction itself can be based on the user's data itself or on the data of others. In words: "Customers that bought product A also were interested in product B" vs. "You could also be interested in B".

The advantage of such systems cut both ways: For the vendor, it increases the number of items sold and the user satisfaction is increased. He/she can also sell more diverse items and he/she can better understand what the user wants [34]. On the other hand the user might feel

[1] http://en.wikipedia.org/wiki/Web_analytics
[2] http://en.wikipedia.org/wiki/Recommender_system
[3] http://www.amazon.com/

pursued because the system knows many details from the user's previous navigation sessions and as the system will not forget over time it might be surprising and uncomfortable for the user. Within this work, I am not interested in selling items to the user, but in finding similarities between categories and fields to guide the user to the information he/she searches. So there is a huge difference in how the following methods get applied and whether the user might draw profit from it or not.

3.1 Methods for Retrieving Click Data

There are three different basic types for retrieving click data. The methods are heavily dependent on the target of your research and what possibilities you have. This task is quite related to the topic of web analytics. "Web analytics itself deals with the methods for measurement, data collection, data analysis and providing the related feedback on internet for the motive of understanding behavior of the customer using web site. The benefit of studying behavior of the customer leads to optimize the usability of a web site." [25]In their survey, Kumar et al. propose, that there are mainly three pillars of web analytics: Data collection, data storage, data evaluation. In this chapter the main question will concern the data collection. In the game I created in this thesis, I can excessively use the data from the user as it is no secret to the user that I want to process that data. In this case I have an advantage to most web analysis tools, as they need to hide themselves from the user and need to stay in the background. So these are two different, but technically similar approaches to the same topic.

3.1.1 Web Server Log Files

If one is the owner of the site that should be investigated, one has the opportunity to process the log files of his web server to determine on which links the users clicked. It is essential, that the retrieved file names are sufficient distinguishable, so there is no doubt what content the user saw. This method is completely unnoticed by the user and therefore needs no awareness from him. It also can be processed at a time afterward as long as the log files are available. See an example for a log file in Listing 3.1. In the first line, we have the IP-address of the accessing computer (or proxy server), followed by date and time of access. The important part is the path of the page which was requested by the `GET` method. The next entries show, which client the user used to access this page. In this case it was Google Chrome version 33 on a Windows NT 6.3 machine, that normally refers to a Windows 8 operating system. The referred page is also declared as seen in line two. So the file `emailsettings.png` was referred to from the `index.html` file.

```
1 129.27.2.3 - - [10/Mar/2013:08:41:59 +0100] "GET /index.html HTTP/1.1" 200
    1385 "-" "Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML,
    like Gecko) Chrome/33.0.1750.146 Safari/537.36"
2 129.27.2.3 - - [10/Mar/2013:08:42:00 +0100] "GET /emailsettings.png HTTP/1.1"
    200 21125 "http://server/index.html" "Mozilla/5.0 (Windows NT 6.3; WOW64)
    AppleWebKit/537.36 (KHTML, like Gecko) Chrome/33.0.1750.146 Safari/537.36"
3 129.27.2.3 - - [10/Mar/2013:08:42:00 +0100] "GET /favicon.ico HTTP/1.1" 200
    3569 "-" "Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML,
    like Gecko) Chrome/33.0.1750.146 Safari/537.36"
4 129.27.2.3 - - [10/Mar/2013:08:46:09 +0100] "GET /webdav.png HTTP/1.1" 200
    38270 "-" "Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML,
    like Gecko) Chrome/33.0.1750.146 Safari/537.36"
```

Listing 3.1: These are some logfile entries generated by an Apache HTTP server. This is a typical output that a server generates from a web request by someone on the Internet. It contains the IP address, a timestamp, the request parameters and a user agent string.

In a smaller scale, a kind of statistical analysis is already done for years on most web sites to determine page view counts and page impressions. An often used software for that is Webalizer[4], although the particular path of one user can not be tracked with this software. Problems arise when users are connected via proxy servers, which can cache web pages. In this case, the user retrieves a page, which is not detected by the server's log file. The same problem arises, when the browser itself caches pages and a web site is called for the second time, so the browser does not initiate a separate request to the server. The problem diminished lately, as many content pages are nowadays server generated instead of static HTML files. Anyway generated pages need to be accessed separately and are a big problem for caching algorithms. But for log file analysis this comes in handy. When dealing with caches it is generally hard to predict exactly which links the user has followed.

3.1.2 Modify Site Content

If you are not the owner of the web site, you usually cannot modify the content. In this project, I have the possibility to load the target web site ourselves and present it on our web site. Additionally it is possible to make some changes meanwhile. As the browsing address line then points to our server's site, it is quite obvious to the user, that he/she is not browsing the original site. In our case, the whole process is wrapped into a game, so it is transparent for the user. See Section 5.1 for that. This is important, as this method of giving the user a modified page, could have the flavor of betraying the user if it is not clear for him/her what is going on. In practice, all link tags will be extended to send additional click information to the server. Also all local links need to be modified to point to an address, where the subsequent page can be loaded and modified again. This is the chosen method for this software project. Although I find it appropriate to use this method within this project, I would not recommend this to be used

[4] http://www.webalizer.org/

in commercial sites as there might arise legal issues because I am presenting external content which could be misinterpreted as being my own intellectual property.

3.1.3 Page Tagging and Cookies

Page tagging became very popular these days because it is probably the easiest method for tracking users. It is done by inserting a small JavaScript in every page of a web site. This means that every time a visitor opens the page, this script is activated and the visitor information and actions can be tracked [25]. This solves the problem of inaccuracy of log file analysis and the presence of caching (browser and proxy). It is also possible to outsource the service to companies specialized on the topic, although much company dependent information is leaking then to the other company. I would consider this to be a huge drawback and it is in fact a little ironic, because companies that spy on their users can then be spied on by the company that does their web site evaluations.

Cookies are also commonly used and provide some interesting identification methods. Cookies are created on the user's computer when the browser accesses the web site. There are mainly two types of cookies. One is "session cookies" and the other "persistent cookies". Session cookies get deleted once the visitor leaves the web site, persistent cookies remain on the visitor's computer. The reason why cookies are being used is to identify visitors. Each cookie holds a unique ID number, which makes it possible to differentiate between returning visitors and new visitors [25].

3.1.4 Browser Plugins

Browser plugins can handle local modifications of loaded web sites inside the web browser before they get displayed. An example for that is Greasemonkey[5]. Written in JavaScript and XUL (User Interface Language), it can handle different scripts at the same time. In the case of retrieving click data information, one can build a script, that automatically adds similar information as described in Subsection 3.1.2 above. The advantage is, that the page the user is browsing still has the original address, and one can track the whole browsing behavior also an different sites. The disadvantage is, that the user will have to install the script and that he is giving away very much of his privacy, as it is not transparent for what purpose the data will be used. So this is quite a risky method. An example for a Geasemonkey script that performs such a task is Wiki Paths (Subsection 3.5.1).

[5] https://addons.mozilla.org/de/firefox/addon/greasemonkey/

3.2 Use of Click Data

As mentioned above, the data itself can be used in numerous ways. In our case I use the data to determine the path a user chooses from one topic to another. It is to find out neighboring topics and categories and more general a little view inside the brain of the user and how he/she connects knowledge. More generally, the acquisition of click data has a much wider important role. For web site providers, it can be information about the popularity of his/her offers and he/she can probably predict the intention of his/her users. For advertising companies, it will categorize the interests of the users to give them tailored advertisements and offers. In fact, it is a dangerous tool that can help the user or that can be used against him for the profit of others.

In information theory, knowing the path between two articles (or nodes) can improve the understanding of the meaning of neighboring nodes and one can gain additional information out of that knowledge. It can help to predict and understand on which basis nodes are connected together. This can be used to guide the user more precisely through the process of finding information. Refer to Subsection 2.4.2 for a better understanding of the process of information retrieval. On the one hand it can improve the ranking of the documents by affecting the relevance, on the other hand it can present information that was not explicitly searched for because it meets the criteria for similar search terms.

3.3 Security And Privacy

An important question about security and privacy arises. As mentioned above: If the user is aware of the fact, that he/she is taking part in a game, privacy is not violated at all. This applies in my case of the developed software. In the case of scanning log files and performing analytics on web sites, the user gets traced without knowing. This can also be unwanted, but is reality for years now. The good thing is, that it is limited to the provider of the used service. As there are many providers, the data cannot be merged that easily across different providers. More dramatically, it is today's state of the art to use Google Analytics[6] which gathers information far beyond a single web site or provider. Advertisement companies aggregate every piece of information they get from hundreds of web sites only for the purpose of giving the user precise recommendations to articles. Using a browser that happily gives away all information about the user can be very dangerous and affects the user's privacy massively. Also cookies are in suspicion to be a problem for the user's privacy. Recently there were some changes in browsers and which cookies web sites have access to. See also Section 4.9 for further issues.

[6] http://www.google.com/analytics/

3.3.1 Avoid Getting Tracked

All of the above can be used against the intention of the user. So the user should take actions against being spied on. There are several browser addons (sometimes also called plugins) that can be installed to control the code of a web page. There are plugins are available for each browser type. The bigger projects are available for all commonly used browsers like Firefox, Internet Explorer, Chrome, Opera and Safari. Also mobile browsers are supported as there is a big movement towards mobile devices like mobile phones or tablets. The more successful addons are:

- Ghostery[7]
- NoScript[8]
- Request Policy[9]
- Adblock Plus[10]

Shortly summarized, these addons control the content of the websites you are loading. Most of them block unwanted requests to known tracking sites like the above mentioned Google Analytics. Generally, JavaScript requests to an address that is not within the domain of the web page are suspicious. Most addons block these requests and they typically have a list of domains they block generally. To block advertisements is also a related task, as advertisements are embedded in a similar way. Normally the user has control of the blocking feature by the possibility of adding whitelists and blacklists and he sees an icon when pages got blocked and has a way to unblock it directly. Furthermore it is to mention, that the user cannot prevent being tracked entirely. Web site owners always have many possibilities (e.g. logfiles) and big companies aggregate so many service offers that you can hardly slip through. Also cookies are not bad themselves, but their use can be exploited when separated services are connected together to exchange their knowledge about a certain user or computer.

3.3.2 Firewalls and Proxies

Another technical way to prevent tracking is prohibiting or mangling network traffic. Firewalls installed on a device or on a network node like a router can deny traffic being sent to certain domains or companies. This is a good way to ensure that no device behind this firewall can have access to a certain tracker. Firewalls are complicated to manage and are difficult to adapt to the personal need. Proxy servers[11] have the ability to investigate the (HTTP) traffic and can mangle it or block it entirely. Proxies can have a more sophisticated filter set and mangling rules than firewalls as they operate on a higher network layer. With proxies the management of

[7] https://www.ghostery.com/
[8] https://noscript.net/
[9] https://www.requestpolicy.com/
[10] https://adblockplus.org/
[11] http://en.wikipedia.org/wiki/Proxy_server

the rules can be made easier, but sadly it is still a problem to gather an appropriate rule set as there are so many companies and URLs and addresses may change over time and need to be looked after. So the whole idea of firewalls and proxies are mostly not used in this way except to block whole services like Facebook for the employees of a company.

Additionally both can be used to protect a local area network from services outside. So you have to trust whoever maintains these servers as all your network traffic runs through these nodes. Remote proxies can also be used for that, which is not recommended, as there would arise many security issues. This can also be used to protect oneself against computer viruses and malware. Another possibility is to use domain name servers (DNS[12]) that filter out malicious domain names and redirect them to empty pages. A well known service of this type is called OpenDNS[13]. This is also to be used very carefully as the remote system controls a central part of your communication system and may redirect not only malicious services. Actually the very same technique is also used for hijacking credit card information and other frauds. The technical idea behind these services is good, but you should only rely on them if you can totally trust them as your systems are totally depended on them. In case of misuse there is absolutely no way to find out that you were misdirected. Furthermore the operator of such service can see all addresses you are accessing from your browsers. This is normally only to be discovered by your Internet service provider (ISP), usually a company you have a contract with.

3.3.3 Geo Location and Devices

When connecting to the Internet, the your provider assigns your computer or router an available and unique IP[14] address. As every provider is registered by the Internet Assigned Numbers Authority (IANA[15]) and has certain IP ranges available for assigning, it is an easy task to query an IP address to find it's provider. Even providers have to segment their networks in smaller parts geographically. Therefore IP addresses can normally be resolved down to country/state level and most times even down to city level. The geolocation[16] then are longitude and latitude coordinates on the global map. There are online services like "whois"[17] that can directly tell the user's location for a given IP address. Devices can be identified and located not only by their IP but on a variety of different device dependent items:

- Internet Protocol (IP) address
- MAC address
- RFID
- hardware embedded article/production number, embedded software number (e.g. UUID, Exif/IPTC/XMP)

[12] http://en.wikipedia.org/wiki/Domain_Name_System
[13] https://www.opendns.com/
[14] http://en.wikipedia.org/wiki/IP_address
[15] http://en.wikipedia.org/wiki/Internet_Assigned_Numbers_Authority
[16] http://en.wikipedia.org/wiki/Geolocation
[17] http://whois.arin.net

- Wi-Fi positioning system
- device fingerprint (e.g. operating system level)
- canvas fingerprint (browsers)

Investigating requests from the same IP can lead to the conclusion, that certain devices belong to the same person. The data mined for these different devices can then be connected to each other. On the other hand, IPs can be linked together, when the same user name is logged in simultaneously from two different IPs at the same time. So the more the user connects to services, the easier it gets to be identified. Only once logging in into a commercial site, the site can fingerprint the user's device and permanently add it to his/her personal profile. From then on, the user can be easily traced in all future, even though the user is not logged in anymore. Even across browser changes or updates. The data can be correlated even to match a certain time of the day, where the user normally is at work or at home. The more connected these services are, the more dangerous it is for the user that the data can be used against him. Fingerprinting the browser (canvas fingerprinting) is a method that gathers all kind of available browser parameters to put them together into a unique ID that is surprisingly distinguishable from other browsers. If the measured ID changes because the user changed his system or installed a new browser, the new profile can be linked to the old one by only having one service accessed that distinguishes him/her from other users. If a company controls enough important services this will happen very early. Read Subsection 3.3.4 and Subsection 3.3.5 for further explanations. Finally it has to be pointed out, that these methods can be used for benefit as well as against a user. The fact, that each user gives away much more data than he is aware of remains.

3.3.4 Commercial Interests and Aggregated Data

Whenever you obtain data from a web server, the server knows what data you accesses. It might also know your login credentials or cookies the site stored on your computer for identification. It knows your IP-address which can be geographically resolved quite well with tools like GeoIP-Tool[18] within city range. Refer to Subsection 3.3.3 to get more information about that issue. This is not further surprising, but even these data most people are not aware about. The real problem arises, when a web page uses external links for commercials or statistical reasons (e.g. Google Analytics). These pages can collect far more data than the original web site, as they have your data and know the original site. These connection are often caused by advertising images or (interactive) videos which is then visible to the user. But it can also be a hidden frame or pixel non visible just for the analysis of your browsing behavior. Figure 3.1 shows a network graph, that was built up when visiting eight web sites. The visited sites were:

- http://www.nytimes.com
- http://www.latimes.com/

[18] http://www.geoiptool.com/

- http://orf.at/
- http://derstandard.at/
- http://diepresse.com/
- http://www.krone.at/
- http://www.wetter.at/
- http://www.sms.at/

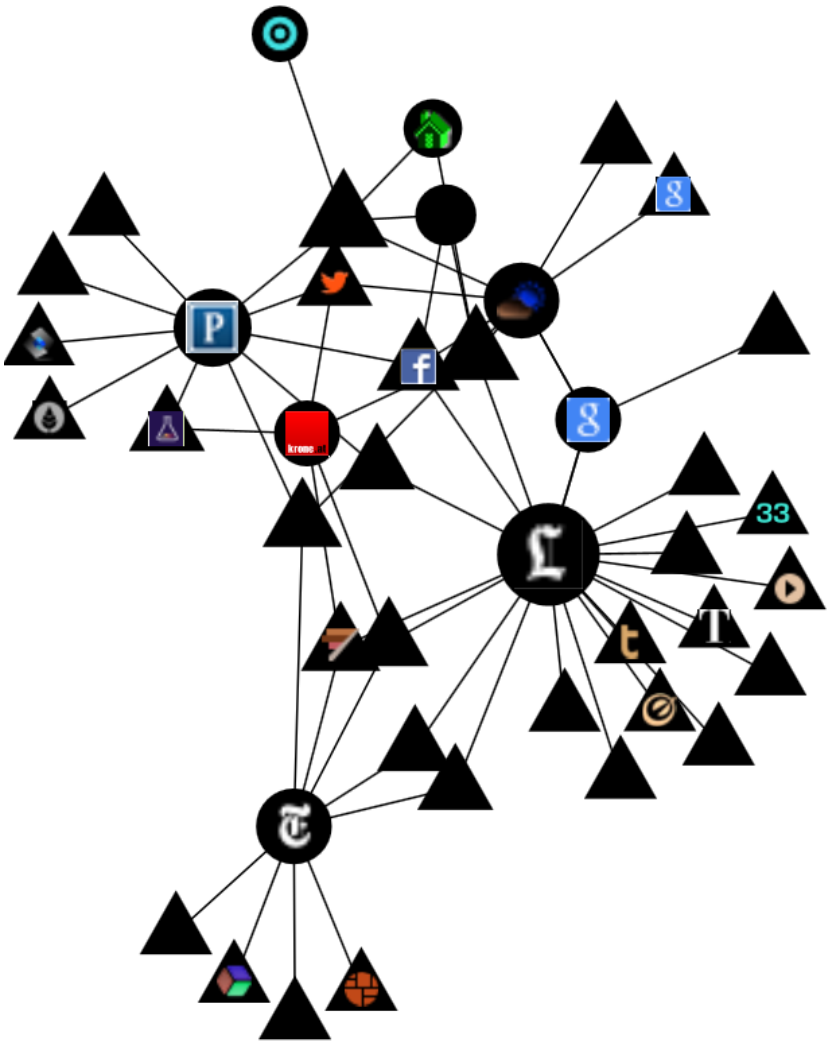

Figure 3.1: Graph of data connections to external web sites built up by the browser from visiting only eight different web pages. It shows 42 servers that were visited. Some servers like Twitter, Facebook and some advertisement servers are referred to by more than one sites which might expose browsing information.

The graph was produced with a Firefox add-on named Lightbeam[19]. It shows mainly two different aspects: First is, that some sites have really many external connections. For example

[19] http://www.mozilla.org/en/lightbeam/

the LA-Times has 19 references outside. DiePresse has 12 references outside. Even though the page was just opened and nothing had been done there at all. These pages have the circle symbol behind. Second is, that some sites use the same third party sites. For example Facebook occurs from 5 pages although it was not even opened once. Google has 6 incoming links although it was not directly opened. The third party sites, that are not directly accessed but referenced from other sites are denoted by the triangle symbol. So there are many sites, that know about us accessing the specific pages right now. In many cases third party sites even know the exact article you open from a news web site. So these third party sites are also able to estimate your political flavor or other personal details. They will use this information for advertising and if they can make money out of it, they probably will. See also Subsection 3.3.1 for the issue of user tracking.

3.3.5 Big Data

"Data was no longer regarded as static or stale, whose usefulness was finished once the purpose of which it was collected was achieved, such as after the plane landed or in Google's case, once a search query had been processed. Rather, data became a raw material of business, a vital economic input, used to create a new form of economic value. In fact, with the right mindset, data can be cleverly reused to become a fountain of innovation and new services. The data can reveal secrets to those with the humility, the willingness, and the tools to listen." [29] In his book, Viktor Mayer-Schöneberger describes big data as collecting a broad variety of data as a method to find correlation. Every action gives us a number of signals that might not even be directly related to the action itself. Combining the signals can lead to statistical tendencies which lead to better predictions. So data has a hidden value that can be exploited by who has the data available.

As an example he mentions a system for vehicles that can predict car accidents by evaluating seat sensors. An accident could be completely avoided, if for example, signs for tiredness can be successfully evaluated, or an emergency call could be made. On first sight, the system seems to be a good idea. But when gathering more and more data to increase the accuracy of prediction, one may run into privacy issues of the concerned persons. Health care data is probably a good example for very personal data. Implementing these kind of predictions in a general store might reveal when a customer gets pregnant even before she buys pregnancy test but as an inference of the bought food. Obviously the shop owner now possesses very personal data of a customer. It is now an ethical question, how he may use this data. There are no regulations so far that prevents him from using this data for his advantage. Also there seem to be a variety of privacy levels from the mentioned shopping analysis to e.g. health or medical care.

The gathering of personal information has already begun and will increase within the next years. The usage of the data is widely unregulated as well as the right of ownership. Fact is, that the data can only be used by who has the data available. Who mines the data, who stores the data and company, businesses or states might sell/buy them for their profit. An important

notice is, that correlation does not necessarily implicate causality. An example here would be in the field of law enforcement: What should a society do, when it predicts, that 90% of the released prisoners are predicted to recommit crime again? Is there a benefit for society to keep all of them imprisoned only because of the likelihood? Perfect predictions cannot be made, because the used data might be from last year. The handling needs to be regulated to protect human volition, free will, responsibility and moral. Transparency will be hard to implement, as the prediction might be based on hundreds of strong values, where every single value might be absolutely insignificant.

In connection with browsing data, the gathering of data is very easy. All browsing relevant data can be stored on servers and can be analyzed. Browser scripts can send additional data from the client to the servers. The kind of analysis of such huge amount of data sets can be very time and memory consuming. The process of collecting data in a big scale originally not knowing which information of the collected data might be relevant is also called big data. It is also not important from which source the data is collected. Large companies that control many services have hundreds of thousands browsing sessions every day and also may trade them. So this is the top scale of security issues we all have to deal with as long as governments do not restrict this kind of surveillance. The problem for the privacy of each Internet user is, that the aggregation of many small pieces of data can be put together to amazingly detailed picture of an individual. Also anonymized data is problematic, because when correlating many (sometimes over hundred) properties, the correlation of the data might still identify one single person. For all these reasons, users should be very concerned about their privacy and might think of methods to protect themselves while there are no strict rules for companies (and governments) to protect them.

"Big data erodes privacy and threatens freedom. But big data also exacerbates a very old problem: relying on the numbers when they are fare more fallible than we think. [...] The quality of the underlying data can be poor. It can be biased. It can be mis-analyzed or used misleadingly. And even more damningly, data can fail to capture what it purports to quantify." [29]

3.4 Further Applications and Related Publications

I want to introduce some interesting works from other people that might lead further into the topic of dealing with click data. They basically have the same origin as my work but cover different aspects in using the data.

3.4.1 Giving Up on Navigating

An interesting work from researchers from Google (Aju Thalappillil Scaria, Rose Marie Philip, Robert West and Jure Leskovec) "The last click: why users give up information network

navigation"[36] analyzed data collected from Wikispeedia[20], which also take paths into consideration that users created by searching through articles on Wikipedia. They focus on the question, *why* the users give up while navigating and try to find out certain characteristics (e.g. out-degree, page rank, shortest path length to the target) of the pages that caused the giving up: "An important part of finding information online involves clicking from page to page until an information need is fully satisfied. This is a complex task that can easily be frustrating and force users to give up prematurely. An empirical analysis of what makes users abandon click-based navigation tasks is hard, since most passively collected browsing logs do not specify the exact target page that a user was trying to reach. We propose to overcome this problem by using data collected via Wikispeedia, a Wikipedia-based human-computation game, in which users are asked to navigate from a start page to an explicitly given target page (both Wikipedia articles) by only tracing hyperlinks between Wikipedia articles. Our contributions are two-fold. First, by analyzing the differences between successful and abandoned navigation paths, we aim to understand what types of behavior are indicative of users giving up their navigation task. We also investigate how users make use of back clicks during their navigation. We find that users prefer backtracking to high-degree nodes that serve as landmarks and hubs for exploring the network of pages. Second, based on our analysis, we build statistical models for predicting whether a user will finish or abandon a navigation task, and if the next action will be a back click. Being able to predict these events is important as it can potentially help us design more human-friendly browsing interfaces and retain users who would otherwise have given up navigating a website."

3.4.2 Click Predicting

Georges E. Dupret and Benjamin Piwowarski state in their proceeding "A user browsing model to predict search engine click data from past observations"[16], that click logs are not a good way to determine the relevance of documents in the result list, because the user might have already seen them before (or after). So there will be a bias which will rate important documents lower than they should be rated: "Search engine click logs provide an invaluable source of relevance information but this information is biased because we ignore which documents from the result list the users have actually seen before and after they clicked. Otherwise, we could estimate document relevance by simple counting. In this paper, we propose a set of assumptions on user browsing behavior that allows the estimation of the probability that a document is seen, thereby providing an unbiased estimate of document relevance. To train, test and compare our model to the best alternatives described in the Literature, we gather a large set of real data and proceed to an extensive cross-validation experiment. Our solution outperforms very significantly all previous models. As a side effect, we gain insight into the browsing behavior of users and we can compare it to the conclusions of an eye-tracking experiments by Joachims et al. In particular, our findings confirm that a user almost always see the document directly

[20] http://www.wikispeedia.net/

after a clicked document. They also explain why documents situated just after a very relevant document are clicked more often."

The described problem is a general problem when evaluating log files. This is also the case when we track clicking links directly. When making assumptions, one always has to know circumstances, why the user was not clicking on a certain item. Please read Subsection 3.1.1 for further information. In our case, for the game task, this is only partly relevant, as it is not important if links were already clicked or not. But still there might be other reasons why people do not click on certain links, for example when the link is not recognized by the user.

3.4.3 Probability Estimation for Future Clicks

Yujiu Yang, Xinyi Shu and Wenhuang Liu propose a model in their proceeding "A Probability Click Tracking Model Analysis of Web Search Results"[45] that determines future clicks out of historical click data. They check their model on a real world data set from a commercial search engine: "User click behaviors reflect his preference in Web search processing objectively, and it is very important to give a proper interpretation of user click for improving search results. Previous click models explore the relationship between user examines and latent clicks web document obtained by search result page via multiple-click model, such as the independent click model(ICM) or the dependent click model(DCM), which the examining-next probability only depends on the current click. However, user examination on a search result page is a continuous and relevant procedure. In this paper, we attempt to explore the historical clicked data using a probability click tracking model(PCTM). In our approach, the examine-next probability is decided by the click variables of each clicked result. We evaluate the proposed model on a real-world data set obtained from a commercial search engine. The experiment results illustrate that PCTM can achieve the competitive performance compared with the existing click models under standard metrics."

The probabilities would also help us to find out the next item on the path to the finishing node. They could be seen as additional information for the graph on the links between the items. It is to be questioned if the additional property can improve the applicability of the system or not.

3.4.4 Random Walk on the Click Path

Nick Craswell and Martin Szummer also consider search engines in their proceeding "Random walks on the click graph"[8]. They did a random walk on a large click log and produced a probabilistic ranking of documents for a given query. They also regard previously clicked links and experimented with back- and forward random walks:

"Search engines can record which documents were clicked for which query, and use these query-document pairs as "soft" relevance judgments. However, compared to the true judg-

ments, click logs give noisy and sparse relevance information. We apply a Markov random walk model to a large click log, producing a probabilistic ranking of documents for a given query. A key advantage of the model is its ability to retrieve relevant documents that have not yet been clicked for that query and rank those effectively. We conduct experiments on click logs from image search, comparing our ("backward") random walk model to a different ("forward") random walk, varying parameters such as walk length and self-transition probability. The most effective combination is a long backward walk with high self-transition probability."

This is interesting for my work as I can also provide a graph of connected nodes: On the one hand probably have it from the crawling process (see Section 4.6 for details) or I can build it from your received click data. Both ways a walk on the graph to predict future paths would be interesting to undertake.

3.4.5 Enhance Navigability with Tax Taxonomies

In the proceeding "Enhancing the navigability of social tagging systems with tag taxonomies"[39] from Christoph Trattner, Christian Körner and Denis Helic shed a light on tagging systems and why they are not that comfortable as we would like to have them. Tagging definitely enhances the probability of finding information. The Click Path software could easily be used on a system that connects the tags of items instead of the categories. But does the navigability become easier? Probably not, as the authors state that navigating in social tagging systems are hard to navigate. In their work they introduce tag-resource taxonomies to enhance navigation within tagging systems:

"Tagging introduces an intuitive and easy method to organize resources in information systems. Although tags exhibit useful properties for e.g. personal organization of information, recent research has shown that the navigability of social tagging systems leaves much to be desired. When browsing social tagging systems users often have to navigate through huge lists of potential results before arriving at the desired resource. Thus, from a user point of view tagging systems are typically hard to navigate. To overcome this issue, we present in this paper a novel approach to supporting navigation in social tagging systems. We introduce tag-resource taxonomies that aim to support efficient navigation of tagging systems. To that end, we introduce an algorithm for the generation of these hierarchical structures. We evaluate the proposed algorithm and hierarchies from a theoretical, semantic and empirical point of view. With these evaluations we are able to show the high performance and usefulness of the proposed hierarchies."

The advance from categories to tagging is generally interesting for this kind of work. It helps to find content and it helps to connect items together. Although tagging might not be superior to categories for many reasons it is eligible as additional tool to be considered and used.

3.4.6 Eye Tracking

Probably interesting further work would be to track users not only by recording their clicks, but also recording what they were looking at. On the one hand, eye tracking today is quite easy to do, but on the other hand evaluating the data is complicated. There are some interesting papers on this, although they concentrate much on the issue of where search results are to be placed that they are convenient to be found by the user. Anyway it is an essential part of the user's navigation process and needs to be considered.

Mari-Carmen Marcos, David F. Nettleton and Diego Saez-Trumper in "A user study of web search session behaviour using eye tracking data"[28] try to find different behavior patterns that lead to success or failure: "In this paper we present and empirically evaluate a user study using a web search log and eye tracking to measure user behaviour during a query session, that is, a sequence of user queries, results page views and content page views, in order to find a specific piece of information. We evaluate different tasks, in terms of those who found the correct information, and in terms of the query session sequence itself, ordered by SERP (Search Engine Result Page), number and return visits to the results page for the same query. From this we are able to identify a number of different behaviour patterns for successful and unsuccessful users, and different trends in user activity during the query session. We find that a user behaves differently after the first query formulation, when we compare with the second formulation (both queries being for the same information item). The results can be used to improve the user experience in the query session, by recognizing when the user is displaying one of the patterns we have found to have a low success rate, and offering contextual help at that point. The results may also contribute to improving the design of the results page."

Also Hilal Al Maqbali, Falk Scholer, James A. Thom and Mingfang Wu deal with the same subject in their proceeding "Using eye tracking for evaluating web search interfaces"[2].

Also to mention is the paper from Christoph Trattner, Philipp Singer, Denis Helic and Markus Strohmaier named "Exploring the differences and similarities between hierarchical decentralized search and human navigation in information networks"[40]. It is the group of researchers around the supervisor of my thesis that investigate the problem of decentralized search because the user only has certain local knowledge and do not know the whole network graph: "Decentralized search in networks is an activity that is often performed in online tasks. It refers to situations where a user has no global knowledge of a network's topology, but only local knowledge. On Wikipedia for instance, humans typically have local knowledge of the links emanating from a given Wikipedia article, but no global knowledge of the entire Wikipedia graph. This makes the task of navigation to a target Wikipedia article from a given starting article an interesting problem for both humans and algorithms. As we know from previous studies, people can have very efficient decentralized search procedures that find shortest paths in many cases, using intuitions about a given network. These intuitions can be modeled as hierarchical background knowledge that people access to approximate a networks' topology. In this paper, we explore the differences and similarities between decentralized search that utilizes hierarchical background knowledge

and actual human navigation in information networks. For that purpose we perform a large scale study on the Wikipedia information network with over 500,000 users and 1,500,000 click trails. As our results reveal, a decentralized search procedure based on hierarchies created directly from the link structure of the information network simulates human navigational behavior better than simulations based on hierarchies that are created from external knowledge."

3.5 Related Software Projects

There are projects that want to retrieve the user's navigation data. A short overview about the methods and how they are different to my approach follows.

3.5.1 Wiki Paths

WikiPaths[21] is a hyperlink scavenger hunt game played on Wikipedia.org in which players try to find the shortest path between two, seemingly unconnected articles. The game runs as a Greasemonkey Firefox Add-on, allowing users to start a new game at their leisure as they navigate Wikipedia. This is similar to my project as users are encouraged to play a game to gather information about their behavior. The difference lies in the technical realization, as the user needs to install a piece of software. My project adapts the destination web site before it is presented to the user. See Subsection 3.1.4 for further information about this technique. The Greasmonkey script certainly would perform better in a large scale scenario, as the processing is fully done by the host computer and not on the server. This should be considered if the program would be used for a bigger group of testers at the same time.

3.5.2 The Wiki Game

The Wiki Game is a fancy game[22] that can be played via web browser or an iPhone application. On the site one can choose between several modes of the game. Start and target pages are chosen by random on Wikipedia articles and everybody starts at the same position. They also have a time-counter to increase the challenge. The current page is displayed within a frame. Navigation was very slow as the pages are changed on loading time and the servers might have been under heavy use. The design is quite well adapted for children what might be a very good idea especially when trying to find out more about the browsing behavior differences between children and adults.

[21] http://playwikipaths.com/
[22] http://thewikigame.com/

3.5.3 Wikispeedia

The web site "Wikispeedia"[23] runs a similar game on Wikipedia articles that are taken from an archive version of Wikipedia 2007. It offers several games where users can join finding the target article. The links to categories are highlighted, so that it is even easier to get the possible outgoing links from a certain page. The game has no time limit and you see no competitors that fight directly against you. The navigation is quite fast, as the pages are already prepared for this game. It generally looks very clean and intuitive, but often links are not marked so that continuing browsing gets sometimes interrupted. That might be because the copy of the data is quite old and probably got improved meanwhile. When you succeed the game, you can note down how difficult it was for you and you are presented the number of links it took you. This implementation of the game has the advantage, that the game takes place on own hosted sites where the code can be prepared instead of interacting on generic pages like in my project.

[23] http://cs.mcgill.ca/~rwest/wikispeedia/

4 Web Development

The available methods for retrieving click data were described in Chapter 3. In this chapter, I will go into more detail which technologies are available for web development. After that, the technologies that were used within the software project are explained in particular and why certain technologies are more suitable than others. The embedding of the parts (where it all comes together), the considerations behind the choices and the importance of the corresponding software parts are also discussed.

4.1 Web Application Programming

A web application framework (WAF[1]) is a piece of software, that deals with dynamic web pages, web applications or web services. It runs on a web server and handles incoming requests that usually come from browsers. Web application frameworks are designed to deliver fast results but still force the developers to use pre-built object oriented classes, reuse templates and often implement the model-view-controller architectural pattern for separating the data model from logic and user interface (read Subsection 4.5.3 for further details). The persistence layer handles the data that can be stored inside a web application. As many applications rely on versatile data, connections to commonly used database management systems are usually integrated in the frameworks which is very convenient for a developer. So the persistence layer can be seen as a mapping from database objects to accessible data inside the application. High performance frameworks also provide pooled database connections to increase datatransfer from the database to the application. Important properties of a web application framework besides the chosen programming language and persistence layer are properties like: Availability of caching algorithms, performance limits, scalability, security and authorization possibilities, how much server load they cause and how much memory they use.

4.1.1 ASP, C# and .NET

Microsoft's Active Server Pages (ASP) provide dynamic web pages and web applications using the their .NET[2] framework and C# as programming language. It is the preferred choice of most web programmers behind PHP. It runs on IIS (Internet Information Server[3]) under Windows

[1] http://en.wikipedia.org/wiki/Web_application_framework
[2] http://en.wikipedia.org/wiki/.NET_Framework
[3] http://de.wikipedia.org/wiki/Microsoft_Internet_Information_Services

operating systems which needs to be licensed in order to use it. It includes the Framework Class Library (FCL) which provides user interface, data access, data connectivity, cryptography and network communication. The framework comes with an own convenient editor for the Windows platform called "Visual Studio".

4.1.2 C and C++

Although C and C++ are commonly used languages, they are not that often used in web development. A downside of this technology seems to be the lack of library support. There are only a few projects supporting these languages. Wt[4] is one of it. Wt has an integrated application server and auto detects the browser's capabilities to provide proper rendering at the client side and avoid browser bugs. Positive facts are the slim and memory saving architecture and a good performance, which can be a very important combination on embedded systems. Due to the close to hardware language, it uses little server resources. It comes with Sqlite, PostgreSQL, MySql and Oracle support. Another framework is CppCMS[5] which is designed for extremely high loads. It also supports all common databases like Sqlite3, PostgreSQL, MySql and generic Odbc.

4.1.3 Java

Java provides a number of frameworks like Spring and Google Web Toolkit. Java is very widespread and therefore it has probably the biggest community that share code over the Internet. This can be very helpful and is a big advantage over other languages. Java itself is not entirely an interpreted language. A just-in-time compiler translates the Java bytecode on executing. This has many performance advantages but is not as fast as C or C++. Java has advantages in profiling and debugging. There exist many editors like Eclipse and Netbeans which are very comfortable to use. Spring[6] is open source framework and aims to simplify development of Java Enterprise Edition. It focuses on business logic and provides solutions for large scale business applications. The framework supports all common databases that have a JDBC implementation available. Anyway, many developer prefer to use Java EE and Apache Struts over Spring. Google Web Toolkit[7] is a framework many Google services use internally. It has a special feature, that client side JavaScript code can be implemented directly in Java. This is possible because of the JavaScript Native Interface (JSNI) that compiles the Java code into JavaScript. This has the advantage, that the entire application can be developed in Java. It has an interface for remote procedure calls as well as widgets for developing graphical user interfaces similar to Swing. Hibernate[8] is a persistence framework that can store Java attributes

[4] http://www.webtoolkit.eu/wt/
[5] http://cppcms.com/wikipp/en/page/main
[6] https://spring.io/
[7] http://www.gwtproject.org/
[8] http://en.wikipedia.org/wiki/Hibernate_(Java)

and methods directly in relational databases. It can be used in addition to other frameworks for simplifying the handling of database access. It lets database objects be treated like Java objects. This can be very convenient but is not always an advantage, because special database operations that include many different data records cannot be handled this way.

4.1.4 Perl

Perl[9] originally developed from C, awk and other Unix shell commands. It has advantages in text processing like parsing text with regular expressions. Perl is a popular language for system administrators because short programs are easy to write and one-liners can be entered directly into the command line. Many people like Perl as CGI scripts on servers. However also large projects are written in Perl: Bugzilla, Request Tracker, Craigslist, IMDb, DuckDuckGo and Slashdot. Known frameworks are Catalyst[10], Dancer[11] and Mojolicious[12]. The projects support model-view-controller and internationalization and connect to SQL databases using the DBI module.

4.1.5 PHP

PHP is commonly used in web development. The language itself is quite simple and easy to learn. The downside is, that the source code of larger projects tends to get confusing. Most small pages are written in plain PHP without any complicated framework. Continue reading from the language itself in Section 4.2. However there exist many frameworks that are based on PHP. Cake PHP, CodeIgniter, Drupal, FuelPHP, Joomla, Solodev CMS, Yii, Zend Framework and Zikula. All of them implement a model-view-controller framework and have database interfaces and caching methods. They also implement internationalization which is an important feature in web development. PHP has the advantage of a huge online community and many good editors. Profiling and debugging are not convenient although some frameworks come with debugging tools. The performance is average and the scale is limited. In exchange, it features fast development times.

4.1.6 Python and Django

"Python is a programming language that was created by Guido van Rossum in the early 1990s. Django is a free and open source web application framework, written in Python. Its initial release was in 2005. Benefits of using Python and Django are, that a developer has more control in choosing layout and configuration options and it is transparent and minimalistic, but all things need to explicitly included. Compared to Ruby on Rails, many find updates to be less painful

[9] http://en.wikipedia.org/wiki/Perl
[10] http://www.catalystframework.org/
[11] http://www.perldancer.org/
[12] http://mojolicio.us/

and less frequent. Python has a clean syntax that resembles English. It is stronger in areas like data manipulation, analytics, system administration, and scientific programming. This is why it is often used in academic and science world."[26] It runs on web servers like Apache, NGINX, Lighttpd or Hiawatha as a Python module. Database access is ensued explicitly and can handle Sqlite, PostgreSQL, MySql and Oracle. Limited access to Microsoft SQL Server also exists. Many big and well-known sites use Django including Pinterest[13], Instagram[14], Mozilla[15] and The Washington Times[16]. Django can handle a decent amount of concurrent requests and simultaneous logged in users. So this seems to be one of the top range professional frameworks. Still it remains an interpreted language.

4.1.7 Ruby on Rails

"Ruby is a programming language that was created in the mid-1990s in Japan by Yukihiro Matsumoto. Rails is an open source web app framework, written in Ruby, that appeared in 2005. The main selling point of Rails is the philosophy called convention over configuration (CoC) - meaning it is a framework that has a structured layout with defaults (read: it's easy to get up and running - sort of like a web app starter kit). Ruby syntax is ideal for those who prefer pattern-matching characters because it uses differing characters as analogues to keywords. It's built for speed and adaptation and has strong support community both online and offline."[26] Database access ensued implicitly over introspection and all commonly used database systems are supported. It has a considerable online community, probably not in the amount of Java or PHP, but much better than the smaller projects. Editors and tools for Ruby are not that common and succumb in debugging and profiling options. It has very fast development time because it is an interpreted language. In terms of speed and performance it ranges in the middle of the field and cannot compete with compiled languages. The framework itself is quite large and lost a little of it's importance over the years since 2010. It also runs on an Apache web server as well as Lighttpd with FastCGI or any other CGI engine.

4.2 Applying Web Service Technologies

Web applications not always are the first choice. They come with a huge overhead of libraries that are not used and sometimes with an inherent logic or programming layer that has to be learned first. So I will continue focusing on basic programming without framework and will continue investigating what needs to be done by the server:

The displayed content needs to be generated dynamically and for more users at the same time. The performance issues lead me to compare the different technologies. Commonly used

[13] https://www.pinterest.com/
[14] https://instagram.com/
[15] https://www.mozilla.org/en-US/
[16] http://www.washingtontimes.com/

web servers run Apache[17] with CGI[18] (Common Gateway Interface) or PHP. For Windows it is mostly the Microsoft Internet Information Server with Microsoft Active Server Pages[19] (ASP). As I developed parts of the project on a Linux system only the Apache solutions came into question. As performance and scalability are important points, the different solutions needed to be investigated. As Emmanuel Cecchet et al.[7] pointed out in their performance study, Java has some essential advantages over PHP. For professional high-end web sites one should consider Enterprise Java Beans (EJB[20]) as it is top of the range in performance and maintainability. PHP instead is more a hypertext preprocessor and runs as standard Apache web server, where PHP code can be directly embedded into the HTML code. See Listing 4.1 for the PHP code. The web server then invokes the PHP module and executes the script. Requests to the database are explicit and are performed using an ad hoc interface. Usually the program code itself is relatively short.

In contrast to that concept, Java servlets run separated from the web server. A Java servlet is a Java class that can be dynamically loaded by the servlet engine and runs in a Java Virtual Machine (JVM[21]). After the initial load, the servlet engine invokes the servlet using local calls, but since the JVM is a separate process from the web server, inter process communication (IPC[22]) takes place for each request. Servlets access the database explicitly, using the standard JDBC interface, which is supported by all major databases [7]. For professional and large scale use, Enterprise Java Beans (EJB) abstracts the application business logic from the underlying middle ware. An EJB server provides a number of services such as database access (JDBC[23]), transactions (JTA[24]), messaging (JMS[25]), naming (JNDI) and management support (JMX[26]). The EJB server manages one or more EJB containers. The container is responsible for providing component pooling and life cycle management, client session management, database connection pooling, persistence, transaction management, authentication, and access control [7]. Although EJB seems to be the top quality choice, it is a very big framework for serving a relatively small amount of data in this case. The Listings 4.1, 4.2 and 4.3 illustrate the difference in the code between these languages.

[17] http://www.apache.org/
[18] http://en.wikipedia.org/wiki/Common_Gateway_Interface
[19] http://www.asp.net/
[20] http://www.oracle.com/technetwork/java/javaee/ejb/index.html
[21] http://en.wikipedia.org/wiki/Java_virtual_machine
[22] http://en.wikipedia.org/wiki/Inter-process_communication
[23] http://www.oracle.com/technetwork/java/javase/jdbc/index.html
[24] http://www.oracle.com/technetwork/java/javaee/jta/index.html
[25] http://docs.oracle.com/javaee/6/tutorial/doc/bncdr.html
[26] http://www.oracle.com/technetwork/java/javase/tech/javamanagement-140525.html

```php
<?php
  header("Content-Type: text/html");
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE>Sample</TITLE>
 </HEAD>
 <BODY BGCOLOR="#FDF5E6">
  <H1>
<?php
    printf(getHeading());
?>
  </H1>
<?php
   printf(getContent());
?>
 </BODY>
</HTML>
```

Listing 4.1: This is an example of a PHP source code. It prints out a valid HTML page and references a function for the header and the content. It looks clear as the main format is still HTML and the code sections are between special tags.

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class SampleServlet extends HttpServlet {

  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
  throws ServletException, IOException {

    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    String docType =
      "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 " +
      "Transitional//EN\">\n";
    out.print(docType +
                "<HTML><HEAD>\n" +
                "<TITLE>Sample</TITLE></HEAD>\n" +
                "<BODY BGCOLOR=\"#FDF5E6\">\n" +
                "<H1>" + getHeading() + "</H1>\n" +
                getContent() + "</BODY></HTML>\n");
  }
}
```

Listing 4.2: An example for a Java Servlet source code. It is clear to see, that the programming language is Java and the HTML tags that should be printed out are uncomfortable to use. The strings are within quotes and sometimes need to be escaped.

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
2 <HTML>
3  <HEAD>
4   <TITLE>Sample</TITLE>
5  </HEAD>
6  <BODY BGCOLOR="#FDF5E6">
7   <H1>
8    <%= out.print(getHeading()); %>
9   </H1>
10  <%= out.println(getContent()); %>
11 </BODY>
12 </HTML>
```

Listing 4.3: Java Server Pages combine the two advantages from Java Servlets and PHP. The code is clear as the main format is HTML and the Java language commands are between special tags. This helps not to lose the focus for web development.

4.3 Java Servlets and JSP

Java Server Pages (JSP[27]) is another way of writing Java Servlets [19, Chapter 1.5]. JSP pages get translated into servlets, compiled and then run at request time. The code of JSP can be embedded in the HTML code (similar as in PHP), which makes it more easy to build web pages and to layout them. In JSP one has all advantages of Java, e.g. JDBC connection and threads, only to mention a few advantages that are relevant for this project. The differences between the syntax for the Java servlet code and for Java Server Pages is quite obvious, as the HTML output is not very convenient in Java servlets. It is also important to understand, that even if JSP technology and servlet technology are essentially equivalent in power, it is a question of convenience which one to take [19, Chapter 1.5]. Servlets are best for tasks oriented towards processing, whereas JSP is best for task oriented towards presentation. In many cases it is useful to use them together at the same time and benefit from the advantages of both technologies.

The decision was made for Java and its web application technology JSP. As servlet engine Apache Tomcat[28] was deployed, as it is the most popular used system and also was available on the Graz University of Technology server. As programming environment Netbeans[29] was used as it works together with its built in Apache Tomcat engine, is open source and available for free. Netbeans can also export WAR files. This is the format for Java web applications. Files of this type can be directly deployed into the Tomcat Web Application Manager and started there. A web application, as defined in the servlet specification, is a collection of servlets, Java Server Pages (JSP), HTML documents, images and other web resources like JavaScript files

[27] http://www.oracle.com/technetwork/java/javaee/jsp/index.html
[28] http://tomcat.apache.org/
[29] http://netbeans.org/

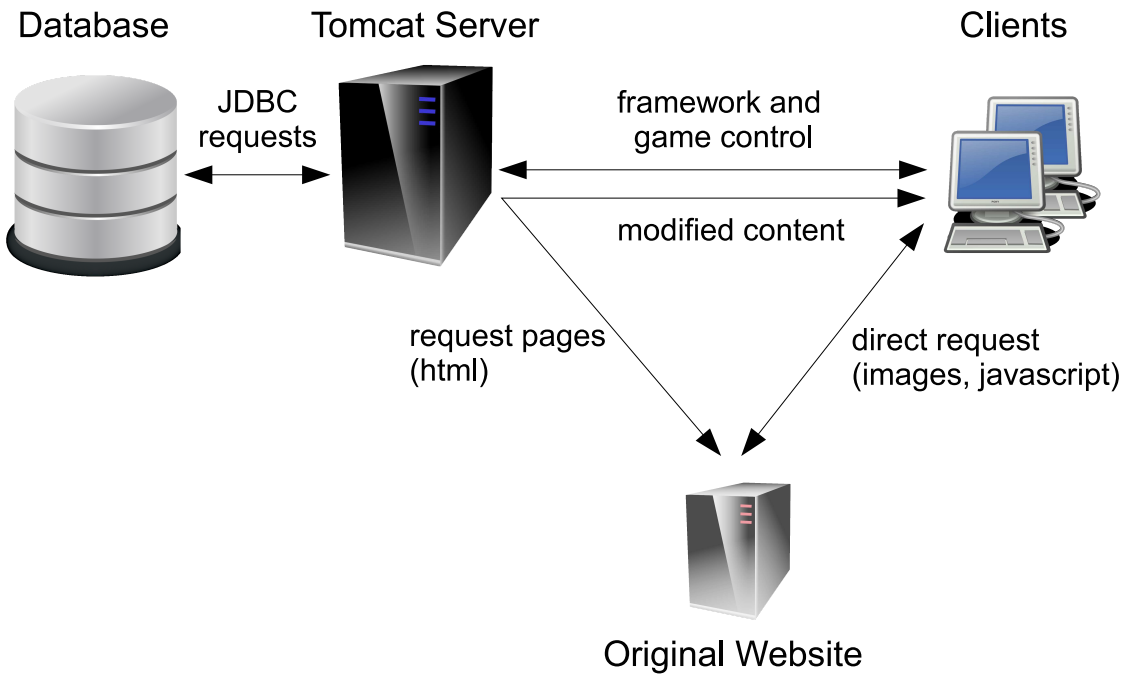

Figure 4.1: The connection of the technical parts of the software project. The client computer requests the game web site (or a content page) and contacts the Tomcat server. The server sends a request to the original web site and sends back a modified page to the client. Only pictures and embedded scripts can be requested directly from the original site. The Tomcat server takes control over the games and stores all relevant client activities inside the database.

that are set up in such a way that they can be deployed across any servlet-enabled web server [43]. The file is packed similar to the commonly used JAR[30] files and can be digitally signed as well.

The role of the Java application on the Tomcat server in the big picture is, to serve data to the clients and store the client's interactions inside the database. It is the link between all software parts. See Figure 4.1 for an overview of the different components. The clients on the right side are users that take part in the game. They receive a web site directly from our server and some JavaScript code that later runs inside their browser. They also get content from the original web site, which is modified by our server, as well as data directly from the original web site like pictures or scripts. The JavaScript running inside the client's browser always checks for game control commands and initiates the loading of a new game if the listener is told to do so. The Java application does the following tasks:

- serving the framework itself
- controlling the actual game task
- modifying the pages loaded by the users

[30] http://download.oracle.com/javase/6/docs/technotes/guides/jar/jar.html

- recording the clicks from the clients
- send game control parameters to clients
- storing the click data inside the database

Read more about the Java implementation in Section 5.7. Moreover, the servlet has to create the games the users take part in. Every game has a lifetime of around three minutes which is of course configurable. Best values for the game lifetime must still be found out, as it is sometimes quite hard to finish the given task and the lifetime depends highly on the underlying web page content. On the other hand, when there is time leftover, users get bored if they have to wait too long for the next game. When a game expires and there are users actively playing, a new game is created. The game itself only consists of the starting site and the destination site. These entries are generated randomly from the site sources table in the database.

4.4 JavaScript

In the opinion as Douglas Crockford, JavaScript[31] is the world's most famous and the world's most unpopular programming language [11]. As he describes in his book "JavaScript: The Good Parts" [9] that JavaScript was originally a fusion of Java's syntax and conventions with Self and Scheme, two languages which had no commercial success. It has properties from Self[32] like prototypal inheritance and dynamic objects, and from Scheme[33] lambda expressions and loose typing. It is also influenced by Perl[34] for using regular expression notations. First it was named LiveScript, then it was renamed to JavaScript. JavaScript nowadays is the language of an open system, but it is not open itself. It is a proprietary trademark which now belongs to Oracle[35]. So, unfortunately the language of the open systems is not open itself. JavaScript is a prototype-based, object-oriented scripting language that is dynamic, weakly typed and has first-class functions. It is also considered a functional programming language like Scheme mentioned above and OCaml[36] because it has closures and supports higher-order functions. All these properties are described below. JavaScript can also be called ECMAScript and is not interpreted Java. JavaScript is a different language [9].

4.4.1 Language Characteristics

Back around the year 2000 where the Internet bubble burst and the Internet failed being a document delivery system, JavaScript could have vanished. But within the next years, a paradigm change took place. The Internet changed from document retrieval to distributed

[31] http://en.wikipedia.org/wiki/JavaScript
[32] http://selflanguage.org/
[33] http://www.ccs.neu.edu/home/dorai/t-y-scheme/t-y-scheme.html
[34] http://www.perl.org/
[35] http://www.oracle.com/
[36] http://ocaml.org/

programming. This is why JavaScript survived and got a second chance with the invention of Ajax in the year 2005 [11]. In the view of Crockford, there are some major disadvantages within the language:

- *The name:* Because it suggests that it is somehow related to Java. But it is not. It is syntactically similar to Java but not more than Java to C. JavaScript was not developed at Sun Microsystems, the home of Java. It was developed at Netscape[37] which was then bought by AOL[38].
- *Syntax:* The syntax is C-like including curly braces which makes it look like an ordinary procedural language. This is misleading, as it has more in common with functional languages like Lisp or Scheme. It has arrays instead of lists and objects instead of property lists. Functions are first class. It has closures and you get lambda expressions without having to balance all those parentheses.
- *Bad DOM model:* Actually this is more a problem of the browser's DOM[39] (Document Object Model). In many Ajax libraries, the DOM model is corrected by the library which makes it possible to write good applications in JavaScript. It would be a good solution if this correction would already take place in the browser, but the libraries work well for now. A lot of performance is wasted inside the DOM interface as well, making JavaScript look slower than it could be.
- *Design errors and typecasting:* The "+" operator means both, addition and string concatenation. The *with* statement manipulates variables where you cannot tell, in which scope the variable is modified. The *typeof* operator has sometimes wrong or unhelpful results. The keys of *Array*s are hashed strings which does not perform well. It has too many "falsy" values like *false*, *null*, *undefined*, *NaN*. You can have block-less statements (without curly braces). Additionally there are some really bad issues in connection with typecasting. See the examples in Listing 4.4 and Listing 4.5, which is a compiled list from [37] and [9].
- *Number format:* This problem is based on the decimal system we are using. The commonly used floating point representation is not sufficient in the case when we want to add floating point numbers like this: $0.1 + 0.2 == 0.3$. When evaluating in JavaScript, this returns the value 0.30000000000000004, which is subsidiary for a general problem in nearly all programming languages we are using. As floating point representation is the only one available in JavaScript, it is quite hard to do e.g. financial calculations.
- *Lousy implementations, bad books:* Especially earlier implementations where quite buggy, which reflected badly on the language. Nearly all books are quite awful. They contain errors, poor examples and promote bad practices. Important features of the language are often explained poorly, or left out entirely.

[37] https://isp.netscape.com/
[38] https://www.aol.com/
[39] http://www.w3.org/DOM/

- *Standard:* The official specification for the language is published by ECMA. The specification is of extremely poor quality. It is difficult to read and very difficult to understand. This caused authors to be unable to use the standard document to improve their own understanding of the language. There are many versions, which leads to confusion.
- *Amateurs:* Most of the people writing in JavaScript are not programmers. They lack the training and discipline to write good programs. JavaScript has so much expressive power that they are anyway able to do useful things in it. The code looks familiar, so people tend to claim that they do not need to learn the language. This has given JavaScript a reputation of being strictly for the amateurs, that it is not suitable for professional programming. This is simply not the case.
- *Object oriented and global variables:* It has objects which can contain data and methods that act upon that data. Objects can contain other objects. It does not have classes, but it does have constructors which do what classes do, including acting as containers for class variables and methods. It does not have class-oriented inheritance, but it does have prototype-oriented inheritance. Objects cannot have private variables and private methods. All members are public. On the other hand global variables can be introduced. This is convenient, but also very dangerous when using other libraries that probably work with the same variable names.

```javascript
//javascript '+' as adding and string concatenation
7 + 7 + 7;     // = 21
7 + 7 + "7";   // = 147
"7" + 7 + 7;   // = 777
7 + "7";       // = 77

//javascript 'with' statement
some.variable.start = 1;
some.variable.end = 2;
with (some.variable) {  //using 'with', the
    start = 1;          //global variables 'start' and 'end'
    end = 2;            //are be affected too!
}

//automatic semicolon insertion
42; "hello!"      //valid
42\n"hello!"      //valid (where \n represents line break)
42 "hello!"       //invalid
if (x) { y() }    //valid

//troublesome slash token, problem with semicolon insertion
var i,s
s="here is a string"
i=0               //this creates no regexp which is evaluated by exec,
/[a-z]/g.exec(s)  //instead divides 0 by [a-z] and then by g.exec(s)

//silent errors
return
{
  ok: false
};                //returns undefined
return {
  ok: false
};                //returns object containing ok=false

//number calculations with floating point
0.1 + 0.2 !== 0.3       //it is 0.30000000000000004

//very often seen: two errors that cancel each other out
value = myObj[name];    //returns 'undefined' if object does not exist
if (value == null) {    //correct: if (value === undefined) {}
   alert(name + ' not found');
}

//no new scope for each code block
for (var i = 0; i < 10; i++) {
   ...
}
return i;             //returns 10
```

Listing 4.4: There are many confusions inside the JavaScript language. This is a list of examples showing what the JavaScript interpreter might resolve other ways than the programmer would expect it to be. The comments inside the code explain the results.

```
1  //typeof operator
2  var i = null; typeof(i);          //returns 'object', which is wrong
3  var a = new Array(); typeof(a)    //returns 'object', not helpful
4
5  //more type conversion confusions, equality operator
6  false == 0    //true, because 0 is typecasted to false
7  false === 0   //false, zero is 0, not false
8  '' == 0       //true
9  '' == '0'     //false
10 '0' == ''     //true
11 0 == '0'      //true
12 false == 'false'    //false
13 false == '0'        //true
14 false == undefined  //false
15 false == null       //false
16 null == undefined   //true
17 " \t\r\n " == 0     //true
18
19 //breaking statements
20 var x =             //allowed
21 "Hello World!";
22
23 var x = "Hello      //will not work
24 World!";            //solution would be a backslash
25
26 function myFunction(a) {
27     var
28     power = 10;
29     return a * power;
30 }                    //function will return a*power correctly
31
32 function myFunction(a) {
33     var
34     power = 10;
35     return           //here a ";" will be inserted
36     a * power;
37 }                    //function will return undefined
```

Listing 4.5: Especially type castings in connection with false, null and undefined values are irritating. Line breaks are supported but may not be used safely.

4.4.2 Good Things about JavaScript

If JavaScript was that bad, it would not have succeeded with Ajax as well. So it must have many good sides on it as well. As it is a functional language with dynamic objects, you can add properties to an object at any time. This is very convenient as you do not have to inherit from another object and add the property then. It has familiar syntax which it is quite easy to learn, so it can serve beginners as well as experienced programmers or scientists. No other language has this broad range of programmer skills. It is very effective in an event-driven application model, which is quite suitable for the tasks running on the browser side. The code size is significantly smaller than in other languages, but only if you use it the way it was meant to be.

It already showed that it performed much better than Java as a client programming language for Internet application. Also Douglas Crockford points out good features in the language: It has object inheritance, and functions can be members of objects. A good application is the *for .. in* statement, which mixes inherited functions with the desired data members [9]. As already mentioned, it has lambda statements, which are anonymous functions (or subroutines) defined, and possibly called, without being bound to an identifier. This anonymous functions are convenient to pass as an argument to a higher-order function and are ubiquitous in languages with first-class functions. This is an example for a lambda statement:

```
alert( (function(x) { return x*x; } )(10)); //returns 100
```

Prototypal inheritance means, that objects inherit from other objects. No classes are needed. The classical way is to define classes and create objects belonging to a class. Most programming languages do the classical way. In JavaScript objects are created directly from other existing objects which is called delegation. This can be a very powerful tool [10]. Another interesting thing are closures: "A variable reference binds to the lexically enclosing definition for that name that was active at the time the enclosing lambda form was evaluated. To explain the semantics in terms of the implementation, evaluating a lambda expression was said to produce a closure. This is a function value represented as an object that contains references to the current bindings for all the variables used inside the lambda expression but defined outside it. These are called the free variables. When this closure object, or function, is applied to arguments later, the variable bindings that had been captured in the closure are used to give meaning to the free variables appearing in the code. The term closure describes more than just the abstract language construct, it also describes its implementation." [17]

Let's sum up the advantages of the language:

- Broad range of programmer skills
- Objects inherit from objects
- Functions are members from objects
- Lambda statements
- Closures
- Dynamic objects
- Prototypal inheritance
- Loose typing
- First-class functions
- Global variables

These advantages helped JavaScript to succeed in many different environments. JavaScript is not only used in web browsers. It gained a huge variety of application areas:

- Applications (desktop applications like Adobe)

- Operating systems (widget systems used by Windows or Macintosh)
- Databases (CouchDB[40])
- Mobile devices and applications
- Consumer electronics (TV sets)
- Servers

Redesigning JavaScript now to get rid of the disadvantages is not really possible. You have to face huge compatibility problems with older software. Many programs written in this language rely on the bad habits and you cannot break the functionality of these programs by changing the design. You also cannot renew the syntax of an existing language, because older software will recognize it as syntax errors. In the case of JavaScript, there may be still many versions of old Internet browsers (like Internet Explorer 6) in use. These browsers are deprecated for years now, but there are still many obsolete copies in use. So the only way out is to use another language or stick to JavaScript as it is. A very interesting program called JSLint [12] checks JavaScript code for bad habits or common errors. It is available on a web site and the user can upload his code and see the recommendations of JSLint. This helps to find common mistakes and improves the quality of the JavaScript code.

4.4.3 Field of Application

This software part runs on the client's browser itself. When the framework is loaded, the code contains some scripts that handle load requests inside the content *<iframe>*. This is the part where the requested page is shown, but it has to be replaced by a modified version of the original page. The reason is, that all targets and links must be rewritten to our server's address. Otherwise we would not be able to capture the next click, as there would be a cross scripting error. See Subsection 5.7.1 for more details. A cross scripting error is a browser issue, that the browser's security policy does not allow a JavaScript loaded from another site manipulate the content of the *iframe*. See Section 4.9 and Subsection 4.9.2 for further and more detailed explanations to the cross scripting issue. This rewriting concerns all objects inside the loaded HTML that directly link to addresses with the anchor tag (*a* and *area* with the attribute *href*). Other objects, that refer to scripts or images (HTML tags: *img*, *script*, *link*, *form*, *embed* with the attribute *src* or *href*) must be rewritten as well, as relative path names lead to broken links and have to be replaced with absolute links. As the page loaded in the *iframe* is loaded from the address from our server, all targets must now have absolute URLs pointing to the resource on the original server.

[40] http://couchdb.apache.org/

```
1 <!-- original -->
2 <a href="http://en.wikipedia.org/wiki/Philosophy">
3 Philosophy, <img src="plato.jpg">
4 </a>
5
6 <!-- modified -->
7 <a href="content.jsp?url=http%3A//en.wikipedia.org/wiki/Philosophy" target="">
8 Philosophy, <img src="http://en.wikipedia.org/wiki/plato.jpg"></a>
9 </a>
10
11 <!-- modified with mouse click -->
12 <a href="content.jsp?url=http%3A//en.wikipedia.org/wiki/Philosophy" target=""
13    onmousedown="record.jsp?label=Philosophy&target=http%3A//en.wikipedia.org/
        wiki/Philosophy&rnd=12345">
14 Philosophy, <img src="http://en.wikipedia.org/wiki/plato.jpg"></a>
15 </a>
```

Listing 4.6: When finished loading the web page, a JavaScript modifies all HTML tags the way it is showed here. All URLs become links to our server. Click targets get events added so that our server can record on which target the user actually clicked.

Listing 4.6 shows the modifications made by the JavaScript to the original HTML page. Note, that the code presented here is a little simplified. For example, the *mousedown*-event actually does an Ajax (Section 4.5) request. The *rnd* parameter inside the Ajax request ensures that the request is reliably sent and not cached by the browser, as the information we want to send could then not be received by the server. The *target*-tag is cleared because we do not want to open any sites in a new window. The original URL is passed as parameter *url* to our servlet, so that it can be loaded and modified by our servlet to avoid the cross scripting issue.

4.5 Ajax and Web 2.0

The word Ajax[41] is an abbreviation for Asynchronous JavaScript and XML. It is not a real acronym and not a real piece of software. It is more a combined bundle of technologies that together provide new methods for web technologies. It is widely approved, that Jesse James Garett invented the word Ajax in his article "Ajax: A New Approach to Web Applications" [18]. The basic tools on which Ajax is based on are:

- HTML (or XHTML)
- Document Object Model (DOM, for representing data and content)
- JavaScript (for manipulation of DOM and interconnection)
- XMLHttpRequest-Object (for asynchronously retrieving data)
- REST, SOAP, JSON (get methods and for transferring method names and parameters)

[41] http://en.wikipedia.org/wiki/Ajax_(programming)

- CSS (page output formatting)
- XSLT (data transformation)

Using these methods in modern web sites is generally referred as "Web 2.0". There are also newer undertakings to standardize Ajax within the OpenAjax Alliance[42] in which most big Internet companies take part to put the future of Ajax on a wide basis. There are some important differences between traditional web sites and Ajax web sites that give the user advantages in his experience.

4.5.1 Traditional Web Requests

A traditional web site relies strictly on a request-response-paradigm. Every user action causes a new request to the server which reloads the complete web page and the related response takes some time and is dependent on the network connection. If the connection fails, it can cause a complete breach of the work flow. Figure 4.2 shows the time line of such a conversation between client and server. This was the commonly used technique in the days until the year 2005. An advantage is, that each page represents a stateless version of the page, which can be cached by browsers or proxies. When the content of the page is adapted to prior data, the advantages of caching do not apply any more, because the concept of caching is only designed for static contents or resources e.g. pictures. It is obvious, that the waiting time between the requests seriously interrupt the work flow on the client side.

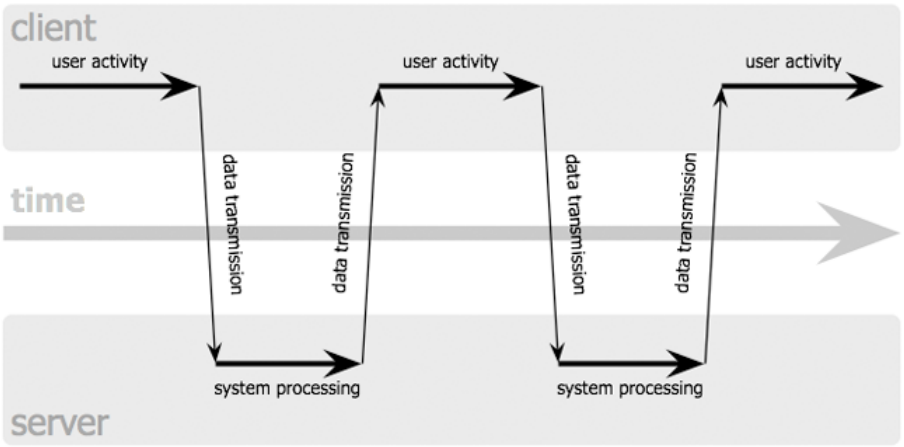

Figure 4.2: This figure is a timing diagram that shows the process flow of a traditional web application and the user's interactions. The user's activity makes the complete web page reload from the server with the new state. The figure is taken from [18]

[42] http://de.wikipedia.org/wiki/OpenAjax_Alliance

4.5.2 Asynchronous Ajax Web Requests

The paradigm shift to Ajax pages came quite quickly, as it increased the usability of web pages and also lowered the data transfer for the servers because more complex but unchanging content has not to be resent several times. Also the server load decreases, as some of the processing can take place on the client side after receiving the data from the server. A possible breach in the work flow mentioned above can be avoided by letting the Ajax part resend the request in order to succeed later. Jesse James Garrett further explains the advantages: Every user action that normally would generate an HTTP request takes the form of a JavaScript call to the Ajax engine instead. Any response to a user action that does not require a trip back to the server - such as simple data validation, editing data in memory, and even some navigation - the engine handles on its own. If the engine needs something from the server in order to respond - if it's submitting data for processing, loading additional interface code, or retrieving new data - the engine makes those requests asynchronously, usually using XML, without stalling a user's interaction with the application [18]. The application only requests data typically via Simple Object Access Protocol (SOAP) or REpresentational State Transfer (REST) when needed and the user interaction can continue while the browser receives the response data. When the data is fully received, a JavaScript function is called, so that the data can be processed and the content of the page can be adapted.

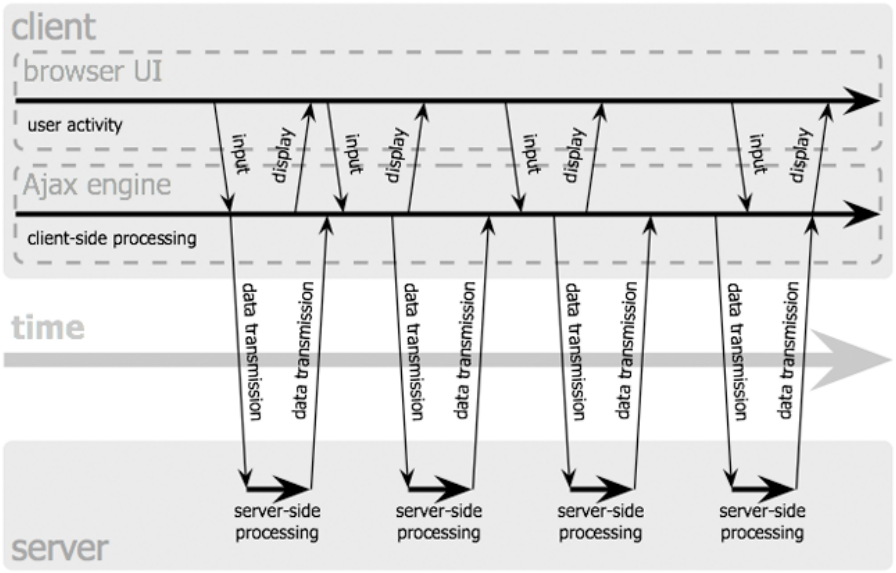

Figure 4.3: Process flow of an Ajax web application. This is the timing diagram from an Ajax web application that shows, that the user's activity directly invokes web requests to the server and after the data transmission is finished, the client updates only the relevant parts of the web page. The figure is taken from [18]

Advantages and disadvantages of an Ajax application:

- No *reloading* of the complete web site needed as content can be changed and data can

be obtained at run time.

- No *browser-plugin* necessary as it runs on every browser that handles JavaScript and DHTML correctly. It's problematic when JavaScript is deactivated by the user within the browser. Comparable technologies like `Adobe Shockwave` or `Flash` depend on plug-ins too, that even are operating system dependent and are only available for certain platforms.
- Pages need to be *tested* intensively with different browser types.
- The use of the *back-button* is problematic and dangerous as the browser normally stores only static pages in the browsing history and will return to the last fully loaded page from the cache. In an Ajax web site this is often not possible, as the condition what used to be the "last state" is not really defined. Mostly the intention of the user is to revise the last performed action. Because of the dynamic of the pages, this intention is hard to fulfill. There are some methods for handling the back button correctly, for example to use a hidden iframe inside the page and to overwrite the back button with a JavaScript function call. This problem is related to the reloading-button problem above. It boils down to the fact, that the state the application is in, needs to be well defined and the required data can be loaded via Ajax from the server into the application.
- *Bookmarks* are intended to store a specific page, but inside an Ajax application the state is not always recoverable as described above. The problem is again very similar to the back-button and the reload. A typical solution is to safe the state within the URL address of the page using anchors ("#"). When calling a bookmarked page, the anchor can be interpreted by the JavaScript application and do the necessary Ajax requests to fill the containers with the requested data. The problem is, that the programmer has to take care of this himself, so developing an Ajax application that fulfills all these kind of needs is not that easy from scratch.
- *Search engines* also need deep links, which is again related to the problems above. Moreover search engines parse the links inside a web site to find further pages which is not really possible within an Ajax application. So crawling this kind of pages automatically (also see Section 4.6) is quite impossible. A solution would be to provide a list of valid deep links for robots to visit on a separate page (extra link strategy) or providing meta-tags to existing links that robots can follow (lightweight indexing).
- *Polling*. The server is not able to send asynchronous events to an Ajax client, if the data is not requested from the client. So the client sometimes needs to poll for data in order to detect changes. Such requests cause completely different work load for the servers. A common practice is to keep back answers from the web server until the data changes or a timeout from the client occurs. This practice minimizes the number of requests, but on the server side, there is always a thread for each client running to hold the connection. So there might be huge scalability issues.
- *Latency*. As the requests from inside the application may not be seen by the user, the user

needs some visual notifications, that the application is waiting for data. Some hourglass icons or "loading" advices are very useful here.

- The support of *Model View Controller* (MVC) pattern can be implemented very well. Ajax also enables cyclic or nested MVC models. See Figure 4.4 for a sketch. In this case, each view element has it's own separate controller and model. Refer to Subsection 4.5.3 for explanation.
- *Barrier-free Internet*. Building an Ajax application that follows the WAI[43] rules is a challenge, as the provided text needs to be accessible through a screen-reader for blind people. This is a problem, as many applications are originally designed for a graphical web browser.

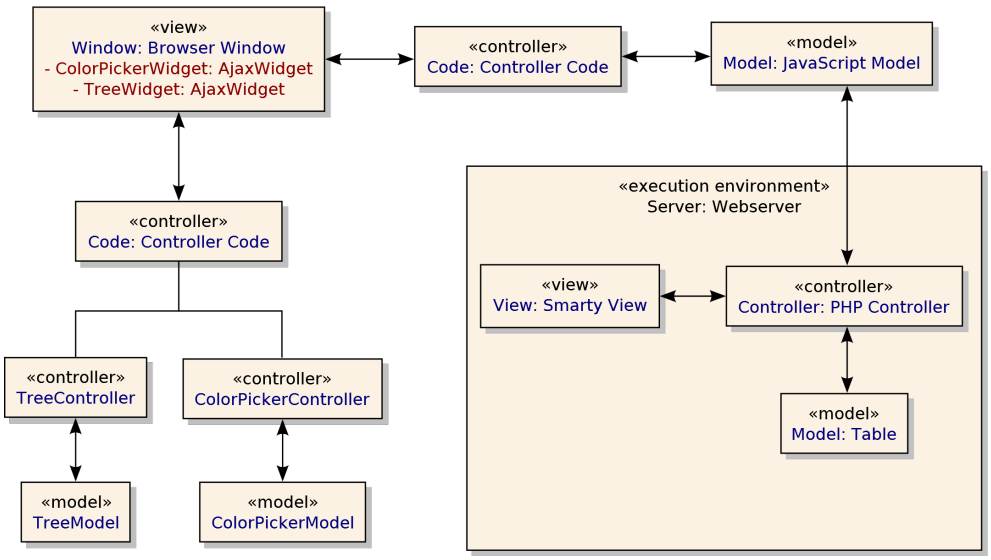

Figure 4.4: Model View Controller pattern (MVC) applied to an Ajax web site. The pattern itself is described below in Subsection 4.5.3.

4.5.3 Model View Controller

This design pattern mentioned above and in Figure 4.4 is very important to know and unfortunately very hard to understand. It obviously consists of the following three parts:

- Model: the pure data
- View: an interface to view the data and probably change it
- Controller: the operations that can be carried out on the data

The separation of the three parts is important, because their tasks should not intersect. The *model* should only represent the data and needs to be independent from the two others. The *view* only displays the model data and should not change the data. Usually this is some

[43] http://de.wikipedia.org/wiki/Web_Accessibility_Initiative

kind of a user interface. The view can be independent from the controller as well as it can be the controller itself. The *controller* only provides the data model to the view and takes commands from the user interface. The important thing is, that the model does not depend on the controller or the view. It therefore can be adapted easily when data types change or attributes are added. Using this design pattern makes the *model* classes adaptable and reusable without modifications. New classes can be added without modifying the original ones and often these classes represent how the data is stored within a database. It also renders the *view* classes reusable for the new classes.

4.5.4 Browsers with Ajax support and Ajax Sites

All these browsers support Ajax in their current version: Microsoft Internet Explorer, Mozilla Firefox, Opera, Apple Safar, Shiira, Google Chrome, Iron, Netscape, Konqueror, iCap. The use of Ajax in ordinary web pages increased a lot lately. On the one hand, responsive design[44] can be easily realized in connection with CSS, on the other hand the user experience is much higher. Commonly known web sites that extensively use Ajax are for example:

- Google Maps
- OpenStreetMap
- AjaxWrite (text processor)
- iRows (spreadsheet)
- nexImage (image processing)
- Flickr (photo management)
- Delicious (bookmarks and tags)
- Last.fm (music network)
- Facebook (social network)
- eyeOS (web desktop)
- Gmail
- Meebo (multi protocol messenger)
- 24SevenOffice (customer-relationship-management)

4.6 Crawling and Indexing Web Pages

Search engines like Google and Yahoo are designed to crawl web pages in order to index their content and make them findable though keywords. A so called *webcrawler*, *spider* or *searchbot* is a computer program, that automatically browses and examines web pages and analyzes them. Normally they build a list of outgoing links and start recursively to visit one after the other. A

[44] http://en.wikipedia.org/wiki/Responsive_web_design

sketch of the architecture of a webcrawler[45] can be seen in Figure 4.5. The programming of a crawler that downloads some pages and stores the content for further investigation might be very easy. Building a high performance system that downloads thousands of pages over weeks including a strategy when to revisit the pages is a big challenge in system design, network efficiency, robustness and manageability. Not at least, the gathered data needs to be processed, indexed and stored efficiently, so that the desired data can be accessed later.

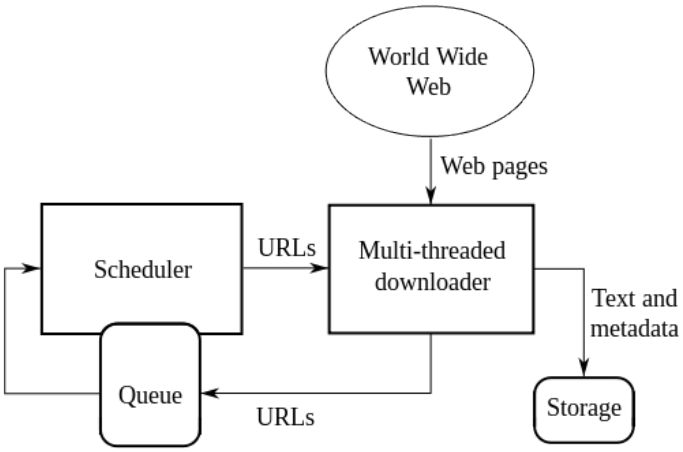

Figure 4.5: Architecture of a webcrawler. Search engines are designed to crawl web pages in order to make them findable by indexing their content for keywords. The figure shows a simple crawler layout, that parses web sites for URLs and queues them for future parsing. Every page is then parsed for keywords and metadata which are stored within a data storage.

Ajax sites are more to be seen like applications then as single web sites and can cause problems here. Currently search engines cannot correctly index Ajax applications because they do not understand the application logic that loads content dynamically [15]. Since the user invokes events on the page, crawling must identify the different application states generated by the client-side logic. According to the authors there are approaches to automatically identifying states by triggering events, but there arise some problems like efficiently crawling application states, avoiding the invocation of potentially very numerous events, scalability in the number of events, duplicate elimination of states, result presentation and aggregation and finally ranking. These are hard problems to solve in the future as it is like scanning a graphical user interface for phrases. To temporary overcome these problems, Ajax pages often additionally provide a static versions of the content. This is for being viewed by older web browsers as well as for being scanned by web spiders which then can successfully index the page.

In this software project, the crawling of Ajax pages is not necessary as long as the target sites are not using Ajax technologies. Within the project we used common spidering software like OpenWebSpider[46] to crawl the target page, to fill the database with a basic set of links

[45] http://en.wikipedia.org/wiki/Web_crawler
[46] http://sourceforge.net/projects/openwebspider

that will be used for as source pages for the games. Moreover navigation data from the user produce can be used to enhance the database.

4.7 Communication and Protocols

Several communication formats and paradigms are commonly used between the software parts within a client-server application. In total there is a big number of these formats, but I will only describe some relevant ones. A service is an abstraction of a computer resource which can be accessed by a client. The client does not have to be concerned about how the server performs the request. The client only has to understand the response from the server based on the application protocol. So the content and the formatting of the data for the requested service [47] needs to be known by both parts. Clients and servers exchange messages in a request-response messaging pattern: The client sends a request, and the server returns a response. This exchange of messages is an example of inter-process communication. To communicate, the computers must have a common language, and they must follow rules so that both the client and the server know what to expect. The language and rules of such a communication are defined in a communication protocol. All client-server protocols operate in the application layer. The application-layer protocol defines the basic patterns of the dialogue. To formalize the data exchange even further, the server may implement an API (Application Programming Interface[48]) such as a web service [5]. The API is an abstraction layer for such resources as databases and custom software.

4.7.1 JavaScript Object Notation (JSON)

"The JavaScript Object Notation (JSON) is a data-interchange format often used in JavaScript, because it is easy for humans to read and write. Additionally it is also easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition, December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language." [24]

Today, often XML is used for this purpose. XML has advantages to describe structured data and to serialize objects. Various XML-based protocols exist to represent the same kind of data structures as JSON for the same kind of data interchange purposes. When data is encoded in XML, the result is typically larger in size than an equivalent encoding in JSON. This is mainly because of XML's closing tags. However, there are alternative ways to encode the same information. See the following XML examples which carry the same information as

[47] http://en.wikipedia.org/wiki/Client-server_model
[48] http://en.wikipedia.org/wiki/Application_programming_interface

the JSON example in different ways. See the JSON example in Listing 4.7 and compare them later with the XML examples in Subsection 4.7.2.

```
{
    "firstName": "Mathias",
    "lastName": "Mayrhofer",
    "uid": 9630747,
    "address":
    {
        "street": "Inffeldgasse 16b",
        "postalCode": "8010",
        "city": "Graz",
        "country": "Austria"
    },
    "phoneNumber":
    [
        {
          "type": "home",
          "number": "+43/316/873-555-1"
        },
        {
          "type": "fax",
          "number": "+43/316/873-555-2"
        }
    ]
}
```

Listing 4.7: JSON notation is a data interchange format often used in JavaScript between the components. It can read and write directly from and to JavaScript objects. It is simple and human readable.

By definition, the JSON text can be evaluated directly by the `eval()` command in JavaScript. Although it is better to use a JSON parser, as a parser will recognize only JSON text and will not evaluate any scripts. To show how easy this is, we take the var `jsonExampleListing` to the text of Listing 4.7 and create and object named `contact` which then has the properties described in the listing:

```
var contact = eval("(" + jsonExampleListing + ")");
contact.firstName === 'Mathias';
contact.phoneNumber[0].number === '+43/316/873-555-1';
```

4.7.2 Extensible Markup Language (XML)

"Extensible Markup Language (XML[49]) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is defined in the XML 1.0 Specification produced by the W3C, and several other related specifications, all free open standards. It is a simple, very flexible text format derived from SGML (ISO 8879).

[49] http://en.wikipedia.org/wiki/XML

Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the web and elsewhere" [44]. The design goals of XML emphasize simplicity, generality, and usability over the Internet. It is a textual data format with strong support for the languages of the world via Unicode. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures like in web services.

```
<?xml version="1.0" encoding="UTF-8"?>
<Object>
  <Property><Key>firstName</Key>   <String>Mathias</String></Property>
  <Property><Key>lastName</Key>    <String>Mayrhofer</String></Property>
  <Property><Key>uid</Key>         <Number>9630747</Number></Property>
  <Property><Key>address</Key>
    <Object>
      <Property>
         <Key>street</Key>
         <String>Inffeldgasse 16b</String>
      </Property>
      <Property><Key>postalCode</Key> <String>8010</String></Property>
      <Property><Key>city</Key>       <String>Graz</String></Property>
      <Property><Key>country</Key>    <String>Austria</String></Property>
    </Object>
  </Property>
  <Property><Key>phoneNumber</Key>
    <Array>
      <Object>
        <Property>
            <Key>type</Key>   <String>work</String>
        </Property>
        <Property>
            <Key>number</Key> <String>+43/316/873-555-1</String>
        </Property>
      </Object>
      <Object>
        <Property>
            <Key>type</Key>   <String>fax</String>
        </Property>
        <Property>
            <Key>number</Key> <String>+43/316/873-555-2</String>
        </Property>
      </Object>
    </Array>
  </Property>
</Object>
```

Listing 4.8: Example for a typical XML object like we could find it for transferring an address book item. The XML notation looks clear and is human readable. With many attributes, the notation becomes extensive and confusing.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<person>
  <firstName>Mathias</firstName>
  <lastName>Mayrhofer</lastName>
  <uid>9630747</uid>
  <address>
    <street>Inffeldgasse 16b</streetAddress>
    <postalCode>8010</postalCode>
    <city>Graz</city>
    <country>Austria</country>
  </address>
  <phoneNumber type="work">+43/316/873-555-1</phoneNumber>
  <phoneNumber type="fax">+43/316/873-555-2</phoneNumber>
</person>
```

Listing 4.9: Example for an XML object that stores the data within the content of the elements. Also attributes are used to denominate more than one entry for *phoneNumber*.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<person firstName="Mathias" lastName="Mayrhofer" uid="9630747">
  <address streetAddress="Inffeldgasse 16b" postalCode="8010"
           city="Graz" country="Austria"/>
  <phoneNumber type="work" number="+43/316/873-555-1"/>
  <phoneNumber type="fax"  number="+43/316/873-555-2"/>
</person>
```

Listing 4.10: Example for an XML object that stores much of the data within the attributes of the elements. The *phoneNumber* elements only have attributes and do not even have a content any more.

The language mainly consists of *elements*, *attributes* and *content*. An element is a logical component which begins with a start-tag and ends with an end-tag. The content is the data between these tags. Attributes are additional variables that can be added to the elements. In Listing 4.8, the elements would be "object", "property" and "key" which are nested hierarchically. There are different possibilities to construct the same object like the address object in this example. In the first example, we have nested elements for the different properties and the data is stored within a key/value scheme. In contrast to this, the second example (Listing 4.9) abstracts the same data with a different XML notation. Here we have the data within the content of the elements which are now named directly after their meaning. This example is much shorter and even better to read. The third example however is even shorter. Listing 4.10 shows the same data where the elements from before are now attributes. The content is empty here. These three examples should point out, that XML is very versatile and the structure can be highly adapted to one's needs. This can also be seen as a disadvantage, as it is up to the programmer to decide which structure he wants to use.

4.7.3 Simple Object Access Protocol (SOAP)

According to John Mueller [31], SOAP relies exclusively on XML to provide messaging services. Microsoft originally developed SOAP[50] to take the place of older technologies that don't work well on the Internet such as the Distributed Component Object Model[51] (DCOM) and Common Object Request Broker Architecture[52] (CORBA). These technologies fail because they rely on binary messaging; the XML messaging that SOAP employs works better over the Internet. After an initial release, Microsoft submitted SOAP to the Internet Engineering Task Force[53] (IETF) where it was standardized. SOAP is designed to support expansion, so it has all sorts of other acronyms and abbreviations associated with it, such as WS-Addressing, WS-Policy, WS-Security, WS-Federation, WS-ReliableMessaging, WS-Coordination, WS-AtomicTransaction, and WS-RemotePortlets. In fact, you can find a whole list of these standards on "Web Services Standards". The point is that SOAP is highly extensible, but you only use the pieces you need for a particular task. For example, when using a public web service that's freely available to everyone, you really don't have much need for WS-Security. The XML used to make requests and receive responses in SOAP can become extremely complex. In some programming languages, you need to build those requests manually, which becomes problematic because SOAP is intolerant of errors. However, other languages can use shortcuts that SOAP provides; that can help you reduce the effort required to create the request and to parse the response. In fact, when working with .NET languages, you never even see the XML. Part of the magic is the Web Services Description Language WSDL[54]. This is another file that's associated with SOAP. It provides a definition of how the web service works, so that when you create a reference to it, the IDE can completely automate the process. So, the difficulty of using SOAP depends to a large degree on the language you use. One of the most important SOAP features is built-in error handling. If there's a problem with your request, the response contains error information that you can use to fix the problem. Given that you might not own the web service, this particular feature is extremely important; otherwise you would be left guessing as to why things didn't work. The error reporting even provides standardized codes so that it's possible to automate some error handling tasks in your code [31]. See Listing 4.11 for an example of a SOAP request.

[50] http://en.wikipedia.org/wiki/SOAP
[51] http://en.wikipedia.org/wiki/Distributed_Component_Object_Model
[52] http://en.wikipedia.org/wiki/Common_Object_Request_Broker_Architecture
[53] http://www.ietf.org/
[54] http://en.wikipedia.org/wiki/Web_Services_Description_Language

```
1 //SOAP request
2 <?xml version="1.0"?>
3 <soap:Envelope
4     xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
5     soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
6  <soap:body pb="http://www.acme.com/phonebook">
7   <pb:GetUserDetails>
8    <pb:UserID>12345</pb:UserID>
9   </pb:GetUserDetails>
10  </soap:Body>
11 </soap:Envelope>
12
13 //REST request query
14 http://www.acme.com/phonebook/UserDetails/12345
```

Listing 4.11: Example for a SOAP and a REST request which differ dramatically in length. In the case of SOAP, the URI parts are connected together by the application, before the same request as the REST query is sent.

4.7.4 REpresentational State Transfer (REST)

Many developers found Simple Object Access Protocol (SOAP) cumbersome and hard to use. For example, working with SOAP in JavaScript means writing a ton of code to perform extremely simple tasks because you must create the required XML structure absolutely every time. John Mueller furthermore [31] describes REST[55] as lighter weight alternative. Instead of using XML to make a request, REST relies on a simple URL in many cases. In some situations you must provide additional information in special ways, but most web services using REST rely exclusively on obtaining the needed information using the URL approach. REST can use four different HTTP 1.1 verbs (GET, POST, PUT, and DELETE) to perform tasks. Unlike SOAP, REST doesn't have to use XML to provide the response. You can find REST-based web services that output the data in Command Separated Value (CSV), JavaScript Object Notation (JSON) and Really Simple Syndication (RSS). The point is that you can obtain the output you need in a form that's easy to parse within the language you need for your application. The Listing 4.11 compares the SOAP query to a much simpler REST query.

4.8 Databases

"A database[56] is usually a large collection of data organized especially for rapid search and retrieval (as by a computer) an organized collection of data." [14] A database stores organized data in a model that usually represents relevant aspects of reality. This can be information like text, numbers, graphics or whole documents. A database management system (DBMS) is a software application that allows access to the data, so that the user or an external program can

[55] http://en.wikipedia.org/wiki/Representational_state_transfer
[56] http://en.wikipedia.org/wiki/Database

access, store or modify the data. The database itself is not generally compatible with other database systems, but most of them can be accessed by using standards like SQL, ODBC or JDBC. This allows an application to work simultaneously with more than one database at the same time and moreover the database can be replaced by another DBMS without changing the application. The following is a list of well known database management systems:

- MySQL
- MariaDB
- PostgreSQL
- SQLite
- Microsoft SQL Server
- Oracle
- Berkeley DB
- SAP HANA
- dBASE
- FoxPro
- IBM DB2
- LibreOffice Base
- FileMaker Pro
- NoSQL
- NewSQL
- CouchDB

Most of these database systems run as isolated services or even on separated servers. This is very convenient, as it simplifies the administration effort and systems can access backup or failover services by changing only the address of the server. Some of these can also be put together into a compound of servers, a database cluster. Typically DBMSs have a similar SQL-like syntax, with only little differences and adaptions. Using standard software databases is very convenient as the setup time is short, and depending on the chosen product they are easy to maintain over time. Databases nowadays come with rich instruction sets and high performance data processing as well as comfortable accessing and designing and high data safety. These advantages are very convincing for outsourcing the task of data storing to a database of your choice and leaves nearly no drawbacks behind also (and probably especially) for large scale data environments.

4.9 Security Issues

We need to distinguish security issues on different levels. First is the level of the application written in this thesis. The second perspective is the security that arises with Ajax and cross site

scripting in general. The third perspective will be on the level of commercial data gathering from the user and big data.

4.9.1 Application Security

This project is frank about being a game that displays content from another web site, that is normally chosen to be a common web site with public access. As the users are taking part in the game voluntary, the impact that the user may not understand what is going on behind the scenes is negligible. It is getting more serious, when the user jumps to web sites, that are more private within my game. The positive thing is, that it is quite obvious for the user, that his history is tracked. A problem would be, if a user copies a link from our manipulated web site and stores it or sends it to a friend. Then the link would originate not the desired web page but our game's web page, which does not do any harm, but might be unwanted due privacy issues.

4.9.2 Ajax and Cross Site Scripting (XSS)

Browsing safely through the Internet is most desirable by the user. With modern technologies that enable web sites to be more like applications which feature a broad variety of functions, this goal can hardly be achieved. Besides the server's ability to use the information from one's access (also see Subsection 3.1.1) directly, a big security question lies on the client side concerning JavaScript and Ajax. As this is a major concern for every browser software, there exist "Content Security Policy (CSP)" guidelines from the W3C Consortium informed by the "IETF websec working group[57]". [13]

Using CSP should prevent the user browsing on a site accidentally from executing (probably malicious) JavaScript code from another site. Malicious JavaScript could be still injected by preparing user content on the original site, so that the browser executes it as it does not violate the same-origin policy (SOP[58]). A known threatening scenario is a bug where JavaScript can be hidden inside a GIF89a formatted picture that would leverage the security of CSP.

Dongseok Jang et al. [22] distinguish four violating methods that compromise the user's privacy: cookie stealing, location hijacking, history sniffing, and behavior tracking. According to the authors, JavaScript enabled the deployment of rich browser based application today, but these scripts are often sourced from different mutually distrusting and potentially malicious web sites. However, JavaScript lacks language-based protection and isolation mechanisms and sports several extremely dynamic language features. Browser-level isolation policies like the same-origin policy for domain object model (DOM) objects are coarse-grained and do not uniformly apply to application resources. Consequently, the proliferation of JavaScript has also opened up the possibility of a variety of security vulnerabilities. [22]

[57] http://tools.ietf.org/wg/websec/
[58] http://en.wikipedia.org/wiki/Same-origin_policy

- *Cookie Stealing*: Code included from a particular site, say for displaying an advertisement, has access to all the information on the hosting web page, including the cookie, the location bar, and any other sensitive information stored on the page. Thus, if the ad code is malicious it can cause the cookie and other sensitive pieces of information to be leaked to the third-party ad agencies, and can lead to a variety of routinely observed attacks like request forgery.
- *Location Hijacking*: The dynamically loaded untrusted code can influence the document's location, by influencing the values stored in URL string variables that are read to dynamically generate HTTP requests. Consequently, the dynamically loaded code can navigate the page to a malicious site that exploit a bug in the browser to fully compromise the machine or mounts a phishing attack.
- *History Sniffing*: In most browsers, all application domains share access to a single visited-page history, file cache, and DNS cache. This leads to the possibility of history sniffing attacks, where a malicious site can learn whether a user has visited a specific URL, merely by inducing the user to visit their site. To this end, the attack uses the fact that browsers display links differently depending on whether or not their target has been visited. In JavaScript, the attacker creates a link to the target URL in a hidden part of the page, and then uses the browser's DOM interface to inspect how the link is displayed. If the link is displayed as a visited link, the target URL is in the user's history. There are companies that sell services that allow a web site to collect the browsing history of their visitors using history sniffing.
- *Behavior Tracking*: The dynamic nature of JavaScript allows a web site to construct a high-fidelity timeline of how a particular user interacted with a web page including, for example, precise information about the user's mouse clicks and movements, scrolling behavior, and what parts of the text were highlighted, simply by including JavaScript event handlers that track mouse and keyboard activity. This information can then be sent back to the server to compute statistics about how users interact with a given web page. Several web analytics companies sell products that exploit these flows to track information about users. For example, `ClickTale` allows web sites to precisely track their users' mouse movements and compute aggregate heat maps based on where users move their mouse, and `tynt` allows web sites to track what text is being copied from them. These services allow web sites to gather fined-grained information about the behaviors of their users without any indication to users that additional information gathering is taking place. The authors believe that users understand that page navigation (as a result of clicking on a link) causes information to be sent to the server, but they do not believe the users understand that other actions, like mousing over an image, can silently do the same.

The term Cross-site Scripting (XSS) denotes a class of string-based code injection attacks on web applications [23]. If a web application implements insufficient input validation and/or

output sanitation an adversary might be able to inject arbitrary script code into the application's HTML encoding. A successful XSS attack can lead to, e.g., compromised authentication information, privilege escalation, or disclosure of confidential data. According to the authors the attacks can be classified in three different kinds: [23]

- *Reflected XSS* denotes all non-persistent XSS issues, which occur when a web application blindly echos parts of the HTTP request in the corresponding HTTP response's HTML. For successfully exploiting a reflected XSS vulnerability, the adversary has to trick the victim into sending a fabricated HTTP request. This can, for instance, be done by sending the victim a malicious link, or by including a hidden Iframe into page controlled by an attacker.
- *Stored XSS* refers to all XSS vulnerabilities, where the adversary is able to permanently inject the malicious script in the vulnerable application's storage. This results in every user that accesses the poisoned web page receiving the injected script without further actions by the adversary.
- *DOM-based XSS* is a special variant of reflected XSS, where logic errors in legitimate JavaScript and careless usage of client-side data result in XSS conditions. The offending data exists solely in the browser and is not sent to the server.

Their solution might be removal filters, that remove problematic keywords like "document.", "script", "javascript" or external links like "http://". Output sanitation that encodes suspicious characters like <, ", ' into HTML before they get interpreted is always good practice although it often fails to be implemented correctly or to the complete set of user-supplied data.

There are many methods that bad guys will use to compromise other web sites to steal login credentials from users. Some of them relate to the attacks described above and some even use the same techniques I am using in my project. In this case, the Java Servlet server hands out content from another server that of course looks like it was directly from the other server. The user cannot distinguish between an iframe that loads content from remote or an iframe that is filled by my server which retrieves the same content from the original server and eventually modifies it. In the second case, I have full control over the site and can probably steal information the user gives us. So a scenario where an attacker loads the content from your online bank so that you think you enter you credentials on the original site technically looks identical to the way my project works. Unfortunately there is no solution for this problem. Another XSS attack could be disguised as an advertisement placed transparently on a web site so that you accidentally click on it (probably it looks like a play button you want to press) invoking e.g. Facebook posts with your name (or just a like).

4.9.3 Database Security

The security of the communication between the Java Servlet and the MySql database should also be considered. Typically the communication takes place via TCP/IP ports. If the Servlet and the database reside on the same server, the database can be addressed by `localhost:port` (e.g. 3372 for MySql) and external access to this port should be completely firewalled. If the database resides on a different server, the transfer should be encrypted and access from the Internet to the database server (and port) should be limited to only the IP address of the Servlet server. This is important, as databases are vulnerable and generally should be behind firewalls.

5 Software Application ClickPath

5.1 Intention of the Software

The idea behind the software was to create a framework that can record the user's navigation behavior. The actual web content needs to be displayed in a separate part of the page, whereas the framework keeps track of the user actions within the page. It records every click and sends it to a designated server. This server is separated from the server which serves the actual content. The framework should use Ajax and Web 2.0 technologies to track the clicks from within the web site for statistical evaluations. Similar technologies as used by Google Analytics for their online evaluations. For security considerations of this method, please refer to Section 4.9. At the server side, the records should stored in a database for later use e.g. evaluations of the click paths. A commonly used database server software should be used in order to be compatible in the future. For large scale test environments one could think of using professional databases that can better handle large amount of data.

To wrap the framework in a useful web application, where users can even take part voluntarily, the following game play was invented. The participants join the game and all start from the same starting point which is a certain page within a web site. They are told to reach a certain destination web site as fast as they can, only following links within the web site. This type of navigating is called *browsing*. The Subsection 2.4.2 explains the difference between searching and browsing. The application shows all participating users of the specific game and the quantity of links they clicked so far to encourage the user to reach the destination fast and probably win the game. After a configurable amount of time, the game is over and the next game starts automatically with a different starting page and a different destination page. The participants that successfully reached the destination page get a winner's ranking.

The system randomly chooses start and destination page out of a set of pages previously known to the system. This might be pages selected by hand or by a previous crawl though the target site. Previously generated click paths from users can also serve as source or destination pages. The best way is to have a complete set of pages of a site and import them into the database of source pages cleaning up unwanted pages by hand or regular expressions, e.g. incomplete pages from Wikipedia, author pages or special pages of the day.

5.2 Software Requirements and Specifications

In software development, the programmer has to choose the appropriate tools to fulfill the requests. What the purpose of the software is, was already explained in Section 5.1. Now it comes to the requirements for the software. Let us distinguish between the following parts of the software:

- Programming language
  - modern language so it will not be deprecated soon and has full functionality and modern programming style
  - fast processing and parsing of text data because the viewed pages must be parsed permanently
  - convenient editor
  - editor with debugging and profiling options for better error handling
  - runs on most operating systems
  - scalable if the project becomes bigger or more complicated
  - library support: if requirements enhance it could be an advantage to use external libraries
  - good database connectivity
  - database connection pooling
  - available for free, open source preferred
  - good community support if there arise problems
  - updates and security fixes when security holes become apparent
- Server requirements
  - reliable data storage
  - separation of database and logic framework
  - high availability
  - resource-efficient for smaller servers
  - fast responding because it will have to serve the content page after every click
  - scalable in case there are many test persons at the same time
  - conveniently manageable
  - reconnects to database when there was a disconnect
  - caching of data
  - updates of the software and the framework (in case of security holes)
  - community if there are configuration issues
- Installation
  - easy setup process
  - easy to deploy the program or an update
  - only one package to handle/deploy, no separate files
  - possibility to import database scheme
- Interface specifications
  - accessible via browser
  - Ajax and Web 2.0 for sending data, controlling the game and for replacing the links

  in the browser
  - reliable sending the click data
  - the data that is recorded contains not only the page address, but also the title of the tag the user clicked
  - runs on different browsers
  - runs without additional software to be installed like addons or extensions
  - a frame should be visible, that shows the task in a side panel
  - it should be clear to the user, that he plays a game and the content page is not the original page
  - there should be an administration interface for adjusting the game variables
  - the administration interface should list previous game results
  - the administration interface should have options for introducing new pages and to import data from external sources (e.g. webcrawler, see Section 4.6)
- Game play
  - automatic game creation if the game time is up
  - user should be able to see the competitors
  - the task in the beginning should be shown
  - see if the game succeeded
  - see if one has reached the destination page
  - see the rank within the other competitors when succeeded

5.3 Technology Selection

Information previously gathered in Section 4.1 can be summarized according to the needs for this software project. This leads to Table 5.1 that shows the amount the technology fits for the project. Be aware, that the measurements may not be objective as personal knowledge and attitude influenced the assessment. Personal weights are on the last row and the result on the most right column. The license of the software can be a problem when it comes to commercial use of the software. As this is not intended, the license was not taken into account, although I highly prefer all types of open licenses.

Not that complicated was the database management system choice. Although there are many database systems available, only a few are commonly used. Please refer to Section 4.8 for more information about database management systems. The following Table 5.2 summarizes the requirements for the database management systems along with the weights and personal preferences. Again licenses are listed but not taken into account. As a result, the MySql[1] database was used this project as back end. It is a reasonable choice as it provides a fast installation and it is already often pre-installed on server systems. MySql provides good data integrity and has the necessary connectors to the frameworks above. It is good in performance and can handle large amounts of data. Although it is no top of the scale application it perfectly

[1] http://www.mysql.com/

			Requirements									
Language	Framework	License	Language convenience, editors and debugging	Performance (especially text parsing)	Persistence support, database performance	Community and library support	Deployment and installation	Resources footprint	Operating system compatibility	Scalability, program and data	Security and updates	Personal points result
C++	Wt	GPL	2	2	2	1	1	3	2	2	1	6.8
C++	CppCMS	LGPLv3	2	2	2	1	1	3	2	2	1	6.8
C#	ASP .NET	proprietary	3	3	3	3	3	1	1	3	2	10.6
Java	Spring	Apache 2.0	3	3	3	2	2	1	3	3	2	9.9
Java	Struts	Apache 2.0	3	3	3	2	2	1	3	3	2	9.9
Java	GWT	free	3	3	3	1	2	1	3	3	1	9.0
Java	JSP	GPL	3	3	3	3	3	2	3	3	3	11.7
Perl	Perl	GPL	1	3	2	1	1	3	2	1	2	6.8
PHP	PHP	PHP (GPL)	2	1	2	3	2	3	2	1	1	7.3
Python	Django	BSD	2	3	3	2	2	2	2	3	2	9.5
Ruby	Rails	MIT	2	3	3	2	2	2	2	3	2	9.5
Personal preference weight			0.5	0.6	0.4	0.5	0.6	0.3	0.2	0.5	0.4	-

Table 5.1: Selection of available technologies and their evaluation for each property to fit the requirements for the software project. The numbers are points from one to three.

fits for our needs.

		Requirements						
Database	License	Database tools	Performance and scalability	Resources footprint	Community support	Installation	Operating system compatibility	Personal points result
MS SQL	proprietary	3	3	1	3	2	1	4.6
MySql	GPLv2	3	2	3	3	3	3	5.4
MariaDB	GPLv2, LGPL	2	2	3	2	2	2	4.4
PostgreSQL	open source	2	3	2	2	2	2	4.6
Oracle	OLSA, proprietary	2	3	1	2	1	2	4.0
SQLite	public domain	2	1	3	2	2	3	4.0
Personal preference weight		0.3	0.6	0.4	0.3	0.2	0.2	-

Table 5.2: Selection of available database management systems and their evaluation for the requirements for the project. The numbers are points from one to three again.

5.4 Software Description

The screen shot in the Figure 5.1 shows the framework with the start page of the current task. In this case the user is asked to navigate from "Technik" (I had German Wikipedia as source site) to "Leonhard Euler". When the user clicks on the starting page, he will be brought to the Wikipedia article and has to navigate to the destination article only using the links served on the page. The search buttons within the page will not work. On the left, a summary of the task is shown, as well as the participating users and the time left for this round. The task is successful, when a clicked link within the page refers to the destination site. This means, that the link must exactly match. This could lead to problems, when links consist of queries or variables. The fragment of the URL[2] is not taken into account, but the path and the search part have to be considered. The technical details how the program works are explained in the Chapter 4.

[2] http://en.wikipedia.org/wiki/Uniform_Resource_Locator

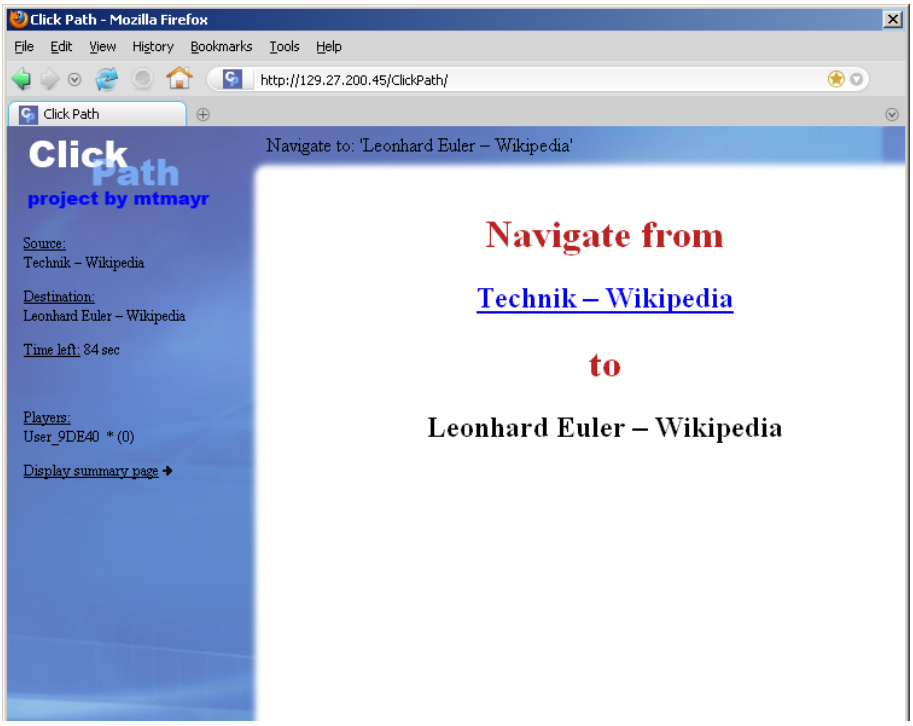

Figure 5.1: Screen shot of the framework showing the navigation task. On the left side, there is the task summary and the participants. The content page is opened inside the center area using an iframe. On the top, the name of the destination page is shown, so that the user knows where he has to navigate to. When the time is up, the browsing will be interrupted and a new game will be created.

5.4.1 Objectives

What is the aim of the whole idea now? When a user searches for a piece of information on the Internet, he is confronted with a massive amount of available data. Without the specific knowledge of the desired domain, it is hard to begin with the search. When searching, he needs to know some catchwords, when browsing he needs to know about the correct categories. As people tend to think very differently, this could be hard task before getting to the desired piece of information.

This is where researchers are challenged to help the user. If we get to know the navigation models, we can guide the user superiorly. A quite new technique is to tag articles and topics with catchwords. In a so called "tag cloud", one can see to which keywords a certain topic is related. This is not only to help the user to see where the current web site is classified, but also to give him the optimal hint to reach the destination site.

5.4.2 Record the User's Navigation

In this framework, the loaded web site gets loaded in an iframe and every link the user clicks on the web site is recorded. Every link target (e.g. `<a href="">`) gets an additional listener. When the target is clicked, it sends an asynchronous HTTP-request (AJAX[3], see Section 4.5 for further explanations) to our site which stores the target link on our database server (see Section 5.8). The framework automatically keeps track of the links the user clicked, and checks if the destination link has already been reached. For later use, it helps to store as much information as we can. This includes the following items:

- current site URL (Uniform Resource Locator)
- targeting site URL
- the label which is referring the targeting link
- user identification
- game identification
- time

Every recorded click path can be viewed afterward by the administrator. You can see an example of a completed task in Figure 5.2. It shows the user identification string of the participating users from this game and below the corresponding navigation path. It can be seen as a sequence of pages loaded and links clicked. The time measurement, displayed in light gray, might also be interesting for later evaluations. The anchor text of the link clicked by the user which brought him to the subsequent page is shown within quotes.

Keeping this kind of data lets us reconstruct the links that were clicked with the intention to reach which destination. Furthermore we can now compare the behavior to the shortest path from the source link to the destination link and to our estimations made by a category or tagging system. The second point requires the knowledge of all links and connections between the available web sites on the target server. This is usually stored in a large graph after having spidered the complete web site with a crawler (Section 4.6). This data can be retrieved before setting up the framework for the game and is to be seen as an isolated task.

5.4.3 Competition and Game Play

To get users to do some tasks for research, you can ask them in a survey and invite them for participation in a test-group. This is a quite time consuming task and the test groups need computers and supervisors to survey the people. The game approach is much more light-weight as people can play where they want (e.g. at home or at the office) and how long they have fun with it. It's also positive, that they react more natural in their familiar environment. So the data will be more representative and with the advantage that we do not spend much money on the test-environment setup and have an online test, where hundreds of participants can join

[3] http://de.wikipedia.org/wiki/Ajax_(Programmierung)

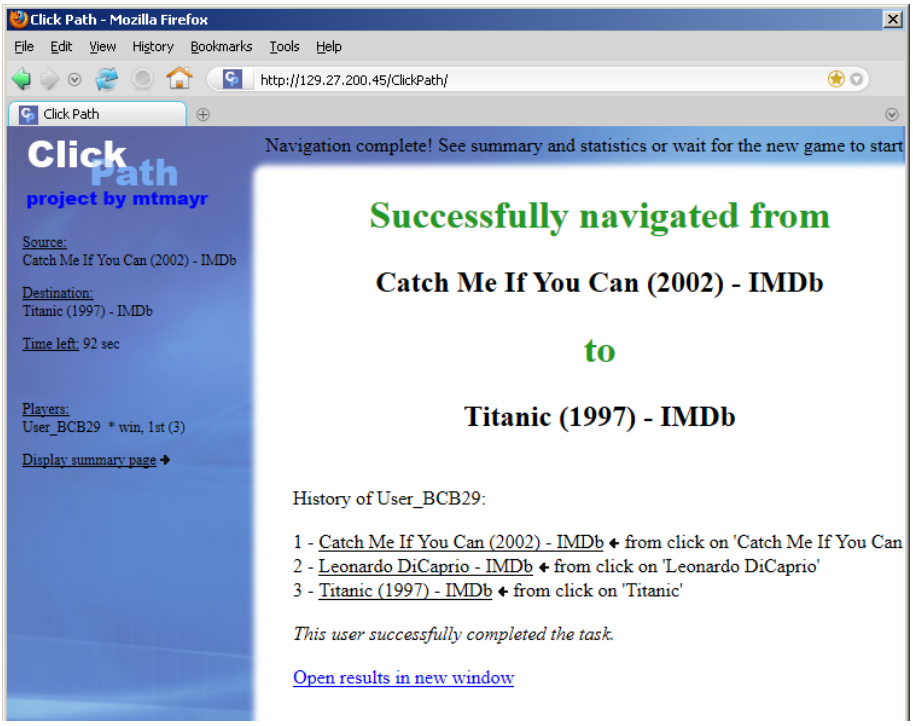

Figure 5.2: Example of a completed navigation path. When the user reaches the destination site, the navigation task interrupts and the details of the user's path are displayed until the next game starts. The page title as well as the anchor text of the clicked link are displayed.

simultaneously. The competition is also for encouraging the users to get to the destination page as fast as possible.

5.5 Used Technologies

The main module is a Java Server Page (JSP) called "ClickPath". It is the technology used at the server side, which loads and modifies the requested pages for the user and receives the click data from the user's web browsers. See Section 4.3 for technical details. The Java Servlet stores the received data into a MySql database which preserves the data for later investigations. Further explanations about the different types of Databases can be read in Section 4.8. The communication from the user's browser to the servlet takes place via Ajax and Web 2.0 requests from within the browser. Explanations about the communications can be read in Section 4.7. These three parts perform the complete work flow from the user's input to where the data is stored and can be extracted for research.

Figure 5.3 shows the state diagram and the transitions between the states for the server part of the program. It starts in idle mode and only continues when clients connect. The received click data, respectively the path of each participant, gets recorded. This state is running until

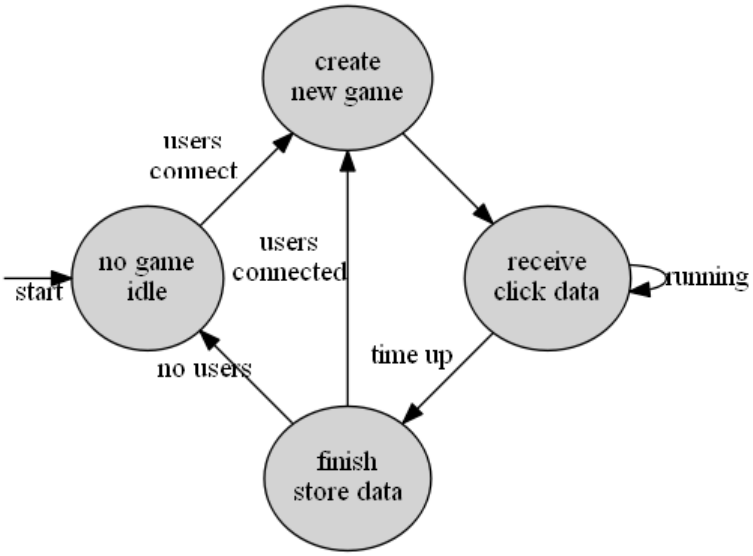

Figure 5.3: State diagram of the program. The program starts at the idle state, while no game is active. If users connect, a game is created randomly and click data is received from the clients until the time is up. All data is stored inside the database and the next game will be created if there are still users connected.

the maximum game time is reached. When the game is finished, the data is stored into the database for later investigation. When there are still users connected, another game is created. If not, the program switches into an idle mode, where no games are created. So there is no data created and no CPU time wasted on the server.

5.6 Using the Software

To install the server, you first need access to an Internet server that has the appropriate bandwidth available. The software does not depend on a certain operating system, as long as a Java Tomcat server can be installed. Then you have to upload the `.war` file. Open the Tomcat Admin Console[4] and upload the `.war` file. When the application starts, we can continue to adjust the settings in the Administration Interface.

5.6.1 Administration Interface

The admin interface can be configured by entering the following URL in any browser window: `http://yourserver.com/ClickPath/admin.jsp`. Figure 5.4 shows a screenshot of the possible settings. Following adjustments can be made:

- *Define Next Game*: When the next game starts, it will derive start and destination page from the source site denoted by `Site`. Newer sites may be introduced by the next

[4] http://tomcat.apache.org/tomcat-7.0-doc/manager-howto.html

setting. Additionally the `time interval` can be adjusted. It is an integer value denoting the seconds the games last. The default value is 120.

- *New Site*: Prepare the program for a new site. Enter `hostname`, `description` and a `starting URL`. After creating a game, sources need to be imported or generated from existing click data. The options are described below.
- *Sources*: Select a site to import click data to sources, fill the table with titles or delete all source entries.
- *Import*: Import a file into the sources table. This can be the output of a prior crawling of the source web site. Import file format is a CSV list with one or two columns separated by a tabulator character (\t). The first column denotes the URL, the second and optional column denotes the title of the page. The line is terminated by a carriage return or end-of-line (\r\n).

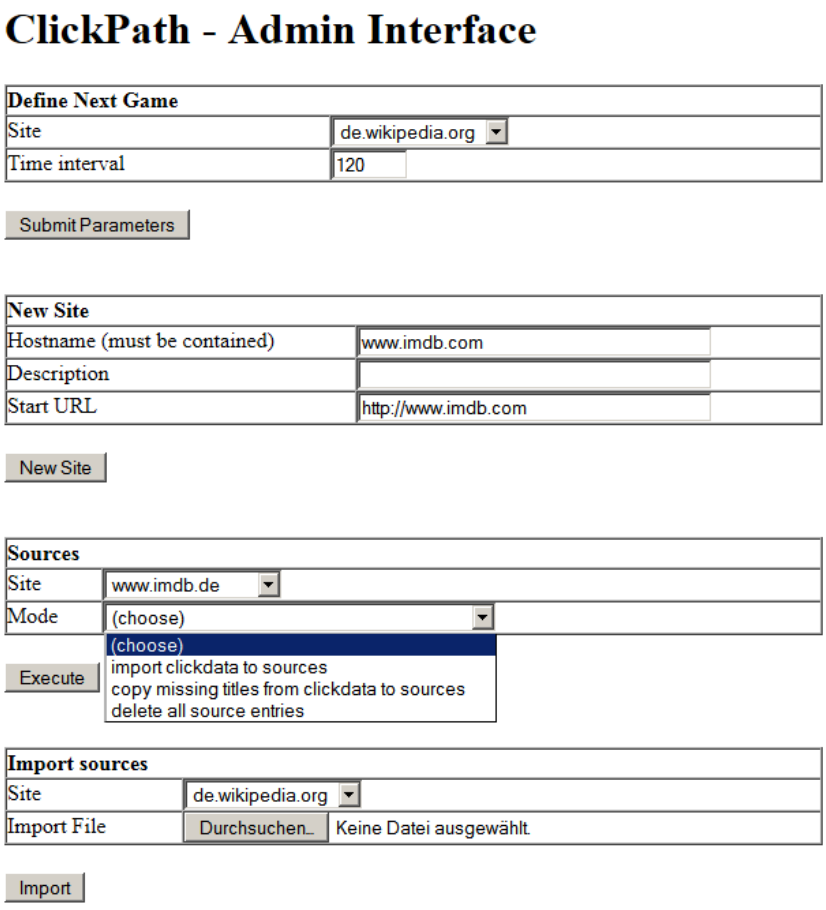

Figure 5.4: In the ClickPath administration interface, the destination site for the next game can be selected, as well as the time period the users have to finish the task. It is also possible to create new source pages and import links lists from a local file.

The sources table contains all pages within a site that can occur as start or destination page. There are two possibilities to fill the sources table with the desired data. First is by externally

crawling the page with another program as mentioned above (see Section 4.6). Any program will do, as long as the output format described above is suitable for import here. The program needs the title of each page, which can be either parsed by the crawler or derived from the click data within the database. This is the second option in the sources entry. It is also possible to import the click data directly to the sources. The process is as follows: We start playing the game at a new page and generate click data to interesting pages. The click data is stored with URL and title inside the database. After a suitable amount of visited pages, we can export the existing click data into the sources table, so the visited pages function as sources now. After people provide more and more click data, the sources list can be extended this way. The only problem is if people clicked on pages that are not suitable for start or destination sites, like for example invalid pages or error messages.

5.6.2 Game Control

The game control is a running thread that creates new games when the game lifetime is over and users were participating in the game. It checks every five seconds for the game status. When a game finishes, it creates a new game from the defined site. A start and a destination page gets randomly selected from the sources table. All client browsers then show the introduction page, where start and destination page are shown in the center of the page. The users can now click on the start page to begin navigating within the page. The destination page is always shown above as seen in Figure 5.1. On the left side, the countdown time is running and the other players are shown that participate in the game. Next to each player, the path count of the player is shown. If the player has already reached the destination page, his/her rank gets displayed nearby. The star sign denotes that this is the player itself.

5.6.3 Investigating Game Results

Within the admin interface, click on `Open Gamelist` to see the list of games where users participated. Figure 5.5 shows the list of games, how many players participated in each game and how many players could finish the task. When clicked on a specific game, the task itself (start and destination page) and all participants are displayed. The users are distinguishable by their session-id and their IP address. Each participant can be investigated separately by clicking on it. See Figure 5.6 for the detailed list of pages the user clicked on. The time spent on each page and the specific link text of the link clicked are listed. Knowing the specific link tag itself is also very important, as it might be interesting, if the user clicks a picture or other symbols. Knowing only the targeting page would not include the information from which link the user came from. If the task was successfully accomplished, it is mentioned below.

Game	Hostname	Date	Players	Finished
Game_2386	www.imdb.de	19.03.2014 02:02	2	2
Game_2385	www.imdb.de	19.03.2014 02:00	2	0
Game_2384	www.imdb.de	19.03.2014 01:58	2	0
Game_2383	www.imdb.de	19.03.2014 01:56	1	0
Game_2382	www.imdb.de	19.03.2014 01:54	1	0
Game_2381	www.imdb.de	19.03.2014 01:52	1	0

Figure 5.5: Inside the administration interface of ClickPath, all results from previous games are listed by date and time and the numbers of players that (successfully) participated.

Host: 'www.imdb.de'
From: Titanic (1997) - IMDb *(http://www.imdb.de/title/tt0120338/)*
To: Leonardo DiCaprio - IMDb *(http://www.imdb.de/name/nm0000138/?ref_=tt_cl_t1)*

Participating Users:

C3CB8856A1EFC4ACD4CB186E6191F47C * 84-115-151-146
BCB2CB4C22DB4FF08B66C9F76DBBB7EE 84-115-151-146

History of User_C3C30:

1 - Titanic (1997) - IMDb ← from click on 'Titanic (1997) - IMDb' +0s
2 - James Cameron - IMDb ← from click on 'James Cameron' +4s
3 - Avatar 4 (2018) - IMDb ← from click on 'Avatar 4' +20s
4 - James Cameron - IMDb ← from click on 'James Cameron' +9s
5 - Titanic (1997) - IMDb ← from click on 'Titanic' +10s
6 - Leonardo DiCaprio - IMDb ← from click on 'Leonardo DiCaprio' +5s

This user successfully completed the task.

Figure 5.6: When clicking on a certain game, the participating users/hosts are listed with their specific path from the source to the destination page.

5.6.4 Playing the Game

Give the test persons the following link: `http://yourserver.com/ClickPath/`. The participating users immediately join the active game. If a user is the first user and the game is already timed out, then a new game will be created instantly. Otherwise the running game is joined. The users keep on competing with each other and try to reach the destination pages. The users only see the other user's performances and if they have already reached the goal. Further statistics are not available for the users. The display on the top of the screen is for the destination page. The information box on the left shows the exact game conditions: The starting page, the destination page, the time left for this game in seconds and the participating players including the number of links they already clicked. Users that successfully made it to the desired page can lean back and watch the other players performing. They have to wait

until the game is over. When the game ends, the new task appears immediately on the content screen interrupting the current browsing. The new game can be started right away by clicking the starting link.

In order to participate in a game, the user needs JavaScript enabled web browser that has permanent connection to the Internet and especially to our game server and the content site(s). The browser itself needs to be capable of Ajax which most of the browsers are. Refer to Subsection 4.5.4 for a list of applicable browsers. The usability is very dependent on fast loading of the pages because there would be unnecessary delays in the competition and unexpected network delays would also influence the evaluation of the results.

5.7 Java Classes and JavaScripts

To give an overview about the used classes in the project we need to get an insight into the program itself. When the game view (main screen) is displayed in the user's browser window, there are mainly three components relevant: The state screen, the content screen. The main screen itself (`index.php`) only defines the layout in CSS and loads the JavaScript sources from the server. Furthermore, the left part gets periodically updated with the status screen reading data from the script `stats.jsp`, and the right part is for displaying the content. The script `linktracker.js` loads the content updater, the link tracker and the periodic updater. The content updater ins called, when a content page is fully loaded. While loading, an image overlay is shown, so that the construction of the page is hidden to the user. When the page is fully loaded, the link tracker starts to add Ajax requests (`record.jsp`) for recording the utilization of the link to all relevant link tags and pictures within the content page. When finished, the overlay gets invisible and the content page gets visible to the user. Whenever a user clicks on a link, two things happen: First the Ajax request to `record.jsp` is called to protocol the click. Second the link target itself gets loaded by the program (`content.jsp`) which loads the original page and prepares it for the use in our framework. The status screen gets information about the game status from the server and recognizes when a new game is created or when the content screen got navigated to the destination URL. When this happens, an overview of the clicked pages is shown and the user has to wait for the game to finish. The `urlparser.js` is a helper script for parsing URLs to determine the passed variables.

The Java classes that have been created can be seen in Figure 5.7. The class `User` represents the properties of a user, that participates in a game. It reads the session-id from browser's cookies, it's IP address and keeps track of the number of clicked links and if the user has already reached his destination. The class `Game` represents all properties about the running game, like when it started, when it ended, how many time is still left, where the navigation started and where it ended. The `Database` class is for accessing the MySql database from the software. It only handles the connection itself, the scripts use the class `Access` to operate the database. The `WatcherThread` is an always running thread that keeps up the connection to

File name	Description
index.jsp	The main screen layout
content.jsp	Getting content from another server and giving back a modified page
record.jsp	Recording a clicked link within the database
stats.jsp	Returns current game status and variables
admin.jsp	Administration Interface to manage the database
display.jsp	Investigating Game Results from previous games
linktracker.js	modifies the links within the web site
urlparser.js	converts absolute and relative URLs
*.gif, *.jpg	navigation elements

Table 5.3: Overview of the accessible files from the Tomcat server and their task. The JavaScript (.js) and the picture files are static files and are not processed by the server. They are directly accessed by the user's browser.

the database server and renews the game if a game expires.

`RequestWebsite` is the class that handles HTTP-requests for the `content.jsp`. It builds HTTP connections to the external server (e.g. Wikipedia) to retrieve the content page. It also parses the content and reads the title-tag for storing it into the database. The parsing identifies relative links and replaces them with absolute links. It also processes the replace-list defined in the database by the server. This replaces e.g. search buttons, external calls or scripts that try to break out from an iframe with non harming values.

Figure 5.7: The Java classes that were developed for this software project including their methods and constants. They are used within Tomcat from the Java Server Pages and consist of seven classes. The *WatcherThread* runs permanently and initiates the starting of a new game when users connect.

5.7.1 Replacing Content

Some interesting point may be to emphasize on how the content of the original page needs to be altered in order to function well in our framework. The Listing 5.1 shows the main changes that need to be done. The first two lines are to replace so called frame-killers[5]. These concern JavaScripts that reside on the content page that want to identify their page being displayed inside an iframe and want to break out by setting the browser URL to the inner location URL. This is a common feature for sites that do not want to be linked or displayed inside other pages. Within our framework this would break our control over the inner site, which is exactly what the frame-killer wants to do. Reasons for that behavior may be that they want to prevent click-jacking within their site, which is a malicious technique of tricking a user to click on something different from what he perceives he is clicking on. In order to keep control over our game, we need to remove these parts of the original code. The next step is to replace all link target (`href`) by "`content.jsp?url=`" appended with its previous target (URL encoded). This is to assure, that clicked links point to our framework again. Otherwise we would lose control over the site after the first clicked link. The last thing to do is to make all other references absolute. This is because the `content.jsp` resides on our server and therefore every relative URL would point to our server and not to the original server. A reference to a picture for example would be replaced by an absolute reference to the original server plus the correct path of the picture as we can load these things directly and do not need to load them across our server. We also need to get rid of links that open another target window. We simply erase the target. Actions that got applied to input elements like submit, button, radio or check box get their mouse- and keypress-listeners erased. This is implemented in the script `linktracker.js`.

```
1 #regular expressions
2 \(\s*(top|self)(.|)(location|)\s*!=\s*(top|self)(.|)(location|)\s*\) -> (false)
3 \(\s*top.frames.length -> (0
4
5 #link targets (a, area)
6 url -> content.jsp?url=absoluteReference(url)
7
8 #absolute references (src, href, action)
9 url -> http://server.com/url
10 http://other.com/url -> http://other.com/url
11 /path/url -> http://server.com/path/url
```

Listing 5.1: Some content from the remote site needs to be replaced for the program in order to work properly. For example frame killers that would delete the game control or special link targets that would open links inside other/new tabs.

[5] http://en.wikipedia.org/wiki/Framekiller

5.7.2 Running Threads

Game control is the only task that needs to be done periodically. Within the server, the `WatcherThread` takes care of the database connection and tries to renew it every five seconds when the connection gets lost. This can happen when MySql gets restarted or a network problem between the two parts occurs. Additionally, it checks if the game is expired to create a new game. This is only done when a user is actually connected, otherwise we would have many useless games with no participants. The old game also gets flagged whether a user completed the game successfully or not and the number of participants is also stored. The creation of a game takes two randomly selected destination pages from the sources table by using an SQL trick to "`order by rnd() limit 2`". So this thread does no time consuming tasks, even if the creation of a new game needs some database queries.

On the client (browser) side, inside the script `linktracker.js` a periodic timer is created that checks the server for status updates concerning the game every two seconds. This check is done by the script `stats.jsp`. To avoid caching by the browser or by a proxy, a random parameter is added. The left part of the screen shows the game information (see Section 5.6). When the server answered back that the game is expired, it also stops the navigation inside the content area and shows the new task.

5.8 Data Persistence

The database back-end is a MySql relational database the Java Servlet is connected to. It is not necessary that both run on the same machine, although it might avoid some headache when dealing with lossy connections or unreliable hardware. I chose MySql to be suitable for this task, as the program does not generate huge amount of data or needs to do high performance calculations on it. The database basically has five tables for the data and one table for storing parameters. The parameters store the current game-id and on which site the next game will be created. Other parameters are the game lifetime and global replacement strings that filter out special commands on the pages. See Table 5.9 for the parameter table layout.

5.8.1 Table Layout

An overview of the table layout is shown in Figure 5.8. Two tables mainly represent the Java classes `Game` and `User`. In the `Game` table, the game specific details like start and destination page, the time the game took place, how many participants and whether the game could be solved is stored. The `User` table consists of the user's session id, the originating IP address and the time the user joined their first game. The table `ClickPath` records every click a user does within a game. It is referred from the script *record.jsp* which is called directly from the user's clicks. It holds references to the specific user and the specific game as well as the link and the exact link text the user clicked. This is sometimes important if there is more than

one link to the same page e.g. a text and a picture. When clicking the picture, the alternate text of the `img`-tag is used. The table `Site` is for describing the source site the game runs on (e.g. Wikipedia). It only has the hostname, a description and a starting URL for fallback. The `Sources` table holds all URLs that can be used as starting or destination points for the games. It can be imported from an external web crawler (read Section 4.6 for more details) or it can be derived from previously stored click data. This is a good option for starting without doing a web crawling before. But the drawback is, that invalid URLs or error pages might be inside the click data, which are not suitable for start or destination points. Of course these kind of pages could be still filtered out by manually revising the data. The advantages of having all possible nodes available in a huge graph are pointed out in the Section 2.2.

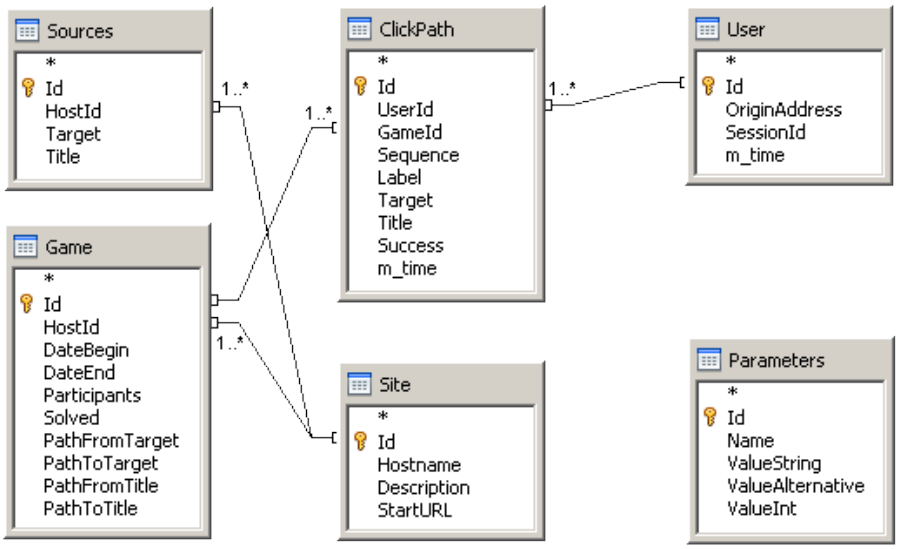

Figure 5.8: The database layout from the backend that stores the click data from the participants as well as the list of possible source and destination pages from the target sites. Also adjustable parameters are stored which can be edited within the administration interface. The table *ClickPath* stores the URL including the label which was clicked by the user as well as the title of the page. This title can then be copied to the *Sources* table.

Field	Type
Id	int(11)
UserId	int(11)
GameId	int(11)
Sequence	int(11)
Label	varchar(250)
Target	varchar(250)
Title	varchar(250)
Success	boolean

Table 5.4: Database layout of the table "ClickPath"

Field	Type
Id	int(11)
HostId	int(11)
Target	varchar(250)
Title	varchar(250)

Table 5.5: Database layout of the table "Sources"

Field	Type
Id	int(11)
Hostname	varchar(250)
Description	varchar(250)
StartURL	varchar(250)

Table 5.6: Database layout of the table "Site"

5.8.2 Data Types and Space

Data types are especially important when dealing with a big amount of data. In this project, every click from a user that participates in a game is recorded. This generates about 500 bytes data every ten seconds per user. The stored data is the target link URL, the label text from the click target and the page title. Refer to Table 5.4 below. The page title can complete the source and destination table is titles are missing there from the web spidering process. When there are ten users playing simultaneously, the total amount of gathered data is about 1.7 megabyte per hour. Dependent on the scale the program should be deployed, these value needs to be considered. Every participant that connects to the game is stored in the table "User" - see Table 5.8. Every game that has been openend is stored in the table "Game" - Table 5.7.

Also a list of available pages needs to be stored in the table "Sources" (5.5). This can be the output of an external spidering process (see Section 4.6) or the list of URLs from previously recorded click data from the program itself. These pages are later used to determine the source

Field	Type
Id	int(11)
HostId	int(11)
DateBegin	datetime
DateEnd	datetime
Participants	int(11)
Solved	boolean
PathFromTarget	varchar(250)
PathToTarget	varchar(250)
PathFromTitle	varchar(250)
PathToTitle	varchar(250)

Table 5.7: Database layout of the table "Game"

Field	Type
Id	int(11)
OriginAddress	varchar(32)
SessionId	varchar(36)

Table 5.8: Database layout of the table "User"

Field	Type
Id	int(11)
Name	varchar(32)
ValueString	varchar(250)
ValueAlternative	varchar(250)
ValueInt	int(11)

Table 5.9: Database layout of the table "Parameters"

and the destination page in the beginning of every game. Anyway this data needs to be stored with URL and page title which is about 250 bytes per page. Additionally, the pages can/must exist for every target site. The sites are stored within the table "Site", as outlined in Table 5.6. So a larger collection of source pages in connection with some target sites may also consume a lot of space. Usually modern database systems like MySql can handle more than 10GB of data size.

5.9 Installation Guide and Data Handling

5.9.1 Installing the Software on the Server

First we need access to a web server to install and configure the following packages on the server. We need to have administration privileges in order to install the software:

- Apache Tomcat server
- Java JDK
- Database management system (MySql)

I used an online installation guide to do so on my Debian[6] Linux server. See listing 5.2 for the commands. When Apache Tomacat is successfully installed, we need to get the configuration of the user authentication right. Insert the section as described in the listing. After that, you should restart the service and reconnect to the web interface. The management application should now be accessible. This is where we have to deploy our application. But first, we need to install the MySql database server if it does not already exist. The shell commands are also provided at the bottom part on the previous listing. When finished installing, the database scheme of the database "clickdata" needs to be installed into the database system.

[6] https://www.debian.org/

The command is the last command at the bottom of the listing. This creates the necessary tables. After that, we need to make sure, that user name and password inside the application can access the database. This is done inside the source file "Database.java" in the method "openConnectionPool()". Change the values as desired and compile the project. After that we return to Apache Tomcat and it's management system. The software project hast to be deployed now by using the Tomcat manager tools. The .war file created from the programming environment can be uploaded via this form and the server installs it. Just click the button named "Select WAR file to upload" on the "Deploy" menu. This uploads the project and runs it. The software is then available on the path to the server appended by ":8080/ClickPath". The application can also be stopped or restarted inside this interface. When everything is working find, the configuration of the game can be made via the application itself: Go to the address on your server ":8080/ClickPath/admin.jsp" and follow Subsection 5.6.1 how to handle the administration interface of the program.

```
1 #assure to be root (sudo su OR su -)
2
3 apt-get update
4 apt-get install tomcat7
5
6 #test connection: http://your_domain_or_ip:8080
7
8 apt-get install tomcat7-admin tomcat7-examples tomcat7-docs
9 apt-get install default-jdk
10 apt-get install ant git
11 nano /etc/tomcat7/tomcat-users.xml
12
13 #insert following section:
14 <tomcat-users>
15  <user username="admin" password="mypassword" roles="manager-gui,admin-gui"/>
16 </tomcat-users>
17
18 service tomcat7 restart
19
20 #manage your applications here:
21 #manager tools: http://your_domain_or_ip:8080/manager/html
22 #virtual hosts: http://your_domain_or_ip:8080/host-manager/html
23
24 #installing mysql
25 apt-get install mysql-server mysql-client
26 mysqladmin -u root -h localhost password 'mypassword'
27
28 #import database
29 mysql -u root -p database < clickdata.sql
```

Listing 5.2: These commands are necessary to get Apache Tomacat installed on a Debian Linux system. It then provides the Tomcat Web Application Manager where your applications can be deployed. The last lines are for installing the MySql database and importing the database scheme for the program.

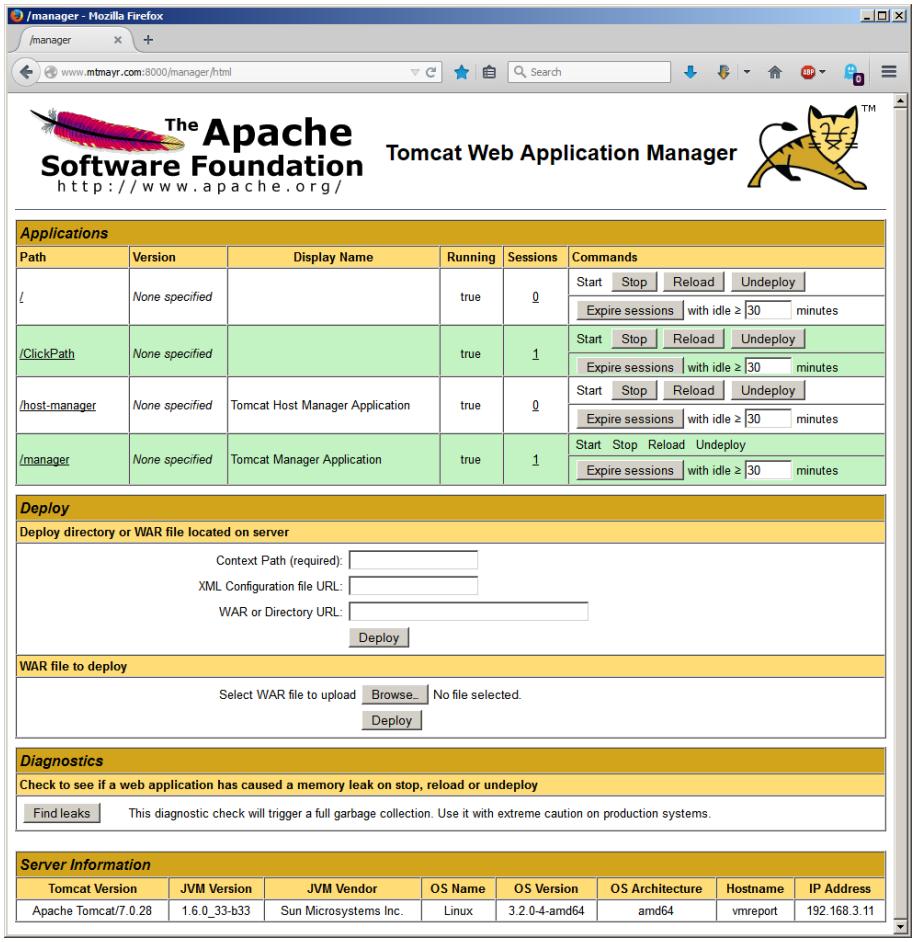

Figure 5.9: When installing Apache Tomcat on the server, the software project can be deployed by using the Tomcat manager tools. The .war file created from the programming environment can be uploaded via this form and the server installs it, runs it and makes the software available on it's specific path. The manager tool can handle more projects at the same time.

5.9.2 Exporting Recorded Data

Investigating recorded click data is normally done via the administration interface which is described in Subsection 5.6.1. When the data is processed further, direct extraction of the table data is very useful. It is also useful to automate this process and probably import the data into another table for the evaluation. When exporting the data, the following tables are relevant. Figure 5.10 shows the data that is created for every single game. It consists of start and destination page and some additional values like if the game was solved and how many participants there were. The start and destination page are stored as web address as well as the title of the content page. It is also recorded when the game started and when it finished. The "Id" of the game will be used in the next table. Figure 5.11 show the table data for every click a user performed. It shows where the user clicked ("Title"), the page address the link lead him to ("Target") and the precise time ("m_time") when the link was clicked. The "Sequence"

field means the number of clicks he already performed during this game. The game number "GameId" and which user "UserId" the click caused are also stored, so that we can investigate which task from the "Game" table he had to accomplish and where he came from. For more information about the table layout, please continue reading Subsection 5.8.1.

Id	HostId	DateBegin	DateEnd	PathFromTarget	PathToTarget	PathFromTitle	PathToTitle	Solved	Part
2.504	5	2014-03-20 03:03:14	2014-03-20 03:05:14	http://www.imdb.de/title/tt0330	http://www.imdb.de/title/tt0338751/?re	Harry Potter und der Feuerkelch (2	Aviator (2004) - IMDb	0	0
2.506	5	2014-03-20 14:19:53	2014-03-20 14:21:53	http://www.imdb.de/title/tt0926(	http://www.imdb.de/name/nm0000138	Harry Potter und die Heiligtümer de	Leonardo DiCaprio - Biography	0	0
2.507	5	2014-03-20 14:21:58	2014-03-20 14:23:58	http://www.imdb.de/title/tt0338	http://www.imdb.de/title/tt0264464/?re	Aviator (2004) - IMDb	Catch Me If You Can (2002) - IM	0	0
2.508	5	2014-03-20 14:24:03	2014-03-20 14:26:03	http://www.imdb.de/title/tt1201(	http://www.imdb.de/title/tt0120689/?re	Harry Potter und die Heiligtümer de	The Green Mile (1999) - IMDb	0	0
2.509	5	2014-03-20 21:15:36	2014-03-20 21:17:36	http://www.imdb.de/title/tt1201(	http://www.imdb.de/title/tt1375666/?re	Harry Potter und die Heiligtümer de	Inception (2010) - IMDb	0	1
2.510	5	2014-03-20 21:17:41	2014-03-20 21:19:41	http://www.imdb.de/title/tt0295	http://www.imdb.de/title/tt0120689/?re	Harry Potter und die Kammer des S	The Green Mile (1999) - IMDb	0	0
2.511	5	2014-03-20 21:19:46	2014-03-20 21:21:46	http://www.imdb.de/title/tt0120	http://www.imdb.de/title/tt0120689/?re	Titanic (1997) - IMDb	The Green Mile (1999) - IMDb	0	1
2.512	5	2014-03-20 21:21:51	2014-03-20 21:23:51	http://www.imdb.de/name/nm0	http://www.imdb.de/title/tt0338751/?re	Daniel Radcliffe - IMDb	Aviator (2004) - IMDb	0	1
2.515	5	2014-03-20 21:28:06	2014-03-20 21:30:06	http://www.imdb.de/name/nm0(	http://www.imdb.de/name/nm0000158	James Cameron - IMDb	Tom Hanks - IMDb	0	1
2.521	3	2014-03-20 21:40:36	2014-03-20 21:42:36	http://de.wikipedia.org/wiki/Gra	http://de.wikipedia.org/wiki/Himmelsk	Graphentheorie Wikipedia	Astronomisches Objekt Wikipedi	0	1
2.523	3	2014-03-30 21:14:48	2014-03-30 21:16:48	http://de.wikipedia.org/wiki/Him	http://de.wikipedia.org/wiki/Platon	Astronomisches Objekt Wikipedia	Platon Wikipedia	0	1
2.524	3	2014-03-30 21:16:53	2014-03-30 21:18:53	http://de.wikipedia.org/wiki/Nat	http://de.wikipedia.org/wiki/Leonhard_	Naturwissenschaft Wikipedia	Leonhard Euler Wikipedia	0	1
2.525	3	2014-03-30 21:18:58	2014-03-30 21:20:58	http://de.wikipedia.org/wiki/Plat	http://de.wikipedia.org/wiki/K%C3%B	Platon Wikipedia	Königsberger Brückenproblem W	0	1
2.526	3	2014-03-30 21:21:03	2014-03-30 21:23:03	http://de.wikipedia.org/wiki/Nat	http://de.wikipedia.org/wiki/Himmelsk	Naturwissenschaft Wikipedia	Astronomisches Objekt Wikipedi	0	1
2.527	3	2014-03-30 21:23:08	2014-03-30 21:25:08	http://de.wikipedia.org/wiki/Him	http://de.wikipedia.org/wiki/Sokrates	Astronomisches Objekt Wikipedia	Sokrates Wikipedia	0	2
2.531	3	2014-04-18 16:51:31	2014-04-18 16:53:31	http://de.wikipedia.org/wiki/Lec	http://de.wikipedia.org/wiki/Graphentl	Leonhard Euler Wikipedia	Graphentheorie Wikipedia	1	1
2.534	3	2014-09-12 20:17:59	2014-09-12 20:19:59	http://de.wikipedia.org/wiki/Nat	http://de.wikipedia.org/wiki/K%C3%B	Naturwissenschaft Wikipedia	Königsberger Brückenproblem W	0	0
2.535	3	2014-09-12 20:20:04	2014-09-12 20:22:04	http://de.wikipedia.org/wiki/Him	http://de.wikipedia.org/wiki/Platon	Astronomisches Objekt Wikipedia	Platon Wikipedia	0	0
2.536	3	2014-09-12 20:22:09	2014-09-12 20:24:09	http://de.wikipedia.org/wiki/Aris	http://de.wikipedia.org/wiki/Sokrates	Aristoteles Wikipedia	Sokrates Wikipedia	0	0
2.537	3	2014-09-12 20:24:14	2014-09-12 20:26:14	http://de.wikipedia.org/wiki/Mar	http://de.wikipedia.org/wiki/Leonhard_	Mars (Planet) Wikipedia	Leonhard Euler Wikipedia	0	0
2.538	3	2014-09-12 20:26:19	2014-09-12 20:28:19	http://de.wikipedia.org/wiki/Sok	http://de.wikipedia.org/wiki/K%C3%B	Sokrates Wikipedia	Königsberger Brückenproblem W	0	0
2.539	3	2014-09-12 20:28:24	2014-09-12 20:30:24	http://de.wikipedia.org/wiki/Aris	http://de.wikipedia.org/wiki/Platon	Aristoteles Wikipedia	Platon Wikipedia	0	0
2.542	3	2014-09-12 20:34:39	2014-09-12 20:36:39	http://de.wikipedia.org/wiki/K%	http://de.wikipedia.org/wiki/Aristoteles	Königsberger Brückenproblem Wil	Aristoteles Wikipedia	0	0
2.543	3	2014-09-12 20:36:44	2014-09-12 20:38:44	http://de.wikipedia.org/wiki/Mar	http://de.wikipedia.org/wiki/K%C3%B	Mars (Planet) Wikipedia	Königsberger Brückenproblem W	0	0
2.544	3	2014-09-12 20:38:49	2014-09-12 20:40:49	http://de.wikipedia.org/wiki/Aris	http://de.wikipedia.org/wiki/Leonhard_	Aristoteles Wikipedia	Leonhard Euler Wikipedia	0	0
2.545	3	2014-09-12 20:40:54	2014-09-12 20:42:54	http://de.wikipedia.org/wiki/Aris	http://de.wikipedia.org/wiki/K%C3%B	Aristoteles Wikipedia	Königsberger Brückenproblem W	0	0
2.546	3	2014-09-12 20:42:59	2014-09-12 20:44:59	http://de.wikipedia.org/wiki/Lec	http://de.wikipedia.org/wiki/Graphentl	Leonhard Euler Wikipedia	Graphentheorie Wikipedia	0	0
2.548	3	2014-09-12 20:47:09	2014-09-12 20:49:09	http://de.wikipedia.org/wiki/K%	http://de.wikipedia.org/wiki/Johannes_	Königsberger Brückenproblem Wil	Johannes Kepler Wikipedia	0	0

Figure 5.10: The game data itself consists of path from one site to the other and additional values like if the game was solved and how many participants there were. The start and destination page are stored as web address as well as the title of the content page. It is also recorded when the game started and when it finished. The "Id" of the game will be used in the next table "ClickPath".

Id	UserId	GameId	Seq	Label	Target	Title	Succ	m_time
4.416	1.537	2.524	4	Geometrie	http://de.wikipedia.org/wiki/Geometrie	Geometrie Wikipedia	0	2014-03-30 21:18:23
4.417	1.537	2.524	5	Geometrie	http://de.wikipedia.org/wiki/Kategorie:Geometrie	Kategorie:Geometrie Wikipedia	0	2014-03-30 21:18:39
4.418	1.537	2.524	6	Algebraische Geometrie	http://de.wikipedia.org/wiki/Kategorie:Algebraische_Ge	Kategorie:Algebraische Geometrie Wikipedia	0	2014-03-30 21:18:45
4.419	1.537	2.525	1	Platon Wikipedia	http://de.wikipedia.org/wiki/Platon	Platon Wikipedia	0	2014-03-30 21:19:02
4.420	1.537	2.525	2	Demokrit	http://de.wikipedia.org/wiki/Demokrit	Demokrit Wikipedia	0	2014-03-30 21:20:33
4.421	1.537	2.525	3	Abdera	http://de.wikipedia.org/wiki/Abdera	Abdera Wikipedia	0	2014-03-30 21:20:38
4.422	1.537	2.525	4	Griechische Kolonisation	http://de.wikipedia.org/wiki/Kategorie:Griechische_Kolo	Kategorie:Griechische Kolonisation Wikipedia	0	2014-03-30 21:20:46
4.423	1.537	2.526	1	Naturwissenschaft Wikipedia	http://de.wikipedia.org/wiki/Naturwissenschaft	Naturwissenschaft Wikipedia	0	2014-03-30 21:21:13
4.424	1.537	2.526	2	Astronomie	http://de.wikipedia.org/wiki/Astronomie	Astronomie Wikipedia	0	2014-03-30 21:21:25
4.425	1.537	2.526	3	Himmelskörper	http://de.wikipedia.org/wiki/Astronomisches_Objekt	Astronomisches Objekt Wikipedia	0	2014-03-30 21:21:39
4.426	1.537	2.526	4	Astronomisches Objekt	http://de.wikipedia.org/wiki/Kategorie:Astronomisches_(	Kategorie:Astronomisches Objekt Wikipedia	0	2014-03-30 21:22:10
4.427	1.537	2.526	5	Liste (astronomische Objekte)	http://de.wikipedia.org/wiki/Kategorie:Liste_(astronomis	Kategorie:Liste (astronomische Objekte) Wiki	0	2014-03-30 21:22:23
4.428	1.537	2.526	6	Astronomisches Objekt	http://de.wikipedia.org/wiki/Kategorie:Astronomisches_(	Kategorie:Astronomisches Objekt Wikipedia	0	2014-03-30 21:22:30
4.429	1.537	2.526	7	Astronomisches Objekt	http://de.wikipedia.org/wiki/Astronomisches_Objekt	Astronomisches Objekt Wikipedia	0	2014-03-30 21:22:37
4.430	1.538	2.527	1	Astronomisches Objekt Wikipedia	http://de.wikipedia.org/wiki/Himmelsk%C3%B6rper	Astronomisches Objekt Wikipedia	0	2014-03-30 21:24:55
4.431	1.538	2.527	2	Sterne	http://de.wikipedia.org/wiki/Stern	Stern Wikipedia	0	2014-03-30 21:25:00
4.432	1.537	2.527	1	Astronomisches Objekt Wikipedia	http://de.wikipedia.org/wiki/Himmelsk%C3%B6rper	Astronomisches Objekt Wikipedia	0	2014-03-30 21:25:04
4.433	1.538	2.527	3	Schwerkraft	http://de.wikipedia.org/wiki/Gravitation	Gravitation Wikipedia	0	2014-03-30 21:25:12
4.434	1.537	2.528	1	Terminator (1984) - IMDb	http://www.imdb.de/title/tt0088247/?ref_=nm_flmg_wr_2	Terminator (1984) - IMDb	0	2014-03-30 21:25:24
4.435	1.537	2.528	2	James Cameron	http://www.imdb.de/name/nm0000116/?ref_=tt_ov_dr	James Cameron - IMDb	0	2014-03-30 21:25:40
4.436	1.537	2.528	3	Titanic (1997)	http://www.imdb.de/title/tt0120338/?ref_=nm_knf_t2	Titanic (1997) - IMDb	0	2014-03-30 21:25:50
4.437	1.538	2.528	1	Terminator (1984) - IMDb	http://www.imdb.de/title/tt0088247/?ref_=nm_flmg_wr_2	Terminator (1984) - IMDb	0	2014-03-30 21:26:03
4.438	1.538	2.528	2	Arnold Schwarzenegger	http://www.imdb.de/name/nm0000216/?ref_=tt_cl_t1	Arnold Schwarzenegger - IMDb	0	2014-03-30 21:26:14
4.439	1.539	2.531	1	Leonhard Euler Wikipedia	http://de.wikipedia.org/wiki/Leonhard_Euler	Leonhard Euler Wikipedia	0	2014-04-18 16:51:35
4.440	1.539	2.531	2	Graphentheoretiker	http://de.wikipedia.org/wiki/Kategorie:Graphentheoretik	Kategorie:Graphentheoretiker Wikipedia	0	2014-04-18 16:52:01
4.441	1.539	2.531	3	Graphentheorie	http://de.wikipedia.org/wiki/Graphentheorie	Graphentheorie Wikipedia	1	2014-04-18 16:52:06
4.442	1.541	2.550	1	Königsberger Brückenproblem Wikipedia	http://de.wikipedia.org/wiki/K%C3%B6nigsberger_Br%(	Königsberger Brückenproblem Wikipedia	0	2014-11-26 02:19:57
4.443	1.541	2.550	2	Leonhard Euler	http://de.wikipedia.org/wiki/Leonhard_Euler	Leonhard Euler Wikipedia	0	2014-11-26 02:20:06
4.444	1.541	2.550	3	Mathematiker	http://de.wikipedia.org/wiki/Mathematiker	Mathematiker Wikipedia	0	2014-11-26 02:20:12
4.445	1.541	2.550	4	Liste bedeutender Mathematiker	http://de.wikipedia.org/wiki/Liste_bedeutender_Mathem	Liste bedeutender Mathematiker Wikipedia	0	2014-11-26 02:20:23
4.446	1.541	2.550	5	Johannes Kepler	http://de.wikipedia.org/wiki/Johannes_Kepler	Johannes Kepler Wikipedia	1	2014-11-26 02:20:29
4.447	1.542	2.552	1	Leonhard Euler Wikipedia	http://de.wikipedia.org/wiki/Leonhard_Euler	Leonhard Euler Wikipedia	0	2014-12-07 16:40:59
4.448	1.542	2.552	2	Zahlentheorie	http://de.wikipedia.org/wiki/Zahlentheorie	Zahlentheorie Wikipedia	0	2014-12-07 16:41:18
4.449	1.542	2.552	3	Mathematik	http://de.wikipedia.org/wiki/Mathematik	Mathematik Wikipedia	0	2014-12-07 16:41:30
4.450	1.542	2.552	4	Naturwissenschaft	http://de.wikipedia.org/wiki/Naturwissenschaft	Naturwissenschaft Wikipedia	1	2014-12-07 16:41:54
4.451	1.542	2.553	1	Leonhard Euler Wikipedia	http://de.wikipedia.org/wiki/Leonhard_Euler	Leonhard Euler Wikipedia	0	2014-12-07 16:43:23
4.452	1.542	2.553	2	Analysis	http://de.wikipedia.org/wiki/Analysis	Analysis Wikipedia	0	2014-12-07 16:43:34
4.453	1.542	2.553	3	Differenzenquotienten	http://de.wikipedia.org/wiki/Differenzenquotient	Differenzenquotient Wikipedia	0	2014-12-07 16:43:41
4.454	1.542	2.553	4	Leonhard Euler Wikipedia	http://de.wikipedia.org/wiki/Leonhard_Euler	Leonhard Euler Wikipedia	0	2014-12-07 16:44:02

Figure 5.11: This is an overview of the data stored inside the table "ClickPath". It shows exactly where the user clicked "Title", the page address the link lead him to "Target" and the precise time "m_time" when the link was clicked. The "Sequence" field means the number of clicks he already performed during this game. The game number "GameId" and which user "UserId" the click caused are also stored.

6 Conclusion

6.1 Project Summary

To summarize the software project, I have to say that it was quite interesting to implement the web application. It was a lot easier to deal with Java Servlets and JSP (Section 4.3) than I thought. As I like Java as programming language, it was not hard for me to get the servlet engine running on my server and do the first steps on the web application. A lot more challenging was the implementation of the replacing procedures within the JavaScript (Section 4.4) and Ajax (Section 4.5) part. JavaScript is a lot more powerful than expected, and manipulating pages inside the browser is surprisingly easy and has deep impact on the functionality of a web page. I would be definitely be interested in having and evaluating the test results from a big number of participants of the game. The used technologies fitted perfectly for my needs. That does not mean, that other technologies would be insufficient, but obviously I applied them correctly.

From the user's point of view it is amazing how many data can be gathered from a normal browsing process (Section 3.1). Usually a functioning JavaScript engine on the client side is sufficient to be able to access sensitive data and probably send them to another server. Cross site scripting (Subsection 4.9.2) attacks appear in various ways and cannot be entirely prevented. So there is a trade-off between functionality of a web page and security for the user. This data is collected by all bigger companies that have web pages we sometimes visit on a daily basis. It is doubtful for me, that this is allowed and unrestricted by law.

From the theoretical view, it is impressive, how information can be classified and put into networks. It is also impressive how the human brain navigates through such kind of networks. We can learn more about our way of thinking and how our brain classifies it's data. Probably this is also related to our language and goes deep into our psyche. And last but not least, to see this information as a small part inside a globally connected network of information and data that was created by mankind. Also the structure and the function of the Internet was interesting to investigate. Especially the fact that we can extract the way we build our social contacts around us (Subsection 2.2.2) can be investigated using Facebook or similar tools. This way, we can learn more about our role in society and the with which people we interact.

6.2 Future Work

The next step to do would be to let volunteers play the game, for example as a task in a lecture at the university, so that we can investigate the data from the collected paths. A visualization of the graph with the weights between the nodes could be applicable as well as lists sorted by weight. In the next step, it would be interesting to predict where the user will click on the page he is currently viewing. There could be an overlay at each page that shows the outgoing link probability percentages or a heat map where most of the people click depending on the destination page or generally.

Further work could be to find additional web sites, where the game can be adapted on, to probably get other aspects or just additional interesting results on the browsing behavior of the users too.

Listings

List of Figures

List of Tables

Bibliography

[1] Russell Ackoff. From data to wisdom. *Journal of Applied Systems Analysis*, 16:3–9, 1989.

[2] Hilal Al Maqbali, Falk Scholer, James A. Thom, and Mingfang Wu. Using eye tracking for evaluating web search interfaces. In *Proceedings of the 18th Australasian Document Computing Symposium*, ADCS '13, pages 2–9, New York, NY, USA, 2013. ACM.

[3] Lars Backstrom, Paolo Boldi, Marco Rosa, Johan Ugander, and Sebastiano Vigna. Four degrees of separation. In *Proceedings of the 4th Annual ACM Web Science Conference*, WebSci '12, pages 33–42, New York, NY, USA, 2012. ACM.

[4] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Publishing Company, USA, 2nd edition, 2008.

[5] B. Benatallah, F. Casati, and F. Toumani. Web service conversation modeling: a cornerstone for e-business automation. *Internet Computing, IEEE*, 8(1):46–54, Jan 2004.

[6] Categories, lists, and navigation templates within Wikipedia, 2014. http://en.wikipedia.org/wiki/Wikipedia:Categories,_lists,_and_navigation_templates.

[7] Emmanuel Cecchet, Anupam Chanda, Sameh Elnikety, Julie Marguerite, and Willy Zwaenepoel. Performance comparison of middleware architectures for generating dynamic web content. In *Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware*, Middleware '03, pages 242–261, New York, NY, USA, 2003. Springer-Verlag New York, Inc.

[8] Nick Craswell and Martin Szummer. Random walks on the click graph. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 239–246, New York, NY, USA, 2007. ACM.

[9] Douglas Crockford. *JavaScript: The Good Parts*. O'Reilly Media, Inc., 2008.

[10] Douglas Crockford. Douglas Crockford - 'JavaScript: The Good Parts, Google TechTalk', 2009. http://www.youtube.com/watch?v=hQVTIJBZook.

[11] Douglas Crockford. Douglas Crockford - 'Really JavaScript?, JSConf 2010', 2010. http://blip.tv/file/3755495?utm_source=player_embedded.

[12] Douglas Crockford. JSLint, code quality tool, 2011. http://www.jslint.com/.

[13] Content Security Policy, W3C technical reports index, 2014. http://www.w3.org/TR/CSP/.

[14] Database definition on Merriam Webster, 2015. http://www.merriam-webster.com/dictionary/database.

[15] Cristian Duda, Gianni Frey, Donald Kossmann, and Chong Zhou. Ajaxsearch: Crawling, indexing and searching web 2.0 applications. *Proc. VLDB Endow.*, 1(2):1440–1443, August 2008.

[16] Georges E. Dupret and Benjamin Piwowarski. A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 331–338, New York, NY, USA, 2008. ACM.

[17] Neal Gafter. The definition of closures, 2007. http://gafter.blogspot.com/2007/01/definition-of-closures.html.

[18] Jesse James Garrett. Ajax: A New Approach to Web Applications. Adaptive Path LLC, 2005-02-18, 2005. http://web.archive.org/web/20080702075113/http://www.adaptivepath.com/ideas/essays/archives/000385.php.

[19] Marty Hall and Larry Brown. *Core Servlets and JavaServer Pages (JSP)*. Pearson Education, 2 edition, 2003. http://pdf.coreservlets.com/.

[20] Helic, Denis and Trattner, Christoph. Network Navigability: "Theorie and Applications", 2009. http://de.slideshare.net/christophtrattner/slides-13583771.

[21] IBM. The Internet of Things, 2010. http://asmarterplanet.com/ and http://www.youtube.com/watch?v=sfEbMV295Kk.

[22] Dongseok Jang, Ranjit Jhala, Sorin Lerner, and Hovav Shacham. An empirical study of privacy-violating information flows in javascript web applications. In *Proceedings of the 17th ACM conference on Computer and communications security*, CCS '10, pages 270–283, New York, NY, USA, 2010. ACM.

[23] M. Johns, B. Engelmann, and J. Posegga. Xssds: Server-side detection of cross-site scripting attacks. In *Computer Security Applications Conference, 2008. ACSAC 2008. Annual*, pages 335–344, Dec 2008.

[24] JSON Group. JavaScript Object Notation, interchange format, 2014. http://www.json.org/js.html.

[25] Lakhwinder Kumar, Hardeep Singh, and Ramandeep Kaur. Web analytics and metrics: A survey. In *Proceedings of the International Conference on Advances in Computing, Communications and Informatics*, ICACCI '12, pages 966–971, New York, NY, USA, 2012. ACM.

[26] Laurence Bradford. Ruby on Rails vs Python and Django: Which Should a Beginner Learn?, 2015. https://www.coursereport.com/resources/ruby-on-rails-vs-python-and-django-which-should-a-beginner-learn.

[27] Jayant Madhavan, David Ko, Lucja Kot, Vignesh Ganapathy, Alex Rasmussen, and Alon Halevy. Google's deep web crawl. *Proc. VLDB Endow.*, 1(2):1241–1252, August 2008.

[28] Mari-Carmen Marcos, David F. Nettleton, and Diego Sáez-Trumper. A user study of web search session behaviour using eye tracking data. In *Proceedings of the 26th Annual BCS Interaction Specialist Group Conference on People and Computers*, BCS-HCI '12, pages 262–267, Swinton, UK, UK, 2012. British Computer Society.

[29] Viktor Mayer-Schönberger and K. Cukier. *Big Data: A Revolution That Will Transform How We Live, Work and Think. Viktor Mayer-Schnberger and Kenneth Cukier*. John Murray Publishers, UK, 2013.

[30] Stanley Milgram. The small world problem. *Psychology Today*, 67(1):61–67, 1967.

[31] John Mueller. Understanding SOAP and REST basics, 2014. http://blog.smartbear.com/apis/understanding-soap-and-rest-basics/.

[32] Mark Newman. The physics of networks. *Physics Today*, American Institute of Physics, S-0031-9228-0811-010-9(2):33–38, November 2008.

[33] Simone Paolo Ponzetto and Roberto Navigli. Large-scale taxonomy mapping for restructuring and integrating wikipedia. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, IJCAI'09, pages 2083–2088, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[34] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. *Recommender Systems Handbook*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

[35] Jennifer Rowley. The wisdom hierarchy: Representations of the dikw hierarchy. *J. Inf. Sci.*, 33(2):163–180, April 2007.

[36] Aju Thalappillil Scaria, Rose Marie Philip, Robert West, and Jure Leskovec. The last click: Why users give up information network navigation. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pages 213–222, New York, NY, USA, 2014. ACM.

[37] Stoyan Stefanov. *JavaScript Patterns*. O'Reilly Media, Inc., 1st edition, 2010.

[38] Christoph Trattner and Denis Helic. Linking related documents: combining tag clouds and search queries. In *Proceedings of the 10th international conference on Web engineering*, ICWE'10, pages 486–489, Berlin, Heidelberg, 2010. Springer-Verlag.

[39] Christoph Trattner, Christian Körner, and Denis Helic. Enhancing the navigability of social tagging systems with tag taxonomies. In *Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies*, i-KNOW '11, pages 18:1–18:8, New York, NY, USA, 2011. ACM.

[40] Christoph Trattner, Philipp Singer, Denis Helic, and Markus Strohmaier. Exploring the differences and similarities between hierarchical decentralized search and human navigation in information networks. In *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*, i-KNOW '12, pages 14:1–14:8, New York, NY, USA, 2012. ACM.

[41] Jeffrey Travers, Stanley Milgram, Jeffrey Travers, and Stanley Milgram. An experimental study of the small world problem. *Sociometry*, 32:425–443, 1969.

[42] Danny P. Wallace. *Knowledge Management: Historical and Cross-Disciplinary Themes*. Libraries Unlimited, 1st edition, 2007.

[43] Web application ARchive, 2014. http://www.oracle.com/technetwork/java/whitepaper-135196.html.

[44] Extensible Markup Language (XML), W3C, 2014. http://www.w3.org/XML/.

[45] Yujiu Yang, Xinyi Shu, and Wenhuang Liu. A probability click tracking model analysis of web search results. In *Proceedings of the 17th International Conference on Neural Information Processing: Theory and Algorithms - Volume Part I*, ICONIP'10, pages 322–329, Berlin, Heidelberg, 2010. Springer-Verlag.

[46] Yiyu Yao, Yi Zeng, Ning Zhong, and Xiangji Huang. Knowledge retrieval (kr). In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, WI '07, pages 729–735, Washington, DC, USA, 2007. IEEE Computer Society.

[47] Chaim Zins. Conceptual approaches for defining data, information, and knowledge. *Journal of the American Society for Information Science and Technology*, 58:479–493, 2007.