TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Informatik XIX

Enhancing Tagging Systems with a Flexible, Faceted Organization Structure

Alexander Sebastian Steinhoff

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. M. Bichler
Prüfer der Dissertation:	
1.	Univ.-Prof. Dr. F. Matthes
2.	Univ.-Prof. Dr. J. Schlichter

Die Dissertation wurde am 19.03.2013 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 06.06.2013 angenommen.

Zusammenfassung

Die Menge an digitalen Informationen, die Menschen täglich erstellen und nutzen, sei es während der Arbeit oder privat, nimmt stetig zu. Um mit diesem Wachstum Schritt halten zu können, werden Werkzeuge benötigt, die es erlauben, effizient auf diese Informationen zuzugreifen und sie zu verwalten. Klassische Organisationsstrukturen, wie zum Beispiel Verzeichnisbäume, sind oft nicht flexibel genug um der Vielfalt der Informationsobjekte gerecht zu werden.

In den letzten Jahren, wurde das sogenannte *Social Tagging* populär. Entsprechende Systeme ermöglichen es Nutzern, gemeinsam große Mengen von Informationsobjekten im Hinblick auf verschiedenste Aspekte mit Hilfe sogenannter *Tags* zu kategorisieren. Dadurch dass es keine Beschränkungen beim Vergeben dieser Schlagworte gibt, sind die entstehenden Strukturen, die auch als *Folksonomien* bezeichnet werden, eher dazu geeignet eine Informationssammlung ziellos zu durchstöbern als systematisch eine bestimmte Information zu finden.

Ziel dieser Arbeit ist, zu untersuchen wie Tagging-Systeme verbessert werden können, damit es mit ihrer Hilfe möglich ist, systematisch und zuverlässig auf Informationen zuzugreifen. Dieses Ziel beschränkt sich auf die Anwendung von Tagging-Systemen in einem vergleichsweise kleinen Maßstab, also zum Beispiel um innerhalb einer geschlossenen Gruppe von Nutzern Informationen zu organisieren.

Beginnend mit einer Analyse der Besonderheiten beim Einsatz von Tagging-Systemen im Kleinen, beschreibt die Arbeit die Entwicklung einer flexiblen Organisationsstruktur, die es erlaubt, zwei Arten von Beziehungen zwischen Tags herzustellen: Zwischen allgemeinen und spezielleren Tags kann eine *Subsumptions-Beziehung* definiert werden und Tags können außerdem zu *Facetten* gruppiert werden. Eine Facette deckt einen gewissen Aspekt der Kategorisierung in einem Teil der Informationssammlung ab. Diese Basiskonstrukte können flexibel zu komplexeren Strukturen kombiniert werden.

Es wurde ein Prototyp entwickelt, der eine integrierte Benutzerschnittstelle für den Zugriff auf Informationen sowie für die Verwaltung der Tag-basierten Organisationsstruktur umfasst. Um die Bedienbarkeit dieser Schnittstelle zu bewerten und den Nutzen des Ansatzes im Allgemeinen zu überprüfen, wurden Nutzerstudien und Interviews durchgeführt. Schließlich wurde das System in einem Experiment einem herkömmlichen Tagging-System gegenübergestellt, indem der Erfolg der Nutzer bei der gezielten Suche und der Exploration von Informationssammlungen verglichen wurde.

Abstract

To keep pace with the increasing amount of digital information being produced and utilized by people in their everyday lives, tools are needed that allow to manage and access information resources efficiently. Traditional organization structures such as folder trees are often not flexible enough to accommodate the great diversity of information resources.

In recent years, social tagging became popular as a means to collaboratively categorize masses of resources with regard to various different aspects. However, due to the uncontrolled nature of tags, the resulting organization structures, so-called folksonomies are rather suited for aimlessly browsing a collection of items than for accessing information resources in a systematic way.

The objective of this thesis is to examine how the utility of tagging systems for the systematic and reliable access to information can be improved when being applied on a comparably small scale, i.e., when being used by an individual or a closed group of users to organize collections of a manageable size.

Starting with an analysis of the particular properties of small-scale tagging systems, the thesis describes the development of a flexible organization structure that allows to define two kinds of relations between tags: between general and more specific tags, *subsumption* relations can be established, and tags can be grouped to *facets*. A facet covers one particular aspect of categorization in a specific subset of a resource collection. These basic constructs can be flexibly combined to create complex organization structures.

A prototype system was developed that includes an integrated user interface for accessing information resources and for managing the tag-based organization structure. To assess the usability of the interface and the utility of the approach in general, interviews and user studies were conducted. Finally, performance measures for search and exploration tasks were quantitatively compared with those of normal tagging systems in an experiment.

Acknowledgments

During the writing of this thesis in the past four years, many people have contributed to its success in one way or the other. At this point, I want to thank all these people and express my appreciation for their support.

First, I would like to thank my doctoral advisor Prof. Dr. Florian Matthes for providing me with the opportunity to write this thesis and for giving me a lot of time and freedom to develop my research ideas. I also owe him valuable hints that had a great impact on the evolution of the prototype that has been developed in the course of this work. I also thank Prof. Dr. Johann Schlichter, the second advisor of this thesis, for reviewing earlier versions of this work and for giving me valuable feedback.

My thanks go to my colleagues of the Chair for Informatics 19 (sebis) at the Department of Informatics of the Technische Universität München. Marta Infante Abreu, Dr. Sabine Buckl, Dr. Thomas Büchner, Matheus Hauder, Yoel Abreu Lee, Ivan Monahov, Dr. Christian Neubert, Sascha Roth, Alexander Schneider, Christopher Schulz, and Dr. Christian M. Schweda were always open for the discussion of new ideas, shared their opinions with me and gave me valuable advice. I particularly appreciate the countless hours of discussions with Dr. Thomas Büchner and Dr. Christian Neubert.

Furthermore, I would like to thank the students Joan Boixadós, Stefan Deser, Tobias Konsek, Felix Michel, and Bernhard Waltl who wrote student theses that I had the opportunity to supervise. We had many productive discussions and each of their theses made an important contribution to the success of this work.

I express my gratitude to all my friends who understood that I had to spend time on this thesis and could not participate in leisure activities as often as I would have liked to. Especially, I thank Florian Gengnagel and Karin Pfister for their positive influence on my nutrition by providing healthy alternatives to convenience food.

In particular, I would like to thank my partner Anna Katharina Brehm for her patience and her constant encouragement. You always had a good sense of when it was necesarry to convince me to take a day off to restore my motivation.

My special thanks go to my parents Dr. Christiane Steinhoff and Dr. Alfons Steinhoff, who accommodated me for many weeks and provided a productive work environment. I am very grateful for your always unconditional support — not only during the past four years.

Garching b. München, 19.03.2013

Alexander Steinhoff

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction and Overview

This thesis covers the development and the evaluation of a novel extension for tagging systems that allows to organize and systematically access tagged information resources. After motivating the work in Section 1.1, we define the goal of this thesis in Section 1.2. It is subsequently rendered more precisely in a set of research questions in Section 1.3. Section 1.4 gives a brief summary of design science research and explains how this work conforms to the research guidelines proposed by Hevner et alii [He04]. Finally, Section 1.5 briefly outlines the structure of this document.

1.1. Motivation

The amount of digital information being produced and utilized by people in their everyday lives is increasing steadily [Ga08]. This applies to information in their work context as well as to private information managed on local hard drives or various online platforms, such as social networks or photo sharing sites. To keep pace with this information growth, people need tools that allow them to manage and access information resources[1] efficiently.

One kind of such tools are full-text search engines. In the last decades, advances on this field together with the availability of cheap computing resources have made it dramatically easier for users to find particular textual resources they are looking for. However, while this technology has become an integral part of most modern information systems, it is not sufficient for the management of digital information resources. In addition to the fact that search is usually limited to textual resources, there are several tasks for which a structured organization of digital contents is essential. When it comes to the management of access rights or the deletion of specific sets of resources, full-text search alone is of little help. It also fails when users cannot specify exactly what they are looking for, either because they do not

[1] For the sake of brevity, we will often use the term *resource* as a synonmym for information resource.

know the very terms occurring in the document or they do not even have a specific target but want to browse a collection of resources in a rather exploratory way [Te04].

Different organization schemes can be applied for making collections of digital resources manageable. One of the simplest and most widespread is certainly the nested folder structure. It effectively represents a tree of labeled nodes to which individual information resources are assigned. Tree structures reflect how items are organized in the physical world: rooms contain shelves, shelves contain folders, folders contain documents, et cetera. This makes them easy to understand by users. However, for the same reason their flexibility is limited and they are hard to evolve and adapt to changing needs regarding the organization of digital contents.

Contrary to such rigid structures, *social tagging* became popular on the web in recent years. The term describes a phenomenon that emerged during the middle of the last decade: masses of users independently assign freely-chosen text labels (i.e., *tags*) to web resources, either in order to retrieve them later themselves or to recommend the resources to others. This is done using *tagging systems* provided through specific web platforms usually focusing on a particular type of information resource, such as *Flickr*[2] for photos or *delicious*[3] for websites. The entirety of the tags on such platforms is commonly referred to as a *folksonomy*. It is a portmonteau word , i.e., a new word being formed by combining two existing words, based on "folk" and "taxonomy" and goes back to the information architect Thomas Vander Wal [VW07]. A folksonomy is a fuzzy structure consisting of implicit, weighted relationships of tags which are embodied in the cooccurrences of tags: the relatedness of two tags depends on the number of resources to which both tags are assigned.

Social tagging facilitates the categorization of large quantities of information resources by millions of independent users regarding a variety of different aspects of these resources.

When the first social tagging platforms emerged on the web, the phenomenon was embraced with great enthusiasm and it induced high expectations [Sh05, We05]. However, it is not without problems. One major challenge is the inconsistent usage of tags. Spelling variations, abbreviations, synonyms within a language or simply the use of different languages lead to the situation that many different representations (i.e., tags) exist for a single concept [GH06, Ma04]. In consequence, this means that a search yields different results depending on which tag is used. Additionally, the same tags are used to represent different concepts due to differing conceptions among users. Again, the quality of search results is impeded because many irrelevant resources are returned. Finally, even if the vocabulary could be harmonized, the resources would not be tagged exhaustively with regard to all aspects being relevant for the users of the folksonomy: Even if everybody agreed upon what for instance the label "portrait" exactly means, not all respective resources would be tagged accordingly. For some users it seems sufficient to assign a tag like "person" since they are not aware of the fact that other users care about different types of images showing persons. All these insufficiencies regarding the tag assignments make it difficult to access resources, particularly in a systematic way.

Despite their shortcomings, folksonomies can be very useful in certain scenarios. The unstructured browsing of a collection of resources stimulates serendipity and lets people discover resources and links between topics that otherwise would have remained concealed [HMHS06, Pe09]. Additionally, the enormously large number of resources can compen-

[2] http://www.flickr.com (accessed 6th of February 2012).
[3] http://delicious.com (accessed 6th of February 2012).

sate for the bad indexing quality: When searching for a particular category of resources it can be assumed that at least some relevant resources are retrieved. When browsing a collection of millions of items, it is usually not critical to retrieve all relevant resources. Finally, the large number of users can improve the indexing quality: When several users independently tag the same resource, it is likely that a tag that someone later uses to access the resource has actually been assigned before [Fu06].

But people do not tag only for a large anonymous audience. Tagging was also adopted by users as a very lightweight and flexible alternative to traditional organization structures such as folders. It is applied on a smaller scale to organize collections of personal resources as well as collections being selectively shared with other people. The bookmarking platform delicious for instance allows both, finding resources tagged by others, and tagging one's own collection of bookmarks. Tagging functionality is also included in tools such as collaboration platforms or email clients as a supplementary feature, although it is usually not the primary means for organizing information.

In such contexts, where a clear organization of resources is required, the abovementioned limitations of tagging turn out to be more severe. While resources can — in theory — be categorized very flexibly, the lack of structure makes it hard to maintain a consistent and extensive categorization. Additionally, the ways in which a collection can be accessed are very limited: Users can usually search for resources having a particular tag assigned or they browse a collection using so called *tag clouds*. Searching for a specific tag is a reasonable way to access an information resource when at least some of its tags are known and these tags are specific enough to yield a reasonably small result set. Otherwise the problems are similar to those of full-text search. A tag cloud is a visualization of the list of the most frequently used tags within a collection of resources. While tag clouds can give users a notion of the general topics of resources, they are of little help when trying to locate a particular piece of information [SC08].

Summarizing the above, it can be said that tagging systems allow for a lightweight and flexible categorization of information resources but the lack of structure mostly leads to a situation where the efforts that have been put into the categorization are not adequately reflected in easy and efficient ways to reliably retrieve particular resources nor is there the possibility to browse a collection systematically.

1.2. Goal of this thesis

The general objectives of this thesis are

1. to examine how the accessibility of information can be improved when tagging systems are applied on a comparably small scale and
2. to develop and evaluate appropriate enhancements for tagging systems.

This means that the thesis focuses on scenarios where tagging systems are applied by an individual or a closed group of users to organize collections of a manageable size. In such scenarios, the purpose of tagging is not to recommend resources to others or to facilitate serendipitous findings. Instead, a tool for the reliable storage and retrieval of information is

required. Reliable in this case means that the users' tagging efforts really make the tagged resources accessible in a way that the users intended while assigning the tags. One measure for this is the precision and recall of searches: With regard to a single tag, it is desirable that a search yields exactly the relevant resources, all of them but as little irrelevant resources as possible. Otherwise, a user who assigns a tag to an information resource and who has a particular concept in mind that is being expressed with the tag cannot be sure that another user searching for the same concept can access the resource later. Accessing the resource means both, directly finding it with a search and narrowing down the collection systematically until the desired resource can be selected from a small set of results.

To achieve the latter one could simply resort to common organization structures such as directory trees. However, most of the flexibility provided by tagging systems would be lost. As a consequence we aim to preserve as much as possible of the strengths of tagging systems, such as the flexible evolution of the organization structure and the ability to categorize resources with regard to various independent aspects. Our vision is that enhanced tagging systems can ultimately replace traditional organization structures in many scenarios by combining the flexibility of tags with the rigidity of tree structures.

The approach developed in this thesis is motivated by the assumption that an organization structure can better suit the needs of its users when it is free to evolve over time in response to changes of the managed information and corresponding access patterns. Contrary to that stand rigid, static structures — being possibly defined by an external authority — and decoupled from the information actually managed. Those structures come in the form of thesauri, classification schemes, ontologies or other knowledge representations. This work can be generally characterized as an attempt to transform folksonomies into clearer structures by giving users the ability to manually sharpen the fuzzy implicit relations among tags and to maintain a once established relation in the further development of the structure.

In addition to the enhancement of tagging systems it shall also be examined whether the contents present in current tagging systems, such as large-scale folksonomies, can be retroactively organized using the new tools and structural concepts in such a way that access to the resources can be measurably improved. This includes targeted access to specific resources as well as rather exploratory tasks. Given the size and extensiveness of folksonomies being created by thousands or even millions of users, it is apparent that it is infeasible to establish an organization structure that is suitable for all contents and all users. Hence, we focus on particular subsets of these collections that are concerned with a certain domain for which the meaning of and the relations among most tags can be assumed to be unambiguous. This way, not only the efforts spent during the categorization of the resources can be harnessed to establish an organization that is useful for any user browsing the collection. But being able to structure a given collection would also facilitate the import of an individual user's personal resources from an existing systems into a new enhanced system.

Improving tagging systems comprises several aspects that are closely connected:

- **Organization structure.** The data managed by tagging systems consists of little more than the assignments of tags to information resources. In particular, tags have no explicit relations among each other but they are only implicitly related through the resources to which they are assigned. Extending this simple structure is one opportunity to enhance tagging systems.

- **Algorithms.** New and better algorithms to analyze the tag assignments in a tagging system can help users to work with a collection of resources in a structured way. This includes statistical techniques, such as clustering, and approaches that take into account the meaning of tags. The latter typically borrow from fields such as natural language processing or the semantic web.
- **User interface.** Changes in the organization structure and new algorithms can both have implications on the user interface being applied for the indexing, i.e., tagging, of resources as well as for later access to the collection. Apart from that, the improvement of the user interface alone represents an option to enhance tagging systems.

Although all of the above aspects are touched to some extent in this thesis, the focus is on the organization structure and the user interface.

1.3. Research questions

The general goal of enhancing tagging systems in order to support the organization of information in small and medium sized collections of resources is broken down into several research questions covering distinct aspects of the research endeavor. These more specific questions guide the development of the tagging system enhancements as well as their subsequent evaluation and thus are intended to allow an assessment of the success of the research efforts undertaken.

As stated before, the central question is the following:

> ***Research question 1:*** Can tagging systems be enhanced in a way that makes them suitable for the management and structured organization of collections of information resources?

The decisive point in this question is that tagging systems are not only used to annotate collections of resources being later aimlessly browsed in order to make serendipitous findings. Instead, it shall be possible to use them to store information in a structured way so that it can be reliably and systematically retrieved later.

The goal of enhancing tagging systems clearly leads to the question in how far and to what extent an improved system is superior to conventional tagging systems:

> ***Research question 2:*** Is it possible to provide a more reliable and efficient way to access information? Can users find specific information resources faster than with conventional systems?

Since accessing a particular piece of information is an important and frequently performed task in organized collections of information resources, it shall be examined whether the suitability of tagging systems for this task can be improved.

The following question focuses on the better exploitation of the contents of present tagging systems:

> ***Research question 3:*** How can conventional folksonomies or parts thereof be transformed to benefit from the developed enhancements?

It will be examined to what extent such a transformation is possible and which tools are required to facilitate it.

One of the main advantages of tagging systems is their simplicity. The average user of such a system cannot be expected to be familiar with conceptual modeling or ontologies. Yet, structuring a folksonomy, e.g., by relating tags to each other, is in fact the creation of a certain kind of knowledge representation that very likely exceeds the complexity of more common structures that users are familiar with. Therefore, the development of a user interface that allows users to apply the tagging system enhancements developed in this thesis without prior training is challenging. This is addressed by the next research question:

> ***Research question 4:*** In how far are non-expert users able to understand and apply the structuring concepts developed in this thesis?

A meaningful organization of a collection of resources is not possible without knowledge about the respective subject area and this knowledge is to some extent reflected in the organization structure. While the purpose of an organization structure is primarily the organization of the knowledge expressed in the information resources, the representation of knowledge beyond the contents of individual resources is an interesting aspect. This leads to the next research question:

> ***Research question 5:*** To what extent does the structuring of the contents of tagging systems allow to express knowledge regarding the relation of the categories being used?

The answer to this question is embodied in the design of the tagging system enhancements developed in this thesis. While it is desirable to employ structural concepts with a high expressiveness, this goal conflicts with the attempt to design a system that can be easily understood by users.

Additionally, it will to be examined whether the modifications of tagging systems have a negative impact on their flexibility, one of their greatest advantages:

> ***Research question 6:*** In how far does the structured organization of resources limit the flexibility of tagging systems?

This includes the flexibility users have while tagging new resources as well as the way collections of resources can be accessed.

Finally, the development of a prototype system is an integral part of this research endeavor.

> ***Research question 7:*** How can a system be technically realized that allows the structuring of collections of tagged information resources as well as browsing these structured contents in a systematic way?

1.4. Research design

The research presented in this thesis follows the conceptual framework for design science research in information systems as presented by Hevner et al. in [He04].

Rather than studying the application and impact of existing solutions, design science research is characterized by the creation and application of new artifacts. These artifacts "extend the boundaries of human problem solving and organizational capabilities by providing intellectual as well as computational tools" [He04]. This means that theories are not only developed by observation and the consultation of existing knowledge but they are derived from the practical experimentation with new tools, systems, models or methods. Further, the assessment of the new artifacts is based on the utility they provide with respect to solving a certain problem or class of problems.

The framework presented by Hevner et al. is meant to serve as a frame of reference for all information systems research activities including the evaluation and presentation of the research. It is graphically depicted in Figure 1.1. The research process is described as a loop that repeatedly switches between the construction and refinement of the new artifact and its evaluation. These two distinct phases are referred to as "build and evaluate".

The research process is framed by and interacts with the environment and the knowledge base. The environment consists of people, organizations and technology. It represents the domain of the problems and business needs that are addressed by the research. Further it is the context in which the new artifacts are applied and evaluated.

During both phases of the research cycle, the foundations contained in the knowledge base are applied. The development phase builds on existing frameworks and models being the result of prior information systems research. Similarly, the evaluation is based on well-proven methodologies and guidelines. The outcome of the research contributes to the knowledge base and can itself inform future research.

Along with the conceptual framework, Hevner et al. provide guidelines for understanding, executing, and evaluating design science research. In the following, these guidelines are briefly summarized together with a short description of how they were taken into account in the research presented in this thesis.

1. Design as an artifact. Hevner et al. briefly summarize their first guideline as follows: "Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation" [He04]. In this thesis, an extension for tagging systems is developed. Since it is not only conceptually described but a viable software prototype was implemented as well, the artifact clearly falls into the *instantiation* category.

2. Problem relevance. This guideline states that the research has to address important and relevant business problems. Goal of this thesis is to develop an extension for tagging systems that improves their utility when applied on a small scale, i.e., for personal use or in teams. The general problems connected to tagging systems are well studied (see Section 2.3.5). The recent popularity gain of tags and folksonomies has resulted in the implementation of tagging functionality in many enterprise collaboration platforms [BMN09]. The ubiquitous need to organize information and the simplicity of tags further lead to applications in various software applications, such as tools for personal information management, weblogs, email

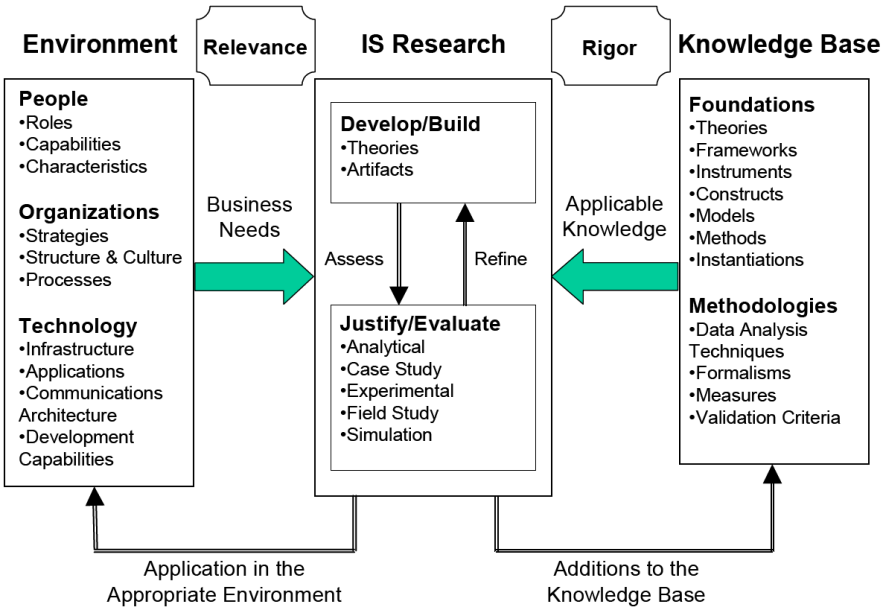


Figure 1.1.: Information systems research framework [He04].

clients et cetera. Nevertheless, the insufficiencies of tagging systems still limit their practical applicability [SC08, HR08]. Thus, an improvement of the effectiveness of such systems can have a major impact on the way information is organized in the business world.

3. Design evaluation. Hevner et al. require that the "utility, quality, and efficacy of a design artifact" is rigorously demonstrated [He04]. For this purpose, several design evaluation methods are suggested. For the evaluation of the tagging system extension, we apply observational and testing methods. In a usability study consisting of a series of user tests, it is observed how users understand, use and apply the developed prototype (see Section 6.1). To demonstrate utility, a controlled experiment is conducted in which users perform tasks using different configurations of a tagging system (see Section 6.2). This allows to compare their performance when using the extension to the baseline, a conventional tagging system.

4. Research contributions. The main research contribution of this thesis is the design artifact itself, i.e., a tagging system extension that improves the accessibility of the system's contents and that allows to establish an organization structure based on the folksonomy tags. The feasibility of the approach is demonstrated by implementing a prototypical extension to a real world enterprise collaboration platform (see Chapter 5). We further empirically assessed the effectiveness of this prototype. Finally, we present additional findings regarding the implicit structure inherent in any tagging systems which were obtained in the course of this research (see Section 3.3).

5. Research rigor. Hevner et al. state that "Design-science research requires the application of rigorous methods in both the construction and evaluation of the designed artifact" [He04]. The theoretical foundations of tagging system as well as previous research on relevant phenomena were extensively studied. They are covered in detail in Chapter 2. The design goals and the guidelines followed during the design process were precisely formulated. Finally, a

comprehensive description of the evaluation is given that includes detailed descriptions of the user study and the experiment as well as performance metrics.

6. Design as a search process. The tagging system extension developed in this thesis is the result of an iterative process that can be roughly divided into two phases. In the first phase it was attempted to establish structure in tagging systems only by harmonizing the tag assignments. When it turned out that implicit tag relations alone were not practical to achieve the research goal, the approach was adapted based on the insights gained so far. The integration of the prototype in a collaboration platform that was also used by the author and his colleagues for their everyday work allowed to quickly evaluate minor modifications of the implementation.

7. Communication of research. This guideline requires that "design-science research [is] presented both to technology-oriented as well as management oriented audiences" [He04]. The former is addressed by providing implementation details and demonstrating that the approach can be integrated in an enterprise collaboration platform. The performance measures obtained during the final experiments serve the latter.

1.5. Outline of the thesis

After giving a brief overview of different knowledge organization structures and their historic roots, Chapter 2 covers tagging system fundamentals and relevant research in this field.

Chapter 3 describes the design process, the goals of the process, as well as principles and guidelines that were followed. The second half of Chapter 3 covers the idea of implicit tag relations. This idea was the basis for the first versions of our prototype and had a major influence on the later development. We further summarize our research contributions in this context.

In Chapter 4, the *TACKO*[4] system, the tagging system extension developed in this thesis, is described. The first part of Chapter 4 covers the TACKO organization structure, the data structure underlying TACKO. It is formally described and its characteristics are illustrated with examples. Subsequently, an integrated user interface being suitable for browsing a collection of information resources as well as for managing the organization structure is presented.

Implementation details are covered in Chapter 5. It is shown how the TACKO organization structure is mapped to appropriate data structures and how queries and updates can be implemented efficiently. Furthermore, it is presented how the tagging system extension fits into the architecture of a web-based enterprise collaboration platform.

Chapter 6 consists of two parts. The first covers a usability study that examines how the TACKO data structure and the user interface are understood and applied by users. Subsequently, the experimental evaluation of TACKO is described. The web-based crowdsourcing service *Mobileworks*[5] is used to recruit more than 150 participants who perform different

[4]TACKO stands for "Tag-based Context-dependent Knowledge Organization"
[5]http://www.mobileworks.com, (accessed 18th of December 2012).

tasks with the TACKO system and with a conventional tagging system that serves as the baseline.

Chapter 7 gives an extensive overview of related work. This includes a variety of different approaches attempting to derive structure from folksonomies, such as clustering approaches or algorithms generating tree structures from tags. Additionally, alternative tag-based knowledge organization systems are presented.

The thesis concludes with Chapter 8 summarizing what could be achieved in this thesis and giving an outlook on further research opportunities.

CHAPTER 2

From Inventory Lists to Tagging Systems

Structures for the organization of knowledge have been applied since the ancient world. While not always perceived as such, they are ubiquitous and permanently applied in our everyday lives. With the advent of the digital age and the associated growth of information, knowledge organization structures gained further importance. It is unimaginable to manage the amount of information an enterprise or even a modern individual is confronted with today without appropriate structures and systems.

To avoid confusion, it has to be noted at this point, that for our purpose the distinction of *knowledge* and *information* is of minor importance. Our usage of the term *knowledge organization* follows Abbas who notes with regard to knowledge organization structures and systems:

> "The systems contain discrete pieces of data or information as representations of individual expressions, but what is being characterized, thereby organized, in these structures is the knowledge base of the *organizer(s)* and how they perceive and make meaningful their knowledge of their environments, as well as the documents or objects within the collection" [Ab10, pp. 12-13].

Thus, when we speak of *information* resources being organized, these resources may nevertheless be representations of *knowledge* as well.

Section 2.1 gives a very brief overview of important, mostly formal, knowledge organization structures that have been developed from ancient history to the present. Subsequently, we cover the fundamental organization patterns underlying these structures in Section 2.2. We specifically highlight the properties of tree structures and facets since they are of particular importance for the TACKO structure presented later in this thesis.

Finally, Section 2.3 is dedicated to the fundamentals of tagging systems. It gives an overview of their basic properties and their usage together with related research results.

2.1. A brief history of knowledge organization

This section can only give a very brief summary of the historical development of knowledge organization structures. The history of knowledge organization is closely connected to natural history, philosophy and cognitive science (cf. [Ab10]). It is inseparable from the fundamentals of classification that reach back to Aristotle. However, not all of these aspects can be covered in this work. We refer to Taylor and Joudrey [TJ09] and Abbas [Ab10] for a more comprehensive overview of the topic. Additionally, Lakoff provides an excellent overview of the different categorization paradigms [La87].

In the following, we characterize the most important structures applied for the organization of knowledge roughly following their temporal progression.

The earliest known forms of knowledge organization structures were applied as means for indexing collections of textual resources such as books or laws[Ab10]. They date back to about 2000 BC. Being merely lists, they focused on capturing the contents of a collection of items. Some of them made the items retrievable by associating each entry with the physical placement of the item. In other words, they represented inventory lists. While this is still an essential function of later more elaborate structures, the increasing size of the collections made it necessary to provide more and better ways of accessing their contents then just by the title of the document.

The invention of the printing press in the 15th century led to a dramatic increase in the number of books and gave rise to a new profession: the bibliographer [TJ09, p. 71]. In the following centuries, the first cataloging standards emerged. The first library catalogs including author and subject indexes appeared. The first known use of cross-references also dates back to the 15th century [Ab10, p. 22].

In 1791, the French government demanded the cataloging of the nation's library collections which led to the first national code and the development of card catalogues [TJ09, p. 73]. New cataloging codes were developed during the 19th century and 1908 the first international cataloging code was created as a result of a British-American collaboration [Ab10, p.23]. Such standards determine which data is contained in bibliographic records and how it is represented. Bibliographic cataloging standards are continuously developed and refined up to the present to reflect changed demands in information access and to accommodate newer kinds of media. Beside the author, year of publication, and other information, the most interesting part of the *metadata* in such a record is concerned with the *subject* of the document.

One way to organize subject terms is to use so-called *controlled vocabularies* which are "simply [...] authorized lists of subject terms that represent the world of knowledge or the terminology within a specific discipline [...]" [Ab10, p. 24]. Common examples are the *Library of Congress Subject Headings* (LCSH) and the *Medical Subject Headings* (MeSH). The latter represents a thesaurus rather than a subject headings lists. Abbas draws the line between subject headings lists and thesauri as follows: "Subject headings lists generally include as their scope the entire world of knowledge [...]" [Ab10, p. 24] while Thesauri "are generally designed for use within a specific discipline, occupation, or proprietary database system" [Ab10, p. 26]. In both, the terms can have hierarchical relationships as well as references to related terms. Taylor and Joudrey name *ontologies* as a third form of controlled vocabularies. According to them, an ontology "[...] is a systematic account of the entities and their relationships found in a

particular domain" [TJ09, p. 336], however, "they do not select one term to be a preferred or authorized term" [TJ09, p. 335].

In contrast to controlled vocabularies, in *classification schemes* not words or terms are assigned to a document but the document is placed into one of a set of classes [TJ09, p. 375]. A prominent example is the *Dewey Decimal Classification Scheme* (DDC), published by Melvin Dewey in 1876 [Ab10, p. 28]. Abbas describes it as follows:

> "The DDC divided the world of knowledge into ten classes, with each of those further divided into ten divisions, and each of those further divided into ten sections for a total of 1,000 potential categories into which books could be classified" [Ab10, p. 28].

The number 536 for instance denoted the category "Heat" in division 500 "Science", section 530 "Physics". DDC is a so-called *enumerative* classification scheme with a fixed set of classes to which objects are assigned. Opposed to that stand *analytico-synthetic* — or *faceted* — schemes. In the 1930s the indian librarian S.R. Ranganathan developed his seminal *Colon Classification* scheme (CC) [Ra33]. A facet can be considered a dimension of classification independent of other facets. Ranganathan proposed five facets usually being referred to by the acronym PMEST:

- Personality (P): the focal or most specific subject;
- Material or Matter (M): a component;
- Energy (E): an activity, operation, or process;
- Space (S): a specific or generic place or location; and
- Time (T): a chronological period, a year, a season, et cetera.

Compare for instance [TJ09]. When classifying a document, the classifier determines the classes the document belongs to with regard to these facets and concatenates the respective codes to one notation. The first component is mandatory, the others can be added as applicable [Ab10, 28]. Facets as a means for the categorization of information resources are covered in more detail in Section 2.2.2.

Beside the organization of library collections, other notable structures emerged in the past centuries in the domain of natural history. Structures for the classification of natural objects were created, as for instance the Linnean Hierarchy that was developed by Carl Linnaeus in 1735 [Ab10, p. 70]. It provided rules for unambiguously assigning animal species and plants a place in a large tree structure. Another frequently mentioned structure in this context is the periodic table of elements which also provides an important frame of reference for scientific communication. Particularly in biology, the term *taxonomy* denotes systematic and usually hierarchical arrangements of objects. However, it has to be noted here, that the term has been used in a broader sense in recent years. Taylor and Joudrey observe a "resurgence [of the term taxonomy] for systems of categorization that are used on the Web and in intranet systems" [TJ09, p. 376]. One of the most comprehensive definitions is given by Lambe who denotes with the term taxonomy "the rules or conventions of order and arrangement" [La07, p. 4]. He further notes, that the variety of the term's usage "reflects the extent to which taxonomies can enter daily life, from classes of people to the disposition of things, ideas, times and places" [La07, p. 4].

While the creation as well as the application of traditional taxonomies has been a field of activity for professionals, nowadays people can hardly avoid to be engaged in the organization of information in one way or the other. Not only are we used to utilize the complex, often faceted, structures that guide us through web-based product catalogues of millions of items. We also organize digital pieces of information regularly, be it emails or documents on our hard drives. The structures being applied are manifold. However, they are mostly variations of the fundamental structures covered in Section 2.2.

Eventually, with the advent of the *Web 2.0* [O'05] it became possible for virtually everybody to share knowledge and information with others at low cost. Social sharing sites, such as for instance *Flickr* or *delicious*, allow their users to contribute items — in these cases photos and links to web resources — to immense pools of information objects that everybody can browse. The size, the diversity and the dynamic growth of the collections, together with the large number of users make it infeasible to apply static and rigid structures for the organization of their contents. In this context, *social tagging* became popular during the last ten years. It allows users to freely categorize information resources with arbitrary text-labels, i.e., tags. As already mentioned earlier, the entirety of the tags in a collection is referred to as a *folksonomy*. Tagging systems will be covered in more detail in Section 2.3. The recent popularity of tags is not limited to the web. Tagging functionality is integrated into various information systems, from email clients to enterprise collaboration platforms, and complements other organization structures.

2.2. Fundamental knowledge organization patterns

When comparing different structures for organizing knowledge, one can observe certain principles or patterns that are not specifically associated with a single structure but occur with slight variations in several of them. We distinguish four such patterns:

- **Keywords:** Associating information resources with an arbitrary number of terms or keywords is a characteristic of different organization structures. In its purest form it is realized in tagging systems that do not allow to define any relationships among terms. In controlled vocabularies, a restricted set of keywords is organized using additional structures.
- **Links:** With the term link, we denote a certain type of connection among two entities that exists independently of other such connections. There is for instance no transitivity property as in hierarchical relationships. Links are often directed and can be represented as cross-references in a set of documents — as in hypertext — or references among terms in a thesaurus or an ontology.
- **Tree structures:** Trees are the determining pattern in many formal and informal organization structures. They can be found in file systems, taxonomies, classification schemes and even in mind maps. Tree structures are covered in more detail in Section 2.2.1.
- **Facets:** Characteristic for faceted organization structures is that information resources are categorized with regard to different aspects or dimensions such as time or location. There is often a fixed set of facets. The categories within a single facet can be structured

themselves, for instance in a hierarchy. The properties of faceted organization structures are covered in Section 2.2.2.

Except for facets, these patterns have corresponding constructs in the *Simple Knowledge Organization System (SKOS)* semantic web standard developed by the W3C[1]. SKOS was created to capture the similarity of "[...] knowledge organization systems, such as thesauri, taxonomies, classification schemes and subject heading systems [...]" and to make them explicit in order "to enable data and technology sharing across diverse applications" [MB09]. It allows to define broader-narrower relationships between concepts as well as associative relationships for "related" terms. Although facets are not directly supported, there are efforts to extend the vocabulary to accommodate faceted schemes as well [PG08].

In the following sections, we cover the properties of tree structures and facets because they are of particular relevance for this work. Subsequently, Section 2.3 gives a broad overview of tagging systems and their properties.

2.2.1. Tree structures

Tree structures are probably the most frequently used structures for knowledge organization, be it in the form of folder trees in file systems or navigation structures on websites. Typically, such a tree consists of a set of categories, to which individual information resources are related. Each such category has exactly one superordinate or parent category, except for the root category. In the simplest case, an individual resource is assigned to exactly one node in the tree structure. A resource is accessed by navigating from the root of the tree to the category to which it is assigned. On each level of the tree, one of the subcategories is selected until the desired category is reached.

It has to be noted, that a tree structure not necessarily represents a *hierarchy* in the strict sense. In her analysis of different classificatory approaches, Kwasnik lists several particular properties of hierarchies, which clearly do not apply to all sets of categories (or classes as they are called in [Kw99]) organized in a tree structure [Kw99]:

- **Inclusiveness:** All sub-categories are included in the root-category.
- **Species/differentia:** There is only one type of relationship among the categories, e.g., an "is-a" relationship.
- **Inheritance:** Everything being true for a category applies to all sub-categories.
- **Transitivity:** Inheritance also applies to sub-categories of sub-categories, and so on.
- **Systematic and predictable rules for association and distinction:** There are clear rules regarding the assignment of categories to supercategories. Categories belonging to the same supercategory are systematically chosen so that they share certain properties and their respective subcategories differ with regard to a certain aspect.
- **Mutual exclusivity:** There is no semantic overlap between two categories on the same level. Assignments to the categories are unambiguous.

[1] The abbreviation stands for the World Wide Web Consortium.

- **Necessary and sufficient criteria:** The properties a category must have to belong to a certain supercategory are also sufficient to determine whether it actually does.

While it is aimed to meet these criteria in the construction of professional taxonomies for certain domains, such as animals or diseases, this is hardly the case for the less formal tree structures being used in personal information management. A typical path in a folder structure, such as "/public/events/photos" clearly contradicts several of the above requirements. Lambe notes that "[...] the 'scientific' sense of a taxonomy as a strict hierarchy is attractive from a distance, but rarely useful in practice when building knowledge management taxonomies" [La07, p. 23].

Kwasnik defines tree structures with a set of less restrictive requirements. One of them is again that systematic and predictable rules are applied in the arrangement of categories. While this is a desirable property of such a structure, our definition is more general: We understand a tree structure as a set of nodes N representing the categories of information resources. This set always contains an element r, being the root of the tree. Each node $n \in N$ is assigned a label $l \in L$ with L being the set of all labels. The label of a node is denoted by $label(n)$. In the following we assume labels are text literals. Each node except for the root node is assigned exactly one parent node, denoted by $parent(n)$. Different nodes can have the same label unless they are siblings, i.e., they share the same parent:

$$\forall n_1, n_2 \in N. \quad n_1 \neq n_2 \wedge parent(n_1) = parent(n_2) \quad \Rightarrow \quad label(n_1) \neq label(n_2)$$

A tree is connected, this means the root can be reached from each node by repeatedly following the parent links. In consequence the tree also contains no cycles. An individual information resource is assigned to exactly one node in the tree. In some cases, resources may only be assigned to the leaves of the tree, but we will assume in the following that they can be assigned to any node, as it is the case in most file systems: A folder can contain files as well as other folders.

The limitations of tree structures have been frequently discussed in literature, particularly in the context of file organization. They were the primary motivation for the development of many new information management solutions [Bl06, Do99, VAS09].

Single-inheritance, i.e., the fact that an information resource has to be assigned to exactly one category (or node) and categories have exactly one parent category, is probably the most severe limitation of tree structures. Assigning resources to categories is cognitively challenging [DL83] and often complicated because the assignment is not unambiguous. When a resource matches several categories in different branches of the tree, additional rules are required where to put it. When accessing resources, the same problem arises.

Their rigidity makes trees very inflexible. When it turns out that new entities do not fit well into the tree, "[...] the entire structure must be rethought and sometimes rebuilt" [Kw99].

The *limited support for facets* is another disadvantage of trees. Although, different facets can be mixed in a tree structure, they cannot be used independently. Consider for instance the path "/events/2013/germany/" in a folder structure. It contains facets for the type of entity, the time, and the location. However, they always have to be selected in this order. This is related to what Kwasnik calls "selective perspective" [Kw99], which means the over-

emphasizing a certain aspect. Furthermore, users have to browse to "maximum specificity" to find a resource [Bl06]. If the year is not known, the resource cannot be accessed.

Implementations of tree structures can contain extensions that allow to circumvent the restrictiveness of the structure. For instance symbolic links in file systems can be used to connect files or directories across different branches of the tree.

However, in spite of all the criticism, it has to be noted that the rigid nature of tree structures can also be valuable for users. In [Jo05], the authors report, that folder trees can serve more purposes than the storage and retrieval of documents: They also reflect the understanding of a certain domain. This means they *represent knowledge* that goes beyond what is captured in the individual resources.

Tree structures are also easy to understand. They naturally reflect the organization of physical items in space, i.e., an item is located exactly in one place and the set of all places can be subdivided into nested regions.

Finally, another advantage that tree structures have over faceted schemes is that the "depth of categorization" can be adapted to the frequency of items in a category. When many resources are assigned to a category, they can be further subdivided. If there are only few, this is not required [Kw99].

2.2.2. Facets

As already noted in Section 2.1, faceted classification was first introduced in the context of library science by Ranganathan [Ra33]. Facets can be understood as independent orthogonal dimensions of classification. While Ranganathan specifically suggested the five facets personality, material/matter, energy, space, and time, the notion of faceted classification is nowadays more abstract.

Taylor and Joudrey define faceted classification in the context of bibliographic classification:

> "Faceted classification, like hierarchical/enumerative schemes, attempts to include all possible subjects, but it does not do this by creating a singular place in a hierarchy for each topic (with its own specified number). Faceted classification is made up of many discrete topics. It is an attempt to divide the universe of knowledge into its component parts, and then to gather those parts into individual categories or facets" [TJ09, p. 387].

With a more general point of view, Lambe notes:

> "A facet then can be defined as a base taxonomy comprising only one of the fundamental dimensions in which content can be analysed. Working with faceted classification schemes or taxonomies simply means working with a number of base taxonomies, where each document or piece of content is analysed according to one or more base taxonomies" [La07, p. 37].

Facets are represented by sets of concepts or categories to which individual information resources are related. These concepts can be hierarchically organized within a facet, e.g., a facet for geographic location can include concepts for continents, countries, and cities. Whether

information resources can be related only to terminal concepts (cities in this case) or also to more general ones depends on the particular application. If an individual resource can be related to several concepts of a facet (for instance, a personnel record can be assigned to "Java" and "C#" of a facet "programming skills") we speak of a *multi-valued* facet. If only one concept or category can be selected, we call it a *single-valued* facet.

A good overview and formalization of faceted search and faceted classification is given by Sacco and Tzitzikas in [ST09a]. They subsume the application of facets to information bases and the exploratory approach to this information under the term *dynamic taxonomies*. Taxonomies are considered dynamic because access to the collection of information resources is usually given to the user in form of an interface that allows the dynamic filtering of the collection using arbitrary combinations of concepts of multiple facets. A user could for example search for all resources in the categories "research project" and "2010" (of a "time" facet) and would retrieve a list of matching resources together with a set of options to further refine the search filter.

Presumably due to this structured yet flexible way of accessing information resources, faceted search interfaces have become the de facto standard for accessing goods and services on modern e-commerce websites. Figure 2.1 shows how facets are applied on amazon.com[2]. The layout is typical for faceted search interfaces. The list of items occupies the most space and the facets are shown at the left border of the screen. The list can be further restricted by selecting facet options.

The most obvious advantage of facets is that they provide several access paths to resources. This becomes obvious when comparing faceted classification to the organization of information resources in trees: Imagine that a person is browsing the website of a broker for used cars looking for a blue convertible. If all the offerings are arranged in a tree structure that organizes them by make and year of construction at the first two levels, finding such a car is very difficult. Instead, with a faceted search interface it is not only easy to find appropriate cars by selecting color and type of car but one even retrieves an overview of all relevant cars that can be further narrowed down by the other facets. However, this is not required. One can even find a resource in the collection without knowing the respective categories in all facets.

Sacco points out, that "dynamic and unexpected relationships" can be discovered when exploring the collection [Sa09, p. 13]. The person looking for cars could for instance find out, that cars of a specific make are only found in a certain price range.

In comparison with tree structures, faceted taxonomies can also be adapted more easily. A facet can be changed or extended without affecting the other facets.

Sacco also notes that dynamic taxonomies (he uses the term for faceted search systems) lead to an "extremely fast convergence to small result sets" during search [Sa09]. This refers to a comparison with classical taxonomies and the number of documents a searching user has to inspect manually [SFT09]. The advantage of facets is that only two facets of 10 concepts each allow for 100 simple concept combinations that can be used to filter the collection. In a classical taxonomy these combinations would have to be represented by 100 compound concepts.

Although the application of facets comes with many benefits, there are disadvantages as

[2] http://www.amazon.com, (accessed 23rd of January 2013).

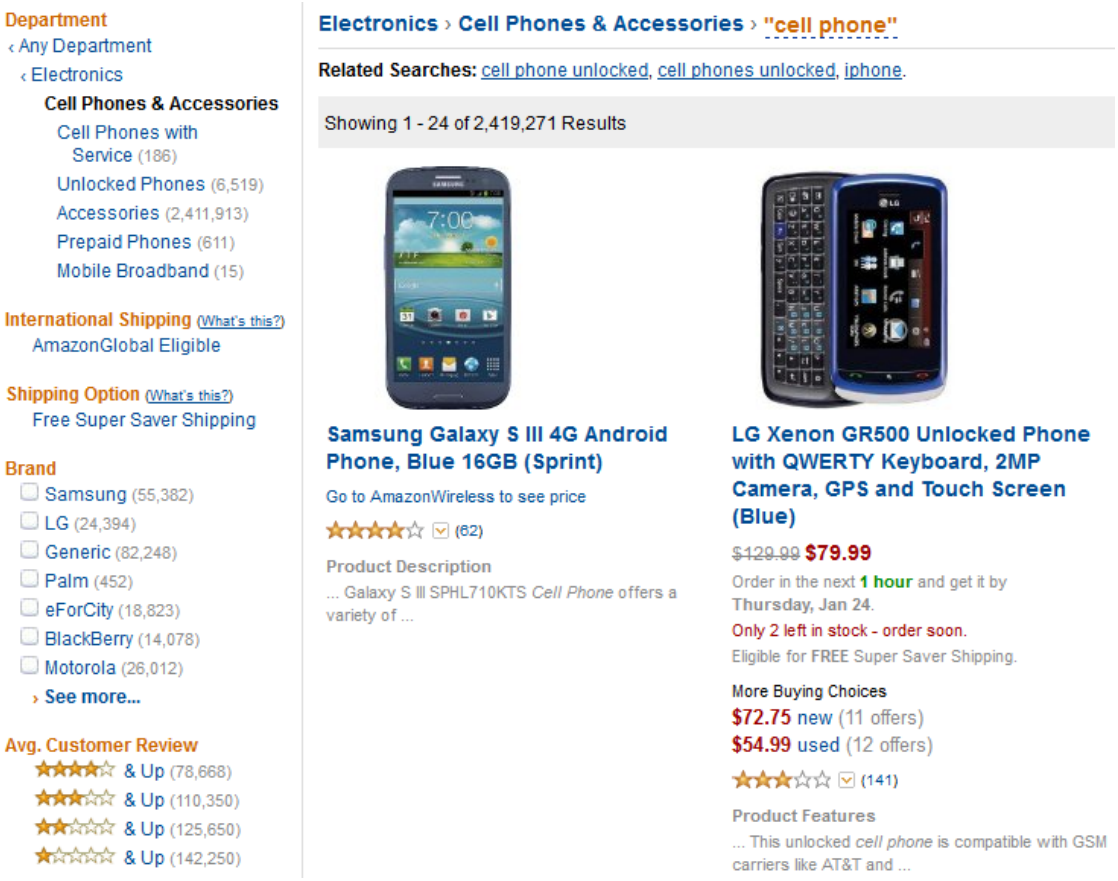


Figure 2.1.: Detail of the faceted search interface on amazon.com after a search for "cell phone".

well. Lambe notes that "faceted schemes often do not give as good an overview of the overall structure of the content collection as trees and hierarchies do" [La07, p. 38]. And further: "Where a tree or hierarchy educates novices and accommodates experts, facets are less informative[...]" [La07, p. 38]. In fact, trees are easier to grasp at first sight and while a two-dimensional faceted taxonomy can still be visualized as a matrix, a greater number of facets is hard to visualize.

From a software developer's point of view, faceted search requires that the collection of resources can be efficiently filtered and it leads to a more complex user interface. To the best of our knowledge, there are no information systems allowing users to easily adapt a faceted organization structure to their needs and to apply it as universally as this is possible for tree structures. In consequence, facets are mostly applied in specialized systems most of which are used to organize resources of the same type, such as hotels, flights, furniture, and so on.

Finally, it has to be stressed that faceted organization of information resources can be implicitly applied in tree structures and in tagging systems. However, in tagging systems, facets are not explicitly distinguished but all tags are mixed, and in tree structures a certain order has to be imposed on the different facets and they cannot be used independently.

2.3. Tagging systems

The notion of tagging, as it is applied and understood today, is closely related to the terms *Web 2.0* and *user generated content*. The term Web 2.0 comprises several principles and practices regarding the way the web is used, which emerged during the first half of the last decade [O'05]. One important aspect is the ability of internet users to collect and produce digital contents and to share these contents with other users on large social web platforms. In the course of this development, tagging became popular as an organization principle for these masses of diverse contents. Characteristic for tagging systems is that the users can choose the tags themselves and there is no authority maintaining a controlled vocabulary.

The first popular services that relied mainly on tags were Flickr[3] for photos or the social bookmarking service delicious[4]. Since users do not only tag their own contents but also see what others are tagging, the terms *social tagging* or sometimes also *collaborative tagging* are used to describe this phenomenon. As already mentioned in Section 1.1, the organization structures emerging from such tagging activities are usually referred to as *folksonomies* [VW07]. A folksonomy is not an explicit structure but it is embodied in the entirety of tags and resources. However, the tag assignments are not completely random but certain usage patterns can be observed (see Section 2.3.4). Weinberger speaks of an "emergent grassroots taxonomy" [We05]. Considerable research efforts are targeted towards detecting and exploiting these hidden patterns.

Recent research on tagging systems was mainly motivated by their popularity as part of the Web 2.0 phenomenon and the resulting availability of very large datasets. However, the flexible nature of tags and their potential to facilitate the organization of large quantities of information resources in scenarios where classical rigid taxonomies are inapplicable makes them an interesting research subject apart from the web.

This section covers the fundamental properties of tagging systems. Section 2.3.1 introduces different types of tagging systems and provides a basic formal definition. Subsequently, Section 2.3.2 gives an overview of the different kinds of tags being used in folksonomies. Section 2.3.3 describes how tagging systems are accessed and Section 2.3.4 covers dynamic aspects as well as user behavior. Section 2.3.5 discusses strengths and limitations of tagging systems and finally Section 2.3.6 gives a very brief overview of popular tagging systems.

2.3.1. Types of tagging systems

The common characteristic of all tagging systems is simply that tags are assigned to information resources. We denote the set of all tags with T and the set of information resources with R. Tags and resources are connected by tag assignments $A \subseteq R \times T$. We call the set of all resources having a certain tag t assigned the extension of t:

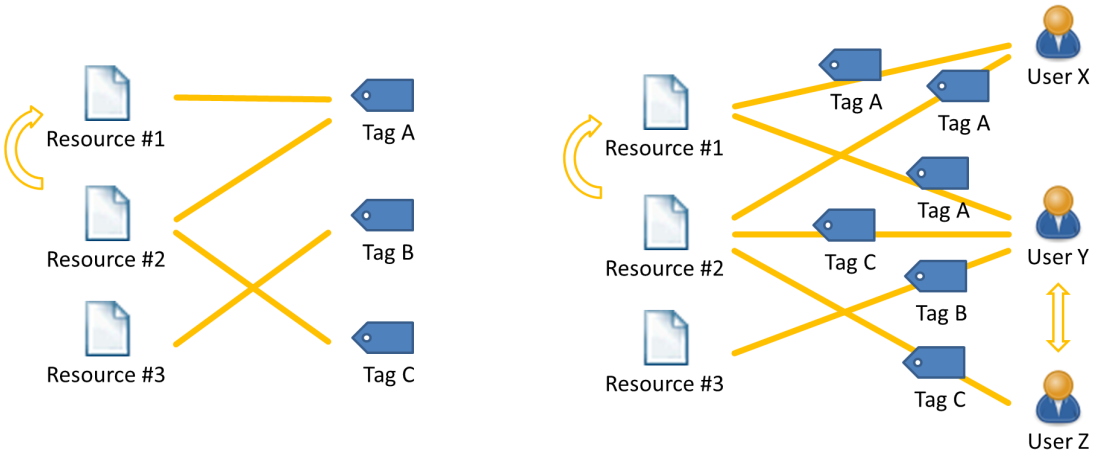
$$extension(t) = \{r \mid (r, t) \in A\}$$

While this represents the of minimum data contained in a tagging system, in the context of collaborative tagging and folksonomies one usually includes the users assigning the tags

[3] http://www.flickr.com, (accessed 11th of February 2013).
[4] http://delicious.com, (accessed 11th of February 2013).

and thus a tagging system is modeled as a tripartite network [LA06] or tripartite hypergraph [Mi07]. This means, there is an additional set of users U, and the tag assignments are defined as $A \subseteq R \times T \times U$ (Wu at al. mention the time of the tag assignment as an additional component [WZY06]). This extended model is called a *broad* folksonomy, the simpler one a *narrow* folksonomy. More precisely, Vander Wal uses these two terms to distinguish whether the same tag can be assigned several times by different users or all users share the same set of tags [VW05]. Although even in the latter case the tag assignments can be associated with users, the essential property of the tagging system is whether a single set of tags is maintained for a resource or if there are different sets of tags depending on the user. Hence Marlow and others use the terms *set-model* and *bag-model* respectively [Ma06]. In the bag-model multiple assignments of tags to the same resource are possible.

Figure 2.2 illustrates the difference between both kinds of tagging systems. Following [Ma06] and [Pe09], the figure indicates that there can be additional connections among documents or users as it is typical for hypertext documents and users in social networks.



(a) Narrow folksonomy **(b)** Broad folksonomy

Figure 2.2.: Types of implicit tag relations

In addition to the distinction of broad and narrow folksonomies — which is referred to as the so-called *aggregation model* —, Marlow et al. list several more attributes in their taxonomy of tagging systems [Ma06].

- **Tagging rights.** This attribute has one of three values: *self-tagging*, *permission-based*, or *free-for-all*. Self-tagging means that users can only tag their own resources. In contrast, a user can tag any resource in free-for-all tagging systems. An example for the latter is delicious.com where all users can tag URLs of arbitrary web pages. In between is permission-based tagging where users can give other users the permission to tag all their resources or certain subsets thereof.
- **Tagging support.** This refers to the support users get when tagging a resource. *Viewable tagging* means that they can see tags already assigned to the resource by others. The system can also compute a list of suggestions based on the present tags or possibly based on other parameters such as the content of the resource. This is called

suggestive tagging. Finally, *blind tagging* means that users neither see the tags of others nor any suggestions.

- **Type of object.** The resources being tagged can be any discrete objects ranging from text documents to URLs or photos and videos.
- **Source of material.** Marlow et al. distinguish tagging systems in which users contribute the resources (e.g., Flickr), and such in which the resources are provided by the system (e.g., the music website *Last.fm*[5]) [Ma06]. Additionally, there are systems like delicious that allow to tag arbitrary web resources.
- **Resource connectivity.** As indicated in Figure 2.2, there can be connections among resources. Marlow et al. distinguish links, such as in hypertext documents, and groups to which resources are assigned aside from tags [Ma06]. An example for the latter is Flickr where users can not only tag photos but also assign them to groups that pool photos of a certain topic, have access rights and are moderated by an administrator[6].
- **Social connectivity.** If the tagging system also offers social networking functionality, users can be connected as well. As for resources, Marlow et al. distinguish three values for this attribute: *linked*, *grouped*, and *none* [Ma06].

Another essential property of a tagging system is its purpose, or from another perspective, the reason why it is used. Regarding users tagging motivations, Marlow et al. state:

> "The motivations to tag can be categorized into two high-level practices: *organizational* and *social*. The first arises from the use of tagging as an alternative to structured filing; users motivated by this task may attempt to develop a personal standard and use common tags created by others. The latter expresses the communicative nature of tagging, wherein users attempt to express themselves, their opinions, and specific qualities of the resources through the tags they choose." [Ma06]

They distinguish the following user incentives [Ma06]:

- **Future retrieval.** This includes not only the retrieval of individual resources but comprises what Golder and Huberman call "task organizing" [GH06] (cf. Section 2.3.2 for an overview of different kinds of tags).
- **Contribution and sharing.** Contributing to pools of resources being shared by potentially unknown other people.
- **Attract attention.** People tag their own resources, such as their blogs, to point others towards them.
- **Play and competition.** Marlow et al. note that either the system is designed as a game, such as the ESP game [AD04], or users treat it as such by developing their own rules [Ma06].
- **Self presentation.** This motivation corresponds with a certain kind of tags. Golder

[5] http://www.last.fm, (accessed 26th of January 2013).
[6] http://www.flickr.com/groups/, (accessed 26th of January 2013).

and Huberman refer to the function of these tags as *self reference* [GH06]. For instance users mark all the movies they have seen or the books they have read.

- **Opinion expression.** Users share their subjective opinion on resources with others.

2.3.2. Kinds of tags

With the purpose of the tagging system and the user motivations vary the types of tags being used. In their seminal paper [GH06], Golder and Huberman identify seven functions of tags in the context of the organization of bookmarks:

1. **Identifying what (or who) it is about.** Most tags denote the topic of a resource. The topic can be as general as "sports" or "politics" or very specific as for instance the name of a particular person, event, organization or place.
2. **Identifying what it is.** These tags describe the type of resource being tagged. For example whether it is a research paper, a blog post, a news portal or a photo.
3. **Identifying who owns it.** These tags refer to the owner or creator of the information resource.
4. **Refining categories.** Some tags do not make sense alone but only in conjunction with other tags. A typical example is that of product names — for instance for cars or cameras. One tag denotes the manufacturer and another one the specific model.
5. **Identifying qualities or characteristics.** Such tags mostly express the subjective opinion of the tagger, for example whether a movie is "funny", "interesting" or "boring".
6. **Self reference.** In a broad folksonomy, where tag assignments include the user who made the assignment, some users use tags like "mymovies", "mybooks", et cetera, for instance to maintain collections of things.
7. **Task organizing.** The most typical tags in this category are probably "todo" or "toread". Golder and Huberman also name "jobsearch" as a typical example. These tags are usually personal.

Based on these functions, Sen et al. define three more abstract classes in [Se06]:

- **Factual tags.** This class of tags subsumes Golder and Huberman's functions 1, 2 and 4. The tags represent facts and are independent from a particular user.
- **Subjective tags.** This class contains all tags that express the users' personal opinions about a resource. They are equivalent to the tags identifying "qualities or characteristics" in Golder and Huberman's classification.
- **Personal tags.** These tags are applied for a user's personal organization of contents. Personal tags subsume tags expressing ownership, self-reference and tags being applied for task organization with regard to the tag functions of Golder and Huberman.

How tags are quantitatively distributed among different categories was examined by Sigurbjörnsson and van Zwol [SZ08]. They could relate 52% of the tags of a sample of Flickr photos to terms in the lexical database *WordNet*. With regard to these tags they found out "that

locations are tagged most frequent (28%); followed by *artifacts or objects* (16%), *people or groups* (13%), *actions or events* (9%), and *time* (7%)".

Schmitz [Sc06b] identifies the "key facets" *place*, *activity*, and *depictions* as well as a certain category of tags referring to "emotion" or "response". For a deeper analysis of the research concerned with tag categories, we refer to [Pe09, pp. 196-203].

Finally, we want to highlight another class of tags, so-called *triple tags* or *machine tags* as they were introduced in the Flickr API[7]. They have the generic form "prefix:property=value" where the prefix can be considered the namespace of the property. On Flickr they are primarily used to attach geo-coordinates to resources. By combining these tags with others, interesting visualizations can be created (e.g., [Ja06]). While users usually do not enter triple tags manually, the existence of these tags again underlines the great flexibility of tagging systems. However, we have also found Flickr photos where triple tags were used to denote subway lines and train models. It is unlikely that these tags have been assigned automatically but users apparently agreed on certain tagging conventions.

2.3.3. Accessing tagging systems

Peters distinguishes two fundamental ways of information access in tagging systems: The *pull approach* and the *push approach* [Pe09]. The former term denotes the process of active search, i.e., the user actively searches or browses the system for a piece of information. The latter can be considered a subscription: The user configures the system to issue a notification as soon as a new resource with certain tags is added. Users are then typically notified via email or RSS/ATOM feeds[8].

Considering only the pull approach in the following, we can still distinguish several different ways of accessing a resource collection: On the one hand, the user's goals can vary in specificity, and on the other hand, different means of access can be applied.

The specificity of user goals can range from trying to retrieve a particular known resource to exploratory search. White and Roth characterize exploratory searchers as:

1. "unfamiliar with the domain of their goal (i.e., [they] need to learn about the topic in order to understand how to achieve their goal);
2. unsure about the ways to achieve their goals (either the technology or the process); and/or even
3. unsure about their goals" [WR09].

In [He09], Hearst gives an overview of the different scientific models of information seeking as well as the strategies being applied by users. One popular model that takes into account that users' goals do not have to be static during the information seeking process is the *berry-picking* model by Bates [Ba89]. Hearst characterizes it as follows: "[...] in the process of reading and learning from the information encountered throughout the search process the searchers' information needs, and consequently their queries, continually shift [...]" [He09, p. 68]. Additionally, "[...] searchers' information needs are not satisfied by a single, final

[7] http://www.flickr.com/groups/api/discuss/72157594497877875/, (accessed 24th of January 2012).
[8] http://en.wikipedia.org/wiki/Web_feed (accessed 26th of January 2013)

retrieved set of documents, but rather by a series of selections and bits of information found along the way" [He09, p. 68].

With regard to tagging systems, one often broadly contrasts the information seeking strategies *browsing* and *searching* (or *querying*) [Pe09, HMHS06]. Peters characterizes browsing as "searching for information by following and pursuing hypertext structures" [Pe09, p. 289] emphasizing rather the particular steps of the information seeking process than the level of specificity of the user's intentions. In the following, we will use the term *navigation* to denote the selection from a list of navigation options in contrast to searching for an arbitrary keyword.

According to Wash & Rader [WR06], the majority of tags used to describe a document — they consider web pages bookmarked on delicious — is not contained in the document itself, and they consider this fact an improvement for access via search. However, tagging is generally rather perceived as a means for facilitating browsing rather than directed search. Hassan-Montero and Herrero-Solana argue that many tags are too abstract and unspecific to access a particular known resource, but tagging systems are rather suitable for facilitating serendipitous findings [HMHS06]. Peters describes the term *serendipity* as a "'happy accident', i.e. the finding of information that the user was not looking for but which are of interest to him anyway" [Pe09, p. 289]. In Section 3.1.3, we will explain in more detail why the indexing quality in tagging systems tends to be bad and why individual resources are hard to find. The limitations of tagging systems are further discussed in Section 2.3.5.

In [HGM06], Heymann and Garcia-Molina identify "three main views" allowing users to browse the contents of tagging systems:

1. "A list of all objects which are tagged with a given tag (or possibly a combination of two or more tags).
2. A list of the most popular tags in the system.
3. A list of tags which have a high degree of overlap with a tag the user is currently investigating" [HGM06].

The last view is probably the one that contributes most to the popularity of tagging systems since it makes visible the implicit relations among tags being central for the serendipitously finding of new interesting contents.

However, this list of views cannot be considered complete. Millen and Feinberg highlight another important way of browsing tagged contents [MF06]. They examined a social bookmarking service called *dogear* that was deployed in a large organization. They found that users more often looked at other users' bookmarks lists than they were searching for bookmarks with a specific tag. They call this behavior *social navigation* and cite Dourish and Chalmers who define it as navigation being "driven by the actions from one or more advice providers" [DC94]. We refer to [Pe09, pp. 291-293] for an overview of research regarding social navigation in tagging systems.

In the first of the three views quoted above, Heymann and Garcia-Molina point out that in some systems the resource list can be filtered by combinations of tags [HGM06]. This means the filter can be incrementally narrowed by adding additional tags. Millen et al. describe another common browsing mode in tagging systems called *pivot-browsing* [MFK06].

It describes the ability to reorient the view with regard to changing topics of interest. Instead of narrowing the search filter, the selection of a tag would replace the current filter with a new filter that only consists of this tag. But pivot-browsing does not only refer to tags. It also includes for instance the selection of a user name to view all resources owned or bookmarked by the respective user. Similarly, individual resources can be focused in order to discover related tags or users for further browsing steps. We consider the detailed view of a particular resource as another main view in tagging systems.

A typical way of presenting a set of tags, be it the most popular tags in a system, tags related to a certain other tag or simply the tags of a certain resource, are so-called *tag clouds*. In a tag cloud, the tags are arranged in a two-dimensional plane and they often differ in font size to express their relative frequency. Alternatively, the size of a tag depends on its current popularity, i.e., how often it has been used in the past hours or days. Figure 2.3 shows an example of a typical tag cloud.

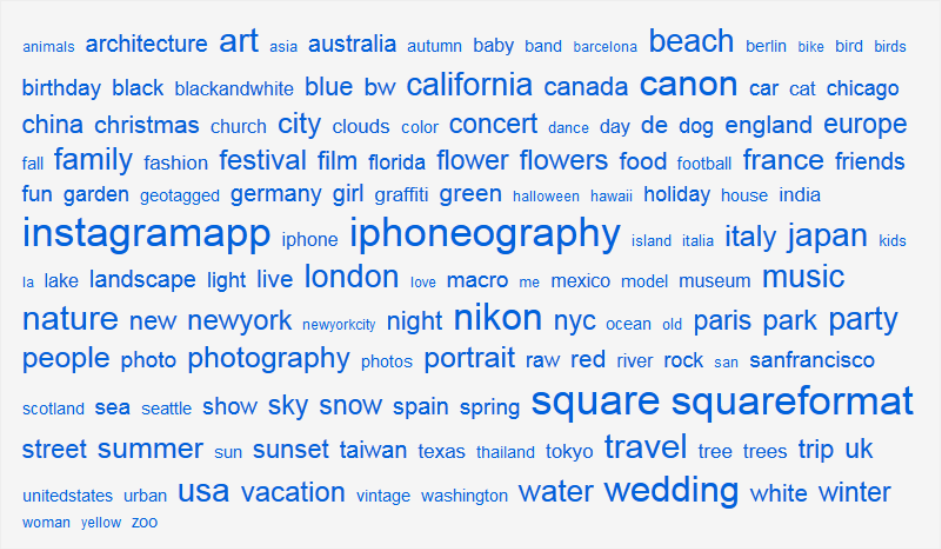


Figure 2.3.: Tag cloud of the most popular tags on flickr.com on January 27th 2013.

Rivadeneira et al. suggest that tag clouds can support different tasks:

- **Search.** Finding a particular piece of information.
- **Browsing.** Navigating in a rather exploratory way.
- **Impression formation or gisting.** Getting a general idea of what the items in a collection are about.
- **Recognition/matching.** Confirming that the resource the user is looking at is what the user expects [Ri07].

The suitability of tag clouds for navigation is controversial. Through interviews with 20 web designers, Hearst and Rosner found that tag clouds are used "primarily because they are perceived as having an inherently social or personal component, in that they suggest what a person or a group of people is doing or is interested in, and to some degree how that changes over time; they are visually dynamic and thus suggest activity; they are a compact alternative

Figure 2.4.: Popular Flickr tags arranged in a tag cloud created with the generator on tagxedo.com.

to a long list; they signal that a site has tags; and they are perceived as being fun, popular, and/or hip" [HR08]. An example for a tag cloud that supports this position is displayed in Figure 2.4. It shows the same tags as the tag cloud in Figure 2.3. Sinclair and Cardew-Hall found in an experiment that users prefer a traditional search interface when searching for specific information but tag clouds can be useful "for browsing or non-specific information discovery" [SC08].

Regarding the layout of tag clouds, it was reported that users profit most from a tag cloud with a sequential layout and alphabetically ordered tags when they are searching for a specific tag [LZT09]. This is supported by Halvey and Keane who found that lists can be scanned faster than tag clouds and alphabetic ordering of terms is advantageous [HK07]. Rivadeneira et al. found that a list that is ordered by frequency is the preferable layout for the task of impression formation [Ri07].

There have been many attempts to improve the layout of tag clouds with tag clusters. We will cover tag clusters in Section 7.1.1.

2.3.4. Tagging system dynamics and user behavior

It is a broadly accepted fact that "in folksonomies, the distribution of indexed tags on a resource level resembles a Power Law curve" [Pe09, p. 171]. *Power law* is defined by Halpin et al. as follows:

> "A power law is a relationship between two scalar quantities x and y of the form:
>
> $$y = cx^{\alpha}$$
>
> Where α and c are constants characterizing the given power law."

Figure 2.5 shows an example graph of an idealized power law distribution for the tag frequencies of a resource. The parameters are $\alpha = 0.7$ and $c = 100$ for the frequency of the most frequent tag, so there are few tags that are used very often and the vast majority of tags is

comparatively rare. A similar shaped curve is obtained when visualizing the frequencies of tags in the whole folksonomy and not only for a particular resource.

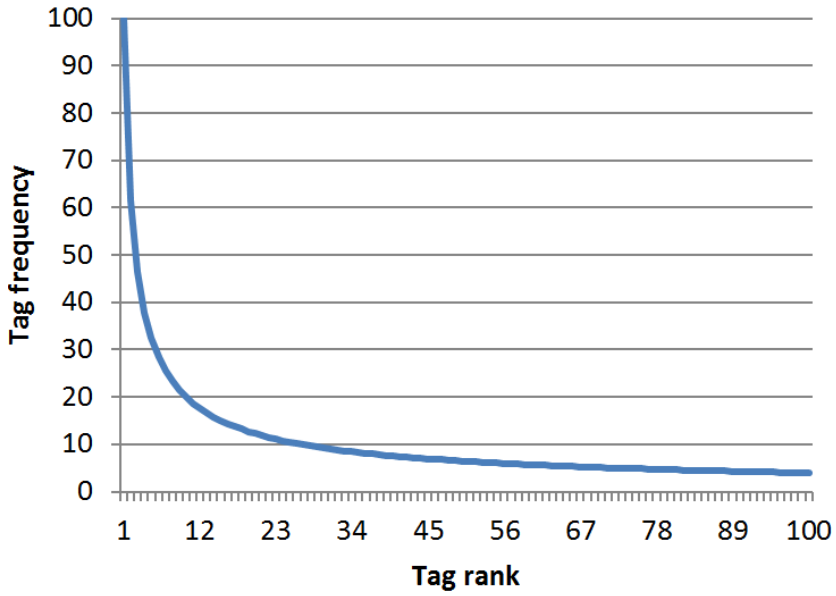


Figure 2.5.: Idealized power law distribution for the first 100 tags of a resource with the first tag occurring exactly 100 times.

That the tag distribution for a resource stabilizes over time was already discovered by Golder & Huberman [GH05]. Halpin et al. found that "given sufficient active users, over time a stable distribution with a limited number of stable tags and a much larger 'long-tail' of more idiosyncratic tags develops" [HRS07]. The term *long tail* is commonly used to refer to the characteristic shape of the right part of such power law distributions since Anderson used it in an article about business strategies in the entertainment sector [An04].

In addition to the tendency of users to assign certain kinds of tags frequently, users also influence each other in their tag choice. Munk & Mork phrase this as follows: "Users often choose broad basic categories, because that requires the least cognitive effort", but there is also "an imitation-dynamic that creates an asymmetry, where a few descriptive metadata are often reproduced and the majority seldom reproduced" [BMM07].

Sen et al. observed that the kinds of tags being used strongly depend on the way users are exposed to the tags of others during the tagging process [Se06]. While users tend to use more personal tags when they cannot see any other users' tags, they rather assign factual tags — such as the topic of the resource — when they see the most popular tags for a resource. However, results of Rader & Wash indicate that the main motivation of users is the organization of resources for themselves: "We find evidence that users choose tags in a pattern consistent with personal information management goals, rather than as a result of social influence" [RW08].

2.3.5. Strengths and limitations

Probably the most frequently stated advantage of tagging systems is that users can categorize contents using their own words and not having to adhere to a terminology or scheme being stipulated by an authority. In one of the earliest publications covering social tagging and folksonomies, Mathes identifies the following problem regarding the traditional creation of metadata: "Author created metadata may help with the scalability problems in comparison to professional metadata, but both approaches share a basic problem: the intended and unintended eventual users of the information are disconnected from the process" [Ma04]. In contrast, a folksonomy "directly reflects the vocabulary of users" [Ma04]. To illustrate this, he compares folksonomies to *desire lines* and quotes Merholz who describes the analogy as follows:

> "A smart landscape designer will let wanderers create paths through use, and then pave the emerging walkways, ensuring optimal utility. Ethnoclassification systems can similarly 'emerge.' Once you have a preliminary system in place, you can use the most common tags to develop a controlled vocabulary that truly speaks the users' language" [Me04].

As already stated earlier, folksonomies are useful for browsing collections in order to discover new interesting content [HMHS06]. Peters notes that "semantic subtleties in the indexed terms can also be better taken into consideration by folksonomies than by controlled vocabularies" [Pe09, p. 213]. The users are not restricted by a limited, predetermined, controlled vocabulary but they can introduce terms where they see fit. This way, they can make accessible resources for which existing controlled vocabularies are inadequate, either due to their coarse level of granularity or because they are not up-to-date. That folksonomy tags are not as well defined as terms in rigid classification schemes can also be seen as an advantage: "The tags' lack of clarity becomes their advantage, as they leave wiggle room, just like human language"[Pe09, p. 212].

In addition to the breadth and specificity of topics, tagging makes it possible to index resources at an unprecedented scale and at very low cost. The users do it themselves and voluntarily. Users do not even see it as a burden but they often enjoy it [Ne07]. Hassan-Montero and Herrero-Solana state that "the low cognitive cost of tagging is one of the main factors of its popularity" [HMHS06].

Tagging is more flexible than other knowledge organization structures. Golder and Huberman compare it to nested directory structures and note that if a resource is assigned to two categories in a classical directory structure, one has to "establish the first as primary or more salient, and the second as secondary or more specific" [GH06]. While users can relate a resource to an arbitrary number of possibly overlapping categories in a tagging system, in tree structures this requires an arbitrary choice of category precedence that is possibly inappropriate in other usage scenarios. To solve the problem several copies of the resource would have to be saved in different places in the directory structure.

Although facets are not explicitly defined as in faceted classification structures, users can easily tag resources with regard to any relevant aspect. However, Peters points out that there are hidden "paradigmatic relations" — such as hierarchical relation — among tags that cannot be exploited [Pe09, p. 223]. Additionally, the context of the tags is lost, i.e., if the

name of a person is assigned to a photo, it is not made explicit whether this tag refers to the photographer or whether the tagging person was concerned with what is shown on the photo.

The unconstrained nature of tags entails additional problems, primarily due to their inconsistent usage. *Precision* and *recall* are typical measures in information retrieval to assess the quality of a search result (cf. [BYRN11]). Precision refers to the fraction of relevant resources in the search result and recall denotes the fraction of all relevant resources that could be retrieved. Thus, precision is low when too many irrelevant resources are retrieved and recall suffers when the user searches for a tag but this tag is only assigned to part of the relevant resources.

Mathes identifies *synonyms* and *homonyms* as the causes of such deficiencies in tagging systems [Ma04]. Synonymy applies not only in the linguistic sense, i.e., a language has two words for the same concept, but also spelling variations can lead to a situation where two different tags have the same meaning. The inconsistent use of whitespace and hyphen characters can be frequently observed, as in "newyork", "new-york", and "new york". A real synonym in this case would be for instance "big apple". Variation in capitalization ("New York" vs. "new york"), spelling mistakes or simply the use of different languages can also lead to problems. Apart from variations in spelling, the use of synonyms is caused by the so-called *vocabulary problem*. Furnas et al. observed in their experiments that the probability that two people independently use the same term to describe an object was below 20% [Fu87]. However, in broad folksonomies the vocabulary problem is less severe: "When a resource has been tagged by many users, it is more likely that the tag used for search will match one of the previously assigned tags" [Fu06].

Obviously the use of synonyms impedes the recall measure if not all versions of a tag are assigned to all relevant resources. We will come back to the problem of synonymy in Section 3.1.1.

In contrast to synonyms stand *homonyms* which have a negative effect on the precision. A homonym is a word with at least two different meanings, for instance "miss" denotes an unmarried woman as well as the event of not hitting a target. In consequence, a search for a homonym is likely to yield irrelevant resources. Golder and Huberman point out that homonyms can be distinguished by adding an additional tag that narrows the search result to the desired domain [GH06]. A search for "bow" and "arrow" would for example rather yield results related to range weapons than for the gesture (bow) or a mark that indicates a direction (arrow). However, they also stress that this is more difficult for *polysems*. In contrast to homonyms, the meanings of polysems are related. Golder and Huberman give the example of the term "window" that may refer to a hole in the wall or a pane of glass [GH06].

Golder and Huberman additionally mention *basic level variations* as another problem in tag usage [GH06]. According to Rosch, the basic level in a category structure, such as a taxonomy, is a certain level of abstraction above which the categories are rather unspecific and general, so their members share only few attributes. Below the basic level, there are only slight differences among the categories, so most of the characteristic attributes of subordinate categories are also shared by the objects in the basic level category [Ro78]. In other words, "the most basic category cuts can be made" at the basic level [Ro78]. Rosch gives the example of "chair" as a basic level category with "furniture" being the superordinate and "dentist chair"

a subordinate category. However, Tanaka and Taylor observed that the usage of basic level categories depends on the level of expertise a person has in a certain domain [TT91]. Experts in a field tend to use more specific categories than laymen. Since this problem is similar to that of synonymy, the use of different levels of abstraction impedes the recall of searches.

A further problem of tags is that navigation options are very limited. Since no explicit links between tags are established, tags can only be related by the frequency of their occurrence in combination with each other. This has already been discussed in Section 2.3.3.

The strengths and limitations of tagging systems have also been examined in studies and experiments that oppose tagging to the organization of information with folders. Pak et al. observed in an experiment that people are faster when organizing information with (digital) folders but they require more clicks to retrieve the information resources [PPI07]. The same could be observed by Ma in another experiment [Ma10]. Civan et al. studied how people apply folders and tags for organizing email and found that less "physical effort" was required to store emails in folders [Ci08]. However, participants of their study reported that it involved more cognitive effort to find the folder that best matches the content of an email than just assigning one or more labels. On the other hand it was found that folders are preferred when items are systematically searched. In particular, participants complained that it was not easy to find all the uncategorized emails when using tags. Finally, the participants pointed out another advantage of tags: In addition to being a means for storage and retrieval the tags of an email also serve as a summary of its content [Ci08].

In this work, we can only cover the most fundamental aspects of tagging systems and important research being related thereto. A plethora of further references to research on tagging and folksonomies has been compiled by Peters [Pe09].

2.3.6. Popular tagging systems

As already mentioned above, the first popular tagging systems were Flickr and delicious. A substantial fraction of the research on tagging systems is based on data obtained from these platforms. The platforms are to some extent complementary: delicious is used for bookmarking web pages, i.e., mostly textual resources, while photos are shared and managed with Flickr. In delicious, several users can tag the same resource whereas in Flickr usually only the owner of a photo assigns tags to it (broad vs. narrow folksonomy). However, there are several other popular platforms that shall at least be mentioned here.

BibSonomy[9] is a platform that allows researchers to share and manage bookmarks and publications [Ho06]. Both kinds of entries can be tagged and publication entries carry citation information. The platform has been helpful during the writing of this thesis, on the one hand to find relevant research and on the other hand to obtain the BibTex[10] records for publications.

CiteULike[11] serves the same purpose as BibSonomy. Tags are a little less prominent on the site.

[9] http://www.bibsonomy.org (accessed 11th of February 2013).
[10] http://http://en.wikipedia.org/wiki/Bibtex (accessed 11th of February 2013).
[11] http://www.citeulike.org (accessed 11th of February 2013)

Technorati[12] allows users to search for blogs and social media based on tags. Most blogging systems allow the authors to annotate their blog posts with keywords. These keywords are indexed by Technorati. The platform lists more than one million blogs.

LibraryThing[13] is a social network and a book cataloging platform. Users can write book reviews and organize their collection of books with tags. Tags are organized in large synonym rings that cover many spelling variations.

The music website Last.fm[14] allows users to tag artists, albums and songs. The resulting music folksonomy can be browsed to discover new music. Additionally, dynamic radio channels are generated based on tags.

Other examples of large websites featuring tagging functionality are the online store Amazon[15] and the video portal YouTube[16]. Although the contents of these platforms can be tagged, tags play a rather subordinate role.

[12]http://technorati.com (accessed 11th of February 2013)
[13]http://www.librarything.com (accessed 11th of February 2013)
[14]http://www.last.fm (accessed 11th of February 2013)
[15]http://www.amazon.com (accessed 11th of February 2013)
[16]http://www.youtube.com (accessed 11th of February 2013)

CHAPTER 3

Design Process

The first section of this chapter illustrates some important observations made in existing tagging systems that led to the key ideas underlying the approach followed in this thesis. This approach will be sometimes referred to as the *TACKO* approach. TACKO stands for "Tag-based Context-dependent Knowledge Organization"[1]. It is the name of the research project framing the research presented in this thesis.

Subsequently, several design decisions, guidelines, and principles that framed the development of the *TACKO system* and the respective *TACKO organization structure* are described in Section 3.2. The term TACKO system — or sometimes just system — refers to the prototype implementation of our approach that has been continuously enhanced and refined in parallel to the development of the data structure in order to assess its practicability.

The design of the TACKO system can be broken down into two major phases. The first was characterized by the attempt to structure a folksonomy with implicit tag relations. This means that tag relations are embodied in the entirety of the tag assignments in contrast to being explicitly expressed in separate data structures. This first development phase is covered in Section 3.3. The idea of implicit tag relations is explained and the user interface that was developed to manage implicit tag relations is presented. The chapter concludes with a discussion of this approach and why it was abandoned during the later development of TACKO.

In the second development phase, tag relations were made explicit. A comprehensive description of the final TACKO organization structure and a user interface to access and manage the structure will be given in Chapter 4.

[1]The term "context-dependent" does not refer to the usage context, i.e., whether a tagging system is used on a mobile device or a desktop computer. It means that the organization of resources takes place in the context of a certain subset of the resources collection. This will be covered in detail in Section 3.3.

3.1. Initial observations

The design of the TACKO system was driven by several observations and assumptions that are detailed in this section. The most popular tagging systems have been applied at large scale and offer the user to browse ever growing collections of an unmanageable size. Taking another perspective on these collections and examining manageable parts of it has led to key insights regarding the utility of tags when applied on a smaller scale.

3.1.1. Synonyms vs. homonyms

As already described in Section 2.3.5, the most frequently stated limitation of folksonomies is the lack of vocabulary control [GH06, Ma04]. Inconsistent usage of tags is inevitably introduced when users independently tag resources without guidance. The use of ambiguous tags, i.e., homonyms or polysems, as well as synonym tags leads to a very fuzzy and noisy categorization of resources and impedes the utility of the system for searching and browsing tasks.

While this applies to large folksonomies in general, the situation is different when narrowing the scope to a certain subset of information resources. When examining a set of resources taken from the same domain, e.g., resources related to a certain topic or place, one can make two essential observations:

- **Homonyms are less problematic.** Since all resources under consideration are taken from the same topical context, it becomes less likely that a single tag has completely different meanings. The term "trial" for example is very ambiguous in general but in the context of "medical research" or "jurisdiction" its meaning is clear. Of course homonyms can still occur in small collections of resources but they are by far not as frequent as in large folksonomies.
- **Synonyms become more apparent.** In contrast to homonyms, the use of synonyms becomes striking when the most frequent tags in a collectively tagged collection are visualized as a tag cloud (see Figure 3.1 for an example).

The second observation requires closer examination. The example presented in Figure 3.1 does not exactly show synonyms in the literal sense but there are several groups of tags with a close semantic relation. Tagging a photo "librariesandlibrarians" can be considered equivalent to tagging it "libraries" and "librarians" separately. There are also several examples of singular and plural tags such as "librarian" and "librarians". While it can be argued that the meaning of both tags can be clearly distinguished, a closer look at the images reveals that many users do not make this distinction: plural and singular versions of a tag are often used together at the same resource.

Additionally, it frequently occurs that only the plural version is used although an image clearly shows only a single entity. One possible explanation is that users understand the tag as the description of a collection a resource belongs to rather than a description of the content of the resource itself. The tag "books" for instance is used frequently for images showing only a single book. However, for users browsing their own image collection it makes sense to search for "books" when they are looking for a particular image since they will retrieve a subset of

their image collection showing books (the usage of plural versions for categories is for example suggested by [GT06]).

Finally, the use of different languages introduces many tags having the same meaning.

In consequence, this means that a user who is browsing the collection is confronted with many tags that are semantically equivalent, at least they are used interchangeably, but lead to different resource sets when selected. While some users might consciously distinguish such tags to express particular semantic nuances, these nuances are not recognizable if not applied uniformly and consequently in the whole collection.

This means that for the user of a folksonomy, i.e., someone who is browsing the collection or looking for a particular resource, the utility of the tags would be improved if synonym tags or tags with closely related meanings were be harmonized. The accessibility of the roughly 40.000 resources in the collection for which the tag cloud in Figure 3.1 was generated would for instance be improved if the tag "librarian" was removed and the tag "librarians" was used instead. This way the tag cloud would be clearer and the user would not be confused by seemingly distinct tags that are equivalent with regard to their actual usage.

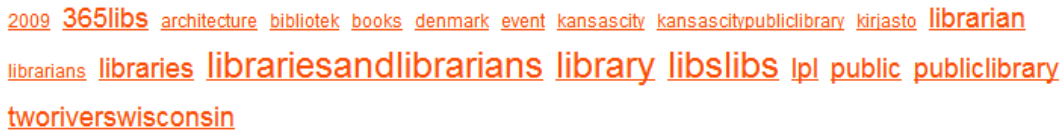


Figure 3.1.: Tag cloud showing the 20 most frequent tags of a Flickr group about libraries and librarians.

3.1.2. Latent tag relations

There have been many attempts to enhance tagging systems by exploiting semantic relations among tags. Such relations can either be obtained by analyzing the tag assignments and deriving tag relations from the folksonomy itself or by mapping tags to concepts in external knowledge representations such as ontologies. An extensive overview of different approaches is provided in Section 7.1. However, both kinds of approaches struggle with the problem of fuzziness in large-scale folksonomies, i.e., inconsistencies in tag usage and tag ambiguity. Neither can clear relations be derived nor do the concepts in an ontology properly reflect the meaning of tags with respect to their usage in the folksonomy.

As described in Section 3.1.1, the ambiguity of tags is of minor relevance in small manageable subsets of a folksonomy, at least if the subsets are not random but contain resources concerned with the same topic. This particularly applies to collections of resources managed and owned by a single user as for instance the bookmarks of a user of the social bookmarking service *delicious*. Such a user usually strives to reuse the same tags for the same categories. Assuming for the moment that the problem of tag ambiguity can be ignored in such a scenario, we assess the potential to improve the presentation of tag clouds in the following.

Since tags are not explicitly related, delicious — as well as most other similar services — presents the tags of a single user in the form of a tag cloud or a simple list. An example

of such a tag cloud is shown in Figure 3.2. The figure only shows the first lines of the tag cloud. The complete cloud comprises more than 2000 tags. Apparently such a tag cloud is of limited help for browsing the collection systematically and it would be preferable to have a clear, well-arranged tag cloud with fewer tags.

If a tag cloud could be directly modified, there would be two fundamental ways to change the tag cloud towards this goal:

- **Hiding tags.** It can be frequently seen in tag clouds that there are tags representing supercategories of categories represented by other tags. The tag cloud in Figure 3.2 shows such examples. Tags in dashed frames can be considered subcategories of the tags in solid frames of the same color. The tag "programming" for instance can be considered more general than the tag "javascript" representing a particular programming language. If the tags "javascript" or "java" were hidden in this tag cloud, it would be reasonable for a user to first select the term "programming" in order to chose a programming language in a second step. It has to be noted that from a semantic point of view the relations between the specific and the more general tag can take many forms. There are examples such as "firefox" and "browser" where the former is a hyponym of the other. Meronymy, i.e., a "part-of" relation, also occurs for instance between the tags "berlin" and "germany". In the example of "java" and "programming" there is no clear hierarchical relationship but it is still reasonable to hide "java". This can be compared to nesting a "java" folder within a "programming" folder in a file system folder structure because "java" belongs to the "programming" category. By hiding very specific tags the set of tags in a tag cloud could be reduced to the most general tags for which no more general categories exist.

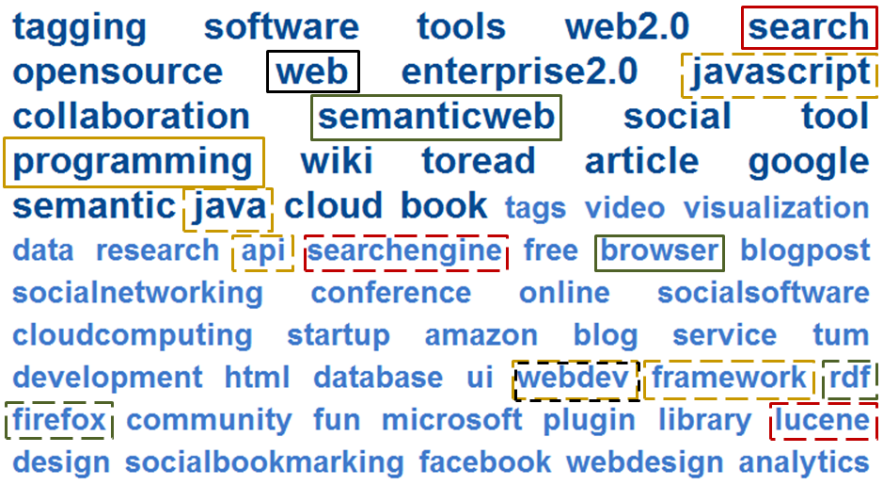


Figure 3.2.: Examples of broader/narrower relations among the most frequently used tags of the author's delicious bookmarks.

- **Arranging tags in a meaningful way.** Tagging allows users to categorize information resources with regard to a large variety of aspects [GH06, Ma04, Bi08]. However, all kinds of tags are usually displayed together in alphabetical order or they are ordered by frequency of occurrence. One way to structure the presentation of tags is to generate

tag clusters containing tags that frequently occur together (cf. Section 7.1.1). While such approaches clearly bear the potential to improve the presentation of tags, there is a major shortcoming: Since clustering is based on tag cooccurrence, tags are separated when they are not used in conjunction although they categorize resources with regard to the same aspect — or facet. Figure 3.3 demonstrates how tags in a tag cloud could be grouped in a meaningful way that resembles the use of facets. The grouping is indicated by colors. For instance the tags "amazon", "microsoft" and "google" stand for companies being related to the tagged resources. If these tags were grouped, the user could easily skip the whole group of tags when looking for instance for a particular resource of which she only knows that it was a blog post. In this case, a group such as "article", "blogpost", "book", "video" would be helpful instead. Such a group could be considered a "content type" facet. Apart from how such an arrangement is established, it shall be highlighted here that grouping tags with regard to the different dimensions of classification results in tag clouds being quite contrary to those obtained with tag clustering techniques (see also Section 3.3.3).

Figure 3.3.: Examples of facets in the most frequently used tags of the author's delicious bookmarks.

3.1.3. Incompleteness of tag assignments

In large-scale tagging systems it is often not required to retrieve all relevant resources for a given query since there are too many results to inspect all of them anyway and the user issuing the query is content as long as most of the results are relevant to the query. The user even cannot assess whether the result list is complete or not. The case is different when managing smaller collections such as personal collections of bookmarks or photos. Then it occurs that users know that particular resources are in the collection and they want to access these very resources using the tags they have assigned earlier. In such a scenario it is essential that a search for a tag yields all relevant resources. Relevant means in this case that a resource falls into the categories that the user associates with the query tags.

Ideally, each tag represents a clearly defined distinct category and all tags used throughout

the collection are assigned to exactly those resources falling into this category. In large folksonomies it is of course not the case that all users share the same interpretation of a tag but let us assume for the moment that a single user as well as a group of people sharing the same professional vocabulary agree on the meaning of the tags used throughout the collections they manage. This is of course idealized but it allows us to examine why recall for searches still tends to be bad even under such perfect conditions. There are basically three reasons for the fact that a tag is not assigned everywhere it applies:

- **Equivalent tags.** Even if users have the same interpretation of a tag, it occurs that they use different tags to represent the same concept. Reasons are spelling variations (e.g., spelling a compound tag with or without spaces), abbreviations, the use of different languages, or simply the fact that there are commonly used synonym terms in the same language. The consequence is a bad recall metric, i.e., not all relevant resources are retrieved.
- **Different levels of abstraction.** It was already mentioned in Section 3.1.2 that the tags in a tagging system vary in their specificity. Rosch et al. have found that a certain *basic level* of abstraction plays an important role in the categorization of objects [Ro76]. In folksonomies this can be observed by examining tag frequencies: The tag "dog" is more frequently used for photos showing for example a terrier than the tags "mammal", "animal", or the more specific tag "terrier" although they clearly apply as well. The tag "dog" is apparently such a basic level tag. However, Tanaka and Taylor could show that this basic level depends on the expertise of the user [TT91]. In effect, users being expert in a certain domain use more specific tags than non-experts. Another reason for the use of tags on different levels of abstraction can be the context in which a resource is considered when being added to the collection. Unless there is appropriate support during the tagging process, the resources are insufficiently tagged on all levels of abstraction, particularly many very abstract tags and very specific tags are missing. Those tags are usually only assigned to a very small fraction of the resources they actually apply to.
- **Different dimensions of categorization.** Finally, the variety of different aspects according to which resources can be tagged makes it difficult for users to bear all these aspects in mind when tagging a resource. Similar to the problem of different levels of abstraction, people are unaware of the fact that there are tags being used in the tagging system that would apply to the resource in question. In collections of Flickr photos, it can be for instance observed that the tag "grayscale" is missing for many resources. This is to some extent due to the other reasons mentioned above, i.e., people use similar or equivalent terms such as "blackandwhite" and "bw" or more general ones such as "monochrome". However, there are many photos which are not tagged with regard to colors at all even though it would have been easy for the tagging users to provide appropriate tags.

Figure 3.4 illustrates the three reasons for missing tags as described above. The small circle in the center contains the resources tagged with a certain tag while the large dashed circle contains all relevant documents. The resources in the blue circles are tagged with related tags and the four resources displayed in the left part of the dashed circle are relevant with regard to the tag in question but are not tagged with a related tag.

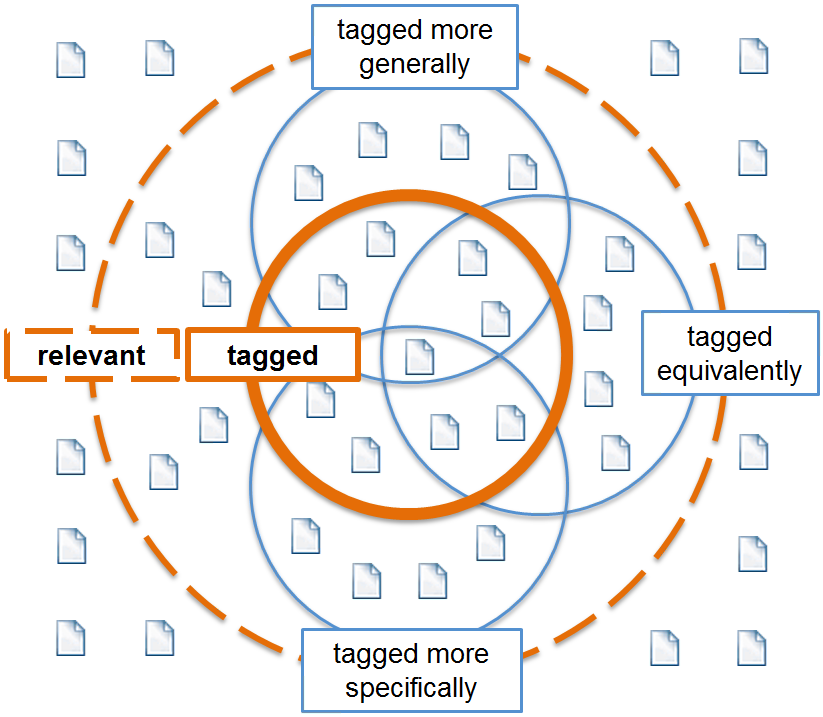


Figure 3.4.: The reasons for the incompleteness of tag assignments.

The problem that tag assignments are incomplete appears to be an inevitable consequence of the continuous growth of the tagging vocabulary. When a tag is used for the first time, there is usually no easy way for the tagging user to find out whether there are already other resource to which this tag applies as well. Even if it is known that such resources exist, the user usually does neither have the time nor the tools to locate them in order to assign the missing tag. If this problem cannot be solved, it would be at least desirable to make the insufficiencies in tag assignments more transparent for the users of the tagging system. If for instance the tag "color photo" was first used in a collection of photos, it would be helpful for users striving to improve the data quality in the collection to see which photos have been tagged with this tag, which photos have been tagged "black and white photo" and to which photos neither of both tags have been assigned.

3.2. Fundamental design decisions and guidelines

This Section covers several fundamental decisions that guided and framed the evolution of the TACKO system. This includes properties of the underlying tagging system as well as guidelines regarding the user interface.

3.2.1. Basic characteristics of the tagging system

Tagging systems can be categorized with regard to a variety of system design features and purposes. Marlow et al. have compiled an extensive list of the different dimensions according to which tagging systems can be classified [Ma06] (cf. Section 2.3.1). In the following, this framework is applied to determine precisely the fundamental properties as well as the purpose of the TACKO system.

Aggregation model. The aggregation model determines whether the tag assignments represent a binary relation involving only tags and resources (set model) or a ternary relation consisting of triples of tags, resources and users (bag model). For the TACKO system we chose the set model. Although our goal is to develop a system which a single user can use as well as groups of collaborating users, we assume that also in the latter case all users of the system agree on the meanings of the tags and on whether a particular tag applies to a resource or not. At this point, we do not want to rule out the possibility to extend our approach to the bag model in the future but since it does not depend on the ability to manage several sets of tags per resource, we chose to keep the system as simple as possible in order to facilitate the development process.

Tagging rights. Again, the necessity to maintain a high degree of flexibility during the development of TACKO led us to the decision to keep the tagging rights as simple as possible. With regard to access rights, there are at most two kinds of users: Users being able to browse a collection and view the contained resources, in the following referred to as *readers*, and so-called *writers* being additionally able to modify all the resources as well as their tags. This can be considered a very simple form of permission based tagging.

Tagging support. When a resource is tagged, the users shall be provided with as much guidance as possible. In order to obtain a high degree of consistency regarding tag usage, users must have access to all required information about tag usage, including the list of all tags and obvious tag usage patterns or conventions. This can be compared to what Marlow et al. refer to as viewable tagging.

Type of object. Although we worked primarily with photos to develop and evaluate our approach, the resource type[2] is generally not restricted. The TACKO system was designed to be applicable for the organization of information objects of any kind, ranging from textual resources to geographic locations. In addition to photos, we apply the approach for instance to organize wiki pages, files and bibliographic references.

Source of material. We assume a closed collection of information resources being managed by an individual or a small group. Resources can be added to the collection by creating new resources within the system or by importing them from other tagging systems. However, since we do not restrict the type of resources, this implies that one can also use the system to manage links to arbitrary web pages as long as they have an adequate representation in the system. Aiming to enhance the capabilities of tagging systems in general, we do not predefine this characteristic.

Resource connectivity. Resources may be linked or assigned to groups being independent

[2]Since we generally use the term “resource” in this thesis, we use the term “resource type” as a synonym for the term “type of object” as it is used in [Ma06].

of the tag assignments but these relations among resources are not taken into account by our approach. They do neither influence the way the collection of resources can be navigated nor are they factored in when generating tag suggestions.

Social connectivity. As for the resource connectivity, links among users are possible but they are not relevant for the TACKO approach.

The characteristics described above are summarized in Table 3.1. In addition to the system design and attributes, Marlow et al. also provide another taxonomy of user incentives for tagging systems. Since the actual application of a system is hard to predict, we do not discuss this point in detail. The primary goal of our approach is to facilitate the future retrieval of the stored resources. This includes scenarios in which resources are shared among a group of other users. However, the other aspects, namely the attraction of attention, play and competition, self presentation, and opinion expression are not explicitly addressed.

Table 3.1.: Characterization of the tagging system underlying TACKO according to [Ma06].

Dimension	Specification
Aggregation model	set model
Tagging rights	simple permission based tagging
Tagging support	viewable tagging / suggestive tagging
Type of object	not restricted
Source of material	not restricted
Resource connectivity	not relevant
Social connectivity	not relevant

3.2.2. User interface guidelines

The design of user interfaces has been extensively studied in the last decades and numerous principles and guidelines emerged [Ma91, Sh92, CL99]. There are guidelines and rules for search and information seeking interfaces in general (e.g., [He09, SBC97] or [MR06, Section 6.3]) as well as for particular kinds of interfaces such as faceted search interfaces (e.g., [He06] or [St09b]) and even for specific interface elements, such as tag clouds [HK07]. Although it was not clear from the beginning that the TACKO user interface in its current form would resemble a faceted search interface, we mention the guidelines for this specific kind of user interface in this section as well, because they influenced the later development of the system.

In this section, a comprehensive summary of all relevant guidelines and design principles cannot be provided. The list presented here is limited to some research findings regarding the design of user interfaces for information seeking interfaces and tagging systems in particular that have had an influence on the design of the TACKO system. More specific issues will be covered in later sections where the user interface of the TACKO system is described in detail. Apart from guidelines, we further include important basic design decisions that had a major impact on the design of our system.

Shneiderman adapted his fundamental usability rules [Sh92] for search interfaces in [SBC97]:

- Strive for consistency
- Provide shortcuts for skilled users
- Offer informative feedback
- Design for closure
- Offer simple error handling
- Permit easy reversal of actions
- Support user control
- Reduce short-term memory load

They act as a frame of reference for several of the following guidelines that informed the design of the TACKO system:

Simplicity. In [He09, Chapter 1], Hearst begins her list of usability guidelines with simplicity. The need for simple user interface design for search tasks is motivated by the following observations:

- Search is usually part of a larger task that the user is trying to accomplish. An intrusive user interface can interrupt the flow of thought.
- "[...] search is a mentally intensive task" [He09, p. 1]. Consequently, there should be as little distraction as possible.
- Accessing information through search interfaces is such a fundamental and common task, that it is performed by a variety of different people with different backgrounds. Thus, each piece of functionality can and will lead to misunderstandings.

The last statement is backed by research indicating that some users even profit from a further simplification of present web search interfaces (e.g., [AK05]).

Efficient and informative feedback. This general principle by Shneiderman is further elaborated by Hearst [He09, Chapter 1]. Among other refinements, she offers the following guidelines:

- Show query term suggestions.
- Support rapid response.
- Allow sorting of results by various criteria.

Although we tried to conform with the first two, we decided to omit the result sorting options, at least in the version of the user interface that was experimentally evaluated, in order to reduce complexity for the users.

Stefaner et al. name two more principles aiming to give the user informative feedback regarding the result of an action when using a faceted search interface [St09b]:

- Animated transitions can help users to understand what happens between two animation steps. Animations can also direct the attention of users towards small changing details.

- A "filter summary" in the form of so-called *breadcrumbs* provides "a central place in the user interface" recording "the sequence of selection actions across all facets" [St09b].

Integrate navigation and search. Hearst advocates the integration of navigation and search in order to reduce the short-term memory load for users [He09, Chapter 1]. Particularly the use of facets is recommended. With facets, users can browse a collection of resources along different dimensions of categorization while still being able to filter by a particular keyword. For keeping track of the navigations steps, breadcrumbs are recommended as a history mechanism. Bergman et al. found that users have a strong preference for navigation when accessing their personal files [Be08]. Barreau and Nardi even report that search was only used as a "last resort" [BN95]. This also suggests that a user interface profits from the integration of navigation options in addition to pure keyword search. A more detailed description of the fundamental decisions regarding the access model is given in Section 3.2.3.

Design for closure. In the context of dialog design, Shneiderman writes: "Informative feedback at the completion of a group of actions gives operators the satisfaction of accomplishment, a sense of relief, the signal to drop contingency plans from their minds, and a signal to prepare for the next group of actions" [Sh92]. Shneiderman adapts this principle to search interfaces and points out that users should know when they have scanned the entire result set [SBC97]. Additionally, he notes that broad shallow navigation trees are preferable to narrow and deep ones. We want to highlight another aspect in this context: It is further desirable that users have the possibility to systematically assert that a certain item they are looking for is not in the collection. It can be very frustrating for users to try several search terms assuming that the searched item is included in the collection without having a systematic way to find out whether this is actually the case.

Reduce empty result sets. In order to reduce errors, Hearst suggests to prevent users from entering queries that yield empty result sets [He09, Chapter 1]. As long as navigation options are derived from the current result set — as it is for instance the case for tag clouds showing the most popular tags in this part of the collection — users cannot take such steps. This is also stated to be a feature of faceted search [St09b]. However, in later versions of our user interface, this guideline became relevant.

Presentation of navigation options. Halvey and Keane evaluated the suitability of tag clouds for the selection of specific options, such as selecting one option from a list of countries [HK07]. They found out that on average an alphabetically sorted list of equal-sized tags yields the best performance. A study by Trattner et al. revealed that users perceive faceted tag-based browsing interfaces as more powerful than traditional tag clouds, however, this was not reflected in actual performance measures, such as success rate or required time for search and exploratory tasks [Tr12].

Integration of the management of tag relations into the search interface. Finally, a fundamental decision informed the design of the user interface. In contrast to related approaches, such as the SOBOLEO system (see Section 7.2), it was decided to integrate the functionality for the management of tag relations into the interface being used for accessing the resources. The rationale behind this decision is complex:

- Tagging systems are simple and lightweight compared to many other knowledge organization systems. This is commonly perceived as one of their most important strengths. Switching between different interfaces adds complexity for users.

- Although integrating all tag management operations into the browsing interface poses challenges for the design of the user interface, it forces the designer to restrict the set of operations to the absolute minimum.
- Having the tools for organizing the tags always at hand, the user is more likely to use these tools and can make ad-hoc changes to the organization when necessary.
- We explicitly aim to facilitate the bottom-up emergence of structure in the sense that users can establish relations among tags when they see a necessity in the part of the collection they are currently working on. Contrary to that stands an organization structure that is developed independently and that is retroactively imposed on a collection of information resources.

3.2.3. Access model

In Section 2.3.3, an overview of different ways to access the contents of current tagging systems is given. In contrast to many such systems being primarily useful for browsing collections of resources without having a precise information need, the TACKO systems aims to facilitate purposeful location of particular information resources. This has several implications on the access model.

We describe the fundamental aspects of our access model using the four-phase framework for search by Shneiderman et al. [SBC97]:

1. Formulation
2. Action
3. Review of results
4. Refinement

Although this framework has been developed for text search, it is abstract enough to be applicable for tag-based information access as well. However, since we chose to integrate search and navigation, the most emphasis is on the refinement step.

At this point, it is necessary, to define how we distinguish the terms *search* and *navigation* in the context of a tagging system. Hearst makes the distinction that

> "[...] search queries tend to produce new, ad hoc collections of information that have not been gathered together before, whereas navigation/browsing refers to selecting links or categories that produce pre-defined groups of information items" [He09, Chapter 8].

In the TACKO system in its basic form, tags are the only means by which information resources are described and accessed. As a consequence, both, issuing a search query and a navigation step, can be seen as the formulation of an information need represented by a *tag filter* (also referred to simply as *filter*) which in the simplest case consists of a single tag. So for our purpose, the difference is only in the way the user selects the filter but not in the characteristics of the set of resources being retrieved. In the TACKO system, the user can select a tag from several initial navigation options or enter a tag manually into a search bar to

formulate an information need. In both cases, choosing a tag can be considered the selection of a pre-defined group of information items, i.e., the group of resources being implicitly defined by the assignment of the tag in question. Ad hoc collections in the sense of Hearst can rather be obtained by combining several tags in the filter which is possible in the refinement phase (see below).

After the formulation phase, the system filters the set of all resources according to the selected tag. This means that the user retrieves the set of all resources having the selected tag assigned as a result set.

Subsequently, the user can review the result list. In addition to the list of matching resources, the system presents a list of navigation options represented by tags that can be used to refine the tag filter. These options do not necessarily have to be displayed in the form of a tag cloud. During the development of TACKO, we varied the way tags are presented as well as how the set of navigation options is selected. However, in any case these tags additionally serve as a summary of the current result set. Thus, they can be considered a *self-adapting exploration structure* in terms of Sacco et al. [SFT09].

The refinement phase is central for the information seeking process in the TACKO system, particularly because it cannot be expected that a tag search leads to the desired information in one step. This is not only because it is unlikely that a query for a single tag yields a sufficiently small result set: In [Te04], Teevan et al. report that users apply a combination of keyword search and "small, local steps" to reach their information target in personal collections of files and emails. They use the term *orienteering* for the latter. The former is called *teleporting*. More specifically, it was observed that orienteering was applied by users "even when they knew exactly what they were looking for in advance" [Te04]. This is in line with other research findings which confirm that users generally prefer navigation over search (see Section 3.2.2). If the user starts the information seeking process by selecting a tag from a list of suggestions instead of typing it in, one can even consider the whole process as a gradual refinement of the – initially empty – filter.

An important design decision that distinguishes TACKO from many other tagging systems was to narrow down the result set as the default effect of the selection of a navigation option. Contrary to this behavior, the selection of a tag leads to a reorientation of the current view in other systems featuring so-called *pivot-browsing* [MFK06]. In the TACKO system, a tag that is selected by the user is added to the filter and does not replace it. In order to obtain a narrower results set, the system interprets the filter as a conjunctive combination of tags: to qualify for the result set a resource needs to have all tags in the filter assigned. It is also possible to remove any tag from the filter in the refinement phase. In general this will lead to a broader result set. Finally, the system always provides the option to add an arbitrary tag to the filter, regardless of whether this tag is included in the displayed navigation options.

This behavior is not unique to the TACKO system. The social bookmarking service delicious[3] has the same behavior for the list of "related tags" shown together with each result set. However, when the user selects a tag being assigned to one of the resources in the result, the filter is replaced by this single tag. Delicious also allows to switch to completely different views during the browsing process, such as the list of all bookmarks of a specific user.

[3] http://delicious.com, (accessed 6th of February 2012).

Narrowing the filter upon the selection of an additional term is also the default behavior in faceted search interfaces. Stefaner et al. name six basic filter refinement operations for such interfaces:

> "Based on an analysis of existing applications, we can distinguish the following navigation modes:
>
> - **zoom-in** makes the query more specific,
> - **zoom-out** makes it more general,
> - **shift** replaces a part of the query by a related concept,
> - **pivot** replaces the whole query by a related concept,
> - **slice-and-dice** allows the disjunctive selection of multiple concepts within a facet,
> - **range selection** offers the options to specify query intervals within ordinal or real value facets". [St09b, p. 79]

While the last one is not applicable for tagging systems, the others could be implemented. However, we chose, to limit the refinement options to zooming in and zooming out in order to keep the interface as simple as possible. It is particularly not possible to express a disjunction of tags.

Very early during the development of TACKO it became obvious that for our purposes it is often required to exclude resources with particular tags from the result set. Hence, the model was extended to also allow so-called *negative tags* in the filter. A negative tag can be considered the complement of a normal tag. Although the negation of tags (or concepts) is not explicitly mentioned in the list above, it represents just a special form of zooming in, since the query becomes more specific.

For the sake of clarity and to allow for future reference, we summarize the notion of a filter formally. For this purpose, we introduce for every tag t a virtual negative tag $\bar{t}$ that is assigned to all resources that t is not assigned to, such that

$$extension(\bar{t}) \cap extension(t) = \emptyset$$

and

$$extension(\bar{t}) \cup extension(t) = R.$$

Let $\bar{T}$ denote the set of all complementary tags. We further define $\hat{T} = T \cup \bar{T}$.

The set of resources matching a filter $F \in \mathcal{P}(\hat{T})$ is called its *extension* and it is defined as

$$extension(F) = \bigcap_{t \in F} extension(t).$$

Finally, we say a filter F is more specific than another more general filter F' if it contains additional tags, i.e., $F \supset F'$.

Navigation in the system can now be modeled as the application of a sequence of filters to a collection of resources under the restriction that each filter is either more specific or more general than the previous one.

3.3. Implicit tag relations

This section covers the first major development phase of the TACKO system. During this phase, it was avoided to store explicit tag relations in order to lose as little flexibility as possible compared to traditional tagging systems. We have briefly covered this approach in [MNS12b].

Section 3.3.1 provides a general characterization of implicit tag relations. We then proceed with a brief formal description of the different kinds of relations in Section 3.3.2. Subsequently, Section 3.3.3 covers a central aspect of our approach, namely the detection of facets in tagged item sets. In Section 3.3.5 we discuss the advantages and limitations of implicit relations and we draw a brief conclusion in Section 3.3.6.

3.3.1. The basic idea

Our approach of implicit tag relations can be systematically derived from the observations described in Section 3.1:

1. Synonyms are more problematic than homonyms.
2. Latent tag relations can be exploited.
3. Tag assignments are incomplete.

To better illustrate the idea, we completely neglect the existence of homonyms for the moment and assume that each tag has a unique and clearly defined meaning. It is not required to explicitly capture the semantics of a tag but it is sufficient to assume that all users of the system interpret a tag in the same way. This means particularly that they come to the same conclusion when assessing whether a tag applies to a specific resource or not and, thus, whether it should be assigned to this resource.

Under such conditions, the incompleteness of tag assignments could be exactly determined in terms of missing — and mistakenly assigned — tags (possible causes are covered in Section 3.1.3). The key idea is now, to give users the tools to correct such deficiencies by allowing them to assign and remove the respective tags with little effort, i.e., without changing the tags of each resource individually.

This way, the usage of tags can be harmonized — more precisely the way tags are used in combination. As a result, the latent and previously "fuzzy" tag relations are sharpened. One example of such a relation is the so-called *subsumption* relationship among terms that can be observed in term frequencies in text [SC99] as well as in the tags of folksonomies [Sc06b]. Let $P(x|y)$ denote the probability that tag x is assigned to a random resource, provided that tag y is assigned. In other words, $P(x|y)$ denotes the fraction of resources tagged x within the set of resources tagged y. If a general tag x subsumes a more specific term y this is often reflected in the cooccurrences of x and y as follows:

$$P(x|y) \geq t, \quad P(y|x) < 1$$

where t is a certain threshold value that can be adjusted. Sanderson and Croft chose $t = 0.8$ for the detection of term subsumptions in texts [SC99]. If users have the tools to specifically

change the tags of many resources at once, they can easily assert that y is never used without x. More precisely, they can assign the missing tag x to all resources having y assigned. This way the threshold t can be set to 1. Figure 3.5 illustrates how a clear implicit subsumption relation is established.

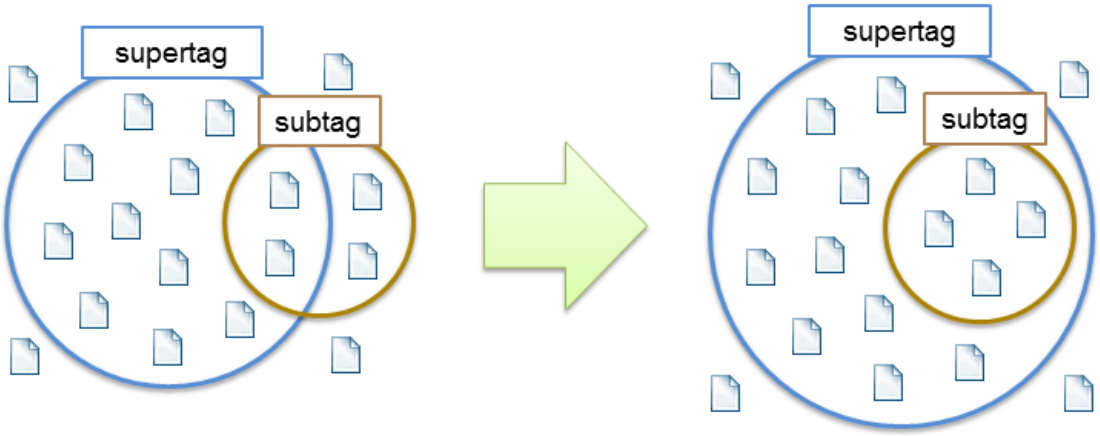


Figure 3.5.: Schematic illustration of how fuzzy implicit tag relations can be sharpened.

As a consequence of the harmonization, not only will the result set for a query for tag x contain more relevant results than before — i.e., recall is increased —, but also the precision of the automatic detection of subsumptions is improved: The threshold t can now be set to 1 eliminating many false positives. Since the subsumption relation among tags is not explicitly expressed, but only embodied in the cooccurrence of tags, we call it an *implicit* tag relation.

Even though the cooccurrences alone do not imply that tags are in fact in a subsumption relationship from a semantic point of view, the implicit relation can for instance be exploited to generate clearer tag clouds: If the general tag x is in the cloud, y can be considered redundant and thus it can be hidden.

In addition to subsumption there are more types of implicit relations that can be expressed through consistent tag assignments. Particularly interesting is the extraction of facets, i.e., groups of tags that categorize resources with regard to the same aspect or dimension of categorization. Section 3.3.2 summarizes all kinds of relations and gives brief formal definitions.

In order to exploit the tag relations for navigation, they have to be continuously derived from the tag assignments of the resources under consideration. Each change in the tag assignments can void existing relations or establish new ones which requires at least a constant monitoring of all assignments. Furthermore, analyzing the current result set dynamically after each search or navigation step provides additional opportunities: The navigation structures can adapt to the subset of resources the user is currently browsing. This allows to provide the user with a potentially better set of navigation options being tailored to her current needs. We generally say, that implicit tag relations may apply only in a certain *context*. Such a context is represented by a set of resources being consistent with the condition defining the relation (e.g., the subsumption condition given above). This set of resources is in turn determined by a filter. We also say that an implicit relation applies in the context of a specific filter.

Another important characteristic of implicit tag relations is their fragility. If users are not guided during the assignment of tags, they can unintentionally break a complex implicit structure that has been consciously established before. Supporting the user with automatic tag assignments is one means to preserve existing relations: When there is a subsumption relation between a general tag x and a more specific tag y, the system can automatically assign x to a resource as soon as it is tagged with y. The user sees the automatic assignment and can remove the tag if the respective implicit relation does not hold anymore. This way it is ensured that implicit relations are only intentionally removed by users.

Having outlined the elementary notion of implicit tag relations, we now give an overview of the different types of implicit relations before we discuss the consequences of this approach in more detail.

3.3.2. Types of implicit tag relations

Figure 3.6 summarizes the different kinds of implicit tag relations being relevant for our approach and illustrates how they are embodied in the tag assignments. Each circle stands for a particular tag and encloses the set of resources the respective tag is assigned to. Figure 3.6a depicts the default case where no particular relation can be observed. The extensions of the tags overlap to some extent but the overlap is not large compared to the individual tag occurrences. In the following, the different types of implicit relations are formally characterized and illustrated with examples.

3.3.2.1. Subsumptions

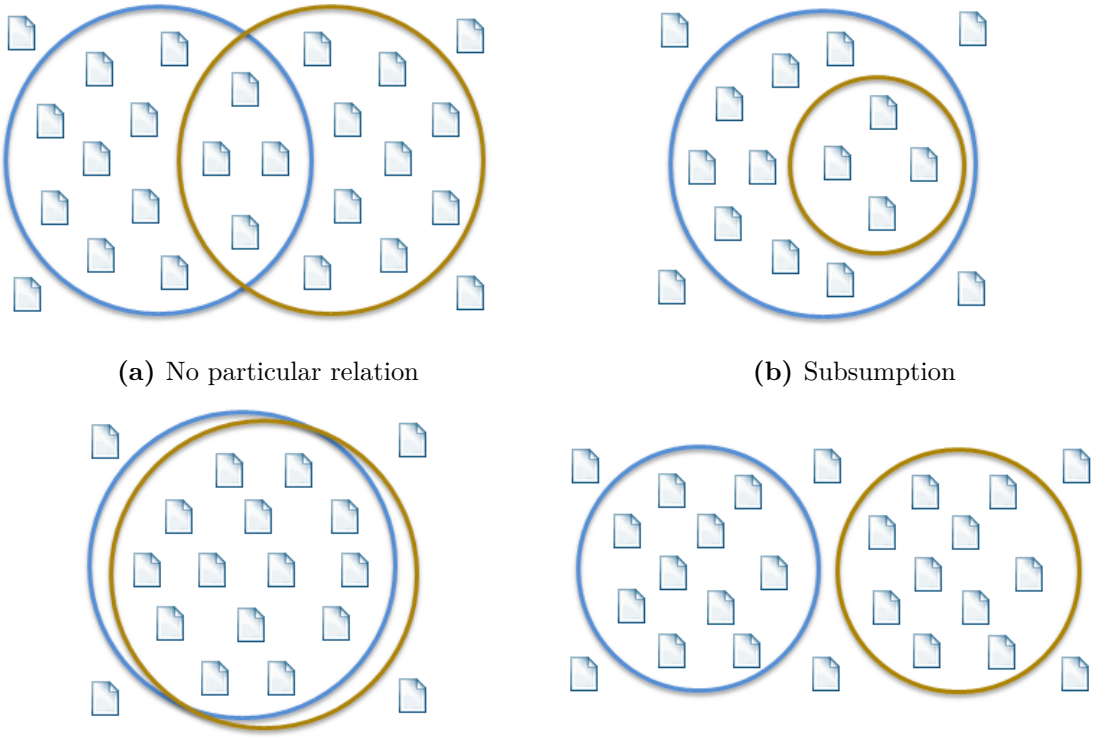
Figure 3.6b depicts the subsumption relation. We say a tag t_1 *subsumes* a tag t_2 ($t_1 \geq t_2$) if the extension of t_2 is a subset of the set of resources tagged with t_1. This may not apply in the whole collection of resources but only within a certain subset. Formally we define tag subsumption between two tags $t_1, t_2 \in T$ in a certain subset of the resources, i.e., a context determined by an arbitrary filter $F \in \mathcal{P}(\hat{T})$:

$$F \vdash t_1 \geq t_2 \quad \text{iff} \quad \mathit{extension}(F \cup t_1) \supseteq \mathit{extension}(F \cup t_2)$$

Subsumption is not defined on negative tags.

A subsumption relation says little about the semantics of the tag. Establishing such a relationship is appropriate between hyper- and hyponyms, such as “project” and “research project”, as well as when there is a “part-of” relationship between holonyms and meronyms, e.g., “europe” and “germany”. It *can* be used to represent generalization-specialization trees comparable to classic taxonomies. However, it is more flexible in two ways:

- A tag can be subsumed by several other tags that are not in a subsumption relation with each other. The tag “research project” can for instance be subsumed by the independent tags “research” and “project”.
- The entirety of all subsumption relations does not have to form one coherent structure but independent structures for different dimensions of categorization can exist.



(a) No particular relation

(b) Subsumption

(c) Equivalence

(d) Mutual exclusion

Figure 3.6.: Types of implicit tag relations

In the context of a specific filter, tag subsumptions form a directed acyclic graph in which the nodes are represented by tags and the edges by subsumption relations. Figure 3.7 shows an excerpt of a fictitious example of such a graph that demonstrates the relation's versatility. It is important to note that this graph may change when the user changes the filter and thus navigates to another context.

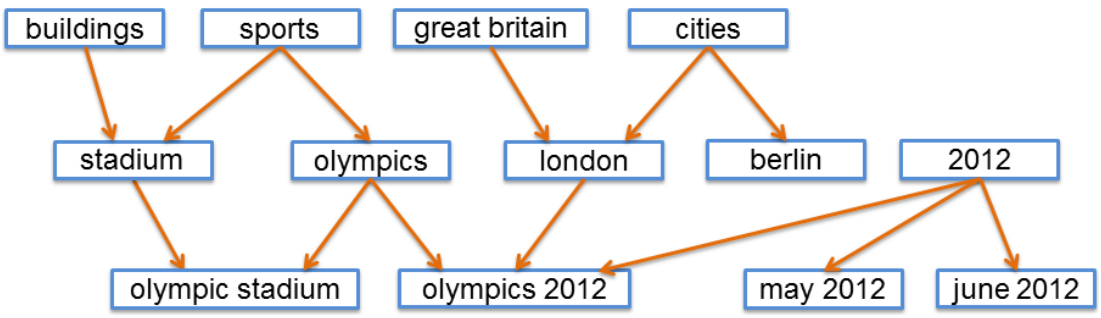


Figure 3.7.: Example of a subsumption structure.

The example shows that the structure offers many possibilities to hide tags in a tag cloud. It would be sufficient to show only the tags not being subsumed by others because each subsumed tag can then be reached by means of one or more additional navigation steps. More precisely, the user can select a subsuming tag from the tag cloud and will then obtain a

new set of navigation options for the respective narrower result set. This way the filter can be systematically narrowed until the desired tag appears (always assuming that the maximum number of tags in the tag cloud is not limited).

While the tag relations depend on the current filter, they do not change in an arbitrary way between two navigation steps. Narrowing the filter preserves all present subsumptions:

$$F_1 \vdash t_1 \geq t_2 \ \wedge \ F_1 \subseteq F_2 \quad \Longrightarrow \quad F_2 \vdash t_1 \geq t_2$$

When the context is very specific, tag subsumption relations can apply to tags that are semantically independent in general: In the personal photo collection of a user it may hold that "2012" subsumes "france" if the user never travelled to France in another year.

3.3.2.2. Equivalence

Equivalence of two tags $t_1, t_2 \in T$ in a certain context defined by a filter $F \in \mathcal{P}(\hat{T})$ simply means that for each resource it holds that either t_1 and t_2 are both assigned or neither of them. In other words their extensions are identical:

$$F \vdash t_1 \equiv t_2 \quad \text{iff} \quad extension(F \cup t_1) = extension(F \cup t_2)$$

An obvious reason for this type of cooccurrence relationship is that t_1 and t_2 are semantically equivalent, i.e., they are synonyms.

There are several reasons why a tagging system contains different tags with the same meaning, be it spelling-variations, multiple languages or simply different words for the same concept in one language. One could argue that this introduces confusion and that the users of the system should instead agree on one term that is used consistently. While this might be easy to achieve in personal tagging systems, eliminating synonyms usually means removing tags from the system that someone else has consciously chosen to describe a set of resources. The equivalence relation offers the possibility to keep both tags and use them consistently. Additionally, using synonyms has the advantage that users do not have to remember which term is the correct one when directly searching for a particular concept for which synonym tags exist.

Harmonizing the use of synonym tags in a folksonomy clearly leads to a better recall for tag searches. In current folksonomies, there is usually a large overlap in the extensions of two synonyms but many resources are only tagged with one of them although both tags apply.

Like subsumption relations, tag equivalence can be detected in a resource set and tag clouds can be made clearer by displaying only one tag of an implicit synonym ring. When implicit tag relations are consequently applied, it is advisable to hide tags from the navigation options when they are assigned to all resources in the current result set. These tags provide no value with regard to the navigability of the system and obscure the more meaningful options.

Since tag equivalence is context-dependent like all implicit tag relations, it is not restricted to synonyms. There may be cases where two tags are used interchangeably in a specific context although all users would agree that they have different meanings in general. Again, we

consider the personal photos of a single fictitious user. It makes sense to use the tags "france" and "paris" interchangeably if the user has no photos of other cities in France. When he visits such cities in the future, tag equivalence will turn into a subsumption relation, but until then the tags are equivalent for the purpose of browsing the photos.

3.3.2.3. Mutual exclusion

We say that two tags $t_1, t_2 \in T$ are *mutually exclusive* in a context determined by filter $F \in \mathcal{P}(\hat{T})$ if the intersection of their extensions is empty:

$$F \vdash t_1 \sim t_2 \quad \text{iff} \quad extension(F \cup t_1) \cap extension(F \cup t_2) = \emptyset$$

For a set of tags $M \in \mathcal{P}(T)$ we define pairwise mutual exclusion *PME* as

$$F \vdash PME(M) \quad \text{iff} \quad \forall a, b \in M,\ F \vdash a \sim b$$

While it is a particular strength of tagging systems that each resource can be associated with several categories, there are tags naturally not being combined with each other. If a photo is tagged "summer 2011", it is unlikely that the tags "summer 2012" or "winter 2011" apply, too. Mutual exclusion indicates that such tags belong to the same facet, i.e., the same dimension of classification (such as time, place, et cetera), particularly if several tags being pairwise mutual exclusive can be detected (see Section 2.2.2 for a more detailed discussion of facets). The number of possible facets in a resource set is not limited. Our notion of a facet comprises all reasonable ways to "slice and dice" the resource set. For instance in a typical collection of Flickr photos, we find tags indicating the make and model of the camera, whether a photo is a grayscale or color photo, whether the photo shows people, whether it was shot at day or at night, which buildings it shows, et cetera.

However, it has to be noted that not all kinds of facets can be reliably detected. Particularly problematic are facets of which typically several concepts are selected to categorize a resource. In collections of photos, a color facet is a good example. Since the extensions of different color tags can overlap, they are in general not recognized as being mutually exclusive anymore. The detection of facets is thus limited to what we call *single-valued* facets in contrast to *multi-valued* ones. Experimental results for automatic facet detection in folksonomies are covered in Section 3.3.3.

Facets are particularly useful for narrowing down a result set when navigating a folksonomy. We therefore suggest to group the tags of a pairwise mutually exclusive set in a column in order to make it easier for users to grasp how the terms are related. If several facets can be detected in a collection of resources, the list of navigation options rather resembles a *faceted search interface* [ST09a, He09] than a tag cloud. Figure 3.8 shows a tag cloud with one automatically detected facet and some other frequent tags.

Not only the presentation of navigation options can be improved with facets but they can also facilitate the indexing process. A list of facets shows users which aspects of categorization are relevant in a collection of resources. Further, facets make it easier to find appropriate tags: Selecting tags from preset lists is cognitively less demanding than making up tags without any guidance. One could also urge users to select a tag from a facet, if all resources in a

Figure 3.8.: Automatically detected facet in a consistently tagged photo collection.

certain context are tagged with at least one tag from the facet and the resource being tagged would be the first one without a tag of the facet. We call a facet *exhaustive* in such a case.

Although we argue that homonyms are rarely found in the kind of resource collections that our approach focuses on, we illustrate, how the detection of mutually exclusive tags can help to disambiguate homonyms if they occur: When tag subsumption relations are consequently applied in a collection of resources, and thus specific terms are subsumed by general categories, it is likely that the search for an ambiguous tag yields a result set that can be partitioned into extensions of mutually exclusive tags. For instance, when a user is searching for the tag "bow", it might turn out that part of the returned resources are concerned with "range weapons" and the rest with "string instruments". When this is automatically detected and revealed to the user, it allows to refine the search by adding one of these mutually exclusive tags to the filter. In effect, this leads to a more precise search result.

3.3.3. Detecting implicit facets

In this section, it is demonstrated, how implicit, latent facets can be detected in real-world folksonomies based on a binary integer linear programming approach. The findings presented in this section are based on a guided research project conducted by the student Bernhard Waltl and supervised by the author of this thesis [Wa12].

As already briefly described in Section 3.3.2.3, the key idea underlying our approach of facet detection is that there are certain facets being reflected in a partitioning of the set of resources, i.e., the extensions of the facet tags are mutually exclusive. We assume that when a set of pairwise mutually exclusive tags can be found, it is likely that it represents a facet, or at least part of the tags belong to the same dimension of categorization.

This stands in contrast to most other approaches aiming to detect and exploit structure in folksonomies. Many of them rely on clustering or the construction of tree structures based on tag subsumption relations (see Chapter 7). In both cases it is focused on groups or pairs of tags having a substantial overlap regarding their extensions.

However, it cannot be expected that there is no overlap at all for facet tags in real-worlds folksonomies. Resources are not tagged consistently and there can always be exceptional resources that justify the assignment of several tags of the same facet even if the "nature" of the facet suggests that only one tag is assigned. In consequence, a small overlap will be tolerated. Similarly, it is not likely, that a facet is collectively exhaustive in the sense that all

resources are categorized with respect to the facet. This was also taken into account when developing the implementation of our approach presented below.

3.3.3.1. Implementation as a linear program

The selection of facet tags in a folksonomy can be treated as an optimization problem. The goal is to obtain a set of pairwise mutually exclusive tags and the set should be maximal in the sense that no more tags can be added without violating the mutual exclusion constraint. To solve this problem, we transform it into a binary integer program.

We chose linear integer programming because it is relatively simple, well understood and sufficient to express our fundamental constraints. Additionally, it allows some variation regarding the objective function.

A linear program has the following canonical form:

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

The term $c^T x$ represents the objective function that is to be maximized by choosing an optimal assignment for the variables in the vector x. In our case, this (binary) vector encodes the selection of tags. Each element of x corresponds to a certain tag. A value of 0 indicates that the respective tag was not selected, a value of 1 means that the tag is part of the facet.

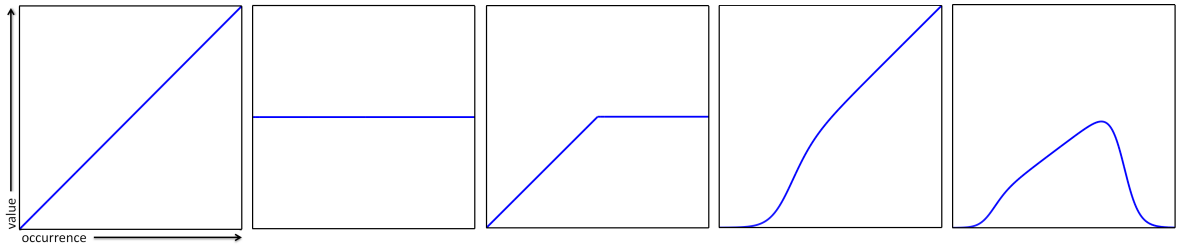
The matrix A and the vector b define the constraints, i.e., the solution space. They are used to encode which pairs of tags are not mutually exclusive and thus may not be part of the same facet. The number of columns in the matrix corresponds to the number of tags. The number of rows to the number of tag pairs with overlapping extensions. A single matrix row expresses that a certain combination of two tags is not allowed. It contains only zeros except for the positions corresponding to the respective tags. These two elements are set to 1. Finally we set all elements of the vector b to 1. In effect, a row in the matrix encodes a linear inequality constraint of the following form:

$$0 * x_0 + ... + \underbrace{1 * x_i}_{i^{th} tag} + ... + \underbrace{1 * x_j}_{j^{th} tag} + ... + 0 * x_n \leq 1$$

with $x_0, ..., x_n \in \{0, 1\}$: and n representing the number of tags (cf. [Wa12]).

The matrix A can be simply generated based on the given folksonomy. Thereby, the restriction that tags are mutually exclusive can be lessened, i.e., no constraint is generated for a pair of tags when the number of resources is sufficiently small. For this purpose, the generation of A is performed with a parameter *allowedOverlap*. The parameter defines the maximum fraction of the resources tagged with the less frequent tag that may be tagged with the more frequent tag as well. If for instance tag a occurs 10 times and b 20 times, a value of 10% would not lead to a constraint if there is only 1 resource having both tags assigned.

Finally, the vector c in the objective function has a major impact on the resulting tag selection. It contains a value for each tag and the solution of the linear program maximizes the total



(a) Number of Occurrences **(b)** Uniform **(c)** Capped **(d)** Logistic **(e)** Logistic with Cutoff

Figure 3.9.: Schematic illustration of the different weight functions [Wa12].

value of the selected tags. These values will be called *weights* in the following. If all tags have equal weights, solving the linear program will simply maximize the number of tags in the facet. However, it turned out that the resulting facets were not very meaningful since a high number of very rare tags are selected this way. Frequently used tags are hardly included in the result. In consequence, several *weight functions* were compared. A weight function computes the weight of a tag based on its relative frequency in the collection.

Figure 3.9 schematically depicts the graphs of the different weight functions:

- **Number of occurrences.** The tag weights are proportional to the tag frequencies.
- **Uniform.** All tags have the same weight.
- **Capped.** Tag weights are proportional to the tag frequencies but a certain maximum weight is not exceeded.
- **Logistic.** A logistic function with two parameters α and β is factored in to lower the weights of very rare tags:

  $$W_{log}(t) = \frac{occ(t)}{1 + e^{(-\alpha * (occ(t) - \beta))}}$$

  where $occ(t)$ denotes the number of occurrences of the tag t.
- **Logistic with cutoff.** The weights of very frequent tags are lowered by factoring in another logistic function accordingly.

These weight functions were chosen during the first tests of the approach with sample datasets. In this process, the respective parameters were varied to fit the specific characteristics of the dataset. It could not yet be determined, how the parameters have to be chosen to obtain good results in general.

3.3.3.2. Experimental results

The approach was tested with different weight functions in two sample folksonomies obtained from the photo sharing platform Flickr. Via the Flickr API, photos of the public Flickr groups "Munich, Germany"[4] and "Product Photography"[5] were obtained. Photos with less than four

[4] "Munich, Germany", http://www.flickr.com/groups/munich/, accessed on October 11th, 2012
[5] "Product Photography", http://www.flickr.com/groups/product/, accessed on November 26th, 2012

tags were ignored. The resulting datasets contained 31,711 photos and 30,860 distinct tags for the Munich group and 29,272 photos with 47,684 tags for the products group.

Canon	(1711)	**Nikon**	(1153)	**Alcohol**	(492)	**Watch**	(844)
1000d	(37)	d100	(26)	**absolut**	(33)	breitling	(22)
300d	(57)	d200	(39)	beer	(88)	festina	(15)
350d	(50)	d300	(69)	gin	(16)	fossil	(38)
400d	(120)	d40	(84)	lichido	(38)	guess	(19)
40d	(37)	d50	(83)	rum	(16)	horloge	(20)
450d	(44)	d5000	(45)	scotch	(13)	**lighting**	(18)
500d	(135)	d700	(97)	wine	(50)	**micro**	(17)
550d	(87)	d70s	(33)			reloj	(18)
5dmarkii	(108)	d80	(97)			rolex	(30)
7d	(60)	d90	(129)			**seamaster**	(14)
ixus	(39)	**film**	(35)			seiko	(29)
moment	(41)	**instantfave**	(122)			**timepieces**	(16)
powershot	(85)	**soninka**	(23)			**watchpho...**	(22)
robert	(35)					**©allright...**	(73)

Table 3.2.: Four automatically detected facets in subsets of two Flickr datasets containing photos of Munich and products.

It was not only tried to extract meaningful facets from the whole folksonomies, but also subsets of resources sharing a common tag were considered. Our goal was to show the general applicability of the approach, so we manually selected illustrative examples. For several subsets, very promising results could be obtained. Table Table 3.2 shows four example facets. All of them were obtained using the weight function "Logistic with cutoff". The tags in the table header define the subset of the collection in which the facet detection was applied. The number to the right denotes how many resources the respective tag was assigned to. The red bold tags do not belong to the facet and can be considered false positives.

It turned out that the results obtained for the complete collections were rather problematic. Although the detected facets contained several tags that could be considered as semantically belonging to the same dimension of categorization, many other tags were included as well or several semantic facets were mixed. Better results were obtained when the collection was narrowed down to a set of resources sharing a particular tag.

However, it has to be noted, that the datasets — being the result of the tagging efforts of several thousands of users — were not modified. The results could be significantly improved when synonym tags were harmonized and missing tags were added. The alcohol facet in Table 3.2 can for example be improved by assigning the tag "vodka" to all resources being tagged "absolut", a specific vodka brand. When this is repeated for the names of other vodka brands, the tag "vodka" appears in the facet instead of "absolut".

It also has to be stressed that the tags in the detected facets are very different from the tags being shown in a normal tag cloud that contains only the most frequent tags. In Table 3.3 it can be seen that there is only one tag, namely "bmw" — in this case representing the bmw museum in Munich — in the facet and at the same time in the ten most frequent tags. It can also be seen that the museum facet is detected although the facet tags cover only about half

Museum (detected facet)	(1039)	**Museum** (top tags)	(1039)
altepinakothek	(38)	architecture	(174)
artgallery	(34)	art	(278)
bmw	(148)	bavaria	(331)
brandhorst	(137)	bayern	(320)
deutsches	(67)	bmw	(148)
glyptothek	(39)	deutschland	(379)
musée	(40)	germany	(554)
neuepinakothek	(18)	munich	(875)
residenz	(17)	museum	(1039)
theatermuseum	(41)	münchen	(684)

Table 3.3.: Comparison of an automatically detected facet and the most frequent tags in the set of photos tagged "museum" in the Munich photo collection.

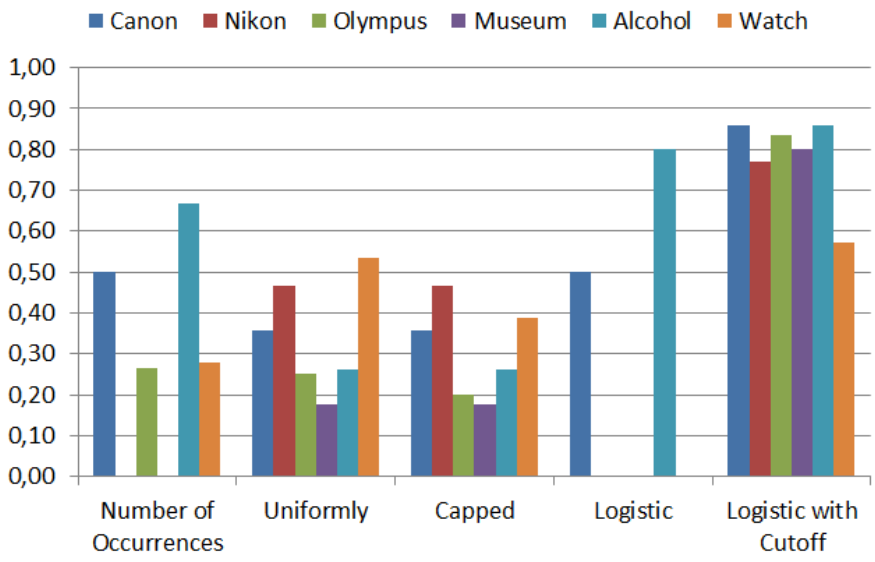
of the 1,039 resources being tagged "museum". This was possible by using the weight function "Logistic with cutoff". In contrast, the weight function "Number of occurrences" would likely lead to a facet that contains a very frequent tag such as "München" and some other very rare tags not overlapping with "München".

The different tag weight functions were assessed by comparing precision and recall of the facets detected in selected resource sets for which the best results were obtained. Thereby our goal was to preliminary evaluate the performance of the different kinds of weight functions and to find out which of them is adequate for our purpose at all. The results are visualized in Figure 3.10. The precision metric represents the fraction of relevant tags in the facet. Recall is the number of relevant tags in the facet divided by the total number of relevant tags. The latter number was obtained by counting the number of distinct relevant tags within the results for all five tag weight functions. It can be clearly seen that the weight function "Logistic with cutoff" yields the best results.

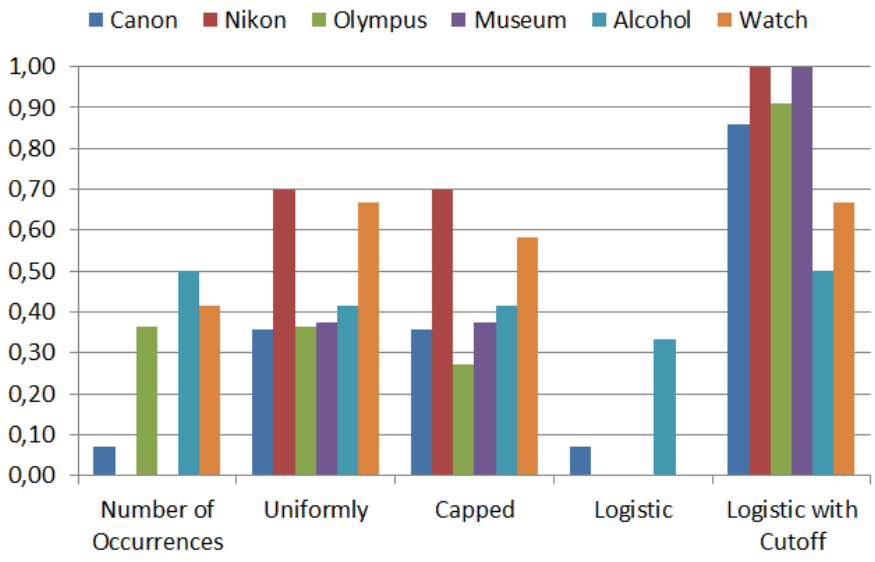
Although this approach of facet detection is far from being mature yet, it has been demonstrated, that it can be used to extract meaningful facets from the tag sets of real world folksonomies. Further research is necessary to evaluate additional tag weight functions and to find generally practicable parameters. Potential for improvement lies also in more complex optimization techniques that allow to express more complex constraints and objective functions. Finally, a combination of the approach with clustering and hierarchy detection could further improve results.

3.3.4. User interface

In the following, we cover the different elements of the user interface for displaying and managing implicit tag relations. Although we strived to avoid a separate interface for the management of tag relations, we first integrated only the browsing functionality into the existing search interface of the Tricia platform, into which the TACKO extension was integrated. This was done in order to make the browsing interface accessible to users before the complete interface was mature enough to be used in production.



(a) Precision



(b) Recall

Figure 3.10.: Comparison of the results obtained for different tag weight functions.

3.3.4.1. Basic search interface

Our prototype implementation was always part of the web-based collaboration platform *Tricia* (see Section 5.1.1 for details on Tricia). In Figure 3.11, the search interface of Tricia is shown. In this view, it is not possible to change the tags of many resources at once, but one can see, how the tagging features are integrated with the rest of the filtering and search functionality. We will describe the enhanced interface for the management of implicit tag relations below but the description of the user interface would not be complete without a brief summary of Tricia's main search screen.

The elements relevant for the tagging functionality are only the breadcrumb (A), the tag cloud above the result list (B) and the list of tags below each search result (C). When facets are automatically detected, the respectig tags are displayed in columns. An example is shown in Figure 3.8. The breadcrumb simply records the tags in the filter in the order they were selected. Tags are added to the filter by clicking them. A click on a tag in the breadcrumb removes all tags from the filter that have been added later than the clicked tag, i.e., all tags right of it. Finally, in the search result list in the lower half of the figure, each resource in the result shows a list of its tags. The tags of an individual resource can be edited directly in this view. The search results are presented on pages of 10 search hits each. The paging controls are at the bottom of the page.

Figure 3.11.: User interface for browsing a set of resources integrated in the search interface of the Tricia platform.

At the left side of the screen, several collapsible built-in facets are shown:

- **Content type.** In general the result can be restricted to files, blog posts, wiki pages et cetera. Options are hidden when no matching resources are in the result set.

- **Workspace.** The Tricia platform in this version could contain different wikis and blogs as so-called workspaces. Each resource is assigned to a single workspace. The workspace has implications on the access rights of a resource.
- **Type.** Wiki pages can have user-defined types and attributes that can be used for filtering.
- **Special.** This facet can be used to find resources containing broken links or invalid attribute values.

Below the tag cloud, the number of results is shown. The search field next to it can be used to combine the current filter with a full-text search. The result list can be sorted by name, relevance or date of last modification. Finally, at the upper-right corner, the online help can be accessed and a specific action can be triggered: By clicking "embed in page" the current result set can be embedded in any wiki page as a list. This list is then updated automatically, e.g., when new matching resources are added.

3.3.4.2. Tag relation management interface

To establish and maintain tag relations, we developed an extension of the standard search interface. It is depicted in Figure 3.12. It provides capabilities to assign tags to all resources matching the current search filter as well as to remove tags from this resource set. As for the search interface, the resources are shown in the lower part of the page and the tag cloud is displayed above of them.

Three colored regions (green, gray, red) are located above the tag cloud. Since tags can be dragged into these regions, they are called *containers* in the following. The left and the right container are part of the filter and can thus be used to specify the set of matching resources.

- The green "required tags" container holds tags a resource must have assigned in order to match the filter. If multiple tags are contained, all of them are required.
- The red "excluded tags" container allows to narrow the filter further. It determines which resources are excluded from the current focus and it is applied in conjunction with the green container. If it contains multiple tags, resources with any of these tags are excluded from the result set.
- The gray "clipboard" container is independent of the filter. It can be used to temporarily store tags that are frequently needed to refine the search. This way users do not have to enter the tag manually but they can use a drag and drop operation to move the tag from the clipboard to another container or a resource.

All the tags shown in Figure 3.12 can be moved by dragging them with the mouse. In order to put a tag into one of the containers it is simply dropped onto it. If a tag is dragged from one container into another one, it disappears from the source container. After each such operation, the page is reloaded to display the updated result set. If a tag is not shown in the interface it can be manually entered via an input field which appears when the user clicks into one of the containers. All input fields for tags in the system offer autocomplete suggestions based on the existing tags while the user is typing. Figure 3.13 shows an example.

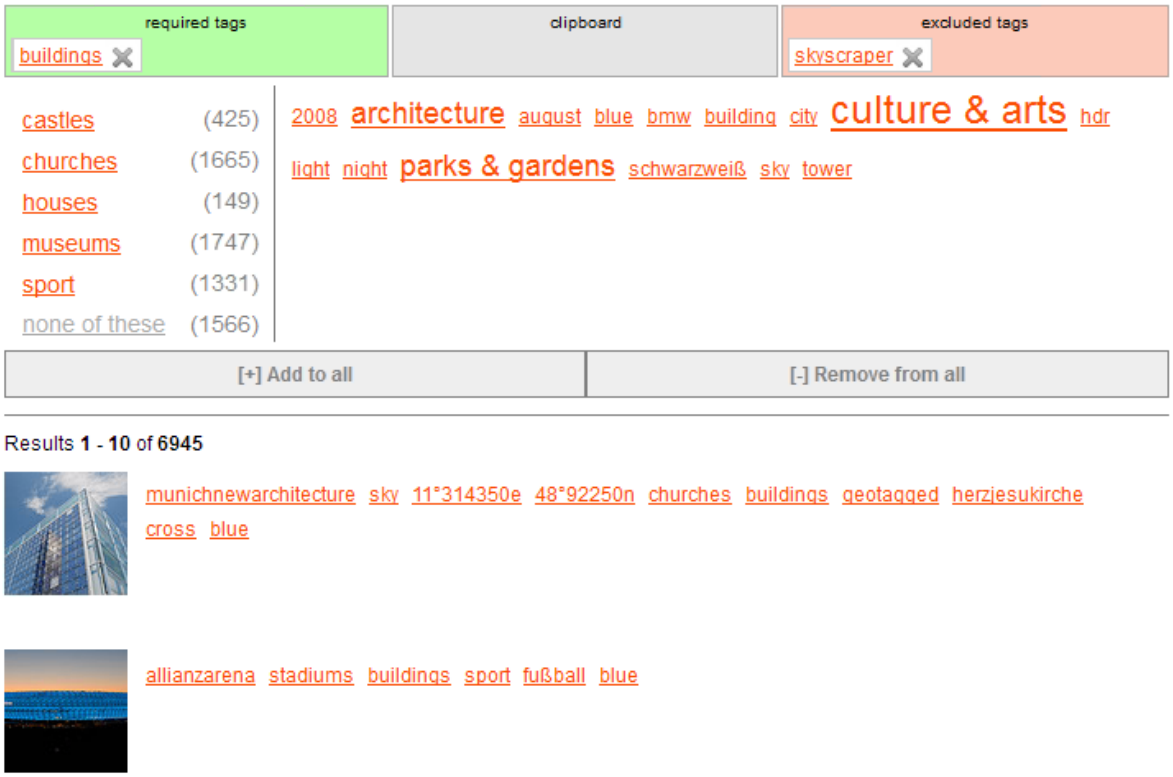


Figure 3.12.: User interface for browsing a set of resources integrated in the search interface of the Tricia platform.

Clicking any tag outside of a container narrows the filter by adding this tag to the green container. A click on a tag within a container removes it.

Below the tag cloud, two gray-colored fields are displayed. They are triggered when a tag is dropped onto them. Alternatively, tags can be manually entered after clicking one of the fields.

- The "Add to all" operation can be used to add a tag to all resources in the current result set. Resources having the tag already assigned are not affected by this operation.
- The "Remove from all" operation removes a tag from all resources in the result set respectively.

In the course of organizing a collection of resources, certain sequences of operations are performed repeatedly. To establish a subsumption between tags x and y so that $x \geq y$, the tag y is added to the filter first. Then x is assigned to all matching resources. It is sometimes practical to drag x into the clipboard container at the beginning of this operation because it might not be shown in the tag cloud after the filter has been narrowed to y. Afterwards y is removed from the green container again.

To establish an equivalence relation between x and y these steps have to be performed twice. The first to assign x everywhere y is assigned, and the second to assert that x does not occur

without y respectively. If only one of the tags shall remain in the system, for instance because one of them contains a spelling mistake, this tag can be removed as soon as it is subsumed by the other one.

Harmonizing tags in a collection that was extracted from a normal folksonomy is a cumbersome task, because many spelling variations and languages can exist. To facilitate this process, we added the possibility to establish a subsumption relation in one step. To accomplish this, a tag can be simply dropped onto another tag, the same way a folder is dropped onto another folder in common file system explorers. This triggers a dialog that asks the user to confirm the operation. The user can also choose to additionally remove the subsumed tag after the operation.

Intentionally creating facets that can be detected by the system is harder, because in general tags cannot be removed from the resources in the intersections of the facet tags: If part of the photos in a collection show cats and there are dog images as well, it would be reasonable to show the tag "cats" and "dogs" in the same facet. However, if there is a sufficiently large fraction of the photos showing cats as well as dogs, the facet is not detected. Sometimes this problem disappears after the collection has been tagged exhaustively with regard to the facet since after all relevant photos have been tagged "cats" or "dogs" the relative size of the intersection of both tags has decreased. But even if this is not the case, it often makes sense to check if one or more of the facet tags should be assigned to resources having neither of the tags assigned yet.

In order to accomplish this, the respective tags can be dragged to the red container one after another to systematically narrow the filter to the resources not being categorized with regard to the facet. If the facet is already detected, this can be accomplished simply by clicking the "none of these" link below the facet.

When categorizing the resulting rest of the resources, it happens often that tags from other facets help to complete the tag assignments quickly. If for example all photos being tagged "churches" shall be additionally tagged with the name of the particular church they show, tags for places or colors can be used to limit the result set to photos of a particular church even if no explicitly church-related tag was assigned yet. If there is for instance a yellow church, people use the tags "yellow" and "church" if they do not know the name. When the photos are organized the tag "yellow" helps to easily assign the correct name to a lot of photos in one step.

Note that all of these operations can be performed in a specific context by adding the respective tags to the filter in advance. For instance, if the user wants to organize only the photos of churches in London, it is sufficient to add the tag "london" to the green container at the beginning and to leave it there during the process.

The user interface control for editing individual resources is opened by clicking the area below the title of a resource where the tags are displayed. This is indicated by a small icon and a change of the background color when the mouse hovers above this area. The control is shown in Figure 3.13.

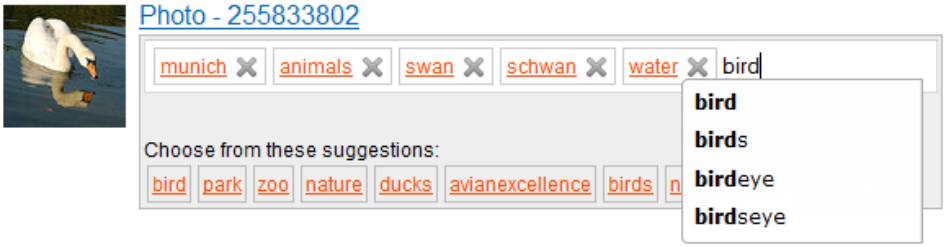


Figure 3.13.: Editing the tags of an individual resource.

3.3.5. Discussion of implicit tag relations

In the following, we provide a structured discussion of the most important aspects and implications of our approach. It is based on theoretical considerations as well as on experiences gained during the work with our prototype implementation. The discussion of these aspects covers the potential of implicit tag relations to improve the accessibility and indexing quality of tagging systems, the limitations of the approach as well as technical implications.

3.3.5.1. Indexing quality

As already pointed out in Section 3.3.1, the most obvious benefit of the application of subsumption relations is that the extension of a tag can be systematically expanded to all resources tagged with subtopics for this tag. This applies to synonyms and equivalence relations respectively. Additionaly, spelling variations can be harmonized as well as plural and singular forms of tags. In effect, the search for the respective tags yields a higher number of relevant resources, so the recall measure is improved.

Particularly for rather general tags the increase in recall is very high. Since people tend to tag on the so-called basic level [Ro78, TT91], general tags such as “buildings” are rare compared to more specific ones for kinds of buildings, e.g., “churches” or “skyscrapers”, or tags for particular buildings, which are even more frequent, when the buildings are well known. For general tags like “animals” we observed increases of several thousand percent in the size of the extension. However, it is hard to generalize such numbers. Where subsumptions shall be actually applied is highly subjective because it depends on the personal interpretation of a tags exact meaning. Does “buildings” mean that a building is visible on a photo or is it sufficient when a photo was taken inside a building or from the top of a high building? Additionally, one can always find tags being rarely used for which even higher increases in the extension size can be obtained. Examples for such tags are technical terms, for instance for animal species, that only a small number of users have used.

In addition to the improvement of parts of existing folksonomies, tag relations can guide the users when tagging new resources. This way, an enduring effect on the indexing quality can be achieved. In the prototype, we implemented automatic tag assignments that are triggered when a tag is added and there exists a synonym or a more general tag that subsumes the new tag. These tags are then automatically assigned.

One can further argue, that grouping tags to facets raises the awareness for the different dimensions of categorization and in effect it motivates users to use more tags (see Section 3.3.2.3). However, only in later versions of our prototype, that went beyond purely implicit tag relations, we actually made the facets visible in the tagging interface.

3.3.5.2. Accessibility

In addition to the aforementioned increase in recall, implicit tag relations offer several opportunities to improve the presentation of navigation options and to summarize a search result with a tag cloud:

- Tags being subsumed by more general tags can be hidden.
- Equivalent tags can be grouped or reduced to one representative tag.
- Mutually exclusive tags can be grouped in facets.

Again, it has to be noted that all tag relations depend on the current context, e.g., the result of the search for a particular tag. While it is usually not true that "irish terrier", "terrier" and "dog" are equivalent in the whole system, this may yet be the case within a particular set of resources. By eliminating two of these tags from the navigation options, the cognitive effort required for analyzing the tag cloud can be reduced. Other frequent tags can be displayed instead which gives a better overall impression of the current context and allows the user to navigate to resources that would otherwise be unreachable via the tag cloud. Finally, tags being assigned to all resources in the current context can be hidden since they are useless for navigation.

In addition to clearer tag clouds, another benefit of the continuous detection of implicit tag relations is that unexpected patterns can be revealed that help the user to understand the contents of the collection. For instance we observed once, that after the tags of a collection of photos taken in the city of Munich were harmonized, the system detected a facet of animal species when restricting the filter to photos tagged "animals". Surprising was, that the system showed the tag "zoo" in this facet in line with other tags for species being frequently encountered in a city, such as dogs or birds. While this is not reasonable from a biological taxonomist's point of view, it represents a useful summary of the animal photos for somebody exploring the collection, at least under the assumption that he knows the photos were taken in a typical European city.

However, the downside of the automatic detection is that it frequently happens that facets are meaningless for most users. We cover this issue in Section 3.3.5.6.

3.3.5.3. Flexibility

Flexibility is probably the most important advantage of implicit tag relations. Tagging of resources is not restricted but only guided by the system. The fact that implicit tag relations always depend on the context allows to create very complex structures that adapt while the user navigates the system.

Implicitly binding relations to a certain context also facilitates the emergence of organization structures in a bottom-up fashion: Users can start to harmonize the usage of tags in manageable subsets of resources (e.g., the resources they are currently working on during a specific task) without considering whether a particular relation is adequate in general. Having established relationships in little, they can be adopted in other contexts or they can be gradually generalized. This way, complex structures can emerge that can hardly be expressed using classical knowledge representations such as monohierarchical taxonomies.

Additionally, implicit relations flexibly adapt when new resources are added to the folksonomy. For example, if an existing tag equivalence relation is not appropriate for a new resource (e.g., the first photo in the collection showing a dog that is not a terrier), the equivalence relation disappears in all contexts containing the new resource.

Since the users are not restricted to navigate along a fixed structure but the structure is derived from the current result set, it is easy to combine the approach with other kinds of filters. Examples for additional filters are access rights, date of last modification, creation date, file size, et cetera.

3.3.5.4. Portability and universal applicability

Since tag relations are not explicitly stored but embodied in the tag assignments, they can theoretically be represented in all systems that allow the assignment of tags to resources. One could say that the tag relations "stick to the resources". This is convenient when data has to be exchanged between tagging systems, or part of a collection is made available to other parties.

However, the fact that all tagging system are capable of expressing implicit relations does not mean that they are suited for managing and exploiting such relations. Technical implications and user interface requirements are covered below.

3.3.5.5. Expressivity

Although quite complex structures can emerge when implicit tag relations are applied, the basic modeling primitives are very restricted. It is for instance not possible to distinguish part-whole relations from subtype relations since both are expressed as subsumption. This can be compared to the nesting of folders in classical file systems where the semantics of a folder-subfolder relation can be anything that makes sense for the person creating the folder structure. As a consequence, interoperability with more explicit knowledge representations is very limited. Other approaches that map tags for example to semantic web ontologies, can profit from the fact that there is a bindingly defined meaning of a tag across platform borders.

The expression of facets through pairwise mutual exclusion of tags is limited to single-valued facets. If for example a set of publications is tagged with the names of the authors, it cannot be recognized that these tags belong to the same facet the extensions of the tags are generally not disjoint. Distinguishing this from the partial overlap that naturally occurs among most of the frequent tags in a folksonomy is not possible.

3.3.5.6. Detection accuracy

One of the major challenges resulting from the lack of explicit relations is that incidental, meaningless cooccurrences of tags have to be distinguished from such relations that have been consciously established. While there is no difference from a formal point of view, it is obvious that in a set of five resources with many tags there are several random subsumption, equivalence and mutual exclusion relations. In contrast, it is very unlikely that an equivalence of two tags being assigned to the same 10,000 resources occurs by pure chance.

An apparent solution for this problem is to define a threshold for the frequency of the respective tags within a context below which implicit tag relations are ignored by the system. This reduces the probability of *false positives* but on the other hand introduces the problem of *false negatives*, i.e., relations that were intentionally established but are not recognized. However, one can argue that this is tolerable as long as thresholds are low enough. For instance in a set of ten resources, tag relations are of limited value anyway.

One can attempt to find appropriate thresholds by considering the problem from a theoretical perspective. We therefore assume that two semantically unrelated tags x and y are randomly distributed in a set of n resources. Let further $P(x)$ and $P(y)$ denote the probabilities that the respective tags are assigned to a randomly chosen resource. We simply assume that these probabilities are reflected by the relative tag frequencies in the current context.

For the probability of an equivalence of the tags in a result set of size n we write $P_n(x \equiv y)$ and obtain

$$P_n(x \equiv y) = (P(x)P(y) + (1 - P(x))(1 - P(y)))^n.$$

If we choose $P(x) = P(y) = 0.5$ and $n = 10$, this probability is slightly smaller than 0.001. If we change $P(x)$ or $P(y)$, an implicit equivalence relation becomes even less likely.

For the probability that x subsumes y we obtain

$$P_n(x \geq y) = (1 - P(y)(1 - P(x)))^n.$$

Apparently, in the worst case $P(x)$ is very high and $P(y)$ is low. Let $P(x) = 0.95$ and $P(y) = 0.05$ respectively[6]. To obtain a similarly low probability for a false positive as for the equivalence relation (i.e., $P_n(x \geq y) < 0.001$), under these circumstances the number of resources n is required to be at least 2760. For the more common case that $P(x) = 0.2$ and $P(y) = 0.1$ we still require more than 82 resources to reduce the probability of a false positive respectively.

We covered the detection of facets in Section 3.3.3 in more detail but at least a basic impression of the underlying probabilities shall be given here. The probability that x and y are mutually exclusive in a set of n resources is

$$P_n(x \sim y) = (1 - P(x)P(y))^n.$$

Thus, for $P(x) = P(y) = 0.1$ we obtain for example $P_n(x \sim y) = 0.99^n$. It is obvious that it is rather a common case than an exception that two tags are mutually exclusive if their

[6] It has to be noted that this is a rather uncommon scenario because it means we are considering a context in which x is assigned to 95% of the resources.

relative frequencies are not very high. However, an exhaustive facet with two tags — i.e., each resource is either tagged x or y — can be detected more reliably. For $P(x) = 0.5$ and $P(y) = 0.5$ the probability equals the probability of the equivalence of x and y (which is about 0.001 for $n = 10$).

Although the numbers for equivalence and for exhaustive facets look practical and one could argue about whether a fault rate of 0.001 is actually required, there are two more problems in practice:

- The probabilities above refer to a single pair of tags. Since the number of such pairs depends on the number of distinct tags in the result set, the probability that there is at least one false positive among the detected relations is much higher. In fact it is near one for realistic scenarios.
- The assumption that tag probabilities are independent is wrong in many cases. Even if tags belong to different facets, there can be correlations.

In summary, it can be said that implicit relations can be established in large and diverse collections and promising results could be obtained with our prototype implementation. However, for the application in small collections, the detection accuracy is not sufficient.

3.3.5.7. Scalability and technical implications

The dynamic detection of implicit tag relations for each result set is technically challenging. Not only is it necessary to count the individual tag frequencies but it is required to determine the frequencies of pairs of tags, i.e., how often each tag occurs in combination with each other tag.

Furthermore, the consequent application of implicit tag relations entails a massive increase in the total number of tag assignments. Especially the fact that synonyms are stored redundantly for each resource leads to a multiplication of the average number of tags per resource. As it can be observed on the book cataloging website *librarything.com*[7] which allows the explicit definition of synonym rings, users make extensive use of this feature when it is directly supported.

In our prototype, we used the information retrieval software library Lucene[8] to determine the resources matching a given filter and to count the tags and tag pairs in a result set (see also Section 5.1.2). During our experiments with implicit tag relations, we worked with collections of up to 30,000 resources (mostly photos obtained from Flickr) and worked on commodity hardware, such as notebooks with 2GHz processors and 4GB of RAM. The time required to analyze a query result increased up to several seconds when more tags were assigned in the course of tag harmonization. A profiler revealed that the critical step is to determine the frequencies of tag cooccurrences. We thus limited the cooccurrence detection to the 100 most frequent tags in the result set, so the set had to be analyzed once for counting the tag frequencies and then again for the cooccurrences of the frequent tags. Less frequent tags were not interesting because they would not have been shown in the tag cloud anyway. We do not give exact performance measures here, because they are rather meaningless given that

[7] http://librarything.com, (accessed 6th of February 2012).
[8] http://lucene.apache.org, (accessed 13th of February 2012).

our implementation was not particularly tuned for low response times. However, we can state here, that tag cooccurrence analysis entails a significant additional computational effort compared to only retrieving matching resources and counting tag frequencies.

To establish implicit tag relations, tags have to be assigned to or removed from a potentially large number of resources. This turned out to be even more challenging than the analysis of a resource set since the changes have to be persisted on the hard drive. Since this was still relevant for later versions of our prototype, we describe our solution in Section 5.3.4.

3.3.5.8. Usability and Complexity

Developing a user interface that is understood by users turned out to be very hard for different reasons. First, the implicit nature of tag relations is hard to communicate to the users. Even if a relation can be successfully established, the result may not be instantly visible. For the same reason, relations are also destroyed by accident. Together with the fact that unintended relations occur frequently, e.g., meaningless facets, this can give the user the feeling of not being in control of the system at all.

While subsumptions can be easily established with one drag and drop operation in a tag cloud, creating facets is much more difficult and requires deep knowledge of how the system works.

Another severe problem is the overall complexity of the emerging structures. Since tag relations always apply in the context of a certain filter, there is a whole set of tag relations for each such context. The different combinations of tags in the system lead to so many different contexts that the precise effects of a single tag assignment or removal are very hard to foresee. Even if they were explicitly listed, and the user could check the effects, it would not be clear how to proceed if the effects were not what the user expected.

The problem of complexity becomes even more severe when the tag assignments of several resources are changed at once. The resulting changes in the implicit tag relations are even challenging to determine programmatically and we did not implement automatic tag assignments for this case. This means that existing relations were not preserved but could be unintentionally voided this way.

3.3.6. Conclusion

On the one hand, our approach of implicit tag relations has some very promising characteristics, such as the flexibility, the portability and the opportunity to generate clearer tag clouds. We could also produce some promising examples that demonstrate its strengths, as it can be seen in the figures above. On the other hand, there are two critical unsolved problems:

- Implicit relations are too fragile and cannot be detected reliably in small result sets.
- The overall complexity of the implicit organization structures is too high to be easily understood by users.

In summary, it has to be stated, that our goals could not be met with pure implicit tag relations. However, the experiences gained during our work with the prototype system and during

its development had a major impact on future versions of the system. We tried to preserve the positive aspects and adapt the approach to remedy the problems where necessary.

CHAPTER 4

The TACKO system

This chapter covers the final version of the TACKO system. In Section 4.1, the explicit TACKO organization structure is formally described and illustrated with examples. In Section 4.2 the characteristics of the structure are highlighted by comparing it to tree structures and faceted organization. Subsequently, the user interface is presented in Section 4.3. The section includes a detailed description of all user interface controls and the operations users can perform with it.

We have presented TACKO to some extent in [MNS12a].

4.1. The explicit TACKO organization structure

Based on the experiences made with implicit tag relations, it was decided to radically change the approach with regard to one of the central aspects: The attempt to keep all tag relations implicit was abandoned. Instead, different types of explicit relations can be defined. However, these relations are in fact explicit versions of their implicit counterparts. It was attempted to preserve the flexibility of implicit relations, while mitigating their shortcomings.

By making relations explicit, the two main problems of implicit relations are addressed:

- Implicit structures cannot be detected reliably in small collections of resources.
- Users find it hard to understand implicit tag relations and the complex structures resulting from their application. In consequence, it is difficult for them to exploit their full potential.

Not only to reduce complexity for the users, we decided further simplifications. Turning our implementation, that previously focused on the dynamic detection of tag relations, into a system capable of managing explicit relations involved considerable effort. To maintain flexi-

bility and to facilitate further modifications and experiments, we restricted the functionality to a minimum. As a result, it was decided to provide no support for equivalence relations. When different tags with the same meaning are used in the system, users have to unify them by deciding which tag to keep and by deleting the other variations. Although there are some good reasons for equivalent tags, such as the convenience of abbreviations or support of different languages (cf. Section 3.3.2.2), we also saw drawbacks connected to their implementation. Different variations of a tag can lead to confusion of the users. Furthermore, it introduces additional complexity into the user interface which we tried to avoid as much as possible. Finally, with regard to the topology of the organization structure, abandoning equivalence relations is not a severe limitation.

Thus there are only two kinds of explicit tag relations: subsumption and facets. Explicit facets now represent what was previously called "mutual exclusion" and could be interpreted as a facet. Both types of relations are covered in the following sections.

4.1.1. Subsumption

An implicit subsumption relation between two tags is characterized by the fact that the extension of one tag is a superset of the extension of the other one. This is still a requirement for an explicit subsumption relation but it is not sufficient. The user has to additionally establish an explicit link between both tags. This way, it is avoided that random patterns in the tag assignment are mistakenly interpreted as subsumptions. Thus for subsumption of a tag t_2 by another tag t_1 we write $t_1 \geq t_2$ and it always holds that

$$t_1 \geq t_2 \quad \Longrightarrow \quad extension(\{t_1\}) \supseteq extension(\{t_2\})$$

Note that a sumpsumption relation is now defined independent of any filter. Much of the complexity of implicit tag relations resulted from the fact that subsumption relations were context-dependent. They could disappear when the user widened the filter by removing a tag and they could turn into equivalence of two tags when the filter was narrowed. Furthermore, the management of different sets of subsumption relations for different contexts is very complex — from the implementation perspective and as well from a usability point of view.

As a consequence it was decided to make the scope of subsumptions global, i.e., there is only one set of subsumptions that applies to the whole collection.

4.1.2. Facets

In TACKO, facets are simply groups of tags. Usually those tags are related in so far that they are concerned with the same aspect of categorization. One could for example define a facet for tags specifying a location and another one for time-related tags. However, from a formal point of view, a facet is just an arbitrary set of tags.

In contrast to subsumption relations, facets are defined within a certain context. While context-dependent subsumptions introduced unmanageable complexity, it turned out that the possibility to define a facet only for a certain part of the collection is advantageous. So a

facet applies within a subset of the collection that corresponds to a certain filter $F \in \mathcal{P}(\hat{T})$. For a set of n tags $\{t_1, t_2, ..., t_n\} \subseteq T$ that represents a facet in the context of F we write

$$F \vdash \text{Facet}(\{t_1, t_2, ..., t_n\}).$$

Within a single context, several different facets can be defined.

Facets have no implications on the tag assignments. In particular, it is not required that the tags of a facet are mutually exclusive in the context of F. This is an important difference to implicit facets based on mutual exclusion. Since the facets are explicit and do not have to be detected automatically, an overlap of tags is no problem. This makes it possible to create multi-valued facets, i.e., facets of which usually several options apply to the same resource as in the example of the "authors" facet for publications: When organizing resources representing publications, it is now possible to create a facet containing tags representing the names of authors even though these tags are frequently used in combination. Or in other words the extensions of these tags can overlap and do not have to be disjoint.

Since the facets are independent of the tag assignments, there are no restrictions with regard to the membership of tags in facets. The same tag can be member of several facets. It is even possible that a tag is contained in two different facets applying in the same context. A tag representing a famous building in a city can for instance be part of a facet for places of interest and another one for buildings.

In an earlier intermediate version of the TACKO organization structure, the facets defined in one context were entirely independent of the facets in other contexts. This means that after each navigation step all the facets could change and each facet was only visible in one particular context. Our aim was to maximize the flexibility of the organization structure. For each view on the collection of resources, i.e., for any filter, the set of available navigation options could be individually configured. Assume that the following facets apply:

$$\{\text{“buildings”}\} \vdash \text{Facet}(\{\text{“churches”}, \text{“skyscrapers”}, \text{“stadiums”}, \text{“towers”}\})$$

$$\{\text{“buildings”}, \text{“sports”}\} \vdash \text{Facet}(\{\text{“diving platforms”}, \text{“stadiums”}\})$$

In the earlier TACKO structure, these facets only applied in the respective contexts. Thus a facet could be copied to another related context and tailored to better suit the respective domain as in the example of buildings and sports related buildings. When navigating from {“buildings”} to {“buildings”, “sports”}, it would look like the facet adapted to the new context.

While it first appeared that navigation could be facilitated this way, it turned out soon that managing distinct sets of facets for all contexts was not practical. There are simply too many possibilities to combine tags to filters and whenever it happened that a user combined filters in a new way, the set of facets for the respective context was always empty. As a consequence, the system was changed to show not only the facets of the current context but also all facets defined in more general (less specific) contexts. Figure 4.1 illustrates how filters are related by the subset relationship. The arrows point from general to more specific filters.

This change made it possible to define basic facets in the root context, which corresponds to the empty filter {}, but these facets are available everywhere in the collection. A typical

example of such a facet is

$$\{\} \vdash \text{Facet}(\{\text{“color”}, \text{“grayscale”}\})$$

in a collection of photos. The general rule is that facets are defined once in a certain context and then apply to all contexts that are more specific:

$$F' \supset F \quad \wedge \quad F \vdash \text{Facet}(\{t_1, t_2, ..., t_n\}) \quad \Longrightarrow \quad F' \vdash \text{Facet}(\{t_1, t_2, ..., t_n\})$$

We call this rule *facet inheritance*. The most general context in which a facet applies is called the facet context. Note that $F' \supset F$ means that F' is more specific than F because it contains more tags.

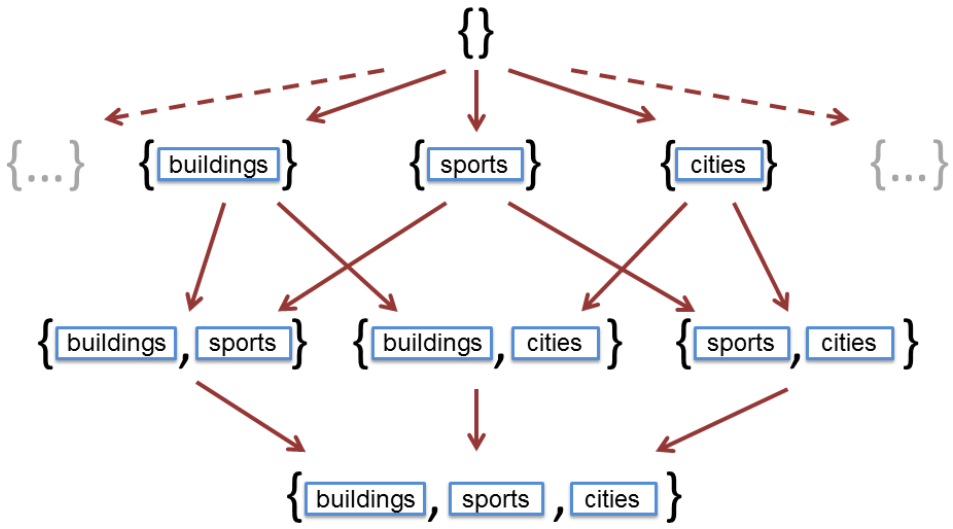


Figure 4.1.: Small excerpt of the lattice structure induced by the $\subset$ relation on filters illustrated with example tags.

This also means that facets can be independently used for filtering the collection. Selecting a tag from one facet makes the filter more specific, but the other facets are not affected because they are still inherited by the more specific context. So they are still available for narrowing the filter further. However, additional more specific facets can apply in narrower contexts to provide more specific navigation options. How facets are integrated in the user interface is covered in more detail in Section 4.3.

We also experimented with more complex solutions allowing the users to adapt an inherited facet to a specific context. For instance when a tag was added to a facet in a context in which it was not originally defined, a modified copy of the facet was created, that overrides the original facet. This worked better than having individual facets for all contexts, but it still led to confusion.

Finally, it has to be noted, that facets cannot be defined in the context of a filter containing negative tags. Nevertheless facets can apply in such a context according to the inheritance rule described above. The facets applying in the context of a filter $F \subseteq \hat{T}$ are equivalent to

those applying in the context of a filter F' being obtained by removing all the negative tags from it: $F' = F \setminus \bar{T}$.

There are basically two considerations that led to the decision not to allow negative tags in facet contexts. First, determining the facets applying in the context of a given filter would involve significantly more computational effort than otherwise. For example if only the tag t_1 is in the filter, the only relevant facet contexts are $\{t_1\}$ and $\{\}$. If negative tags were allowed in facet contexts, not only could facets with contexts such as $\{\bar{t_2}\}$ be relevant but also combinations of negative tags would have to be considered. To solve this problem, one could define that a facet defined for a filter containing negative tags only applies if the current filter also contains these negative tags. This means that in the context $\{t_1\}$ a facet defined in $\{\bar{t_2}\}$ is not visible, but it is in the context $\{t_1, \bar{t_2}\}$. In other words, the user has to explicitly exclude certain tags to see the facet. This leads to the second consideration: Adding negative tags to the filter is by far not as common as selecting a positive tag. Users would rarely encounter the facets in contexts involving negative tags and thus it would be hard if not infeasible to systematically explore the complete navigation structure.

4.1.3. Combining subsumption and facets

Subsumption relations and facets are complementary constructs that can be combined to complex structures. Yet, they are not entirely independent. It is often reasonable to apply facets in combination with subsumption relationships. For instance when constructing simple concept hierarchies such as in a classic taxonomy. Figure 4.2 shows an example of such a hierarchy. Orange arrows stand for subsumptions and dashed boxes represent facets. A tag connected to a facet with a dashed line represents the facet's context. Since a facet context is in fact a set of tags, the actual context additionally comprises the tags subsuming the connected tag.

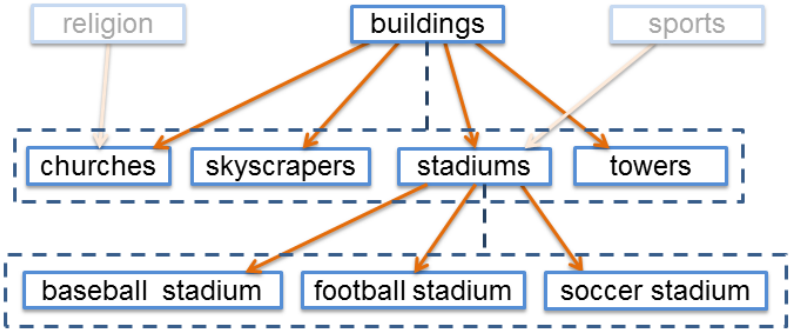


Figure 4.2.: Example hierarchy constructed using subsumption relations and facets.

In the example depicted in Figure 4.2, the definition of the facets seems redundant because the tags in the facets are subsumed by all context tags. Additionally, there are no other tags being subsumed by the context tag but not being part of the facet. However, this is not always the case and it is important to distinguish both, subsumptions and facets as the example in Figure 4.3 illustrates.

Here, a set of specific tags being all subsumed by the same general tag is partitioned into sev-

eral distinct facets that categorize resources with regard to different aspects. This distinction could not have been made with subsumption relations alone. The example also illustrates well that facets can be used to make much more subtle distinctions than that between time- or location-related tags. It also shows that facets why facets are note defined globally but only in a certain context.

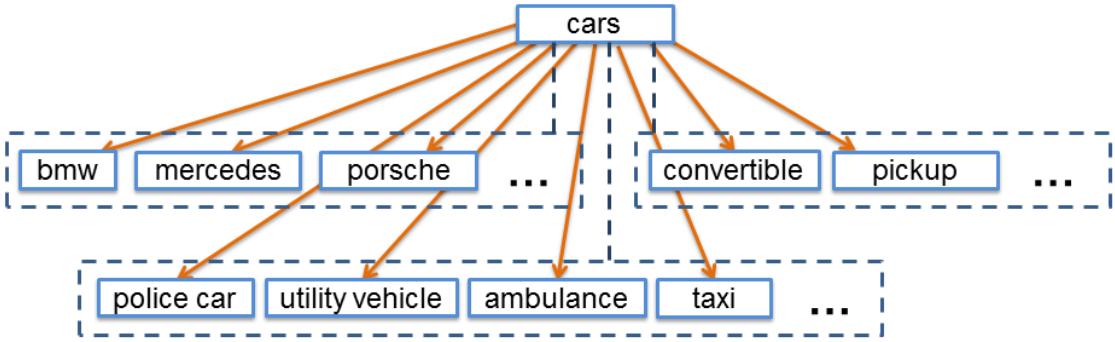


Figure 4.3.: Different facets defined in the same context.

It is not required, that all tags of a facet are subsumed by the same tag although this is often the case. There are for instance the most general tags which are not subsumed by any other tag. On the other hand, a facet can include any tag of the organization structure. Figure 4.4 illustrates both cases. The tags in the facet at the top are not subsumed by any other tags. More interesting is the facet at the bottom right:

$$\{\text{“projects”}\} \vdash \text{Facet}(\{\text{“invoices”}, \text{“project plans”}, \text{“closing reports”}\})$$

Only one of the facet tags subsumed by the context tag. The others are subsumed by completely independent tags. The tag “invoices” is for instance subsumed by “finance” (and also part of another facet defined in the context {“finance”}), but not by “projects”. Otherwise the assignment of the tag “invoices” to a resource would imply the tag “projects” as well. We call a facet in which all tags are subsumed by the facet context tags a *subsuming facet*.

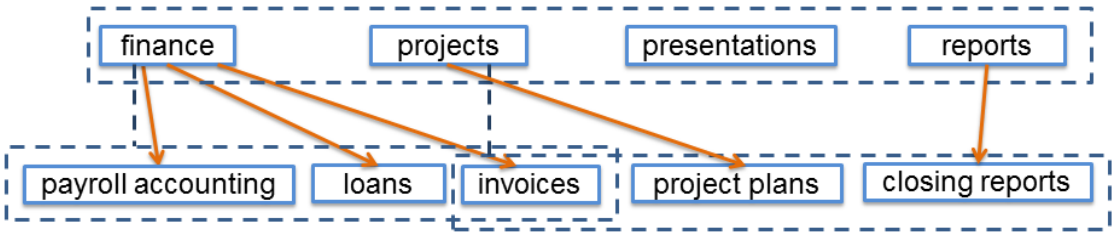


Figure 4.4.: Tags in a facet do not have to be subsumed by the facet context tags.

Finally, another useful combination of facets and subsumption relationships is the intersection of two or more subsuming facets in one or more tags. An example is shown in Figure 4.5. This way it is on the one hand ensured, that all the resources tagged “stadium” can be reached by different navigation paths and that such resources are tagged “sports” and “buildings” as well.

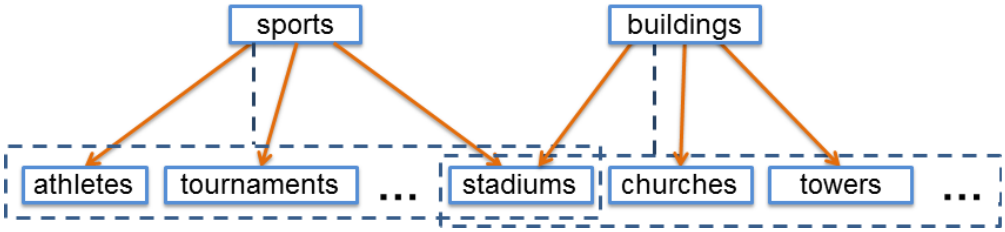


Figure 4.5.: Intersecting facets with subsumption relationships.

4.2. Comparison with other knowledge organization structures

To better illustrate the characteristics of the TACKO organization structure, we directly compare it to tree structures and common faceted organization structures. On the one hand, these structures are closely related since TACKO incorporates elements of both. On the other hand, they are widely applied and the reader is most likely familiar with both them.

In this section, we only compare TACKO to very general alternative structures. More specific solutions are covered in Chapter 7.

4.2.1. Tree structures

At first sight it seems that TACKO supersedes tree structures since they can be modeled with subsumption relationships. However, this is not entirely true: The labeling of the tree is restricted in TACKO. Since the nodes are represented by tags, each label may only be used once. Common directory structures may contain several directories with the same name — at least as long as the directories do not have the same parent directory. On the contrary, a tag in TACKO is unique with respect to its name and the relations to other tags.

However, as long as the directory names refer to the same concept this should not be considered a restriction but rather an advantage of TACKO since the user retrieves all relevant documents when searching for the tag. In a tree of directories, these documents can be distributed among several places. In the case of homonyms, i.e., if the same name refers to different concepts, one would have to tag more precisely in TACKO. While terms can be disambiguated by the path in a directory structure, the disambiguation has to be included in the precise name of the tag: “apple (company)” vs. “apple (fruit)”.

On the other hand, TACKO is more powerful than tree structures in many respects. Intuitively, when compared to tree structures, the advantages of TACKO are similar to those of faceted organization structures: TACKO provides several access paths to resources, the set of resources can be effectively narrowed down to a small set in few steps, and the structure can be extended more flexibly (cf. Section 2.2.2). However, TACKO is not equivalent to common faceted structures and the differences are covered precisely in Section 4.2.2.

The differences to tree structures are perhaps best illustrated at an example. How a concrete TACKO structure can be transformed into a tree and vice versa was examined by Michel in [Mi12], a Bachelor’s thesis advised by the author of this thesis. An exemplary mapping

of a small organization structure is shown in Figure 4.6. In the following, we will briefly describe the respective transformations. Although we make some simplifications for the sake of brevity, the central ideas are the same as in the algorithms described in [Mi12].

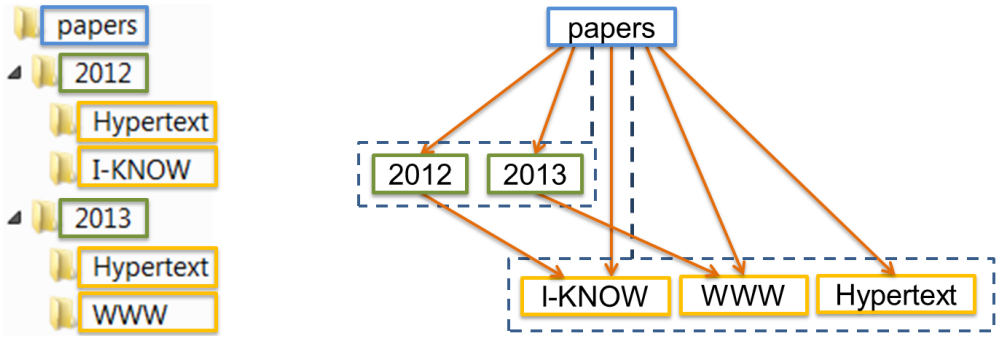


Figure 4.6.: Simple example of a transformation of a directory structure into a TACKO structure.

Mapping the resources in TACKO to a directory structure is rather simple if we assume that the structure is coherent and all tags are assigned to a facet. For each resource r in TACKO, the following steps are performed in a virtual user interface:

1. Start with the empty filter, and an empty sequence of tags, the new path p.
2. In the facets displayed in the navigation bar, select the first tag that is assigned to resource r (if there is such a tag) and append it to path p.
3. If the displayed facets still contain a tag assigned to r (the facets changed after the last selection), go back to step 2, else proceed with step 4.
4. Store resource r in the directory structure so that it has path p. Extend the directory structure if necessary.

Apparently, if a resource is tagged with several tags of the same facet, this will not be reflected in the resulting directory structure. This could be solved by cross-links or by storing copies of the resource in different locations.

Transforming a directory structure to a TACKO structure works as follows:

1. Add each resource in the directory tree to TACKO and tag it with all the path segments it had in the tree. Thus, a resource stored in path "A/B/C" will receive tags "A", "B", and "C".
2. For all pairs of tags t_A and t_B, add a subsumption relation $t_A \geq t_B$ if there is at least one path in which the respective directory "B" is a subdirectory of "A" (not necessarily a direct subdirectory) and if there is no contradictory path, i.e., a path in which "A" is a subdirectory of "B".
3. Form facets by grouping tags for which the respective directories have a sibling relation in the directory tree. Choose the context of the facets as specific as possible while asserting that all facet tags are subsumed by all tags in its context.

Figure 4.6 shows that several sets of sibling-directories can be merged into one facet in the

last step: “WWW” and “I-KNOW” are no sibling directories, but they have a common sibling “Hypertext” in different locations of the tree.

Further, it can be seen that there are subsumption relationships between the years and the conference names. They result from step 2. If all conference names appeared as directories in all of the year directories, no such relationships would exist.

4.2.2. Faceted structures

First of all, the facets in TACKO are simpler than facets in other structures insofar as they are flat and not hierarchical. In TACKO, a facet is just a list of tags that is presented in a certain way in the user interface. Traditionally, facets are rather seen as independent taxonomies that cover distinct aspects of categorization. Nevertheless, TACKO facets can be used to mimic such structures when they are combined with subsumption relationships. In Section 4.1.3, it has been illustrated how a classical hierarchical taxonomy is represented in terms of facets and subsumption relationships (cf. Figure 4.2).

In consequence, a classical faceted organization structure can be emulated with TACKO by combining several hierarchical taxonomies built of TACKO facets and subsumptions. Optionally the roots of the individual taxonomies can be combined in an additional facet in the root context.

A classical faceted structure can be considered a large hierarchy that branches into the different facets at the top level. The characteristic nature of faceted organization rather lies in the way this structure is applied in the sense that resources are related to concepts of all major branches of this hierarchy. Insofar, it can be said that TACKO supersedes classical faceted organization structures if the limitation that two concepts may not have the same name (cf. Section 4.2.1) is ignored.

Apart from the fact, that unconventional structures can be built by combining facets and subsumptions in different ways (see Section 4.1.3), TACKO has three important properties in addition to common faceted organization structures, even when only subsuming facets are used, i.e., facets in which all facet tags are subsumed by the tags of the facet context:

1. TACKO facilitates the bottom-up emergence of the organization structure by allowing to annotate resources with arbitrary tags which can be included in the structure later.
2. The number of facets is not fixed as in most faceted organization structures, but new facets can be introduced in specific contexts. However, it has to be noted, that this also applies to organization structures used in large product catalogs, such as that of Amazon[1] or Ebay[2]. Amazon shows for instance a “metal type” facet in the jewelry category or a “phone compatibility” facet for cell phone accessories.
3. Finally, there is a fundamental difference regarding the topology of the structure: While faceted organization structures can provide several access paths to the contained resources, in TACKO this also applies to the categories themselves. This follows from the fact that a tag can be subsumed by several other tags.

[1] http://www.amazon.com (accessed 14th of February 2013).
[2] http://www.ebay.com (accessed 14th of February 2013).

Figure 4.5 illustrates the last point. It shows the example of the tag “stadiums” that can be reached via the tag “buildings” or via “sports”. The reason is that unlike classical taxonomies the structure that results from TACKO subsumption relationships is a directed acyclic graph but not necessarily a tree.

4.3. User interface

This section covers all relevant aspects of an integrated web-based user interface suitable for searching and navigating a collection of resources organized with the TACKO structure as well as for manipulating the structure itself. After describing and motivating the main layout, the different user interface elements and operations for browsing, tagging and organizing are explained.

Although our prototype was again built as part of the Tricia platform, all elements described in this section were particularly designed to suit the TACKO structure and can be used without interacting with other parts of Tricia. The integration of TACKO with other Tricia functionality is covered in Section 5.2.

4.3.1. Basic screen layout

The development of the user interface for the explicit TACKO structure started with the prototype built for implicit tag relations. The central element in this version was the tag cloud above the search result list (see for instance Figure 3.12). Although this tag cloud could contain facets, meaningful facets were rather rare because they had to be automatically detected. This changed with explicit facets. During the work with the prototype it became apparent, that facets became more useful and important for navigation than tags not belonging to any facet. As the number of facets in the left part of the tag cloud increased, the old layout turned out to be impractical. Facets with many tags could gain a considerable height and in consequence the result list was shifted down.

As a consequence of the increased importance of facets, it was decided to change the basic layout to rather resemble a classic faceted search interface [St09b]. Figure 4.7 depicts the four primary screen regions:

- **Filter.** This region contains a list of all tags in the filter in the order they have been selected. However, the order does not influence the search result. Negative tags can be contained as well. It is also possible to add tags to the filter here, thus the filter region can be considered a combination of search input and breadcrumbs.
- **Navigation bar.** This region contains most of the navigation options, i.e., tags that can be added to the filter. The facets are displayed at the top and further tags are listed at the bottom.
- **Toolbar.** The toolbar shows the size of the current result set. It also contains user interface elements for changing the tags of the resources in this set.
- **Result list.** Here, the set of resources matching the current filter is listed. Although

the resources have no particular order, the order is deterministic and stable in the sense that the order of any two resources does not change after a navigation step given that both are contained in the result set after the navigation step. For this reason, we speak of the result list when referring to the representation of the result set. The user can scroll through the entire list using the mouse wheel or the scrollbar. For each resource, all respective tags are displayed.

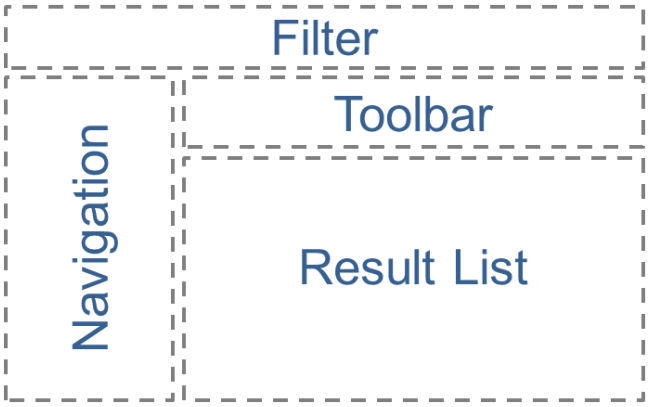


Figure 4.7.: The four major screen regions of the TACKO user interface.

We also considered the option to integrate the path into the toolbar in order to obtain a better screen utilization when the filter contains only few tags. This way, the navigation bar would reach up to the top of the screen. However, the usability study, that is covered later in Section 6.1, confirmed, that the path above the navigation bar helped users to understand the interface [Bo12]. This way, it is obvious that the facets also depend on the filter and thus they can change after each navigation step.

Figure 4.8 shows a screenshot of the user interface. The filter is set to {“buildings”, “churches”} and the result set contains the photos matching this filter. The navigation bar contains a facet, a facet stub and some related tags. The facet is defined in the context “churches”[3]. The facet stub can be used to create another facet in the context “churches”. The user could for instance add tags like “church tower”, “church organ” et cetera.

How this interface is used for navigation, tagging and organizing is described in the following sections.

4.3.2. Search and navigation

As discussed in Section 3.2.3, both, search and navigation steps, can be generalized to a modification of the filter. The only difference is that a navigation step is triggered by the selection of a particular navigation option that has been offered to the users whereas an arbitrary tag can be added to the filter with a search. There are various ways to modify the filter. We start with an overview of how the filter can be narrowed by adding positive

[3]It is actually defined in the context {“buildings”, “churches”} but we will sometimes only refer to the most specific tag for the sake of brevity.

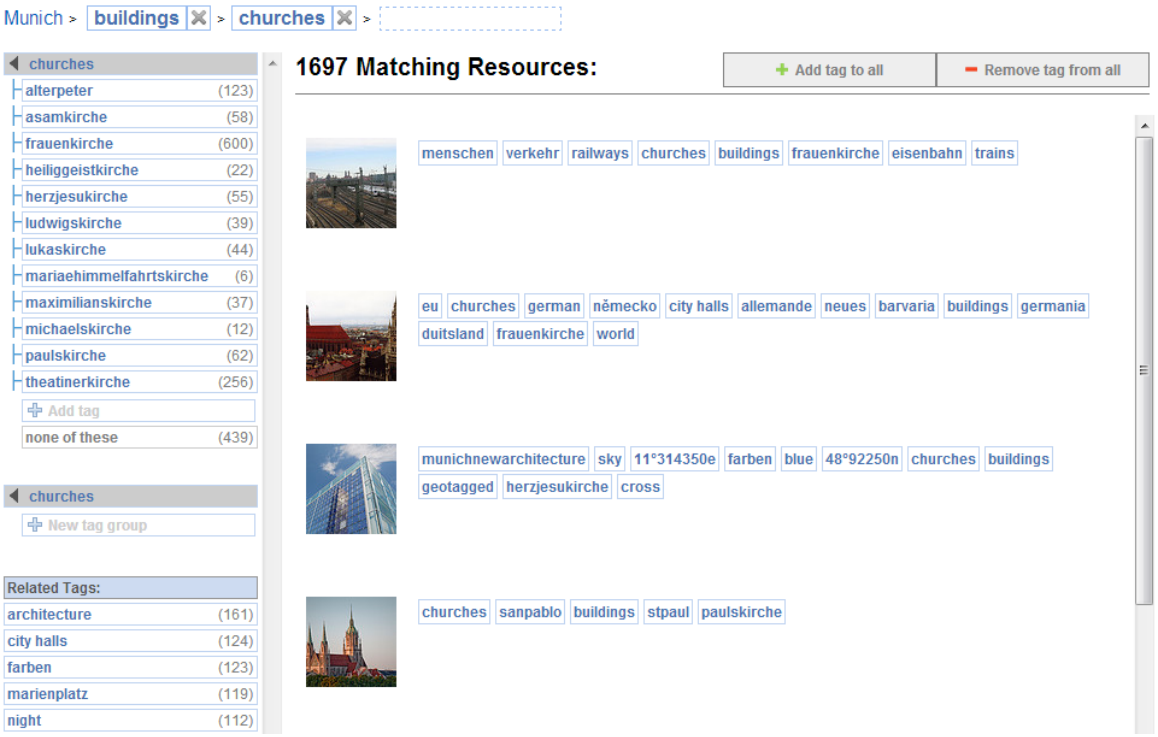


Figure 4.8.: Screenshot of the complete user interface.

or negative tags. Subsequently, we illustrate how tags can be removed from the filter again. Finally, it is described how the user interface responds to changes of the filter.

4.3.2.1. Narrowing the filter

There are three fundamentally different ways to make the filter more specific: Typing in an arbitrary tag — we call this *tag search* —, selecting a tag from the navigation options or a resource and adding all the tags of a single facet as negative tags.

Tag search. A search box is displayed after the last tag in the filter (see Figure 4.9)). Any tag can be appended to the filter by typing it into this box. To make this more convenient and to avoid that users add tags that do not exist in the system, a list of suggestions is shown while the user is typing. This is in the following referred to as the *autocomplete* feature. The suggestions list contains tags that either start with the letters the user has typed in so far or contain a word starting with these letters (tags may contain spaces). The order in which the tags are displayed is random and does not depend on the tag frequency. The suggestions are further based on all the tags in the system and are not restricted to tags that yield non-empty search results in the current context. Finally, it is possible to add a tag to the filter even if it is not shown in the autocomplete suggestions.

Tag selection. Users can select tags either from the navigation bar or from the tags of resources. All these tags are appended to the filter with a single mouse click. Alternatively, tags can be dragged to the end of the filter with the mouse.

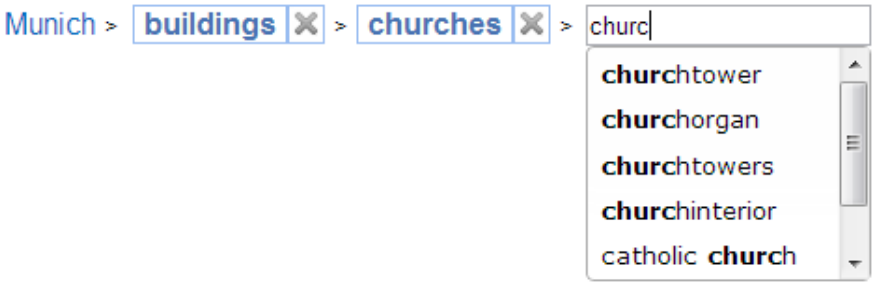


Figure 4.9.: Adding arbitrary tags to the filter.

In the navigation bar, there are two kinds of tags: The tags in the facets and the so-called "related tags". For both kinds of tags it is displayed how many resources in the current context have the tag assigned (see Figure 4.10). In other words, the expected size of the result list after the tag selection is shown. Facet tags that would lead to an empty search result are distinguished by a lighter color.

The "related tags" section lists tags that are assigned to many resources in the current result set but that are not included in any of the currently displayed facets. More specifically, it lists the most general such tags, i.e., specific tags being subsumed by other tags not contained in the current filter are hidden. Assume that there is a facet containing the tags "buildings" and "people". When the user selects the tag "people" the facet disappears (see Section 4.3.2.3) but in the new results set there may be many resources being tagged "buildings", too. Therefore, the "buildings" tag is listed in the related tags section. More specific tags for buildings are not shown.

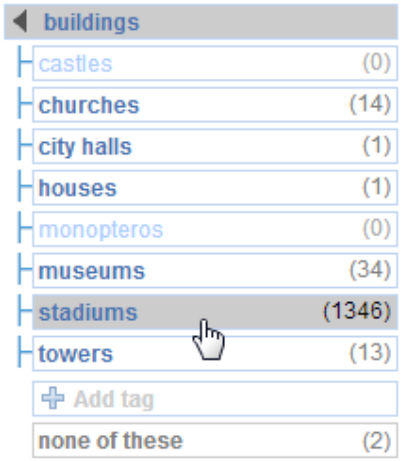


Figure 4.10.: Selecting tags from a facet.

Selecting negative tags. At the bottom of each facet, an additional option named "none of these" is shown (see Figure 4.10). It allows the user to add all the tags in the respective facet to the filter as negative tags. This way, it is possible to restrict the result set to those resources not having any of the facet's tags assigned. In the example of Figure 4.10, the user

would retrieve only the two resources that are tagged “buildings” but for which no specific type of building is specified. It is not possible to add an arbitrary negative tag to the filter. For this purpose a new facet containing only the respective tag would have to be created. Then the “none of these” option could be selected. Figure 4.12 shows the user interface after the selection of a “none of these” option.

4.3.2.2. Widening the filter

The user has several options to remove tags from the filter. The most straight-forward way is to click the “X” that is displayed at the right border of each filter tag (see Figure 4.11). Additionally, it is possible to press the *backspace* key when the input field at the end of the filter has the focus. This way, the respective last tag in the filter is removed. To remove all tags from the filter, the user can click the name of the collection that serves as a kind of “home link”. In Figure 4.11 this is the name “Munich”. It can be considered the root element in the filter and it cannot be removed.

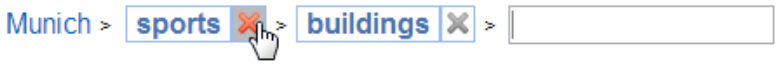


Figure 4.11.: Removing tags from the filter.

The user can also widen the filter by clicking on the headers of facets. Each facet header shows the name of the tag that represents the context of this facet. For instance, the facet shown in Figure 4.10 has the context “buildings”. When hovered with the mouse, the header changes color and the little triangle pointing to the left is highlighted. The actual click removes the tag from the filter.

Finally, when the “none of these” option was selected to narrow down on the resources not being tagged with regard to the respective facet, clicking the option again reverses this navigation step and in effect also widens the filter (see Figure 4.12).

4.3.2.3. Effect of filter changes

In general, a modification of the filter affects all regions of the user interface. In the following, the respective effects are described.

Filter. When a tag is added via the search input or the selection of a navigation option, it is appended to the list of filter tags in the filter region of the user interface. However, since it is possible to enter a tag that is not offered as a navigation option, the user can directly jump to very specific tags. These tags are usually subsumed by several more general tags. To give the user feedback where the added tag is located in the organization structure, these general tags are automatically added to the filter region together with the tag the user entered. If the user enters for example the tag “stadium”, the tags “buildings” and “sports” are prepended — given that the tags are organized according to the examples above. Since the subsumption relation implies that all resources having the specific tag are also tagged with the subsuming tags, the effect on the result set is the same as without the additional tags. The interface

shows the same behavior in case a specific tag is selected from the navigation options and the filter does not contain all the respective more general tags yet.

When the user selects a “none of these” option in a facet, a special segment is appended to the filter region. It is displayed in red color and contains the first two tags of the respective facet prepended by the word “NOT” and followed by three dots to indicate that the other tags are excluded as well. Such a segment can be removed from the filter again the same way as individual tags. An example is shown at the top of Figure 4.12.

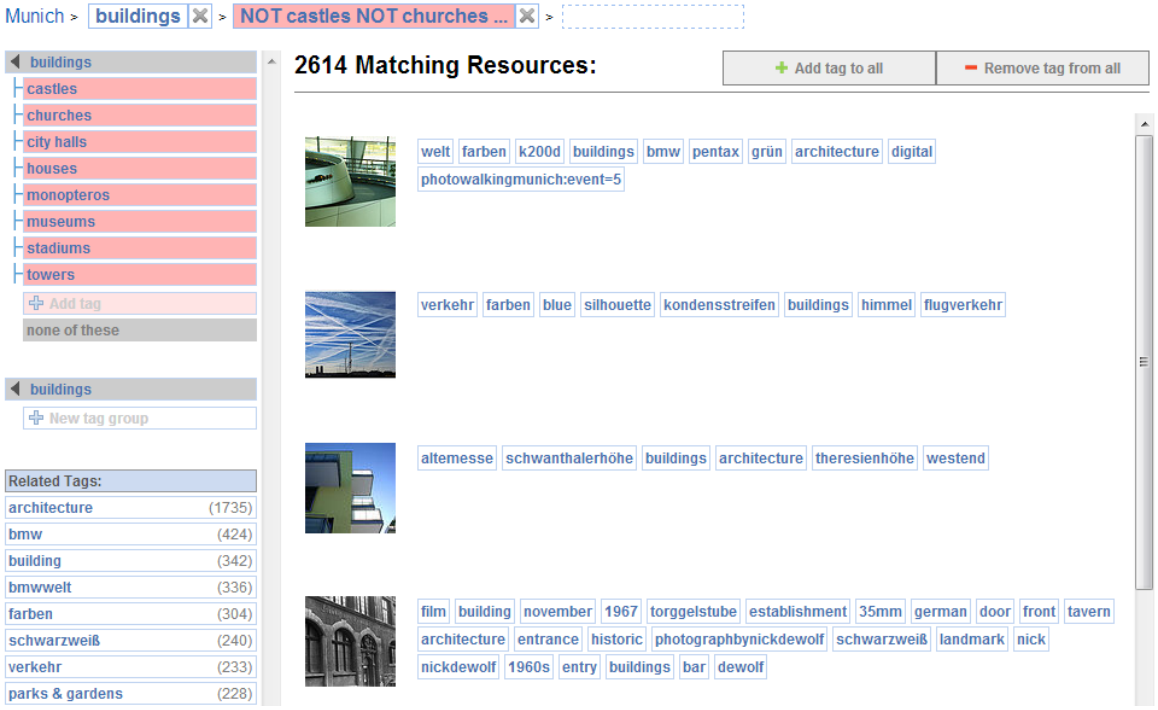


Figure 4.12.: The user interface after the selection of a “none of these” option.

Toolbar. The number of matching resources, that is shown in the left half of the toolbar, is updated according to the size of the new result set.

Result list. The result list is updated according to the new filter. Depending on whether the filter was narrowed or widened, either resources disappear or additional resources are displayed. Note that the order of the segments in the filter has no impact on the result set.

Navigation bar. A change in the filter can entail several changes in the navigation bar: When filtering by a tag that is part of one or more of the currently displayed facets, these facets disappear. Additionally, the facet list is extended by new facets being defined in the new and more specific context. The other facets remain visible. When a tag is removed from the filter, the more specific facets disappear and the general facets containing the tag become visible again respectively.

Similarly, the facet stubs are updated in such a way that only stubs for the most specific tags in the filter are shown (see Section 4.3.4 for an explanation of how stubs can be used to create new facets). In the example of Figure 4.8 this would mean that the stub for the

context “churches” would disappear as an effect of the selection of a particular church tag. A new stub for this tag would be shown instead.

When the user selects a “none of these” option in a facet, the respective facet remains visible but all its tags turn red to indicate that they now represent negative tags (see Figure 4.12).

Finally, the list of related tags is updated as well.

4.3.3. Tagging resources

Resources can be tagged individually or collectively. The latter is also referred to as *batch tagging*. In the so called *tagging mode* a particular resource is edited but changing the tags of resources is also possible in the normal browsing mode via drag and drop operations and context menus. The next paragraphs summarize the different ways tags can be assigned to and unassigned from resources.

Tagging mode. The tagging mode is activated by clicking on a resource. This resource is then highlighted and the other resources pale (see Figure 4.13). Additionally, the toolbar and the filter fade out to indicate that they cannot be interacted with. The list of facets updates according to the selected resource. More precisely, this means that additional facets are shown if the resource is tagged with respective tags. If the user edits a resource tagged with a color tag — such as “red” —, the color facet is shown (if such a facet is defined), regardless of the current filter. Additionally, all the facets that have been hidden in previous navigation steps are shown again: If the user is in the context {“buildings”, “churches”}, the “buildings” facet had been hidden after the tag “churches” was selected. When editing a resource, this facet appears again.

This is required to allow the user to choose more than one tag of a single facet. Tags in facets are selected and deselected with a mouse click. The changes are instantly reflected in the list of tags being assigned to the resource. When the user selects a tag, a more specific facet can appear below the facet the user selected from, giving that such a facet is defined. Respectively, deselecting a tag can have the opposite effect. When all tags are removed from the resource, only the most general facets, i.e., those defined for the empty filter are displayed.

It is not possible to create new facets in the tagging mode but users can extend facets using the input field above the “none of these” option (see Section 4.3.4). A click on the “none of these” option deselects all tags of a facet.

It is also possible to directly enter tags into the input field at the resource. The user is supported with the same autocomplete feature that is available when adding tags to the filter (see Figure 4.9). Individual tags can be removed with a single mouse click. The last tag in the list can also be removed with the *backspace* key. When changes are made in the tag input field, the tag selection in the facets — and possibly also the facet list — is updated accordingly.

Tagging in browsing mode. To make the tagging of resources as convenient as possible, additional features were added to the user interface. They allow to edit the tags of a resource without activating the tagging mode.

Each tag that is visible anywhere in the interface, i.e., the tags in the filter, the navigation

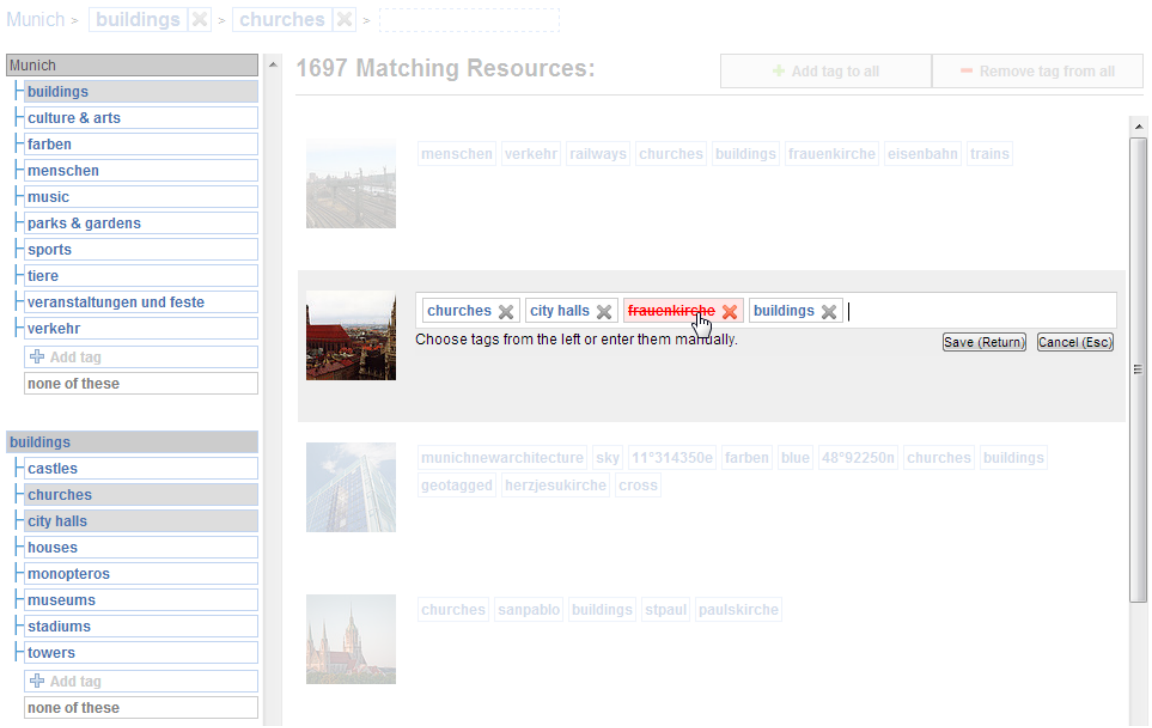


Figure 4.13.: The tagging mode.

bar, or at other resources, can be assigned to a resource with a drag and drop operation. The source of the tag is not affected by such an operation: If a tag in a facet is dragged, the facet is not changed. Similarly, tags are not removed from resources when being dragged onto another resource.

Since a facet does not disappear when its "none of these" option is selected, the user can easily drag the facet's tags onto the remaining resources in order to complete missing tag assignments. In Figure 4.12, the user could for instance assign the types of buildings to the images in the result set. Note that although the red color indicates that the tags are negated, the effect of a drag and drop operation is always the assignment of a normal tag.

A tag can be removed from a resource using a context menu that appears after a right click on the tag (see Figure 4.14). If the resource does not match the filter after such an operation anymore, it instantly disappears from the result list. When the filter contains negative tags, the resources can disappear after tag assignments respectively.

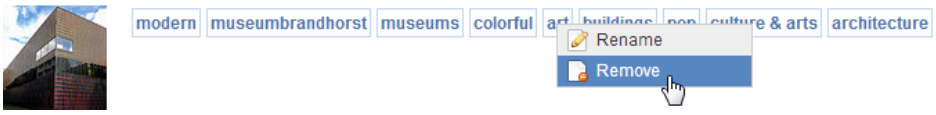


Figure 4.14.: Context menu for removing tags from resources in browsing mode.

It has to be noted here, that the operations described above do not supersede the tagging

mode. Only in the tagging mode the user sees all relevant facets and only there it is possible to assign an arbitrary tag to a single resource. Furthermore, the tagging mode is more convenient when several tags have to be removed because no context menu is required.

Automatic tag assignments.

Similar to the effect of the addition of a very specific tag to the filter (see Section 4.3.2.3), general tags are automatically assigned to resources if a respective more specific tag is assigned. When the tagging mode is active, the user instantly sees this effect in the tag input field of the resource (see Figure 4.13). In a batch tagging operation, the general tags are transparently added to the resources together with the new specific tag.

When a user sees that a tag was automatically added and she removes this tag, the subsumption relation that caused the addition of the tag is voided. This is necessary because the subsumption requires that the extension of the specific tag is a subset of the extension of the more general tag (see Section 4.1.1). Furthermore, the rationale behind this behavior is that this way the organization structure can adapt to new data, that does not fit the existing structure. It has to be noted that the facets are not affected but only subsumption relationships. In consequence, all previous navigation paths to existing resources are still valid.

Batch tagging. To edit the tags of several resources at once, the two fields in the right half of the toolbar can be used. They are labeled "Add tag to all" and "Remove tag from all" respectively and can be accessed in two ways: either a tag is dropped onto them or the user clicks one of the fields and can then enter an arbitrary tag in a popup window. The effect, i.e., adding or removing the respective tag, is applied to all resources in the current result set. As for single tag assignments, automatic tag assignments can be triggered when there are subsumption relationships, that would be voided if the respective more general tags were not assigned (or removed) as well.

4.3.4. Manipulating the TACKO structure

In the following, it is illustrated how facets can be created and modified, how subsumption relationships are established, and how the user can rename or merge tags.

4.3.4.1. Editing and creating facets

In each context, the user has the option to create new facets. For this purpose, so called facet stubs are displayed that only consist of the facet header and an input field with the caption "New tag group". Users are usually not familiar with the term *facet*, so we use the term *tag group* instead. The facet header shows the most specific tag of the context in which the facet will be created (see Figure 4.15a) and facet stubs are only displayed for these specific tags. In the context {"buildings", "churches"} a single facet stub for the tag "churches" would be displayed.

A facet is actually created by adding the first tag. This can be accomplished by one of four ways:

- Entering a tag into the input field,

(a) Context menu and facet stub **(b)** Tag suggestions

Figure 4.15.: Creating and editing facets

- dropping a tag onto this field (e.g., dragged from another facet or a resource),
- selecting a tag from a list of suggestions,
- or as a side effect of the creation of a new subsumption relation (see Section 4.3.4.2 for a detailed explanation).

The suggestions are shown below the input field as soon as it is activated (see Figure 4.15b). They contain the most frequent tags in the set of resources having none of the facet tags assigned yet (i.e., the set of resources reachable via the "none of these" option). The list of suggestions can be extended by clicking the "more suggestions" field at the bottom. When a tag from another facet is dragged onto the input field, it is removed from this other facet[4]. When the user manually enters a tag, he is supported by the same autocomplete functionality being also available for tag assignments and searches.

Tags are removed from facets using a context menu that is activated by a right-click on the respective tag. This is illustrated in Figure 4.15a. To delete an entire facet, all options have to be removed. The tag assignments to resources are not affected by facet changes.

4.3.4.2. Editing subsumptions

During our work with the prototype, it became obvious that facets and subsumptions are frequently used in combination. For this reason, the creation and removal of subsumptions was partly coupled with the management of facets.

Whenever a tag is added to a facet, the question arises whether subsumption relations to the context tags of the facet have to be established. A new subsumption can imply that tags have to be added to resources that are not included in the current search result. If for instance

[4]Note that a tag is not removed from a facet when it is dropped onto a resource.

the tag “skyscrapers” is added to a facet defined in the context {“buildings”} and there are resources being tagged “skyscrapers” but not “buildings” yet, the subsumption condition requires, that the tag “buildings” is added to all these resources. In such a situation, the user is prompted with a dialog as displayed in Figure 4.16. The user can inspect the affected resources by clicking the link at the top of the dialog. This opens a list of the resources in a new browser tab.

In order to make the interface easier to use, it was consciously avoided to ask the user for a subsumption relationship. Instead, it is directly asked if the effect of the subsumption relationship is intended. If there are no resources affected by a new subsumption relationship, it is established without asking the user. The user can void the subsumption later in case it was not intended.

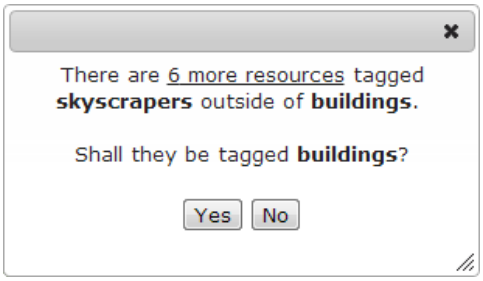


Figure 4.16.: Confirmation dialog for the creation of a subsumption relationship.

If the user decides that no changes should be made to the tags of resources, no subsumption is created, but the respective tag is added to the facet anyway. Figure 4.17 shows a facet defined in the context {“companies and brands”} that contains the tag “beer” but beer is not subsumed by “companies and brands”. This can be reasonable because there are photos related to beer that do not show a particular brand. However, selecting “beer” from the facet, still yields all photos being tagged “beer” and “companies and brands”. It has been observed that the subsumption relation was not well understood by all users (see Section 6.1.3.1), so one could argue that this additional distinction introduces too much complexity and only tags with subsumption relations should be allowed in a facet instead. Nevertheless, the feature can be very helpful while a collection is being organized, even if no facet tags without subsumption are left after the organization.

Figure 4.17.: Facet with a single tag (“beer”) that is not subsumed by the tag in the facet context.

Subsumptions can also be established via drag and drop operations. Therefore, any tag in the filter, the navigation bar, or the search result list can be dragged and dropped onto a tag in the navigation bar. The effect is that the dragged tag is subsumed by the tag it was dropped on. This can be compared to the way folders are handled in common file managers: Dropping a folder onto another one makes the former a subfolder of the latter.

In addition to the subsumption, a new facet is created with such an operation. If the subsuming tag is part of a facet, this works as follows. Let F be the context of the facet that the subsuming tag t_{super} is part of:

$$F \vdash \text{Facet}(\{t_1, t_2, ..., t_{super}, ..., t_n\}).$$

Let further t_{sub} be the tag that is dragged and dropped onto t_{super}. Then, a new facet is created in the more specific context $F' = F \cup t_{super}$:

$$F' \vdash \text{Facet}(\{t_{sub}\})$$

unless there is already a facet defined in F' that contains the tag t_{sub}. If t_{super} is not part of a facet but it is only shown in the list of frequent tags when t_{sub} is dropped onto it, the new facet is created in the context $\{t_{super}\}$ instead:

$$\{t_{super}\} \vdash \text{Facet}(\{t_{sub}\})$$

As already briefly mentioned in Section 4.3.3, subsumption relations are removed as soon as there is a resource that contradicts the subsumption condition: If there is a subsumption $t_1 \geq t_2$ and there is a resource r to which both tags are assigned, the subsumption is voided as soon as t_1 is removed from r. Facets are not affected in this case.

To assert that the addition of a tag to a facet and the subsequent removal of the tag from the facet has no unintended side effects (a remaining invisible subsumption relation), the respective subsumption relation is removed as well, if such a relation has been established. This also allows the explicit removal of subsumptions when they are applied in combination with facets as in the example of “buildings” and “skyscrapers” above: Removing the tag “skyscrapers” from a facet defined in the context {“buildings”} removes the respective subsumption if it is defined.

This does not apply when tags are moved from one facet to another within the same context. In this case, subsumption relations are not affected although a tag is removed from one facet as the effect of such an operation.

4.3.4.3. Renaming and merging tags

Figure 4.15a shows the context menu that appears when the user right-clicks on a tag in a facet. By choosing the “Rename” option, the user can open a small dialog with an input field in which the tag can be edited. This dialog can be reached through context menus of all tags in the interface, i.e., also the tags in the filter or those shown at the resources.

The effect of a rename operation applies to all resources in the collection regardless of the current filter or the facet in which it was triggered. Facets and subsumption relations are changed accordingly.

By renaming a tag t_1 to the name of another tag t_2 that exists in the collection, t_1 is *merged into* t_2. This means that after this operation the extension of t_2 equals the union of the extensions of t_1 and t_2 before the rename. The extension of t_1 will be empty since it is removed from the collection. How existing facets and subsumption relations are affected by a merge of two tags is covered in Section 5.4.2.4.

Renaming tags is not only useful to correct spelling mistakes but it can be applied to simplify the organization structure when it contains distinctions of concepts that are too subtle and not relevant for organizing the collection effectively.

4.3.5. Animations and other visual effects

Early experiments and informal user tests with our prototype revealed that it was very hard for users to understand the organization structure and the operations that can be performed to manage it. These experiences influenced the later development of the user interface and led to the integration of several visual clues and metaphors that are intended to help users understand the system.

4.3.5.1. Dynamic highlighting of interface elements

To indicate which parts of the interface users can interact with, all of them are highlighted when hovered with the mouse or at least the mouse pointer changes. In browsing mode these controls are:

- The name of the collection.
- The "X" icons of the filter elements.
- The input field in the filter.
- All tags in the sidebar including the headers of facets.
- The input fields in facets and the "none of these" options.
- The batch tagging fields in the toolbar.
- All resources in the search result.
- All tags of resources in the search result.

In tagging mode, the only actions being triggered with a click are:

- Adding or removing a tag displayed in a facet.
- Removing a tag from the resource's tags.
- Leaving the tagging mode by saving or cancelling.

To some extent, it was also tried to give users hints regarding the effect of these actions. When a resource's tags are hovered in tagging mode, the text changes color to red and is crossed out. The little triangle in the facet header moves slightly to the left when the header is hovered to indicate that this triggers a step back in the navigation path.

When hovering a tag in the navigation bar, all resources to which this tag is assigned are highlighted in the result list. This way, the user sees which resources would still match the filter when the hovered tag was added. Conversely, tags in the navigation bar are highlighted when the mouse hovers a resource to which they are assigned. This is depicted in Figure 4.18. Hierarchical fragments in the organization structure can tempt users to assume that the facets resemble a classical tree-like folder structure. To better illustrate that a single resource can be associated with several tags of the same facet, the highlighting of tags in the navigation bar was added.

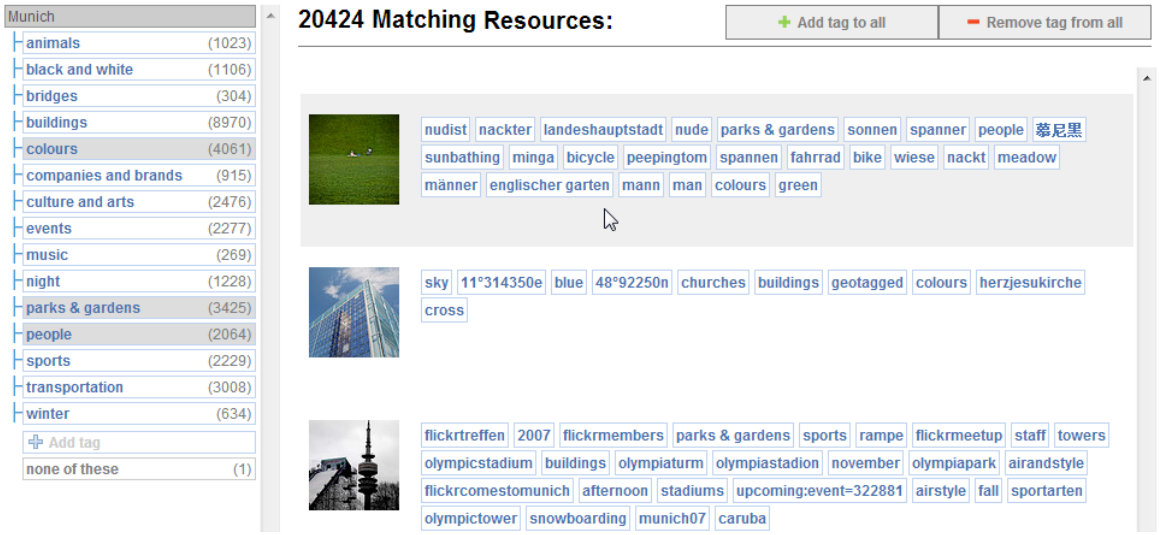


Figure 4.18.: Highlighting of tags in facets.

Finally, when tags are being dragged around the user interface, all elements that they can be dropped on, i.e., other tags, resources or controls, are highlighted when the dragged tag is above them.

4.3.5.2. Transitions after filter changes

To help users understand how the different facets relate to each other and how a change in the filter affects the set of displayed facets, the interface does not change instantly after a modification of the filter. Instead, the search result as well as the facet list are updated in smooth transitions. This is indicated in Figure 4.19.

The figure shows the transition of the interface after the tag “city halls” has been added to the filter. When a tag is added to the filter in any way, all facets that contain this tag, shift out of the screen to the left side. In the example this is only one facet with the context {“buildings”}. New, more specific facets and facet stubs appear below. Facet stubs for contexts being more general than the context of the new facet vanish. In the figure, this is the stub for {“buildings”} since {“buildings”,”city halls”} is more specific. After the disappearing facets have moved out of the screen, the other facets shift to the top.

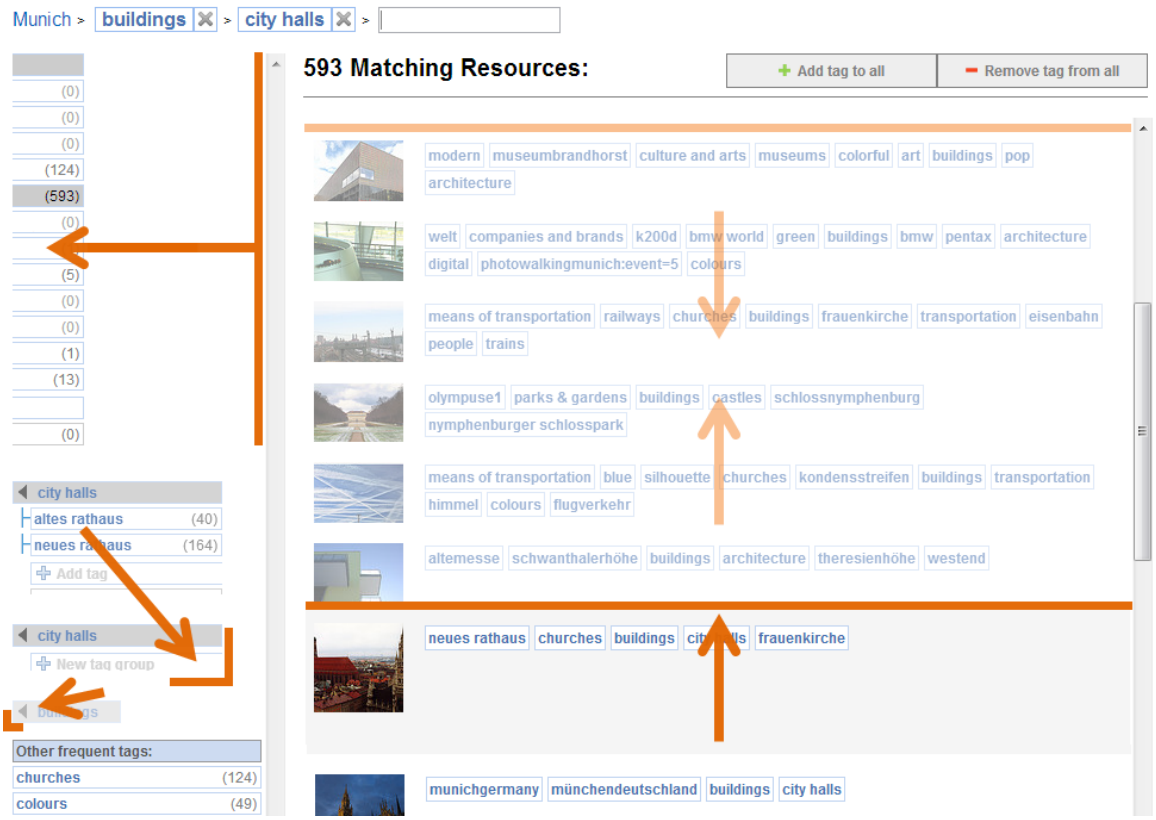


Figure 4.19.: Transitions of elements in the user interface after a change in the filter.

In the result list, all resources that do not match the new filter fade out and their height is reduced to zero. This makes room for other resources entering the screen from the bottom.

When the filter is widened, the same animations run in reverse.

4.3.5.3. Scrolling instead of paging

In the last years, it could be observed on the web that paging in result lists is increasingly replaced by scrollable lists that are only partially loaded in advance but reload transparently, when the user scrolls near the parts of the list that are not loaded yet. For instance *Facebook*[5] applies this functionality in almost all lists.

We applied this principle in our prototype for two reasons:

- It is more convenient than paging through long result lists.
- With this principle, the user has the impression of having immediate access to the entire list although only a small fraction is actually loaded.

The latter point is crucial for our prototype. In order to understand the interface and the

[5] http://www.facebook.com, (accessed 2nd of November 2012).

organization structure, it is important that users know that contrary to folder structures they do not have to browse to maximum specificity to access a particular resource but that the resources they are looking for are in the result list even before any filter is applied. Filters only help to narrow down the search result and to access resources more quickly.

CHAPTER 5

Implementation

In this chapter, we describe the implementation of our approach within the commercial web-based enterprise collaboration platform *Tricia*. Thereby, we cover the advanced tagging functionality that was implemented to realize the initial approach of implicit tag relations as well as the implementation of the explicit TACKO tag relations.

Please note that our implementation represents only one of many possible alternatives how to realize this functionality. Further, the architecture of the Tricia platform often led to design decisions that probably would have been made differently for other platforms. For these reasons, we will omit unnecessary details and sometimes make simplifications when it facilitates the understanding of the key ideas underlying the implementation. However, we aim to provide a level of detail that suffices as a solid basis for the realization of the approach in other software tools.

First, we will introduce, Trica and the Lucene index, the technological basis of our implementation in Section 5.1. In Section 5.2 we will provide an architectural view of our implementation and its integration in the Tricia platform. Section 5.3 covers the realization of advanced tagging functionality that forms the basis for the implementation of the TACKO organization structure which in turn is described in Section 5.4.

5.1. Applied technology

Although it was attempted to separate the basic tagging functionality and the Tricia core as much as possible in order to facilitate software maintenance, the development of TACKO was inseparably connected to the development of the Tricia platform. Therefore, we will introduce Tricia and its architecture in Section 5.1.1.

Tricia utilizes the *Lucene* index to provide full-text search functionality. However, the index

offers much more general functionality and can be used to provide tag-based access to a collection of resources as well. Since it was not only applied for this purpose but also the implementation of tag-autocomplete functionality that takes into account users' access rights (see Section 5.3.3), Lucene is covered separately in Section 5.1.2.

5.1.1. Tricia

The enterprise collaboration software Tricia is commercially distributed by the infoAsset AG[1]. During the development phase of TACKO, the Java source code was available to the public and earlier versions were released under an open source license. Before we cover the Tricia architecture, we will give a brief overview of the most important features of the platform.

5.1.1.1. Tricia features

Tricia is an integrated platform for enterprise collaboration and information management. Users can collaboratively work on wiki pages, share files or write blogs. All contents can be accessed via a web interface. Additionally, Tricia can be mounted as a network drive to access the contained files.

The central content element in Tricia is the wiki page or just *page*. Each page has a title and can contain formatted text. Furthermore, pages serve as containers for files and for other pages, i.e., pages are organized in a tree structure. Pages that do not contain any text can be compared to directories in a file system.

In Tricia, special emphasis has been placed on fine grained access control. The set of pages is partitioned into so-called *workspaces* for which default access rights are defined. However, within a workspace the access rights for all content items can be individually overridden as required. Since the pages within a workspace are organized in a tree structure, it is also possible to define access rights for specific branches of this tree. Access rights are specified by defining a set of readers and a set of editors. Thereby, each editor is a reader as well. An element in these sets is either a single user or a user group. The users in a group can be defined by referring to other groups. This way the management of access rights is facilitated.

An important feature of pages is that their contents can be structured with so-called *type tags* and *attributes*. These structuring capabilities are an implementation of the *Hybrid Wikis* approach which is described in Section 7.4. The textual content of pages can contain hyperlinks, either to external contents or to other content items in Tricia. These links can also be navigated in reverse direction by displaying the set of incoming links for all content items. Furthermore, all content items — this includes for instance also users, user groups, and files — can be annotated with arbitrary tags. Tags can be used to organize contents of different types and across workspaces. They play an important role in the retrieval of resources.

Tricia features a powerful full-text search functionality. All textual contents of all items are indexed. This includes the tags, attributes, comments, and for files also the file content. The search is directly available through an input field at the top of each page. As the user types, the search results are listed in a drop-down menu and can be directly accessed. Additionally,

[1] http://www.infoasset.de (accessed 26th of February 2013).

a more elaborate search interface is available for more complex searches. Figure 5.1 shows a screenshot of the Tricia search interface. For each search result a tag cloud is displayed that can be used to further narrow down the result set. Further filters, for instance for content type and workspace, are displayed at the left border of the screen. A certain combination of filters can be saved and embedded into a page as an "embedded search". The search is evaluated each time the respective page is accessed. This way dynamic pages can be configured that can serve for instance as a dashboard that provides an overview of pages with certain properties. For example if a list of tasks is represented by a set of wiki pages, a page can be configured that lists all completed tasks. Tags, type tags, and attributes are particularly useful for this purpose.

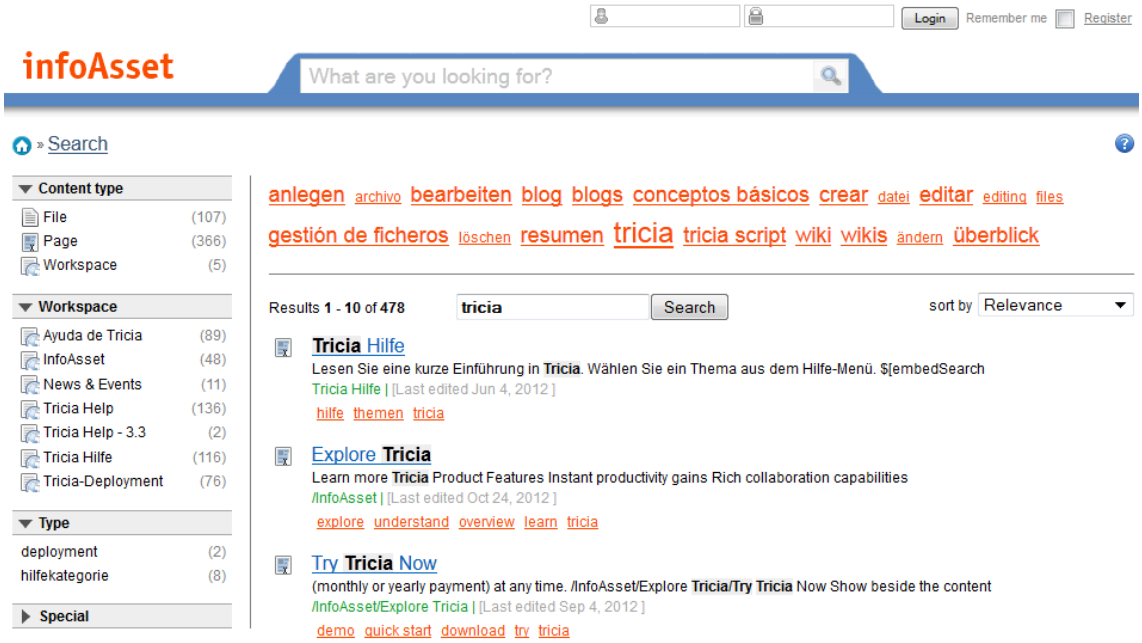


Figure 5.1.: Screenshot of the Tricia search interface.

Tricia's search functionality represents a central component of the platform. It is also applied in all input fields that are used for establishing references among content items. One example is the input field being used to link a page to a parent page, i.e., the superordinate page in the tree structure. Figure 5.2 shows an example. Thereby, all search results are filtered by the access rights of the current user, so a user never sees the title of a page he is not allowed to access.

Finally, the platform allows users to monitor content changes. Users can add content items to their *watch lists* if they want to be informed about changes. Updates to these content items are displayed on each user's personal dashboard and can be additionally received by email.

5.1.1.2. Tricia architecture

In the following, we will cover the Tricia architecture as far as it is relevant for this work. A more detailed overview is given in [BMN10]. Figure 5.3 shows the deployment architecture of

Figure 5.2.: Tricia search functionality integrated in an input field for a page reference.

Tricia. The server component is implemented in Java, the code of the web interface, that is run on the client, is written in Javascript and makes heavy use of the *jQuery* library[2]. AJAX techniques[3] are used to implement dynamic controls on web pages such as instant search and input fields with autocomplete functionality.

A client's HTTP-request is mapped to a so-called *handler* class on the server side. Handler classes contain most of the business logic of Tricia. After a handler has processed a request, it can either forward to another handler or produce a result that is sent to the client. The result is usually an HTML document or it is in JSON format (for AJAX requests).

Data persistence is based on a relational data store (different relational databases are supported) and the file system. Nearly all persisted data objects are also stored in a *Lucene* index to facilitate efficient execution of text searches and structured queries. Lucene is covered in more detail in Section 5.1.2. However, the primary store is realized with a relational database system and the file system, so data persistence does not depend on the Lucene index.

An essential component of the Tricia platform is the data modeling framework that — among other things — performs an object-relational mapping between the data objects and relational database tables [BMN10]. Furthermore, it automatically adapts the database schema after changes in the data model. A UML diagram of the essential concepts is displayed in Figure 5.4. The central class is the `Asset`. It represents a collection of named `Features` which in turn are either `Properties` or `Roles`. A property is a literal value while a role represents a relation to another asset. Important properties are for instance the `IntProperty`, the `StringProperty` or the `RichStringProperty`. As the names suggest, the `IntProperty` allows to store integers and the other two properties store string values. The `RichStringProperty` can include formattings, such as bold text, embedded images and tables, as well as hyperlinks.

An `Entity` is an asset that represents a single independent data object. All content items in Tricia are represented by subclasses of `Entity`. However, there is another kind of asset called `Mixin`. Mixins are used to circumvent the single-inheritance restriction of Java and extend entities with additional features and functionality. This means that mixins allow reuse of certain pieces of functionality in different entities. Important subclasses of `Mixin` are:

[2] http://jquery.com (accessed on 27th of February 2013).
[3] http://en.wikipedia.org/wiki/Ajax_(programming) (accessed on 27th of February 2013).

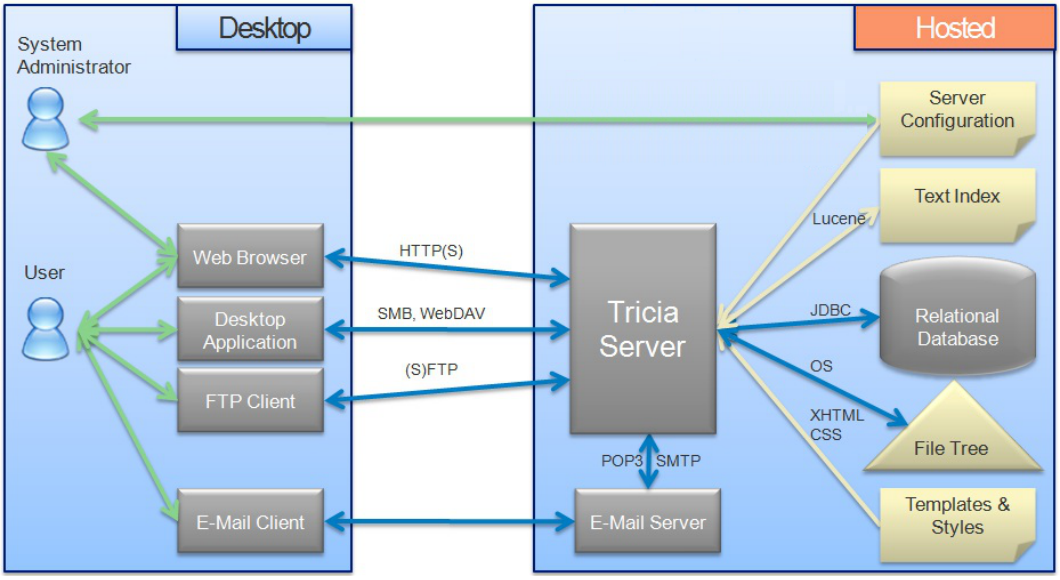


Figure 5.3.: Tricia deployment architecture as shown on the website of the infoAsset AG (`http://infoasset.de/pages/1lr0iafxcx9v6/`, accessed on 27th of February 2013, the image was slightly modified to reflect recent changes).

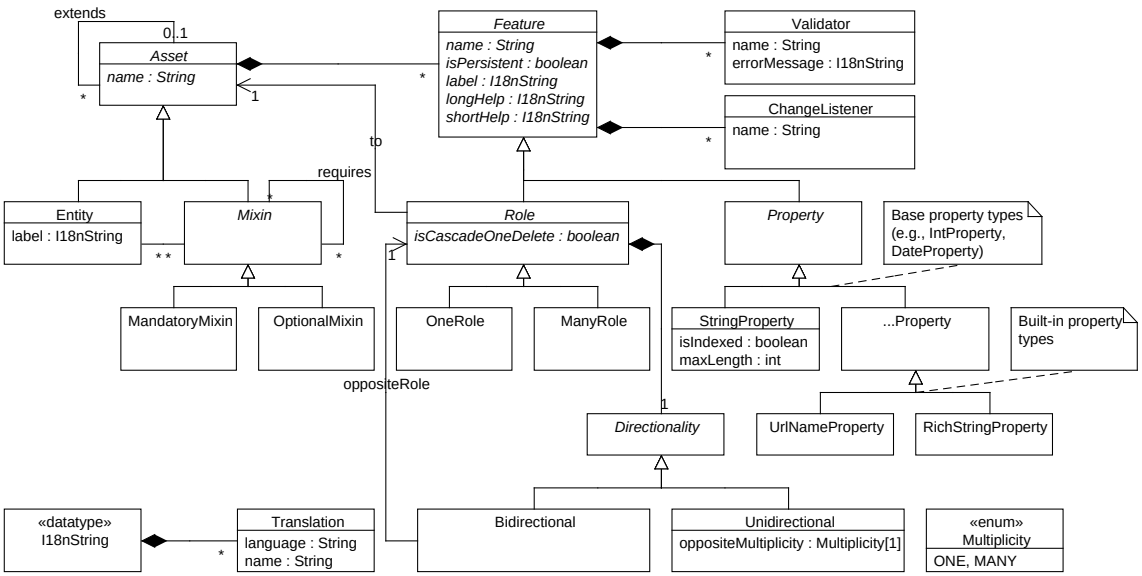


Figure 5.4.: UML diagram of the Tricia data-modeling framework [BMN10].

- `Linkable`: Required for entities that are the target of hyperlinks in text properties. Note that such links are different from roles.
- `Searchable`: Entities with this mixin are indexed and can be found with the Tricia search.

- `Taggable`: Extends an entity with a set of tags.
- `Hybrid`: Adds type tags and attributes, the central structuring concepts of hybrid wikis, to an entity (see Section 7.4 for more details).
- `ReadProtected`: Provides basic access control functionality.
- `Versionable`: Entities with this mixin record a history of changes of their features.

Mixins can have dependencies to other mixins. A searchable entity must for example always be linkable as well. The mixin `Taggable` plays a central role in the implementation of Tricia's tagging functionality and TACKO and will be covered later in more detail.

As depicted in the upper right quadrant of Figure 5.4, each feature of an asset has — possibly empty — sets of `ChangeListener`s and `Validator`s. A validator is used to limit the value domain of a feature or to ensure its consistency with other features. It can prevent an entity from being persisted if the value of the feature is not valid. This way it can for instance be asserted that an email address is valid or a name property is not empty. Change listeners make it easy to directly respond to changes of the respective feature. They can for instance be used to trigger an update of the Lucene index, to reflect changes in the access rights of an entity or to send an email to a user.

5.1.2. Lucene

Lucene is a powerful open source search library that is implemented in Java and supported by the *Apache Software Foundation*[4]. It is highly scalable, very fast and provides software developers with the essential functionality to efficiently index and retrieve documents. Lucene is a mature software that powers such popular services as *Twitter*[5][6].

We will briefly outline the basic functionality of Lucene since it forms an essential basis for the implementation of tagging in Tricia. However, for a more detailed coverage of its capabilities we refer to [MHG10] and the online documentation [7].

Lucene is a reverted full-text index. This means that a Lucene index stores a set of *documents* and these documents can be retrieved based on the *terms* contained in these documents. More precisely, A Lucene document consists of a document number and a set of *fields* that in turn consist of a name and a textual value. The values of the fields are indexed by breaking up their contents into terms. This is done with a so-called *analyzer*. An analyzer can for instance filter out stop words (usually frequent words like "the" or "a" that shall not be indexed), remove punctuation, or convert words to lower-case. Lucene comes with a collection of basic analyzers but also allows to implement custom analyzers if this is required.

Analyzing all the fields of a document results in one set of terms each of which is represented by the field name and a — usually short — text value. At the most fundamental level of indexing and retrieval, Lucene only operates on the terms. Although the raw field contents can be stored and retrieved as well, a query is always expressed based on Lucene terms.

[4] http://www.apache.org (accessed on February 27th 2013).
[5] http://twitter.com (accessed on 27th of February 2013).
[6] http://techcrunch.com/2010/10/06/new-twitter-search/ (accessed on 27th of February 2013).
[7] http://lucene.apache.org/core/documentation.html (accessed on 27th of February 2013).

To convert a textual query string into an expression over Lucene terms, a *query parser* is used. Similar to the analyzer, it transforms the text of the query string, but in contrast to just processing the individual text tokens (e.g., removing punctuation) its output is an expression that is used to find matching documents in the index.

In the simplest case, a query is just a Boolean expression over terms that evaluates to *true* or *false* for each document. Lucene is optimized to find matching documents efficiently for such expressions. For each term in the index Lucene stores a compressed linked list of matching document ids in ascending order. To find for instance all documents containing two specific terms, Lucene runs through both lists in parallel and collects matching pairs of ids. In effect, the intersection of the two document sets corresponding to the terms is determined this way. This is illustrated in Figure 5.5 for the terms "Brutus" and "Caesar". The runtime complexity for this operation is $O(m + n)$ where n and m are the numbers of documents matching the first and the second term respectively [Cu04]. The data structure actually used in Lucene is a bit more complex than illustrated in the figure. It uses so-called *skip pointers* to skip several entries in a list in one step if there is a large increase in the document id in the respective other list [Cu04].

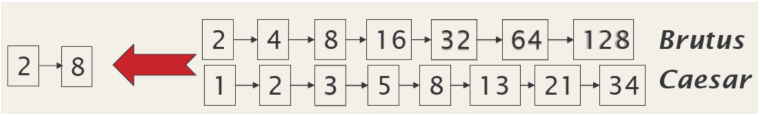


Figure 5.5.: Finding all documents containing two specific terms by running along two linked lists of document ids in parallel [Cu04].

Finding matching documents for a given query is the first part of the retrieval process. Matching documents are obtained with a *collector* while a *scorer* computes a *score* for the collected documents and ranks them accordingly. Collector and scorer are represented by Java classes that can be exchanged to override the behavior of the index. The score of a document typically depends on how often the searched terms occur in the document or the overall relevancy of a document that can be defined through a so-called *document boost* during indexing. The positions of the query terms in the documents can also be taken into consideration.

Finally, a feature that is particularly important in the context of this thesis is that Lucene provides fast access to the other terms of a matching document, i.e., the terms that are not specified in the query, during the retrieval process. This is for instance required to efficiently analyze a search result. Features of the documents, such as the document type or the workspace it is stored in, can be stored in respective fields of the Lucene documents. By counting the terms in these fields during retrieval, one can efficiently provide the user with the number of documents in each category, as it is shown in Figure 5.1. Counting the frequencies of tags also depends on this feature.

5.2. Architecture

When Tricia was extended to support the management of implicit tag relations (see Section 3.3), advanced tagging functionality was required. For instance it was necessary to efficiently add a tag to a large number of resources or to count the frequencies of tags in

the entities of a search result. Later, when the explicit TACKO organization structure was developed, much of this functionality could be reused. Therefore, we distinguish classes and components developed to manage TACKO tag relations and those realizing the underlying tagging functionality.

Figure 5.6 depicts how tagging functionality and TACKO in particular is integrated into the Tricia platform. All classes and dependencies introduced for TACKO are marked displayed in red or surrounded by red dashed boxes. In the following, we will briefly explain the components of the architecture as well as the underlying rationale. The following overview is meant to serve as a frame of reference for more detailed descriptions of the individual parts that are given in later sections.

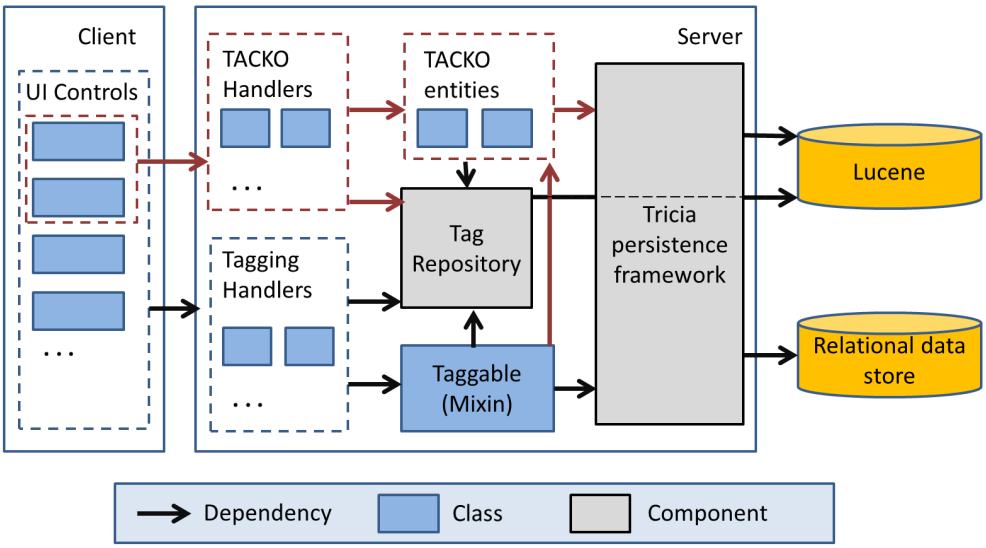


Figure 5.6.: Architectural overview of the integration of TACKO and advanced tagging functionality in Tricia.

One of the central classes in the implementation of the tagging functionality is the class `Taggable`. It represents a mixin type that can be added to arbitrary entities to extend them with tagging capabilities. The mixin basically provides methods for accessing and manipulating a set of tags. The tags are indexed in the main Lucene index that is managed by the *Tricia persistence framework*.

There were two conflicting goals during the implementation of the tagging functionality. On the one hand, it was tried to minimize the dependencies to other parts of Tricia to facilitate the parallel development of the platform. On the other hand, the integration had to be close enough to store tags based on the Tricia persistence framework in order to avoid another separate data store. This problem was solved by introducing a new component, the *tag repository*.

The tag repository provides methods to efficiently access and manipulate all tags stored in the Tricia platform. Particularly it allows to instantly modify the tags of a large number of resources and it allows to efficiently retrieve all tags starting with a certain prefix. The latter is required to provide autocomplete functionality during tag input. For this purpose, the tag repository manages a set of Lucene indexes. However, for the actual storage of tags, the

repository relies on a `StringProperty` of the `Taggable` mixin that stores the tags in serialized form. This storage mechanism can be changed to facilitate testing of the tag repository. It is covered in more detail in Section 5.3.

The taggable mixin and the tag repository are accessed by a set of *tagging handlers* that expose Tricia's tagging capabilities to the *user interface controls* on the client. Each handler provides a small piece of functionality such as retrieving all matching tags for a given prefix or updating the tags of a certain entity. One example of a user interface control is a widget that allows to edit a single entity's tags by using AJAX requests and without reloading the complete web page.

When TACKO was implemented, new special controls and handlers were added for the management of explicit tag relations. Furthermore, new entities were required to model the tag relations and make them persistent. Since the tag relations are implemented based on Tricia entities, persistence is provided by the Tricia persistence framework. All relevant implementation aspects related to the explicit TACKO tag relations are covered in Section 5.4.

5.3. Advanced tagging functionality

In the following, we cover the implementation of the Tricia tagging functionality as well as advanced features that were developed to support the management of implicit tag relations and that were later reused when the TACKO functionality was implemented. Most of this functionality is realized as part of the tag repository, so this section will be mainly concerned with this component.

First, we briefly describe the conceptual model underlying all tag assignments that are managed in the tag repository. Subsequently, we cover the design of the tag repository. Finally, the implementation of particular features such as the support for efficient tag autocompletion or the modification of a large number of resources are described.

5.3.1. Tag assignments and namespaces

Figure 5.7 shows a conceptual model of the data being managed by the tag repository. Basically, the repository stores a set of resources and the associated tag assignments. Each resource is identified by an id. A tag assignment connects a tag to a resource and tags are independent of individual tag assignments.

A tag does not only consist of a name but it is also associated with a namespace. Namespaces were added to allow the representation of additional features of resources in the tag repository. The actual tags that a user assigns to a Tricia entity all share the same *standard* namespace. However, there is for instance another namespace for the type tags of hybrid wikis (see Section 7.4 for an explanation of type tags). In effect, a Tricia entity — represented by a resource in the tag repository — can have the same label assigned as a tag and as a type tag.

Another namespace is used for special tags that indicate which users and groups have read access to the respective resource. In the following, we will call it the *read access* names-

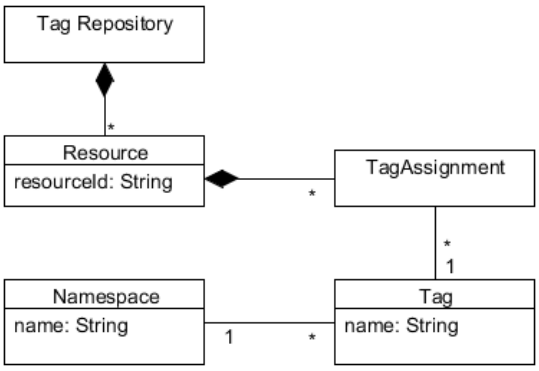


Figure 5.7.: Conceptual UML diagram of the data managed by the tag repository.

pace. Tags with this namespace represent the ids of users and user groups that are explicitly granted read access to the resource (the ids of users being indirectly authorized through the membership in one of the groups are not contained). The mixin `ReadProtected` was adapted to assert that changes in the access rights are properly reflected in an entity's tags if it also has the mixin `Taggable`.

5.3.2. Design of the tag repository

The tag repository was implemented to provide and encapsulate as much of the tagging functionality — apart from the user interface — as possible. It provides services such as the storage of tag assignments or querying for resources matching certain tags. Furthermore, it supports the instant assignment of a tag to a large number of resources and it can manage a set of *tag indexes* that allow for the efficient retrieval of tags matching a certain prefix.

To facilitate the parallel development of the Tricia platform, the tag repository has a well-defined interface represented by the class `TagRepository`. It serves as the *facade* (cf. the facade pattern in [Ga94]) of the repository. It synchronizes access and orchestrates the other classes.

Figure 5.8 shows a UML diagram of the most important classes of the tag repository component. As indicated by the ellipses at the bottom of most classes, the diagram does not show all methods but only those being essential for illustrating the interaction among the classes. The tag repository does not depend on a running Tricia platform but can be used independently. This proved to be particularly beneficial for testing.

However, as already mentioned in Section 5.2, it was aimed to reuse the Tricia data persistence framework for tag storage in order to avoid a separate storage mechanism. For this reason, the tag repository can be configured to use a special `TagStore`, that depends on the Tricia persistence framework. This `TriciaTagStore` uses a certain `StringProperty` of the `Taggable` mixin to store the tags of an entity. For this reason the tag repository depends on the `Taggable` in Figure 5.6. This way, the Tricia persistence framework stores the tags together with the other attributes of an entity in the same relational database record. As a consequence, certain inconsistencies that could occur if tags were stored separately are avoided.

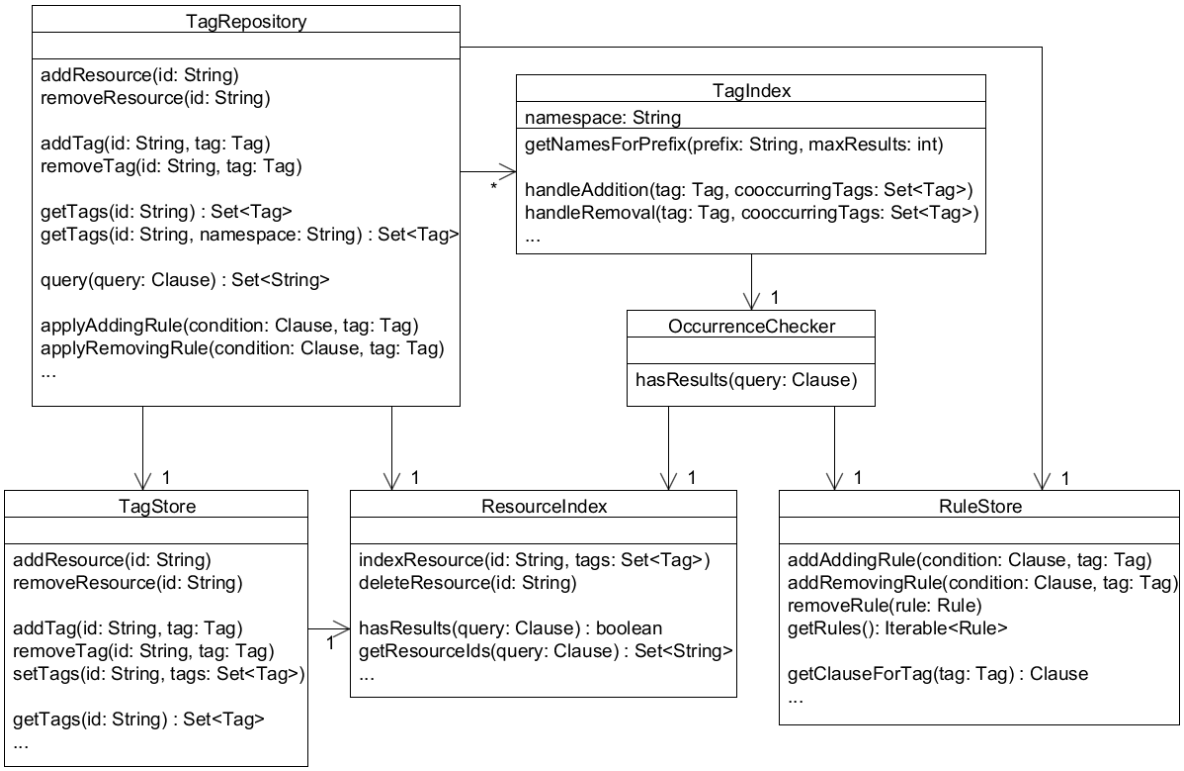


Figure 5.8.: Design of the tag repository represented by the central classes with the most essential attributes and methods.

Since Figure 5.8 only shows part of the methods of the `TagRepository`, we briefly summarize all the services it offers through its interface before we cover the other classes and their interactions:

- **Addition and removal of resources:** New resources have no tags. If a resource is removed, all respective tag assignments are removed as well.
- **Addition and removal of tag assignments:** For each resource, single tags can be assigned or removed. Setting the tag assignments of a resource to a certain set is always established by gradually adding or removing single tags. It is also possible to assign a certain tag to a set of resources or remove a tag respectively. The respective resource set is specified by a Boolean expression over the tags (similar to a query; see below). Such updates are processed in constant time. How this is accomplished is covered in Section 5.3.4.
- **Access to tag assignments of a single resource:** All tags can be retrieved for a certain resource id. Tags can be optionally filtered by a namespace (e.g., to only retrieve the type tags).
- **Querying resources:** A query for a set of resources is represented by a Boolean expression over tags. For instance, if t_A and t_B are tags, then a query "$t_A \wedge \neg t_B$" will yield all resources to which t_A is assigned but not t_B.
- **Counting tag frequencies in a certain set of resources:** As for a query, a Boolean

expression defines a set of resources. For this set, the occurrences of tags are counted. The result is a map that contains all occurring tags as keys and the respective numbers of occurrence as values. This functionality is for instance essential for the generation of tag clouds.

The `TagIndex`es represent an exception to the facade pattern since they can be directly accessed. During initialization, the tag repository is parameterized with a set of such indexes that are then maintained by the repository. In the simplest case, such an index efficiently retrieves all tags in the repository that match a given prefix. However, tag indexes are more powerful. They are explained in detail in Section 5.3.3. Whenever tags are assigned and removed, the tag indexes are notified. Upon removal of a tag it is required to check whether there is still another instance of it, since the indexes do not keep track of the total number of occurrences for each tag. Performing this check is the responsibility of the class `OccurrenceChecker`. This class relies in turn on a `ResourceIndex` and a `RuleStore`.

The `ResourceIndex` is used internally, to process queries for resources. It directly reflects the contents of the `TagStore` and indexes each resource as a Lucene document. The tags are stored in fields of the document (one field per namespace). Tags are not further processed by an analyzer. This means for instance that tags containing spaces are not split but indexed as one term. The resource index provides methods to obtain all results for a query or to check whether a query yields at least one result. In addition, it allows to get aggregated information about the tags of the resources in a search result, for instance the individual tag frequencies.

Finally, the `RuleStore` is used to record changes of the tag assignments that have not been applied to the tag store and the resource index yet. If for instance the tag t_A is removed from all resources having tag t_B assigned, this can involve changes to a very large number of resources and requires respective updates of the index. Since the organization of a set of tagged resources involves many such updates, they have to be processed very fast so that the users can work without notable delays. For this reason, the changes are first stored in the form of a *rule* in the rule store. These rules are transparently applied when the contents of the tag repository are read or queried. The changes implied by the rules are asynchronously applied to the tag store and the index. Eventually, the rules can be deleted. This process is covered in detail in Section 5.3.4.

5.3.3. Tag indexes

The purpose of tag indexes is to determine the set of tags starting with a certain prefix. This is required to provide efficient autocomplete functionality when tags are entered into input fields. Although Lucene automatically creates an ordered list of terms for each field in an index, using this list has two disadvantages:

1. Access to the term list is only possible with the very prefixes of the terms starting with the first character. It is for example not possible to find the tag “tagging system” by the prefix “sys”.
2. The term list allows no further filtering of the terms, for instance restricting the terms

to only those contained in documents stored in a certain workspace or accessible by a certain user.

Both problems can be solved with the tag indexes of the tag repository. The tag index uses a Lucene index to efficiently evaluate prefix queries for tags. The tags are stored as Lucene documents and all prefixes of the tag represent the respective index terms. This is illustrated in Table 5.1. It shows that in the resource index the documents are resources that can be retrieved by tags and that the tag index indexes tags by prefix. Furthermore, both indexes allow to filter the results by so-called access tokens.

Document (resource)	Tags	Access tokens
#1	“foo”	(1), (2)
#2	“foo”, “bar”	(2)
#3	“bar”, “beer bar”	(1)

(a) Resource index

Document (tag)	Prefixes	Access tokens
“bar”	“b”, “ba”, “bar”	(1), (2)
“beer bar”	“b”, “ba”, “bar”, “be”, “bee”, “beer”	(1)
“foo”	“f”, “fo”, “foo”	(1), (2)

(b) Tag index

Table 5.1.: Comparison of resource index and tag index.

To determine the set of prefixes for a tag, a tag index uses a `TokenizationStrategy` that can be configured individually per tag index. It depends on the tokenization strategy, whether the prefixes are always lower-case or whether they include white-space characters and punctuation. In the example of Table 5.1 the tags are first split up at the whitespace characters and the prefixes of the resulting words are indexed. In effect, the tag “beer bar” is also retrieved for the prefixes “ba” and “bar”.

Being able to filter the tags in the tag index by access tokens is required to consequently enforce the Tricia access rights. Users only see a tag in the autocomplete suggestions if they may read at least one entity having this tag assigned. To accomplish this, the tag index has a field “access tokens” that contains the union of all access tokens being assigned to entities with the respective tag. Each access token stands for a particular user or a group of users. For instance, the tag “bar” in Table 5.1 has both access tokens (1) and (2) because it occurs at resource #2 with access token (2) and resource #3 with access token (1).

To determine the tags suggested to a particular user via the autocomplete list, the tag index is queried with the respective prefix and all access tokens of the user, i.e., the user’s personal token as well as the tokens of all groups the user is member of. The index returns all tags matching the prefix and being indexed with at least one of the access tokens.

In fact, queries to a tag index can not only be filtered by access tokens but filtering by read access is only an example. For each tag index, a so-called *context namespace* can be defined. This way, any tag namespace can be used instead. For example, filtering by workspace is also possible if this information is encoded in a tag with the respective namespace. However, we

assume in the following that the context namespace is always the read access namespace in order to provide demonstrative examples.

While this simple mechanism can ensure that a user never sees a tag he or she is not allowed to see, it is not trivial to maintain consistency of the index. To facilitate this, all changes to the tag repository are broken down into the addition and removal of individual tags. Please note that the addition of a tag can affect several resources, so it results in many new tag assignments. However, for the tag index it makes no difference how many resources are affected. When the tag index is notified directly after the addition or removal of a tag, it is also provided with the set of all other tags being assigned to the affected resources. This set of tags is in the following referred to as the *cooccurring tags*. We will now briefly cover how the index is updated.

Addition of a tag. If the added tag is not indexed yet, a new entry is created in the tag index. In any case, if the set of cooccurring tags contains tags of the read access namespace these tags are indexed in the designated field since apparently now at least one document exists that users with the respective read access tokens may read and that has the added tag assigned. Let for instance "beer bar" be the added tag and among the cooccurring tags be the read access token (2). This means that "beer bar" has been added to at least one entity with access token (2) and in the example of Table 5.1, the tag representing token (2) would be added to the "access tokens" field.

Removal of a tag. This step is more difficult since neither can the respective tag simply be deleted from the index nor can the cooccurring tags of the read access namespace be removed. After the removal of a tag t it is only deleted from the index if it is not assigned to any resource anymore after the operation. To check this, the tag index depends on the `OccurrenceChecker`. Similarly, the read access tags have to be checked one by one. For each read access tag t_r in the cooccurring tags it has to be checked whether there still exists another resource with tags t and t_r assigned. An example: If the tag "foo" was removed from resource #1 in Table 5.1, the read access token (1) would have to be removed from the "foo" entry in the tag index. Read access token (2) would remain since "foo" is still assigned to resource #2 which has token (2).

5.3.4. Efficient batch updates based on rules

For the management of implicit tag relations as well as for the implementation of TACKO it is essential that the tag assignments of many resources can be changed efficiently. We call such operations *batch updates*. For instance the creation of a subsumption relationship $t_1 \geq t_2$ requires that the tag t_1 is assigned to all resources being tagged with t_2. Similarly, the tags of many resources have to be changed when a tag is renamed.

However, even in the comparably small resource collections that we worked with during the development of the prototype, such changes often involved modifications of the tag assignments of thousands of resources. Since Tricia is based on a relational data store in which the tags of each resource are stored in a separate record, such batch updates can take several seconds. Additionally, the index has to be updated which entails an additional delay. Finally, changes to a Tricia entity trigger several change listeners — for instance to create a new entry in the version history. The total time required for modifying a single resource stored with

the Tricia persistence framework was more than 50ms on a machine with a 1.73GHz CPU, 4GB of RAM, and a solid state drive. It is obvious that this is impractical for our purposes and a naive implementation of batch updates is prohibitive. Even if Tricia was not used for the storage of tag assignments, it is clear that batch updates would entail a notable delay in very large resource sets if they were not performed asynchronously.

5.3.4.1. Overview

The problem is solved by adding an additional layer of abstraction between the storage of the actual tag assignments and the interface exposed to the clients of the tag repository. The idea is that instead of instantly changing all resources affected by an update, the update operation is temporarily stored as a so-called *rule*. Rules are transparently applied during subsequent access to the tag repository. This means that even if the effect of a rule has not been applied to the tag store yet, this is transparent for clients of the tag repository because rules are taken into account in the new abstraction layer. If the user assigns for instance the tag A to all resources being tagged B, a respective rule is saved. When the tags of a resource r with the tag B are read in a subsequent operation, the tag A is added before the tags of r are returned to the client. Thus, for the client the previous tag assignment appears to be effective.

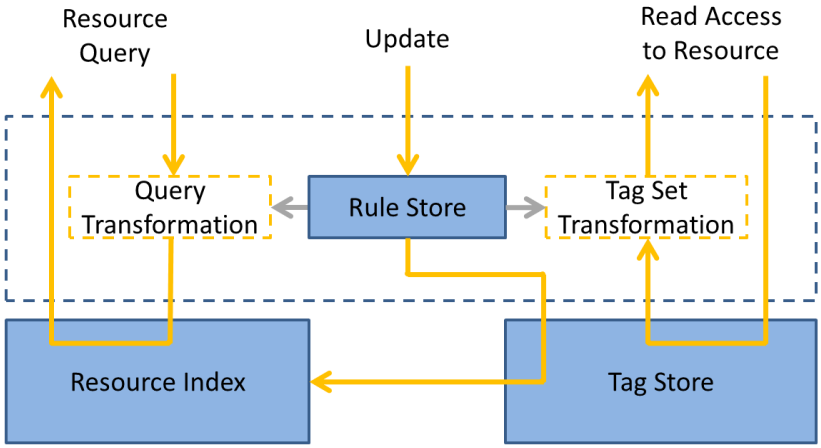


Figure 5.9.: Conceptual view of how rules are taken into account when the tag repository is accessed.

Figure 5.9 depicts how rules affect access to the contents of the tag repository. All update operations are saved as rules in the *rule store* (how rules are represented is covered below). The effects of rules are then gradually applied to the tag store which in turn instantly triggers an update of the resource index. These updates are performed in a background task that is run periodically. If a rule has been *materialized* completely, i.e., all affected resources have been updated in the tag store, it is removed from the rule store. Until then, a rule is taken into account whenever the resource index is queried and when the tags of individual resources are read: Queries are rewritten and a resource's tags are transparently changed before they are returned to the caller of the respective method. How the tag indexes are affected is covered separately in Section 5.3.5. The arrows indicate the flow of information. The gray arrows illustrate that both transformations depend on the rules in the rule store.

In our implementation of the rule store, all rules are always held in memory. This is reasonable since the number of rules is small compared to the number of resources and it significantly boosts performance of the tag repository. The rules are not persisted with the Tricia persistence framework although this would have been easy to implement. However, since most rules are materialized several seconds after each update, this was acceptable for our purposes. In case the system is shut down before all rules are materialized the respective updates are lost.

Each rule consists of two parts, the condition and the effect. The condition specifies which resources are affected by the update and the effect denotes which tag is to be added or removed. The effect is always the addition or removal of a single tag. If Γ is the set of all rules, a single rule $\gamma \in \Gamma$, has the form $[\kappa \rightarrow \delta]$. The condition κ is a formula in propositional logic with the atomic formulae or variables representing the tags and the resources, i.e., $T \cup R$. In the following, Arabic letters denominate tags. For a resource we write the symbol "#" followed by a natural number denoting the resource id. The effect δ is denominated by a tag with a prepended $+$ or $-$ depending on whether the tag is added or removed.

An example rule γ could look like this:

$$\gamma = [A \wedge B \wedge \neg\#1 \quad \rightarrow \quad +C]$$

To determine whether a specific resource matches a condition, the tag variables are replaced by the truth values *true* and *false* depending on whether the respective tag is assigned to the resource or not. All resource identifiers evaluate to *false* except for the one matching the very resource. Hence, the above rule applies to all resources tagged A and B except for resource #1. The effect of the rule is the addition of tag C. Note that updates affecting individual resources can be expressed with rules as well. In this case, the condition consists only of a single resource identifier.

We have to distinguish two kinds of rules that we call *external* and *internal* rules. An external rule represents a modification of the tag assignments as they are exposed to the tag repository's clients. These *external tag assignments* represent the *external state* of the tag repository. The client of the tag repository specifies an update with an external rule since the client only sees the external state. On the contrary, internal rules specify how the *internal tag assignments* being currently stored in the tag store — and representing the *internal state* — have to be altered in order to match the external state. The internal rules are stored in the rule store. They are also required to transform an *external query* posed by a client into an *internal query* that can be processed by the resource index. While the effect of both kinds of rules is always the addition or removal of an external tag, the condition of an external rule refers to external tag assignments and the internal rule to internal tag assignments respectively. To make this explicit in the notation of rules, we use lower-case letters for the tags in conditions of internal rules.

When the client of the tag repository performs an update, the respective external rule is transformed into an internal rule before it is saved in the rule store. This transformation is covered in Section 5.3.4.4.

We require that at any point in time no pair of contradictory (internal) rules exists in the rule store. We say two rules $\gamma_1 = [\kappa_1 \rightarrow \delta_1]$ and $\gamma_2 = [\kappa_2 \rightarrow \delta_2]$ are contradictory if one of them adds a tag t, the same tag t is removed by the other rule, and the formula $\kappa_1 \wedge \kappa_2$

is satisfiable. In other words, there can exist a resource r that matches both conditions. Whenever the external or internal state of the tag repository changes, it has to be precluded that such contradictions are introduced. This means an external update may not lead to a contradiction in the rule store (Section 5.3.4.4) and neither may a contradiction be introduced by the materialization of a rule (Section 5.3.4.5). How it is ensured that no contradictions are introduced is explained in the respective sections.

To maintain consistency of the tag repository, read access, external updates and internal changes required for the materialization of rules are synchronized. At any time at most one such operation is performed.

5.3.4.2. Accessing individual resources

To understand how the external tag assignments of an individual resource are determined by combining the internal state and the internal rules, an important difference with regard to the interpretation of internal and external rules is essential: While the order in which external rules are applied to the repository is of course important, the set of internal rules is not ordered and there is no transitivity. This means that if there are two internal rules $[a \rightarrow +B]$ and $[b \rightarrow +C]$ this does *not* imply that all resources with the internal tag a have the external tag C. The effect of a rule always refers to the external tag assignments.

The tags of an individual resource r are obtained by applying the effects of all matching rules in a single step. More precisely, after all internal rules that match the resource r have been determined, the effects of these rules are applied to its set of internal tag assignments at once. This is important since reciprocal effects have to be avoided. The rule effects are unambiguous since there are no contradictory rules. When the rules matching r are determined, it is not possible that two rules with contradictory effects apply. The combined effects of all matching rules result in the addition of certain tags and the removal of certain other tags. The sets of added and removed tags are disjoint. If the effect of a single rule cannot be applied, for instance because a tag that has to be added is already assigned, the rule is ignored.

We illustrate this with an example. We assume that the following rules are stored in the rule store:

$$\begin{aligned}\gamma_1 &= [\#3 \;\; \rightarrow \;\; +B] \\ \gamma_2 &= [a \wedge \neg\#3 \;\; \rightarrow \;\; -B] \\ \gamma_3 &= [c \;\; \rightarrow \;\; -D] \\ \gamma_4 &= [\neg d \;\; \rightarrow \;\; -F]\end{aligned}$$

Let further #3 denote the resource that is accessed by a client of the tag repository and let $\{a, c, d, f\}$ be the set of tags saved for resource #3 in the tag store, i.e., the internal tag assignments. Obviously the rules γ_1 and γ_3 match. The respective effects are the addition of B and the removal of D. Thus, the set $\{A, B, C, F\}$ is returned to the client of the tag repository. The last rule γ_4 does not match because the tag d was present before the rule effects were applied.

Note that a and A represent the same tag but that the lower-case letter refers to the internally stored tag while the upper-case letter expresses that the external state is meant. We write

the tags in the effects of rules in upper-case letters because they refer to externally triggered updates of the tag repository that have not been materialized in the tag store.

The tag repository offers methods to obtain an aggregate view of the tag assignments of a set of resources matching a certain query. One example is the set of distinct tags assigned to all resources matching a certain query. To obtain the result, the external tag assignments for each such resource are computed as described above.

5.3.4.3. Rewriting queries

When external queries are evaluated, they have to be transformed into internal queries before they are passed on to the resource index. Just as the condition of a rule, a query is represented by a formula in propositional logic with tags and resource identifiers as atomic formulae. The rules in the rule store are used to substitute the tags in the query by formulae specifying the resources to which the tags are assigned according to the external state of the tag repository.

Let for instance

$$\gamma_1 = [a \quad \rightarrow \quad +B]$$

be the only rule in the rule store. If now an external query $\sigma = B$ is to be evaluated (it retrieves all resources with tag B), it has to be first transformed into an internal query $\sigma' = b \vee a$.

In general, when an external query is transformed into an internal one, each tag in the query is replaced by a formula that evaluates to *true* for precisely the resources to which the tag is assigned according to the external state of the tag repository. For a tag t, we denominate this formula with $internal(t)$ and it is obtained as follows:

$$internal(t) \;=\; (t \;\vee \bigvee_{[\kappa \rightarrow \delta] \,\in\, \Gamma_{+t}} \kappa) \;\wedge\; \neg \bigvee_{[\kappa \rightarrow \delta] \,\in\, \Gamma_{-t}} \kappa$$

Thereby Γ_{+t} denotes the rules in the rule store that add tag t and Γ_{-t} denotes the set of rules removing t respectively. Apparently, the resulting formula evaluates to *true* if t is assigned to a resource or if the resource matches a condition of a rule adding t but only if it does not match a rule removing t. As stated above, it is not possible that contradictory rules are contained in the rule store. Note that it is only possible to transform a query this ways because the internal rules are considered as not being transitive.

If the external query is an atomic formula representing a single tag t, the resulting internal query is apparently correct: If there are no rules affecting t, the query is not changed at all because with regard to t the internal and the external state of the repository are identical. If there are rules adding t to some resources and the rules have not been materialized yet, the query is changed to match these resources as well. Similarly, the internal query will not match resources from which t is removed by a rule. The correctness of the transformation of a more complex query can be shown my means of structural induction on the structure of Boolean expressions.

5.3.4.4. Updating the tag repository

Similar to a query that is rewritten according to the current rules, an external rule, representing an update operation, needs to be transformed into an internal rule before it is saved in the rule store. In a rule $\gamma = [\kappa \to \delta]$, the condition κ specifies the set of resources to which the rule applies. As for a query, each tag t is replaced by $internal(t)$ in this formula.

Additionally, another small extension is added. Let $\bar{\kappa}$ denote the condition after the replacement. We transform it into a new condition $\kappa' = \bar{\kappa} \wedge \neg x_\gamma$. We call x_γ a *rule tag*. Its namespace is a special *rule namespace* and there is an individual tag in this namespace for each rule. The rule namespace is used internally and its tags are not exposed to the clients of the tag repository. A rule tag can be assigned to a resource to prevent a particular rule from matching this resource. This can be required when new resources are added to the tag repository (see below) and when rules are materialized (see Section 5.3.4.5).

Finally, modifications have to be made to the existing rules in the rule store in order to prevent contradictory rules. It has to be ensured that no resource being matched by the condition of the new rule is matched by conditions of rules having the opposite effect. Therefore, the conditions of the existing rules are modified in a way that the new rule takes precedence over the existing rules. More precisely: The effect of the new rule takes precedence over the effects of the existing internal rules representing previous updates. To achieve this, the following steps are followed whenever a new external rule $\gamma = [\kappa \to \delta]$ is applied to the tag repository in an update operation.

1. Condition κ is transformed to $\bar{\kappa}$ by replacing each tag t occurring in κ by the respective formula $internal(t)$.
2. The formula $\bar{\kappa}$ is transformed to $\kappa' = \bar{\kappa} \wedge \neg x_\gamma$ and we obtain a new rule $\gamma' = [\kappa' \to \delta]$.
3. If δ is of the form $+t$, transform each existing rule $\hat{\gamma} = [\hat{\kappa} \to \hat{\delta}]$ with $\hat{\delta} = -t$ into a new rule $\hat{\gamma}' = [\hat{\kappa}' \to \hat{\delta}]$ where $\hat{\kappa}' = \hat{\kappa} \wedge \neg\kappa'$. If δ is of the form $-t$, apply the same transformation to all rules $\hat{\gamma} = [\hat{\kappa} \to \hat{\delta}]$ with $\hat{\delta} = +t$ respectively. This way it is ensured that the new rule γ' takes precedence over the existing rules. Further, γ' clearly does not contradict any of the transformed $\hat{\gamma}'$ since
$$\kappa' \wedge \hat{\kappa}' = \kappa' \wedge \hat{\kappa} \wedge \neg\kappa'$$
which is obviously not satisfiable.
4. The transformed rule γ' is added to the rule store.

A simple example shall illustrate this process. Let
$$\gamma_1 = [a \wedge \neg x_{\gamma_1} \quad \to \quad -C]$$
be the only rule that is currently stored in the rule store. Let further
$$\gamma_2 = [A \wedge B \quad \to \quad +C]$$
be a new rule that is added to the rule store. The first two steps are trivial because there are no rules affecting tags A or B: The rule γ_2 is transformed into an internal rule ${\gamma_2}'$ with
$${\gamma_2}' = [a \wedge b \wedge \neg x_{\gamma_2} \quad \to \quad +C].$$

Since the new rule γ_2 adds tag C and γ_1 removes C, the existing rule γ_1 has to be transformed to ${\gamma_1}'$ with

$$ {\gamma_1}' = [a \wedge \neg x_{\gamma_1} \wedge \neg(a \wedge b \wedge \neg x_{\gamma_2}) \quad \rightarrow \quad -C] $$

The same steps are followed when the tags of individual resources are changed, for instance with a rule like $[\#1 \rightarrow +A]$. If there is an existing rule that removes A from a certain set of resources, it is obvious that an exception has to be made for resource #1. This means that resource #1 is not matched by the condition of the existing rule.

Finally, resources can be added or removed. To remove a resource it is sufficient to delete it from the tag store and from the resource index. However, when a resource is added, the addition of a new entry in the tag store and in the resource index is not sufficient. Although the new resource has no tags yet, it is possible that it is affected by an existing rule. In this case the respective rule tag is automatically assigned, so the rule does not match anymore. Let for instance $\gamma_1 = [\neg a \wedge \neg x_{\gamma_1} \quad \rightarrow \quad +B]$ be an internal rule in the rule store. To prevent that a new resource is affected by this rule, the resource is assigned the rule tag x_{γ_1}.

5.3.4.5. Materialization of rules

The materialization of rules is done asynchronously in a background task. Thereby, rules are applied to only one resource at a time[8], so the rule materialization and further updates as well as read access to the tag repository can be interleaved.

In each materialization step, a random rule $\gamma = [\kappa \rightarrow \delta]$ is selected from the rule store. The resource index is used to find a resource to which the effect δ has not yet been applied. If the rule adds for instance a tag t, the respective resources can be found with the internal query $\kappa \wedge \neg t$. If there are no such resources, γ is removed from the rule store.

If resources are found, a random resource r is selected from these resources and the effect of γ is applied to r. Additionally, if there are other rules in the rule store matching r, they are applied as well. In consequence, the set of tags saved for r in the tag store reflects exactly the external state.

However, it is possible that after the rules matching r in the initial state have been applied there are other rules matching r in the updated state. Therefore, the rule tags corresponding to these matching rules are assigned to r as well. Eventually, r is not matched by any rule and the internal tag assignments of r are equal to the external tag assignments — apart from the rule tags.

A short example illustrates when it is necessary to assign additional rule tags: We assume that the rule $\gamma_1 = [A \rightarrow +B]$ has been applied to the tag repository. Subsequently, another rule $\gamma_2 = [C \rightarrow +A]$ is applied as well. Both rules are not materialized yet and are saved in the rule store as ${\gamma_1}' = [a \wedge \neg x_{\gamma_1} \rightarrow +B]$ and ${\gamma_2}' = [c \wedge \neg x_{\gamma_2} \rightarrow +A]$. A resource r with tag set $\{c\}$ would only be matched by ${\gamma_2}'$. However after the materialization of ${\gamma_2}'$, the new tag set $\{a, c\}$ would match ${\gamma_1}'$ which is wrong because γ_1 was applied first and did not affect

[8] Several resources can be updated in one step but for the sake of simplicity we assume that only one resource is changed.

resource r. The problem is solved by adding the rule tag of γ_1 to the tag set. Eventually, the internal tags of r are set to $\{a, c, x_{\gamma_1}\}$.

Finally, when a rule γ is removed because it has been completely materialized, the respective rule tags in the condition of γ can be removed from all resources unless they still occur in the conditions of other rules. If they do not occur anymore in rules, a new rule $[x \to -x]$ can be added to the rule store for each such tag x.

Generally, the materialization of rules is accomplish by repeating the following steps:

1. Select an arbitrary rule $\gamma = [\kappa \to \delta]$ from the rule store.
2. Find a resource r that matches κ but to which the rule has not been applied yet. If such a resource is found, proceed with step 4, otherwise proceed with step 3.
3. For each rule tag x in κ add the rule $[x \to -x]$ to the rule store, but only if x is assigned to at least one resource(the resource index has to be queried) and x is not contained in the condition of any other rule. Remove γ from the rule store. Skip steps 4 and 5.
4. Determine the set of all rules matching r and apply their effects to the set of tags saved for r in the tag store.
5. Repeat as long as there is at least one rule γ^* matching r: Assign the respective tag x_{γ^*} to r.

The described procedure does not change the external state of the tag repository: The resource that has been modified is not affected by rules anymore and reflects exactly the external state. Additionally, the changes in the rule store are limited to the removal of materialized rules — not having an effect anyway — and the addition of rules that affect only rule tags that are not externally visible.

5.3.5. Updating the tag indexes

The tag index play a special role because in contrast to the resource index, they are instantly updated whenever the external state of the tag repository changes. As described in Section 5.3.3, an update of the tag index is triggered whenever a tag is added or removed. This does not only apply to single tag assignments but also to batch updates. However, an update of the tag index is based on the set of cooccurring tags, i.e., the other tags that are assigned to the affected resource, or, in the case of batch updates, the affected resources. Due to the way a tag index works, it is not required that individual resources are distinguished but it is possible to just pass the union of the tags of all affected resources to the tag index in case of batch updates.

We assume for instance that a tag t was removed in a batch update and the set of cooccurring tags is $\{a, b, x\}$. Let further be x a tag in the read access namespace. Then the fact that t was removed from *any* resource with tag x requires to check whether there are further cooccurrences of t and x and to update the index accordingly if this is not the case.

5.4. Implementation of the TACKO organization structure

In this section, we cover how the TACKO organization structure is realized in our prototype system. In Section 5.4.1, we first cover which data structures are implemented and how they relate to the abstract structure covered in Section 4.1. Subsequently, in Section 5.4.2, we describe how updates of the structure are performed and how the state of the tag repository and the separately stored tag relations are kept consistent.

The following description is simplified to some extent with regard to the actual implementation of our prototype. Since the final data structure was not known during the development, part of the implementation is more generic than necessary. For the sake of clarity, we omit irrelevant generalizations and peculiarities being specific to Tricia. In particular, we assume that entities are neither assigned to a workspace, nor do they have content types, type tags, or other attributes since all these properties can be represented as tags with appropriate namespaces.

5.4.1. TACKO data structures

A conceptual model of the central classes underlying the implementation of TACKO is shown in Figure 5.10. The lower half of the figure shows the classes that are required to represent the TACKO tag relations, i.e., subsumption relations and facets. The classes in the upper half determine how users can access the tagging system.

The `Path` consists of a sequence of `Filter`s that the user has selected to narrow down the set of resources. It corresponds to the filter region in the user interface (see Figure 5.11). Being a conjunction of primitive filters, also referred to as the path segments, the path can be considered a compound filter itself. However, the name was chosen to indicate that the order of the primitive filters reflects in which order they were selected. Nevertheless, a path does not necessarily represent an exact trace of navigation steps since the user can remove filters. Note that it is even possible to remove a filter from the middle or the beginning of the path.

Each primitive filter is either a simple `TagFilter` or a `NoneOfTheseFilter`. The name of the latter corresponds to the "none of these"-option displayed at the bottom of each facet. A tag filter matches all resources with the corresponding tag. A "none of these"-filter refers to a set of tag filters and matches all resources to which none of the respective tags is assigned.

Whenever a tag is entered or selected by the user, a new tag filter is appended to the path. Similarly, a new filter is added when a "none of these"-option is selected from a facet. Such filters are displayed in red color in the user interface (cf. Figure 5.11). Since the path is interpreted as a conjunction of filters, each new filter further narrows down the set of resources matching the path.

A `Context` represents a point of reference for tag relations. In combination with the class `ContextConfiguration` it is — among other things — used to determine the set of visible facets for a given path. A context can be considered an anchor for a context configuration which in turn contains the facets and the subsumption relationships. According to the data structure described in Section 4.1, facets cannot be defined in the context of a filter containing

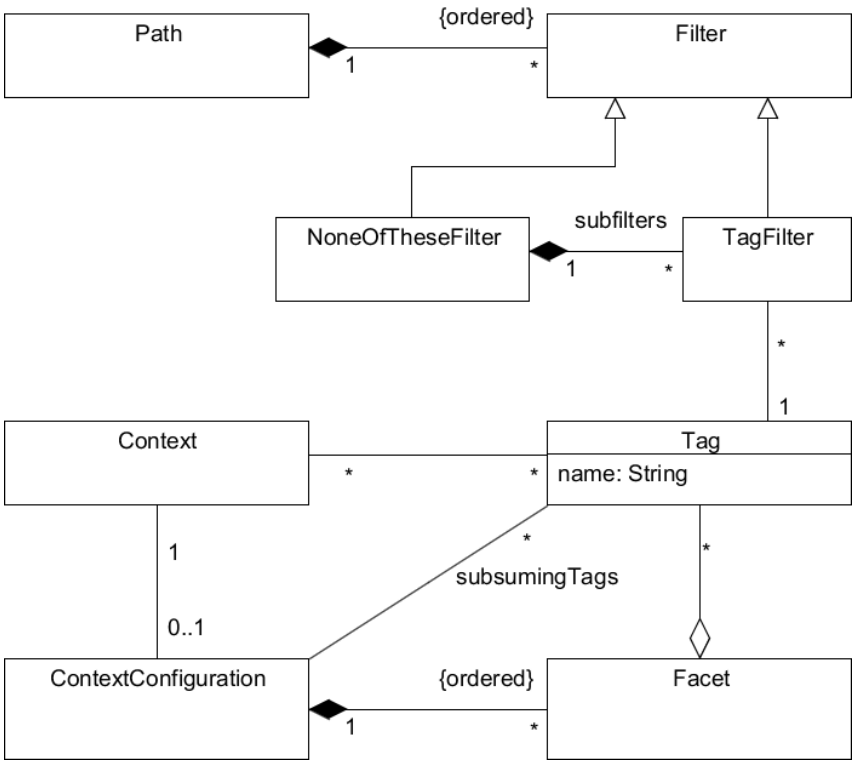


Figure 5.10.: Conceptual UML diagram showing the fundamental classes of the TACKO implementation.

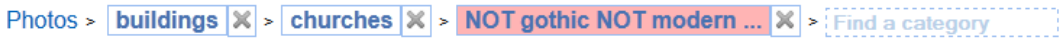


Figure 5.11.: Representation of a `Path` in the user interface.

negative tags[9]. This is reflected in the `Context` class since it is only defined by a set of normal tags that cannot be negated. For each path there is a corresponding context that is obtained by combining the tags of the tag filters in the path to a set. The filter order and the "none of these"-filters are ignored. Apparently, different paths can map to the same context. This context is then used to determine the set of visible facets.

We illustrate this with an example. We assume the tagging system contains the following facets:

$$\begin{aligned} \{\} &\vdash \text{Facet}(\{\text{“buildings”}, \text{“colors”}\}) \\ \{\text{“buildings”}\} &\vdash \text{Facet}(\{\text{“churches”}, \text{“skyscrapers”}\}) \\ \{\text{“colors”}\} &\vdash \text{Facet}(\{\text{“green”}, \text{“red”}\}) \end{aligned}$$

Each facet is represented in a context configuration and all context configurations have different contexts. We assume further that a user navigated to the following path by selecting

[9] In this case the term *filter* does not refer to the `Filter` class but to a set of tags as according to the formalism used in Section 4.1.

appropriate filters:

[“colors”][NOT “green”, NOT “red”][“buildings”][“skyscrapers”].

Each pair of square brackets denotes a filter.

For the path in the example we obtain the following context by omitting the “none of these” filter:

{“colors”, “buildings”, “skyscrapers”}.

To determine the set of facets for a given path p the following steps are required:

1. Determine the context c corresponding to path p.
2. Find all context configurations with contexts being equal to or more general than c. A context c_1 is more general than another context c_2 if all tags in c_1 are part of c_2 as well (cf. the subset relation on filters in Section 4.1).
3. Select all facets from these context configurations.

When the system is in browsing mode, facets are hidden if they contain a tag that belongs to c.

In our example scenario, the facet {“green”, “red”}) defined in context {“color”}) is displayed. However, the other two facet are hidden because both contain tags that have already been selected and are part of the context corresponding to the current path.

`ContextConfiguration` is the only class being implemented as a persistent Tricia entity. For each `ContextConfiguration`, the respective context, the facets, and the subsumptions are serialized in fields of this entity. The `Context` functions as a key of the `ContextConfiguration`. Subsumptions are expressed via the `subsumingTags` field of a context configuration. When a tag t_{sub} is subsumed by another tag t_{super}, the context configuration for the context $\{t_{sub}\}$ contains a reference to t_{super} in this field.

5.4.2. Handling modifications

In the following, we will describe important issues regarding the modification of the organization structure and changes in the tag assignments. We will limit this description to aspects being either not obvious or critical for maintaining consistency. The latter refers primarily to the state of the tag repository and the separately stored subsumption relations.

5.4.2.1. Modifying facets

The implementation of most operations for the modification of facets is straightforward. Tags can be added or removed simply by updating the `ContextConfiguration` containing the respective facet. Complete facets can be deleted as well.

With regard to the creation of facets, the simplest solution would be to save them in the context corresponding to the current path. If we assume for instance that the user navigated

again to the path

[“colors”][NOT “green”, NOT “red”][“buildings”][“skyscrapers”]

in our example of Section 5.4.1, a new facet would be added to the `ContextConfiguration` for the context {“colors”, “buildings”, “skyscrapers”}.

However, during our work with the prototype we found that another solution is preferable for the management of facets. We assume that there is one subsumption among the three tags in our example context: “buildings” $\geq$ “skyscrapers”. Rather than creating a facet in the context containing all three tags, it is more likely that one either wants to create a new facet in the context {“colors”} or in {“buildings”, “skyscrapers”}. Additionally, it turned out that combining unrelated tags in a facet context results in a facet that is hard to find. To find all facets in the organization structure, one would have to try all possible combinations of tag filters since the respective contexts might contain facets.

For these reasons, the user is offered several context options when facets are created. They are obtained by

1. selecting only the most specific tags from the current context, i.e., the tags not subsuming other tags,
2. creating a context for each such tag, and
3. adding all tags subsuming the respective tag.

In our example this results in the two context options {“colors”} and {“skyscrapers”} after step 2. The tag “buildings” is added to the latter in step 3. Figure 5.12 shows how the two options from the example are displayed to the user[10]. For each option, only the most specific tag is displayed.

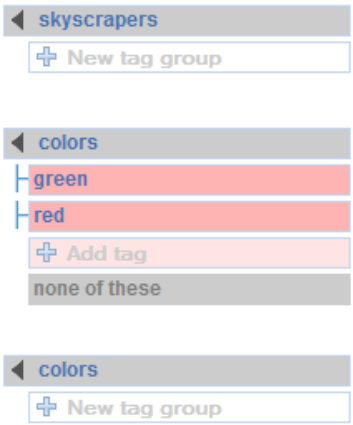


Figure 5.12.: Two different options for the creation of a new facet.

[10] As explained in Section 4.3.4 we use the term *tag group* in the user interface since users are generally not familiar with the term *facet*.

5.4.2.2. Adding and removing tags

In contrast to facets, subsumptions are not independent of the actual tag assignments. A subsumption $t_{super} \geq t_{sub}$ implies that all resources being tagged t_{sub} are tagged t_{super} as well. Hence, when a new tag is added to one or more resources, it has to be checked whether the tag is subsumed by any other tags. If this is the case, these tags are assigned as well.

Similarly, when a tag t is removed from one or more resources, it has to be ensured that none of the tags subsumed by t are assigned to any of the affected resources. In case such tags are found, they are not removed but instead the respective subsumption relations are voided.

However, it has to be stressed that this only happens if the state of the tag repository would otherwise contradict the subsumption relations. The complete organization structure consisting of facets and subsumptions can well exist even without any tag assignments and resources.

5.4.2.3. Adding and removing subsumptions

The removal of a subsumption relationship entails no changes in the tag repository. If the tag assignments were valid before, they will be valid after the removal as well.

On the contrary, when a new subsumption relationship is established, this may require to assign additional tags. As stated above, a subsumption $t_{super} \geq t_{sub}$ requires that the tag t_{super} is assigned to all resources being tagged t_{sub}. Please note that thereby the transitivity of subsumptions is also taken into account. The user is prompted with a dialog asking to confirm the additional tag assignments before the respective subsumption relation is established (cf. Section 4.3.4.2).

5.4.2.4. Renaming tags

Finally, renaming a tag is an operation that can result in non-trivial changes of the organization structure. With regard to the tag repository, renaming a tag t_1 to t_2 is treated as the removal of t_1 and the subsequent assignment of t_2 to all the resources to which t_1 was assigned before. However, the first complication arises if t_2 is subsumed by other tags. Then, additional tag assignments may be required (cf. Section 5.4.2.2).

In the organization structure, the subsumptions involving t_1 have to be adapted: All tags that were previously subsumed by t_1 are now subsumed by t_2. If a tag subsumed t_1, it now subsumes t_2 accordingly.

Facets can also be affected by a rename operation. In the simplest case, the tag t_2 never occurs in a facet together with t_1. Then, all occurrences of t_1 can be exchanged for t_2. If both tags do occur in the same facet, t_1 is deleted.

Finally, t_1 may occur in a context c_1 for which a context configuration is saved. Again, if t_2 is not part of the same context, t_1 may simply be replaced in c_1. Otherwise it is removed from c_1 as it is removed from a facet if the replacement would result in a duplicate tag. However,

after the modification of the context c_1 it may be equal to another context c_2. If this is the case, the respective context configurations have to be merged.

Let for instance $c_1 = \{$“entertainment”,“movies”$\}$ and $c_2 = \{$“entertainment”,“films”$\}$. When now the tag “movies” is renamed to “films”, both contexts are identical and the facets defined in the respective context configurations have to be merged.

CHAPTER 6

Evaluation

This chapter covers the evaluation of the TACKO user interface and the TACKO organization structure. The evaluation can be separated into two phases: In the first phase, a usability evaluation was conducted based on a late version of the developed prototype. In the second phase, quantitative measures were obtained to assess in how far the application of TACKO can improve the access to information.

The development of a novel system that employs new and uncommon organization principles bears the risk that users do not understand how the system is used or that the user interface is deficient from an ergonomic point of view. The goal of the usability evaluation was to find usability issues and to assess whether users are able to apply TACKO for accessing and organizing information with the present user interface. Since the user interface was still developed further in parallel to the tests, it was possible to get early feedback on modifications and to gain valuable insights into how TACKO is perceived by users. The results are summarized in Section 6.1.

Subsequently, it was attempted to quantitatively measure the benefits of TACKO. In a large-scale experiment, more than 150 participants, that were recruited via a web-based crowdsourcing service, performed two different kinds of tasks in a collection of several thousand photos. On the one hand it was examined to what extent the organization of photos with TACKO improved the performance measures. On the other hand it could be assessed whether the application of the faceted TACKO user interface had an effect apart from the better indexing quality. The complete experiment is covered in Section 6.2.

Section 6.3 concludes with a discussion of the results of both phases of the evaluation.

6.1. Usability evaluation

The work presented in this section was conducted with the support of the student Joan Boixadós who also covered the findings presented here in much more detail in his master's thesis [Bo12] which will be referred to several times below. Before we present a summary of the most important results of the evaluation in Section 6.1.3, a brief introduction on the topics usability and usability evaluation is given in Section 6.1.1. Based on these fundamentals, the evaluation method was chosen. Section 6.1.2 covers which methods were applied and how the evaluation was conducted.

6.1.1. Fundamentals of usability evaluation

The International Organization for Standardization (ISO) defines usability in the ISO 9126 standard for product quality in software engineering: "A set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users"[IS01].

The concept of usability is commonly broken down into several characteristics or attributes. In the following, we refer to the broadly accepted decomposition of the concept which was proposed by Nielsen [Ni93]. Nielsen describes usability as a combination of five attributes:

- **Learnability** refers to the speed and success of new users learning to work with the user interface.
- **Efficiency** denotes the level of productivity that users can achieve when working with the interface.
- **Memorability** refers to the extent to which users remember how the interface works after a period of time in which it was not used.
- **Errors** are actions that do not have the effect intended by the user. An error rate can be calculated by dividing the number of errors by the total number of actions.
- **Satisfaction** stands for the user's attitude towards the system, e.g., whether it is perceived as being pleasant to use or helpful.

There are trade-offs between these attributes. For instance can the attempt to lower the error rate impede other attributes, such as satisfaction or efficiency.

Measuring these attributes and finding problems in the user interface that impede the usability in terms of these attributes is the goal of usability evaluation. There are basically two classes of usability evaluation methods: *Inspection methods* and *test methods* (cf. [Ni95] and [Ni93]). The former involve usability experts assessing a user interface or even an early prototype thereof. In contrast, the latter category contains methods that require real users and a working system.

In the following paragraphs, we will very briefly characterize different methods that can be applied to evaluate an existing user interface and that are based on real users. We provide no further details on inspection methods because they are of minor relevance for this work. Thereby, giving a comprehensive introduction on the topic of usability evaluation is not our

goal. Instead, the brief summary of the methods is meant to serve as a basic frame of reference for the evaluation of TACKO that is covered in Section 6.1.2. For a more detailed description of the methods, we refer to the work of Nielsen [Ni93].

The central characteristic of *usability testing* is that real users perform tasks with a working prototype or the final system. Thereby they are observed by the evaluator. This allows to discover problems users have in understanding and applying the interface. Additionally, measures such as the time needed to complete a task or the number of actions can be collected. The sessions can optionally be recorded. Usability testing is time consuming and requires unbiased users that are not familiar with the system under consideration. It is important to establish a setting in which users apply the interface naturally and are not intimidated by the fact that they are observed.

A variation of usability testing is the *thinking aloud* method. The evaluator asks the participating user to continuously speak about what he or she is thinking while using the interface. It makes it easier to locate the sources of errors than when the users are interviewed later when the task is already completed and important details were possibly already forgotten. The method is very flexible, simple, and requires no additional equipment. However, it should be combined with other usability methods [Ni12].

Not literally a *test* method, but also an approach requiring real users and a working system is *observation* in the field. Users are observed during their daily work in their natural work environment. The method is therefore less intrusive than methods performed in a usability lab. The observer may interrupt the users when it is required to understand a specific action but these interruptions should be rare.

Another way to gather data about the actual usage of a system in practice is *logging*. It requires that the user interface or other parts of the system are instrumented to record all actions users perform. This way it can be cheaply applied to a large number of users. It allows to quantify precisely how often certain problems occur given that a problem can be identified in the log files. However, it is not easily possible to find out the causes of problems but it can be a valuable complement for other methods.

Interviews and questionnaires can be used to collect the subjective opinions about a user interface. In contrast to objective measures, these methods can be used for instance to find out which features of the system a user particularly likes or dislikes [Ni93]. Questionnaires can be answered by a large number of users, while interviews are more time consuming and therefore the number of users is limited. However, the interview questions can be flexibly adapted to unexpected answers and discovered problems, so interviews are better suited for exploratory evaluations where the exact questions are not known in advance (cf. [Ni93]). Finally, *user feedback* can be collected by allowing users to submit issues to the developers by means of a special functionality integrated into the system.

6.1.2. Execution of the evaluation

Since we tried to adhere to common usability guidelines during the design of TACKO (cf. Section 3.2.2), the value of inspection methods involving usability experts was rather limited compared to methods allowing to get direct feedback from real users. For the evaluation

of TACKO it was decided to apply a combination of usability testing — including thinking aloud — and interviews. On the one hand, this offered the possibility to observe users how the apply TACKO in order to find usability problems. On the other hand, the causes for the problems could be clarified further in the interviews. Additionally, users could be asked for their subjective impression and suggestions for enhancements.

Questionnaires and log analysis were not applicable because not enough users were already applying TACKO in practice. However, a log analysis was done to some extent during the experiments covered later in Section 6.2.

The usability evaluation involved 16 participants aged between 24 and 30. Among all participants there was one woman. All were very familiar with computer work and had an understanding of the concept of tagging. Seven participants stated to use tagging on a weekly basis. However, only a minority of 4 participants had accounts on either Flickr or delicious [Bo12].

The participants were divided into four groups:

- Participants in group 1 were given no background information of the system. They knew that they would use a tagging system, but neither facets nor subsumption relations were explained. These relations were even removed from the dataset so they could not infer their usage from examples.
- Participants assigned to group 2 also did not get any instructions on facets or subsumptions but they could see example data where the concepts were applied.
- Group 3 was explained the concept of faceted organization and the ideas underlying TACKO. However, the dataset did only contain tags without relations.
- Group 4 did get instructions and a dataset in which the TACKO organization concepts were applied.

The users could participate from home by using the desktop sharing software *TeamViewer*[1]. They connected to the PC of the evaluator which was running an instance of the Tricia platform. This allowed the participants to work in an environment where they felt comfortable but also allowed the evaluator to closely observe all user actions. The voice services *Skype*[2] and *GoogleTalk*[3] were used to talk to the participants during the sessions.

The dataset was mostly based on a collection of several hundred Flickr photos. However, some participants could work with their own delicious bookmarks. The Flickr photos had been prepared by the evaluator and had been tagged and organized extensively so that the strengths of TACKO could be utilized (cf. [Bo12]).

At the beginning of a session the participants had about 10 minutes to get familiar with the user interface without accomplishing a specific task. Subsequently a brief interview was conducted. The questions depended on the user group since not all users had the same information about the system. The uninformed users were subsequently guided through the interface before the actual tasks started.

The participants had to complete tasks of different kinds:

[1] http://www.teamviewer.com, (accessed 15th of January 2013).
[2] http://www.skype.com, (accessed 15th of January 2013).
[3] http://www.google.com/talk/, (accessed 15th of January 2013).

- searching for certain resources,
- organizing resources in certain ways,
- or exploring the collection, e.g., to find their favorite photo.

The tasks were conducted by applying the *thinking aloud* method so users were asked to verbalize their thoughts. Depending on the available time, different aspects of the interface were discussed or users were shown more advanced examples of applications of TACKO.

Since some users participated several times — using different versions of the interface — 23 sessions were conducted in total. Figure 6.1 shows the version of the interface being closest to the final version that is covered in Section 4.3.

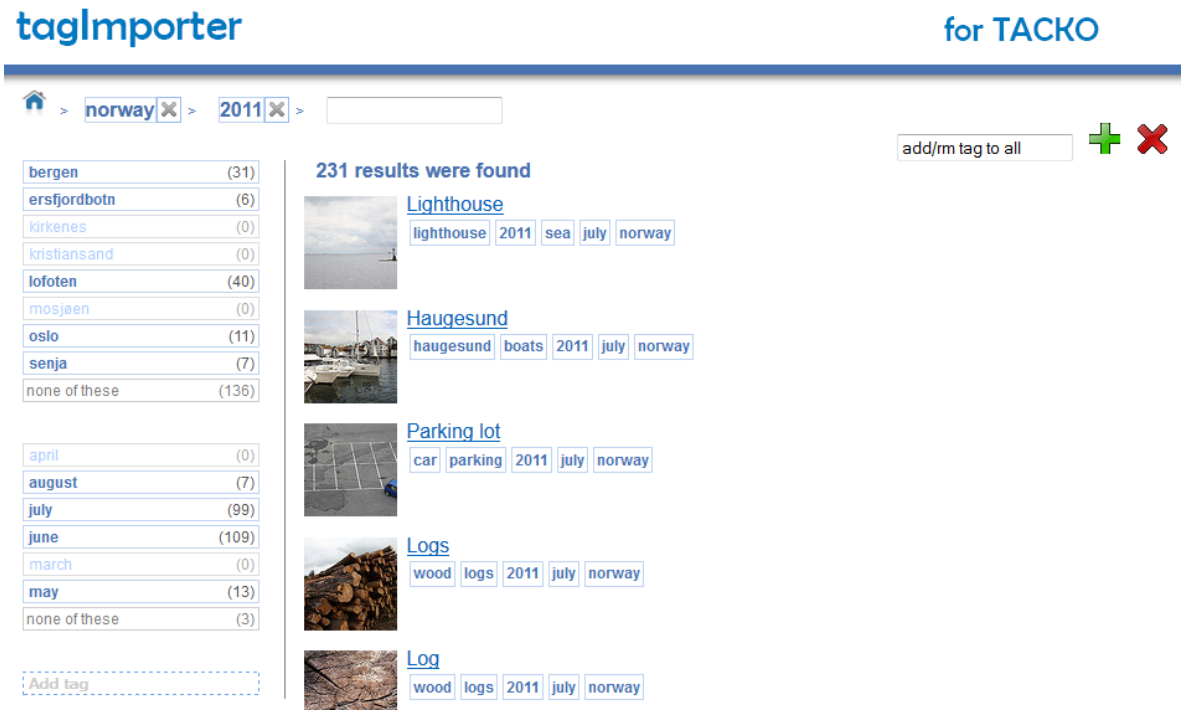


Figure 6.1.: TACKO user interface used for the usability evaluation (cf. [Bo12]).

Since our focus is on the findings of the work in this thesis, we refer to [Bo12] for more details on the planning and execution of the evaluation.

6.1.3. Results and findings

We first cover some general observations and then present a list of specific usability issues in Section 6.1.3.2. Again, we refer to [Bo12] for a more detailed report.

6.1.3.1. General observations

A central finding of the evaluation was that the amount of information users get before they use the interface for the first time is critical for their understanding. Additionally, it is essential that users are given examples of how the different kinds of tag relations are applied. Even the users in group 3, who actually knew that facets can be created, had problems applying them without seeing an example before. On the other hand, particularly in the case of facets, a good example helps users to understand the concepts behind TACKO, although it is difficult for users to explain them. Providing two example facets with at least three tags was mostly sufficient to give users a basic idea of facets and how they are applied.

It was also found that it is essential for the recognition of a particular facet that it does not change too much between two navigation steps. In previous versions of the TACKO system, a tag was hidden from a facet if there were no respective resources in the current result set, so the same facet changed its appearance in different contexts. After the usability evaluation it was obvious that this is very irritating for users and that it therefor had to be changed. We decided to always display the whole list of tags in a facet even though it is often much longer than necessary. Confusion particularly arose when users changed the facets themselves. It was often difficult for them to predict the exact effects of their changes since the facets changed from context to context in a seemingly unpredictable way. To make clear how the facets displayed in different contexts relate to each other, facets were animated during a navigation step as a consequence of this evaluation.

Understanding subsumption relations was even more difficult since they did not have a visual representation comparable to that of facets. This can be seen when the interface in Figure 6.1 is compared to the final interface covered in Section 4.3 (e.g., Figure 4.8). In the final version there is a facet header that is represented by a tag that usually subsumes most of the facet tags. Additionally, an icon was added to each tag indicating whether the respective tag is subsumed by the tag in the facet header or not.

It has to be acknowledged that users could not take advantage of the full flexibility and power of the TACKO organization structure without previous instructions. However, most users could accomplish to navigate through the collection without help as long as they did not have to change the organization themselves.

Nevertheless, the usability evaluation can be considered very successful insofar as many important usability issues were revealed. This led to a significant improvement of the final version of the prototype that was used in the experiments measuring the effect of TACKO on the performance of users working with the system (cf. Section 6.2).

6.1.3.2. Specific usability issues

In total, more than 40 usability issues were found in the usability study [Bo12]. In the following, we present a list of the most important issues being revealed. Many of the found issues are not relevant to the latest version of TACKO anymore because respective improvements were made. Such issues are not included in the following list:

- It is too difficult to undo some operations, e.g., creating a subsumption relation. The system should provide a generic undo mechanism.
- Some users were irritated by the fact that the same tag can be part of several facets.
- Users did not recognize that they can perform drag & drop operations with tags. If they knew it, it was not clear where they could drop tags.
- The effects of drag & drop operations are not consistent. Dropping a tag onto a resource leads to the assignment of the tag to the resource. Dropping it onto another tag creates a subsumption relationship and possibly changes the tags of several resources as a side-effect.
- Users did not know that they can right-click on a resource.
- Users complained that it is not possible to assign a tag to several resources in one step unless they already have a common tag, so that the filter can be changed accordingly. It should be possible to select an arbitrary set of resources as the target of an action.
- Many users requested that one should be able to negate individual tags in the filter. Currently it is only possible to select the "none of these" option to view all resources not being assigned any of *all* tags of a facet.
- Many users misunderstood the tag filter for a search bar. However, after the first try, most users understood that they could only enter tags here.
- The mix of different kinds of relations among tags in the filter is confusing. Some have a subsumption relation and some are independent.
- When assigning a tag to a resource, the automatic assignment of more general tags — according to the subsumption relations — is sometimes not noticed. In other cases it was noticed but the users were confused.
- The autocomplete functionality can be improved by prioritizing the tag suggestions. For instance, in the filter the system should only offer tags being present in the current results set because otherwise users obtain an empty result.
- Some users wanted to change the order of tags in a facet. Currently the tags are always ordered alphabetically.
- The order of resources should be deterministic. For the experiments in Section 6.2, we applied a random order of all resources that was consistent in all contexts.
- The URLs are perceived as cryptic and not helpful.

Despite all these issues there were user interface elements that were well understood and easy to apply. For instance, the users had no problems to understand the basic functionality of the filter. Although its position — i.e., above the facets spanning the whole width of the interface — was initially confusing for some users, it was observed that it helped to understand that the selection of facets depends on the context.

Furthermore, there were hardly any problems related to the controls for making changes to the tag assignments of the resources in the current result set. It was clear that only the resources in the current scope are affected.

6.2. Experimental evaluation of utility

In addition to the qualitative evaluation of the user interface, TACKO was quantitatively evaluated in a controlled experiment. The experiment was conducted to assess in how far the accessibility of contents of tagging systems is actually improved through the application of TACKO. After defining the goal of the experiment, an extensive description of the experiment design and the experimental setup is given. Finally, the results are summarized and interpreted.

6.2.1. Objective of the experiment

In accordance with the guidelines proposed by Montgomery [Mo08], we start with the problem statement and a definition of the goal of the experiment.

The abstract objective to assess whether TACKO can improve the accessibility of collections of tagged items can be broken down into two subgoals:

- Assess whether the TACKO user interface can be used to improve the indexing quality and the organization of a given collection of tagged items in such a way that users can perform tasks in the collection with a higher success rate and more efficiently than without the retroactively performed reorganization.
- Find out whether the accessibility of such a collection increases when users are able to exploit the TACKO tag relations, i.e., subsumption and facets, by using the TACKO user interface.

As already indicated in the definition of the first subgoal, the vague term *accessibility* has to be defined more clearly. For our purpose, it means the suitability of a collection for performing search or exploration tasks with a specific user interface. This means that the object of investigation is always the combination of a collection of items with a particular user interface. The suitability of this combination can be assessed by comparing the efficiency and the success rates of tasks performed by the participants. A detailed description of the tasks is given in Section 6.2.5.

Finally, we also conduct the experiment to collect detailed usage data that can be analyzed in order to better understand how users use the system as well as to find further usability problems.

6.2.2. Experiment design

The experiment is based on a collection of several thousand photos that were obtained from the photo sharing website Flickr. This raw collection was organized by the experimenter using the TACKO system to obtain a modified collection that contains more accurate tag assignments and tag relations that allow to access the contents of the collection in a structured manner. Details about the collection and how it was preprocessed are given in Section 6.2.4.

As stated in Section 6.2.1, our aim is not only to examine the influence of the availability of a more structured user interface on the task performance but also whether an increase in

the indexing quality alone has a positive effect. Ideally, we would construct a 2^2 factorial experiment and vary both factors, the user interface and the indexing quality of the collection (i.e., organized collection compared to raw collection). However, during the organization of the collection, subsumption relationships are established that have implications on the tag assignments. This means that it is not possible to separate the application of the TACKO structure from the indexing quality. In consequence, we obtain only three factor combinations, that we refer to as *configurations* in the following:

- **Configuration A-RAW:** There are no tag relations and the tags of each photo are exactly as they were obtained from Flickr.
- **Configuration B-ORGANIZED:** The tags have been organized using the TACKO system but the structure is not reflected in the user interface, i.e., the tags have changed but facets and subsumptions cannot be exploited.
- **Configuration C-TACKO:** The organized collection can be browsed with the TACKO user interface.

For the sake of brevity we often refer to the configurations only with the letters A, B and C respectively.

The participants of the experiment perform two different kinds of tasks with these configurations: Searching for a specific photo and selecting a small set of photos based on certain instructions. For a search task, it is determined whether it was completed successfully and for both kinds of tasks the time until completion is measured. The different tasks, the exact measures and the differences in the user interface are described in more detail in Section 6.2.5. After completion of the tasks, the participants answered a very short survey consisting of two questions regarding the perceived indexing quality and their comprehension of the user interface (see Appendix A). When users performed both kinds of tasks with one configuration they *could* participate again. They were then assigned slightly different tasks and another configuration (see Section 6.2.5.3 for details). However, since users could also abort, we cannot speak of a whithin-subjects design. The fact that some users performed tasks with all three configurations is not exploited in the statistical tests.

Several hypotheses were formulated to test statistically whether TACKO can enhance the accessibility of tagging systems. One of the most obvious metrics that can be compared for the different configurations is the success rate of photo searches. We expect that both, the organization of the photos and the availability of the TACKO organization structure affect the probability of finding a photo.

Let p_A, p_B, and p_C denote the probabilities of success for the three configurations. To isolate the effects of the retroactive organization of the collection and the application of the TACKO user interface, we can compare configuration A with configuration B and B with C respectively. The combined effect can be examined by comparing A with C. We illustrate the statistical test with the example of p_A and p_B.

We cannot rule out the possibility of a negative effect of TACKO, so we choose a *two-sided*

test, i.e., our alternative hypothesis is that p_A and p_B are different:

$$H_0 : p_A = p_B$$
$$H_1 : p_A \neq p_B$$

We use a simple Z-test to test the hypotheses based on the statistic:

$$z = \frac{\hat{p}_A - \hat{p}_B}{\sqrt{\hat{p}(1-\hat{p})(\frac{1}{n_A} + \frac{1}{n_B})}}$$

The estimators $\hat{p}_A$ and $\hat{p}_B$ represent the proportion of successful photo searches in the samples obtained for the respective configurations. The size of the samples is denoted by n_A and n_B. Finally, $\hat{p}$ stands for the overall proportion of successful searches in the union of both samples. Given that the null hypothesis is true, i.e., p_A and p_B are equal, the distribution of z is a standard normal distribution.

Since H_1 is a two-sided alternative hypothesis, we reject H_0 if the absolute difference in the observed sample means is high, i.e., if the statistic z has a sufficiently small value. The thresholds depend on the significance level of the test. For a certain probability of type I error α, the thresholds are $+Z_{\alpha/2}$ and $-Z_{\alpha/2}$ where $Z_{\alpha/2}$ is the upper $\alpha/2$ percentage point of the standard normal distribution (cf. [Mo08, Chapter 2]). For instance, we obtain thresholds of approximately ± 1.96 for z if $\alpha = 0.05$. Given a certain value for z we can also calculate the *P-value* indicating exactly how likely the observed result is apart from fixed significance levels, based on the assumption that the null hypothesis is true.

We further examine the time required for task completion when the tasks are based on the organized collection and when the TACKO user interface is used. We treat both kinds of tasks — searching and collecting photos — separately. For search tasks, we further distinguish the time required for successful tasks and the time required regardless of the success, since it is likely that the success rate has a strong influence on the required time and thus an interpretation of the results is difficult if this distinction is not made.

Again, we compare pairs of configurations to isolate the effects of the organization and the TACKO interface. Unfortunately a t-test (see below) is not suitable for comparing two sample means. Although we can drop the assumption that both samples are normally distributed because the sample size is large enough, the variance is very high which makes it unlikely to obtain significant results. Instead, a *Mann-Whithney U-test* is applied. This test can be used to compare samples not being normally distributed and it is not affected by the variance of the results. However, it has to be noted that it does not compare the sample means but it is based on an ordinal comparison of sample values. The test relies on a statistic U which is computed based on the ranks of both samples in an ordered series of *all* observations. It is known that U is normally distributed for large n which makes it possible to relate the computed statistic to the z-distribution as described above. For more details on the Mann-Whithney U-test we refer to [Co98].

Finally, we examine the perceived quality of organization by evaluating the answers to the small survey that participants completed after each task (cf. Section 6.2.5). A t-test is used to compare the answer means. We can compare the combination of search and collection tasks as well as each task type separately. It can be argued that a comparison of the means is not appropriate for Likert scales, so we also provide a visualization of the answer frequencies.

However, a t-test is at least practical for testing whether there is a significant difference in means regardless of how this fact is interpreted. The t-test is very briefly summarized below.

Let for instance l_A and l_B denote the "real" average Likert score for configurations A and B respectively. Our hypotheses are:

$$H_0 : l_A = l_B$$
$$H_1 : l_A \neq l_B$$

If $\bar{t}_A$ and $\bar{t}_B$ are the observed means for these variables, the t-test builds on the assumption that the static t_0 with

$$t_0 = \frac{\bar{l}_A - \bar{l}_B}{S_p \sqrt{\frac{1}{n_A} + \frac{1}{n_B}}}$$

follows a so called t-distribution. The S_p stands for the root of S_p^2 which is an estimate for the common variance of both samples (cf. [Mo08]). The variables n_A and n_B stand for the individual sample sizes.

Similar to the z-test, the obtained value for t_0 can be compared to the t-distribution to obtain the significance level for the deviation in means, i.e., whether the observed value is extreme enough to reject H_0. Although the t-test depends on the assumption that both samples are normally distributed and have equal variances, this requirement can be dropped if the samples are sufficiently large [Mo08, Chapter 2]. Since we compare samples consisting of at least 30 measurements, we ignore this assumption. For more details on t-tests we refer to respective textbooks (e.g., [Sp11] or [Mo08]).

The comparison of the main metrics described above is central for answering the questions underlying the experiment. However, several more measures are extracted from the recorded log files (cf. Section 6.2.5.4). They were selectively analyzed as well when it was expected that they give valuable hints or explanations regarding the outcome of the experiment.

6.2.3. Participants

The participants of the experiment were recruited using the web-based crowdsourcing platform *MobileWorks*[4] The more popular platform *Amazon Mechanical Turk*[5] could not be used because the service was not offered outside of the US when the experiment was prepared. Such crowdsourcing platforms allow customers to submit large quantities of small simple tasks which are then processed by a large number of workers throughout the world. Customers usually pay per successfully completed task and the platform manages the payment of the workers. Usually the tasks have to be broken down into small units that can be completed within a few minutes or even seconds. Typical examples are the classification of photos or the digitalization of handwritten text.

Using a crowdsourcing platform to recruit participants for scientific experiments has clear advantages: Once the infrastructure is established, the experimenter has access to a very

[4] http://www.mobileworks.com, (accessed 3rd of January 2013).
[5] https://www.mturk.com/mturk/welcome, (accessed 3rd of January 2013).

large number of participants with a high availability. A set of several hundred tasks is usually completed within hours. This way it is possible to find more participants in shorter time and for less money than for instance if the participants are students, particularly if they have to be gathered at the same place in a certain time period. Additionally, the workers are not biased in any way regarding the outcome of the experiment. They have no clue what the goal of the experiment is and can be expected to simply try to complete the tasks as fast as possible.

However, there are disadvantages as well. Since the tasks have to be very short, the experiment tasks have to be broken down into small units each of which only takes at most a few minutes to complete. It is possible that some participants complete several tasks and spend more time on the experiment in total but this cannot be guaranteed. This uncertainty poses challenges for the experiment design and requires a very flexible setup. Furthermore, it is impossible to control external influences on the participants. It is not clear whether they work on tasks at a desktop PC in an office or on a mobile device in a public place. Finally, the quality of the results is often very bad. Possible reasons are that workers do not read the instructions properly, they have insufficient language skills, they use old hardware or have a very slow internet connection or they are simply distracted during the tasks and do other things in between. In effect, many results had to be rejected (see Section 6.2.6.1).

Before participants started a task, they were asked to enter a name as well as their gender and age. In total, 152 people participated in the experiment, 82 men and 70 women. The age ranged from 16 to 67 (average 29.8).

The cultural background of the participants is not known. However, the names indicate that many of them are Asian. To participate in the experiment, English skills were required but this could not be enforced via the MobileWorks platform. To successfully complete a task, the instructions had to be read and the participant needed to exploit the photo's tags. If participants never used a single tag and just scrolled through the list of photos, the task was rejected (cf. Section 6.2.6.1).

6.2.4. Preparation of the dataset

Since our participants were recruited using an international crowdsourcing platform, we could not assume that they have any specific knowledge. It is even not possible to rely on a certain cultural background. Therefore, it was required to choose a subject not being related to a certain place or a specialized knowledge domain. At the same time, it had to be diverse enough — with regard to the number of concepts as well as facets — that it makes sense to apply a non-trivial organization structure. Trivial in this sense would for example be a single facet containing the names of famous actors. Therefore, we decided to base our dataset on the set of photos managed in a Flickr group called "Product Photography". Groups are an additional way to share photos on certain topics in Flickr. The difference is that groups can have a discussion board, explicit members and administrators[6]. Therefore, the quality of photos and tags tends to be higher within a group than it is in a random sample of Flickr photos being tagged with a tag corresponding to the subject of the group. However, it has

[6] http://www.flickr.com/groups/, (accessed 8th of November 2012).

to be noted that photos can be shared among several groups and there is no organization scheme connected to a group.

When we obtained the metadata of the photos via the Flickr API on November 8th, 2012, the group contained 46,888 photos. The photo collection was then processed in two phases: First, it was manually organized and structured using TACKO. Subsequently the resulting collection was reduced to a substantially smaller set of photos by removing photos that are obviously very hard or even impossible to find (e.g., photos without tags). This smaller set represented the basis for the three different configurations (cf. Section 6.2.2). In configuration A the tags were reverted to the original tags obtained from Flickr. However, the selection of photos was the same in all three configurations.

We will now provide some details on the manual preparation and the automatic filtering of the dataset.

Manual preparation step. To obtain a dataset that is organized with the TACKO structure, the collection was manually organized by the experimenter. During the preparation, it was attempted to harness the existing tags as much as possible. In particular, it was avoided to assign tags to or remove tags from individual resources. Instead, the scope of all operations was based on existing tags. This means, that new tags, such as "food and beverages" could be introduced, but they were only assigned to a batch of resources being already tagged with a related tag.

More precisely, the following kinds of changes were applied:

- **Removal of tags carrying no additional information.** Such tags usually apply to the whole collection of photos (in our dataset "product photography" is such a tag). They are useless for narrowing down the result set in a meaningful way.
- **Removal of non-evident tags.** These tags are not obviously related to what is shown on the photo. They represent for instance Flickr usernames, technical terms of the photography domain, camera models, et cetera.
- **Removal of tags with a low recall.** The tag "reflection" is an example of such a tag. It applies to many photos because they show some kind of reflection but only a small fraction of them is tagged with regard to this aspect. Such tags can mislead users looking for a photo because selecting them usually leads into a dead end. If it was easily possible to improve the recall for the tags through additional batch assignments, this practice was preferred.
- **Removal of tags with unclear meanings**. Examples of such tags are "design", "architecture" and "art". For each such tag it is hard to assess whether it applies to a particular photo or not. Some users use the tag "architecture" for all buildings, others only for interesting architectural aspects. In effect, either the recall is very low, or the precision is bad, depending on the interpretation of the tag.
- **Renaming of tags.** Obvious spelling mistakes were corrected. Synonyms and tags in languages other than English were unified. Singular tags were changed to the plural form. The autocomplete feature was used to find similar tags.
- **Disambiguation of homonyms and polysems.** Tags with several meanings were split into more specific tags as far as possible. For example, the tags "apple (fruit)"

and “apple (company)” were introduced to distinguish these two meanings of the tag “apple”. Additional tags like “fruit” or “ipod” could be used to assign these new tags to the correct photos.

- **Creation of subsumption relationships.** Subsumptions were established among existing tags where appropriate. This often entailed many new assignments of the more general tag. Sometimes, new general tags were introduced as supercategories of existing tags.
- **Batch assignment of tags to improve recall.** As pointed out above, for many tags the recall measure is very bad. Obvious correlations with other tags help to increase recall. For instance, the assignment of the tag “red” to all photos being tagged “coca cola” increases the recall for “red” since most of these photos show the coca cola emblem. Depending on the initial indexing quality, this operation does not even necessarily impair the precision for searches for “red”.
- **Batch removal of tags to improve precision.** Similar to the batch assignments, tags were removed when it was obvious that they were mistakenly assigned to a set of photos that could be easily isolated using existing tags.

The manual organization of the collection took approximately 5 hours. During the first half of this time the most frequent tags were harmonized and structured. Later, random photos were selected and it was checked whether they could be found using the organization structure. If this was not the case, the respective tags were integrated into the structure.

All manual steps were automatically logged and are summarized in Table 6.1. It has to be noted, that sometimes a single user operation leads to several logged operations. This is for instance the case, when a tag is added to a facet. Then, a subsumption relationship is implicitly established which results in an additional log entry.

Type of operation	Frequency
Adding a tag to a facet	401
Removing a tag from a facet	216
Assigning a tag to all resources matching the filter	78
Removing a tag from all resources matching the filter	155
Establishing a subsumption relationship	496
Renaming a tag	41

Table 6.1.: Frequencies of manual operations during the preparation of the dataset.

Automatic filtering. After the manual preparation of the collection, some automatic changes were made. First, all tags being longer than 30 characters were removed since most of these tags obviously resulted from mistakes made during data entry (tags were not separated by spaces). Subsequently, certain photos were removed from the collection until all photos matched the following two requirements:

- at least three tags of the photo are contained in at least one facet.
- there are at most 14 other photos having exactly the same facet tags assigned as the photo under consideration.

The first rule eliminated photos that are hard to find because too few tags were assigned. For instance, the only way to find a photo without any tags is to scroll through the whole collection of photos. The second rule assured that there are no sets of more than 15 photos that cannot be systematically narrowed down using facet tags. We chose 15 as the maximum size because this means users have to scroll down 1–3 screens depending on their screen resolution which we considered reasonable.

The application of these rules reduced the size of the collection from 46,888 to 9,518. Table 6.2 summarizes the basic properties of the resulting dataset. The collection had to be reduced this way in order to achieve reasonable success rates for photo searches. Please note that photos were not only removed in the organized collection but also in configuration A-RAW in order to obtain comparable conditions for all configurations. The right table column refers to the resulting subset of the raw collection. The number of subsumption relationships does not include relations being implied by the transitivity property of subsumption.

Property	Value after organization	Value before organization
Number of photos	9,518	9,518 of 46,888
Number of distinct tags	19,828	19,927
Number of distinct facet tags	291	0
Number of tag assignments	121,029	112,260
Number of facet tag assignments	52,230	0
Number of facets	82	0
Number of subsumption relations	293	0
Average tags per resource	12.72	11.79
Average facet tags per resource	5.49	0

Table 6.2.: Basic properties of the dataset.

We consider a collection size of roughly 10,000 photos reasonable for our experiment since it is on the one hand large enough to make it impractical to find a photo by scrolling through the entire list. On the other hand, choosing a larger collection of photos of a certain domain increases the problem that there are large subsets of similar photos that cannot be distinguished by means of their tags.

6.2.5. Tasks and measures

The participants performed two different kinds of tasks: Searching a specific photo and collecting a set of photos of a certain kind according to given instructions. The user interface varied slightly. In the following, we will briefly describe the tasks and the respective user interface adaptation. Subsequently, it is outlined how the combinations of tasks and configurations were assigned to participants in order to obtain unbiased results. Finally, an overview of the measures being captured during the experiments is given.

6.2.5.1. Searching photos

Figure 6.2 shows the user interface applied for search tasks. In the top right corner of the screen, the participant sees the photo that has to be found. It can be enlarged with a mouse click. The target photos were selected at random but were the same for all configurations. One participant never searches for the same photo twice. The only difference between the three configurations is the navigation bar. Only in configuration C, tags are grouped to facets. Figure 6.2 shows configuration A-RAW with the unorganized Flickr tags.

Before the participants start the search, they are given basic user interface instructions on a separate screen. See Appendix A for details.

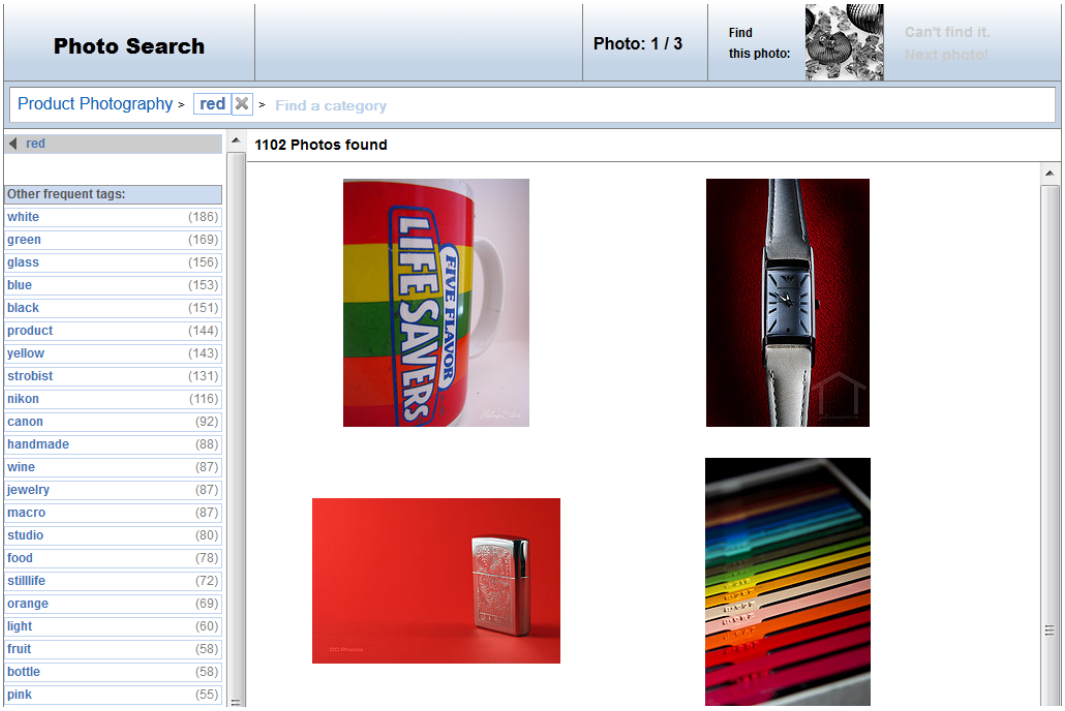


Figure 6.2.: User interface adapted for search tasks.

During a search task, participants have to search three photos in a row. If the participant fails to find the photo, it can be skipped after 45 seconds by clicking the button in the top right corner. However, there is no time limit, so participants can continue to search the photo. If the photo is found in the list, the participant clicks the photo to select it. Subsequently the next target photo appears at the top.

After searching three photos, the participants answer a very short questionnaire which is also illustrated in Appendix A.

6.2.5.2. Collecting photos

The user interface applied for the photo collection tasks is shown in Figure 6.3. The figure shows the organized collection with facets of configuration C. There are three different in-

structions determining which photos have to be collected in a collection task. They are listed in Section A.2.2 in the appendix. For instance the participants have to collect photos of cars but the cars have to have three different colors and four different makes. Additionally, two of them have to be toy cars or models.

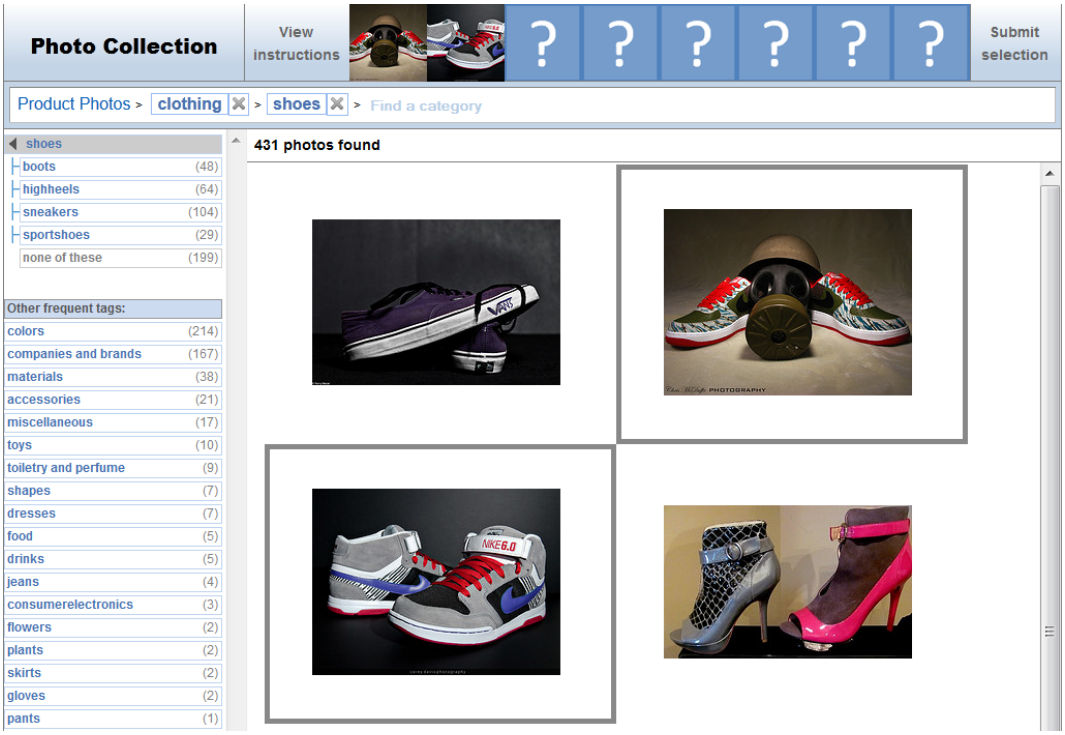


Figure 6.3.: User interface adapted for collection tasks.

In all collection tasks, eight photos have to be selected. When a photo is clicked, a thumbnail picture is added to the bar at the top of the screen and replaces one of the eight question marks. The photo in the result list is marked with a grey frame. To unselect the photo, the participant can click the thumbnail image or the larger image in the list. When the participant has selected eight photos, the task can be completed by submitting the selection using the button at the top right corner of the screen. Subsequently the same short questionnaire as for the search tasks has to be answered.

6.2.5.3. Task assignment

The combination of kind of task and configuration is determined dynamically each time a participant is forwarded to the experiment server after accepting a task on MobileWorks. This is required since it is not known how many tasks a participant will complete. Participants can abort after the first one or they can complete up to twelve tasks. They were identified via a workerId encoded in the URL of the link they followed on the MobileWorks platform to start the task. Additionally, a browser cookie was set when they accessed the experiment server for the first time.

In the assignment of tasks, several aspects had to be regarded

- Search and collection tasks have to be mixed in order to obtain results for both.
- Participants must not be assigned the same task twice, i.e., not search for the same photo or get the same collection instructions.
- When participants perform several tasks, they should work with different configurations to reduce the impact of participants with particular characteristics on the results. For instance, if there is a very fast participant completing twelve tasks, these tasks should be evenly divided among the three different configurations to avoid a bias.
- The order in which participants work with different configurations has to be varied.

The last point is important because otherwise a bias could be introduced. When all participants see the configurations in the same order, it is likely that they perform better with the later ones because they already learned how the tasks work in general. So the position in the sequence of configurations is a factor influencing the result of the experiment. To balance the effect of such factors on different "treatments" — in our case the configurations A, B and C —, one commonly applies a so called *Latin squares* setup in experimental design (cf. [Ba08]). In our case, such a design would ensure that the configurations occur equally often at each position in the task sequences of the participants. Although this is desirable in our case, it was not sufficient: With a Latin squares setup, only three different sequences would have been applied, e.g., ABC, CAB, and BCA. It can be seen that in two of them configuration B was positioned before C. Since we directly compare performance measures for B and C, this could introduce a bias.

In consequence, we applied a more sophisticated task assignment. We additionally balanced the relative order of configurations. This means that configuration A is equally often positioned before configuration B as B occurs before A et cetera. The following approach was applied to achieve this:

- The first two tasks are always a search and a collection task with the same configuration. The system automatically choses the configuration that currently occurs least as the first configuration of a participant.
- Task 3 and 4 are also a search and a collection task but with another configuration. The system counts how often the remaining two configurations were applied as the second configuration in the set of all task sequences starting with the configuration chosen for the first two tasks of the current participant. The less frequent configuration is selected.
- Task 5 and 6 are performed with the remaining configuration. They also include one search and one collection task.
- All subsequent tasks are search tasks. The configuration is changed for each task by repeating the sequence determined before.

6.2.5.4. Measures

As already stated above, we measure the success rate of search tasks and the time required to complete both, search and collection tasks. For search tasks we further distinguish time

spent on successful tasks. Additionally, several other measures can be extracted from the log files. The following actions were logged:

- begin and end of a task,
- selection of a tag, i.e., narrowing the filter,
- removal of a tag from the filter, i.e., widening the filter,
- scrolling (one log entry per 10 photos),
- finding and skipping photos in search tasks,
- and selection and deselection of photos in collection tasks.

For the actions that modified the filter it was distinguished how the modification was achieved, for instance whether a tag was selected from a facet or whether it was entered manually. Based on this log, the following measures were be computed:

- Number of scroll actions (10 photos) per task.
- Number of tags selected in the navigation bar.
- Number of tags being entered manually to narrow the filter.
- Total number of navigation actions, i.e., narrowing and widening the filter.

Finally, the participants answered a short questionnaire consisting of only three questions. The first two were answered on a Likert scale with seven options (see Appendix A for details). These answers are saved per task and can thus be analyzed to compare differences between the configurations.

6.2.6. Evaluation of the results

The participants worked on 490 tasks which incurred costs of only 98 USD. After it had been verified that the setup worked with 6 initial tasks, the tasks were submitted to MobileWorks in batches of size 20 to 90. After submitting a batch, it took only a few minutes until the first workers started to work on the tasks. It was necessary to submit the task in batches to be able to review the results and block certain workers immediately if they did not work properly, either intentionally by cheating, which was rare, or more frequently because they completely misunderstood the task or they had no appropriate device or internet connection to participate in the experiment. The criteria which led to the rejection of a result or blocking of a participant are listed in Section 6.2.6.1. The results of the statistical tests are given and discussed in Section 6.2.6.2.

6.2.6.1. Filtering of results

Roughly 50 percent of the tasks were not finished by the participants or were rejected due to one of the following reasons:

- The total time spent on the task, i.e., either collecting photos or searching three photos,

took more than 20 minutes. Such a long time indicates that either the participants did not understand that they can skip photos if they cannot be found or that their internet connection is too slow to use the interface properly.

- The participants did not use the tags at all but tried to find the photo by scrolling through the entire list of more then 9000 photos.
- There were breaks of more than 60 seconds between two actions. Apparently, the participants were distracted or had technical problems. The only exception to this rule is the first action within a task which often took longer. The participants probably clicked the target photo to enlarge it or reviewed the instructions of collection tasks (these actions were not recorded).
- It was obvious that participants did not try to complete the task properly. Either they waited until they could skip a photo without trying to find it (no actions at all) or they obviously cheated by entering tags they could not have known without searching for the photo id on the web (e.g., they entered the name of the Flickr user which the photo belongs to although the photo did not show the name). These cases were very rare.

The fraction of aborted and rejected tasks is similar for all three configurations (between 49.7 and 53.2 percent). In extreme cases the participants worked on a single task for more than an hour, mostly trying to find a photo by scrolling through the entire collection.

For the photo search tasks, another filter was applied: Since a complete search task consists of a sequence of three photos, the time for a single photo can be very long while the sum of the three search times is still below 20 minutes. For this reason, we also ignored photo searches taking longer than 6 minutes. With 23 out of 411 total searches these outliers were rare but had a significant impact on the average and the variance of the search time. The samples for all three configurations contained at least 5 such cases. The distribution of the search times below 6 minutes is visualized in Figure 6.4.

The results of the photo collection tasks indicated that many problems are due to the fact that participants did not read the instructions properly (see Appendix A for the detailed instructions participants were given before they started a task). Problems with the English language might also have played a role.

When the participants tried to cheat or there were several tasks in a row that had to be rejected for any of the above reasons, they were blocked and could not work on further tasks.

6.2.6.2. Experiment results

Table 6.3 summarizes the results for the primary measures. It reveals a large difference of the success rates for photo searches. The two sided z-tests showed that the difference between configuration A-RAW and C-TACKO is significant ($p = 0.03846$), this means that the combination of the TACKO interface and a better organization of the resources has an impact on the success rate. However, for the individual factors the effect was not significant.

The time spent on search tasks varied from a few seconds to several minutes. This spread is reflected in high standard deviations and is illustrated in Figure 6.4. The very short tasks

Measure	Configuration A-RAW	B-ORGANIZED	C-TACKO
Search tasks			
Number of searches	130	119	139
Found photos	69	75	91
Success rate (in percent)	53.08	63.03	65.47
Search time (found)	53.19 ± 60.23	58.07 ± 61.22	77.08 ± 73.28
Search time (all)	88.78 ± 65.87	73.29 ± 49.98	91.73 ± 58.74
Collection tasks			
Number of tasks	38	31	31
Collection time	332.19 ± 247.42	269.42 ± 252.64	229.63 ± 173.66
Questionnaires			
Total answers	83	74	80
Answers after search	45	43	49
Answers after collection	38	31	31
Org. quality, overall (1-7)	5.40 ± 1.70	5.39 ± 1.74	5.60 ± 1.56
Org. quality, search (1-7)	5.24 ± 1.75	5.09 ± 1.84	5.14 ± 1.76
Org. quality, collection (1-7)	5.58 ± 1.62	5.80 ± 1.49	6.32 ± 0.74

Table 6.3.: Results for the primary experiment measures.

are due to the fact that the beginning of the task was set to the time of the first action. This means that participants could examine the target photo and if they found a very precise tag describing it — such as a brand name —, a search for photos with this tag yielded a short list from which the target photo could easily be selected. Opposed to that are cases where participants only narrowed the photo collection to a rather broad category — e.g., "food" — and tried to find the target by scrolling through all results for several minutes.

The average time spent on a photo search is surprisingly high for all configurations. Since participants were able to skip a photo after 45 seconds, it was not expected that they would try to find the photo much longer. However, more than 25% of the search tasks took more than 2 minutes.

To compare the search time for the successful photo searches, it was first asserted that the variances of the samples are similar with a Bartlett test. Subsequently, the samples were compared using a Mann-Whithney U-test. Again, a strong significance was obtained for the difference of the samples for configurations A-RAW and C-TACKO ($U = 2475$, $p = 0.022$, two-sided). As suggested by the differences in means, this deviation seems to be primarily caused by the TACKO user interface: The difference between samples B-ORGANIZED and C-TACKO is nearly significant with a significance level of $\alpha = 0.05$ ($U = 2820$, $p = 0.055$, two-sided). The difference between A-RAW and B-ORGANIZED is not significant ($U = 2463$, $p = 0.619$, two-sided). The differences in the search times of the samples are visualized in Appendix B.

The average search time for all photo searches including those in which photos were skipped is also given for comparison. The numbers show that the average time spent on searches

in configurations A-RAW and C-TACKO is very similar. However, the average time for configuration B-ORGANIZED is much lower. It is in fact more than 20% lower than the average for configuration C-TACKO. A Mann-Whithney U-test showed that this difference is significant ($U = 6894$, $p = 0.021$, two-sided). A comparison of the times for A-RAW and B-ORGANIZED only resulted in a p-value of 0.169 ($U = 8516$, two-sided).

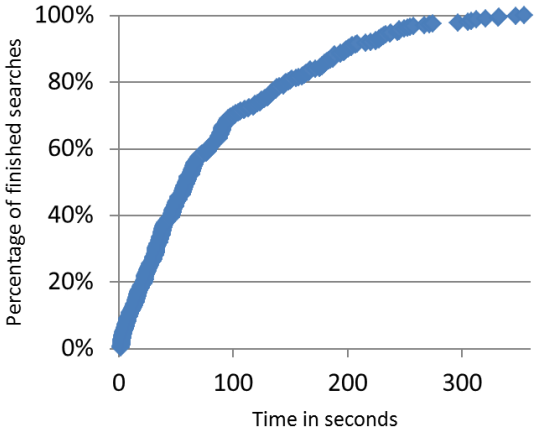


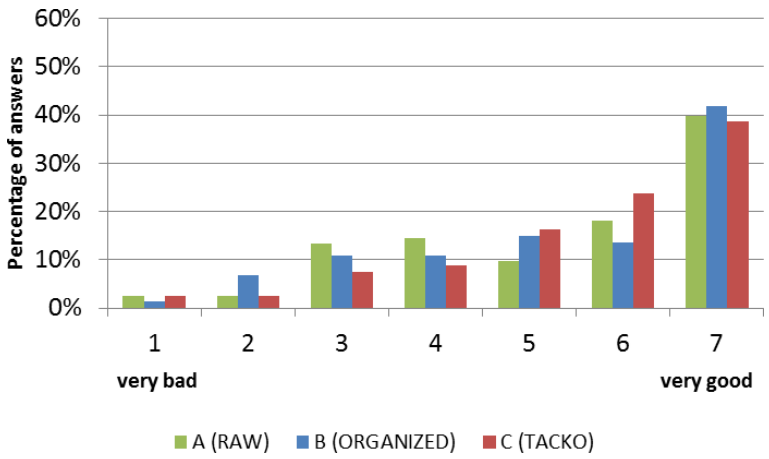
Figure 6.4.: Distribution of photo search durations.

The time differences for collection tasks are not as clear as for search tasks. Although the observed means differ by more than a minute, the smaller number of tasks and the high variance makes a comparison difficult. The greatest difference is between the mean time in configurations A-RAW and C-TACKO, a decrease of more than 30%. However, the Mann-Whithney U-test showed no significant difference ($U = 716$, $p = 0.126$, two-sided). Nevertheless, the visualization of the time distributions allows to assume that this is due to the fact that the distributions differ in shape (cf. Figure B.2 in Appendix B). It can be seen that only one of 31 tasks took more than 8 minutes in configuration C-TACKO. In B-ORGANIZED this applies to 5 in 31 and in configuration A-RAW to 10 out of 39 tasks. An additional t-test was conducted to compare the sample means. Despite the high variance, a p-value of $p = 0.069$ was obtained.

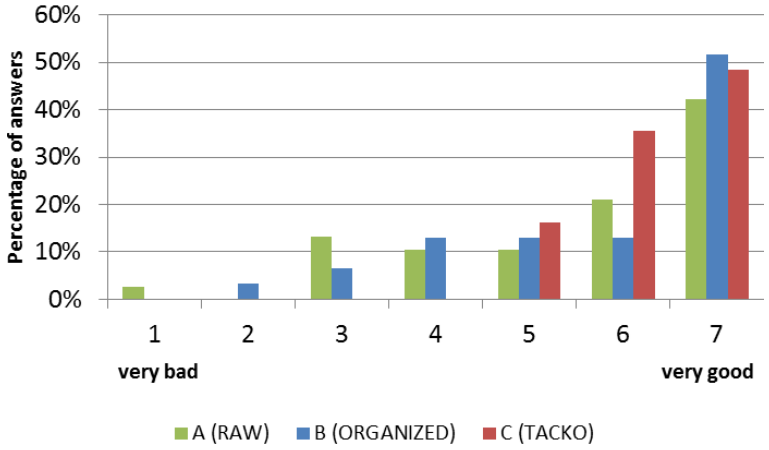
In all configurations it happened several times that participants misunderstood the collection tasks. Particularly the set of instructions for the “food and drinks” taks (cf. Appendix A) was apparently not precise enough. Many participants did not understand that a single photo had to show both, food and a drink. However, the selected photos were reviewed and the task was accepted when it was obvious that the participants tried to complete the task according to their understanding. When this was assessed, it was not visible which configuration the participant had used.

The perceived quality of the organization of the photos being measured in the post task survey is very similar for almost all samples. The overall average is 5.46 with a standard deviation of 1.67. Only the answers selected after a collection task with configuration C-TACKO differ strongly from this value. Figure 6.5 shows the distributions of answers for all tasks in comparison to those for collection tasks. The distribution of answers for search tasks and the result for the other question are visualized in Appendix B.

A t-test showed that the difference in means is significant between configurations A-RAW



(a) All tasks (237 answers)



(b) Collection tasks (100 answers)

Figure 6.5.: Distribution of answers to the question "How were the photos categorized?" for all tasks compared to the collection tasks.

and C-TACKO ($p = 0.023$). A power of $p = 0.094$ was obtained when testing for a difference between B and C.

The increase in perceived quality of organization is also reflected in the results obtained for the additional measures listed in Table 6.4. During collection tasks in configuration C-TACKO, participants hardly entered tags manually but used the facets and related tags in the navigation bar. They also scrolled less than in other configurations.

Relative to configuration A-RAW, a decrease in all measures can be observed for configuration B-ORGANIZED. This applies to search as well as to collection tasks. This is consistent with the general decrease in time spent on the tasks. For configuration C-TACKO, a shift from search and scroll actions to the selection of tags in the navigation bar can also be observed for photo searches.

Measure	Configuration A-RAW	B-ORGANIZED	C-TACKO
Search tasks			
Navigation actions	6.28 ± 7.12	4.77 ± 4.61	6.83 ± 6.97
Search actions	1.88 ± 2.90	1.50 ± 2.18	1.51 ± 2.58
Tag selection actions	2.08 ± 2.64	1.86 ± 2.18	2.83 ± 2.54
Scroll actions	9.27 ± 12.74	7.72 ± 8.82	7.18 ± 8.21
Collection tasks			
Navigation actions	15.11 ± 15.07	10.65 ± 12.55	10.29 ± 11.79
Search actions	3.37 ± 5.85	2.06 ± 3.62	0.58 ± 1.31
Tag selection actions	4.95 ± 5.08	4.23 ± 4.51	5.06 ± 5.31
Scroll actions	18.68 ± 20.45	14.90 ± 18.97	10.07 ± 8.21

Table 6.4.: Average and standard deviation for the additional measures.

6.3. Conclusion

The two phases of the evaluation targeted complementary aspects of TACKO. The usability evaluation focused on ergonomic aspects of the user interface: Whether it is easy to learn and understand and whether there were problems that lead to errors and reduce the efficiency of usage. The second phase, the experiment, attempted to answer the question in how far TACKO can improve the effectiveness and efficiency of users when performing different tasks in tagging systems.

With regard to the usability of the TACKO system, it can be stated that users have no problems in applying the TACKO user interface as long as it is used for accessing resources. Particularly the filter was very well understood and the discovered usability issues related to the sidebar led to many improvements in the interface. Although no additional usability evaluation was conducted for the final version of the prototype, the results of the experiment suggest that there are no severe usability problems anymore. On the one hand, the performance measures indicate that when TACKO is applied users perform at least as good as in a normal tagging system. On the other hand, in the post task survey users were asked to assess how well they have understood the tool they have been using. The answers were very positive regardless of whether the special TACKO functionality was used or only that of a common tagging system (cf. Figure B.3 in Appendix B).

However, when users had to use TACKO for the organization of resources themselves, it turned out, that they require additional instructions to use the system properly. It was found that particularly the representation of subsumption relationships is insufficient. Users do not recognize these relationships and have problems in applying them. In addition to adding visual clues that help the users to understand how tags are related, a further consequence could be to apply subsumptions only in combination with facets and vice versa. From our experience with TACKO we know, that this means only a minor limitation of TACKO's flexibility. The cases where reasonable facets without respective subsumption relationships

are created are rare. Simplifying the organization structure further is an opportunity for future research.

Nevertheless, it can be argued that, once it is understood, the interface can be used effectively to organize a resource collection: Preparing the set of nearly 10,000 photos for the experiments took only few hours and helped users to find significantly more photos than without this preparation.

To answer the question in how far TACKO is suited to improve access to collections of tagged information resources, we consider several aspects: The kind of task, the success rate, and the required time. For search tasks, an increase in the success rate has been observed after both, the organization of the collection and the additional application of the TACKO user interface. However, although the increase is only significant in combination ($p < 0.05$), the bigger increase in the success rate was observed between configurations A-RAW and B-ORGANIZED.

Users were also faster with configuration B-ORGANIZED than when they used the more complex TACKO user interface in configuration C-TACKO. A log analysis revealed that users change their behavior when they can used faceted search. They tend to select more tags from the facets and enter fewer tags manually into the filter. The total number of actions increased. However, when users had to collect a set of photos, the results showed a tendency towards a faster completion of the tasks with the faceted user interface.

For search tasks, one could state that configuration B-ORGANIZED is preferable to configurations A-RAW and C-TACKO because it yields a better success rate than A-RAW in less time, and the significant time savings compared to C-TACKO compensate for the marginally smaller success rate. In other words: The collection should be organized with TACKO but the search user interface should not be changed. However, in a dynamically evolving collection, the resources have to be organized continuously, and thus a more complex user interface is required. Most tagging systems do not offer functionality to retroactively structure the folksonomy at all. This means that the indexing quality can only be improved by modifying each individual resource which is clearly not practical in large collections. Additionally, it must not be overlooked that the increase in time spent on search tasks is due to user behavior and not to a deficiency in the user interface. Research on search strategies showed that users prefer selecting from a set of navigation options to keyword search, even if they are faster with the latter [Te04]. This could also be observed in the experiment.

If target-oriented search and access to specific pieces of information is not the primary task, but users have to systematically browse the collection, the situation is different. The results for the other tasks where users had to compile a set of photos according to given instructions suggest that configuration C-TACKO, i.e., the faceted TACKO user interface is the best choice. The log analysis showed that users could take advantage of the faceted search interface: On average they used it more than eight times as often as the keyword search. We also observed a reduction in time compared to the other configurations. Finally, users perceived the collection as better organized when they could use the facets.

Particularly the usability evaluation showed that the main challenge in the development of TACKO is not the technical realization but the design of the user interface. On the one hand, users have to grasp the potential of the additional structuring capabilities. On the other hand they must not be intimidated by the complexity of the interface.

As long as users do not apply the TACKO organization structure with the same confidence that they exhibit when working for instance with a folder structure, it remains questionable whether TACKO can replace such established structures in practice. However, the evaluation showed that TACKO can serve as a valuable enhancement for tagging systems. An application in practice would be an opportunity to refine the user interface further and possibly broaden the applicability of TACKO this way.

CHAPTER 7

Related Work

In this chapter, we give an overview of research being related to TACKO in one way or the other. On the one hand, there are many extensions to tagging systems. Very soon after tagging systems became widely popular around the year 2005 the need was seen to organize their contents in a more structured way and a broad variety of approaches with this goal have been developed. We cover different approaches to enrich folksonomies and tagging systems with structure, either by deriving the structure automatically or by giving users the ability to explicitly establish relations among tags. Section 7.1 gives a systematic overview of the former, while Section 7.2 is concerned with the manual structuring of folksonomies.

On the other hand, TACKO, as a system for knowledge organization, stands in line with many other knowledge organization systems being developed to overcome the limitations of hierarchies (or trees) and other traditional structures. There are particularly many personal information management systems being designed with this goal in mind. A selection of such systems is covered in Section 7.3. Although not all of them are based on tagging, the presented solutions exhibit similar features and were developed with the goal of giving users a high degree of flexibility with regard to the organization of and access to information resources.

Finally, in Section 7.4, we briefly present the *Hybrid Wikis* approach that is related to TACKO with regard to several aspects: The author of this thesis was involved in its development and it was also integrated in the Tricia platform. Further, a certain kind of tags called *type tags* plays an important role in hybrid wikis.

Beside scientific approaches we also cover commercial applications throughout the chapter, if they are relevant in the context of this work.

7.1. Automatic structuring of folksonomies

There is a broad variety of approaches for the automatic structuring of folksonomies. However, we have attempted to structure the approaches with regard to the fundamental techniques they rely on. We generally distinguish three different families of approaches:

1. tag clustering,
2. the construction of tag hierarchies, and
3. mapping tags to external knowledge representations.

Each of them will be covered in a separate section. However, not all approaches can be unambiguously assigned to one of these categories. If several techniques are combined in one approach, it is either mentioned in more than one section or in the one being most characteristic for it.

7.1.1. Clustering

The lack of structure in tag clouds (cf. Section 2.3.3) as well as the availability of large folksonomy datasets has motivated researchers and practitioners to apply clustering algorithms to these data. Clustering techniques are widely applicable and are generally used to automatically find groups of related or similar items in large item sets. A related field of application is for instance the clustering of text documents, such as web search results (cf. [ZE98] or the web search engine *clusty*[1]).

A clustering algorithm typically assigns each item to exactly one cluster, so the clusters are disjoint. Additionally, many algorithms require that the number of clusters is specified in advance as a parameter. However, the clusters have no predetermined semantics or names, so often a representative or central item is chosen to represent the cluster. We refer to [Ga07] for the fundamentals of clustering as well as an overview of many clustering algorithms. When applied to tagging systems, clustering can be used to form groups of tags, users or resources.

By recursively applying a clustering algorithm, one can generate hierarchical clusters. However, in this section we focus on approaches that generate only one flat set of clusters. Hierarchical clustering will be covered in Section 7.1.2.

Hassan-Montero & Herrero-Solana were among the first to apply clustering techniques to tagging systems. They aim to improve tag clouds by selecting "tags that better characterize the whole collection of tagged resources" and by arranging these tags in meaningful groups [HMHS06]. Their tag selection strategy favors tags that cover resources not being encompassed by other tags in the tag cloud. In contrast, the most frequent tags usually have a very high overlap. Subsequently, a k-means clustering algorithm is applied to generate a fixed number of tag clusters based on the tag cooccurrences. The fact that tags are assigned by different users is not taken into account, so a broad folksonomy must first be converted into a narrow folksonomy. The screenshot in Figure 7.1 shows an example of a tag cloud gen-

[1] http://clusty.com (accessed 4th of Feburary 2013).

erated with this algorithm based on a dataset obtained from the social bookmarking service delicious [HMHS06].

Figure 7.1.: An improved tag cloud with tag clusters generated by [HMHS06].

Hassan-Montero & Herrero-Solana measure the similarity of two tags with the Jaccard coefficient, i.e., the number of resources tagged with both tags divided by the number of resources tagged with at least one of them [HMHS06]. However, this is not the only metric for tag similarity. Cattuto et al. give an overview of different measures for tag-relatedness and examine their implicit semantics [Ca08].

Begelman et al. propose two different approaches for tag clustering to illustrate the applicability and utility of clustering in tagging systems [BKS06]. The first is based on the construction of a graph of tag relations by connecting each tag to "strongly related tags". Although the decision whether a tag is strongly related to another one in this sense depends on tag cooccurrences, the Jaccard coefficient is not used. Instead, the relative number of cooccurrences is taken into account. This means that to determine whether two tags are strongly related it is not sufficient to consider the frequencies of only these two tags and their cooccurrences but they are compared to other pairs of tags in the folksonomy. The components of the resulting graph are considered the clusters. The second approach proposed by Begelman et al. does not distinguish strongly and weakly related tags but involves the application of a graph clustering algorithm to the graph represented by the cooccurrence matrix of all tags. The graph is incrementally split in a way that increases a certain modularity measure. Begelman et al. demonstrate that for both approaches, meaningful results are obtained when applied to a dataset from delicious [BKS06].

Instead of directly comparing tag cooccurrences, Specia and Motta determine the similarity of tags by generating a cooccurrence vector for each tag — this means each component of the vector contains the number of cooccurrences with another tag — and then calculating the cosine between these vectors [SM07]. In a previous preprocessing step, exotic tags are filtered out and spelling variations are harmonized. Based on the *cosine similarity* the following generative clustering algorithm is applied: A pair of tags with similarity above a certain threshold is taken as the seed of a cluster. Then other tags are added as long as they are sufficiently similar to all tags currently in the cluster. This is repeated for all pairs of similar tags. Since the set of resulting clusters is very large — and not disjoint —, the clusters are merged based on certain criteria in a subsequent step [SM07]. Additionally, Specia and Motta

describe how clusters can be mapped to concepts in semantic web ontologies. However, this will be covered in Section 7.1.3.

The results of Knautz et al. also indicate that clustering increases the utility of tag clouds. They show that users perceive tag clouds as more useful when the tags are clustered based on tag cooccurrences [KSS10].

A fundamentally different approach is presented by Zubiaga et alii. Their tag clusters are based on the content of textual information resources. Each tag is represented by a "super-document" that is obtained by merging "all the documents corresponding to a particular tag" [Zu09]. Self-organizing maps, a form of unsupervised neural networks (cf. [Ko01]) are used to cluster the most frequently used tags in a folksonomy based on the content of the respective super-documents. The result is an arrangement of tags in a 2-dimensional grid of a fixed size. Cells may be empty or contain several tags.

The photo sharing website Flickr also employs a clustering algorithm[2]. For a given tag, related tags are clustered into exactly four groups. These clusters are available — and apparently precomputed — for the most frequently used tags on the platform. An example is shown in Figure 7.2. It can be seen that predetermining the number of clusters can be problematic. Although tags are not arbitrarily assigned to the clusters, one could argue that there are only three "real" clusters and the clusters referring to apple computers and apple phones and music players should be merged. Another problem is that tags of a single facet are spread among several clusters: The tags "red" and "green" are in the "fruit"cluster while "black" was assigned to the "ipod"-cluster.

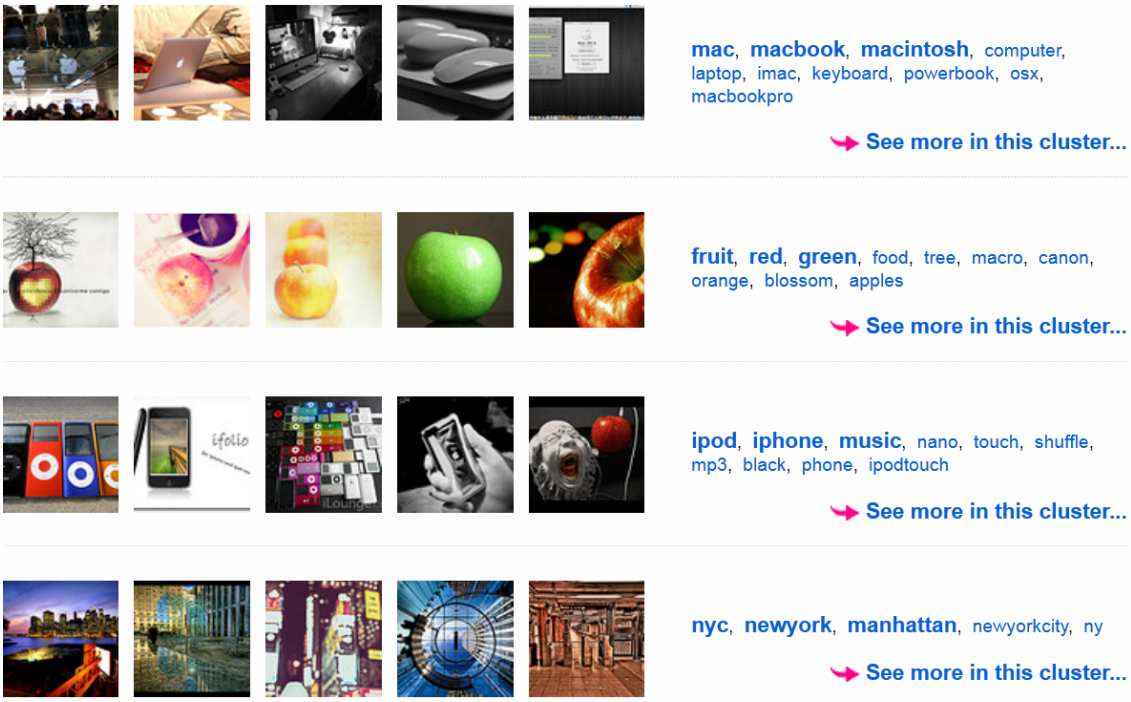


Figure 7.2.: Clusters of tags related to "apple" presented on the Flickr website

[2]Accessible at http://www.flickr.com/photos/tags/TAG_NAME/clusters/ (accessed 4th of Feburary 2013).

Mika shows that the tripartite model of folksonomies allows not only to relate tags based on their cooccurrences at resources but that all kinds of entities in this model — i.e., users, resources, and tags — are implicitly connected by both of the respective other kinds of entities [Mi07]. This means that two tags t_1 and t_2 are not only connected by the resources they are assigned to but also by the users (t_1 is connected to t_2 if there is a user u who assigned both tags, possibly to different resources). Similarly, there is a network of users where connections are based on shared resources et cetera. So it is possible to cluster users, resources and tags based on these different networks. However, for the purpose of illustration, Mika presents groups of related tags being derived from clusters in a graph that was obtained by filtering resource based tag relations in several steps [Mi07].

Based on a certain kind of mixture model, Zhang et al. map users, resources and tags to vectors in a multi-dimensional conceptual space [WZY06]. The components in the vectors represent the latent semantics of the entities. The number of latent semantic dimensions is fixed. After iteratively fitting their model to a delicious dataset, they determine for each dimension in the conceptual space the five tags having the highest probability associated with it. The result is illustrated in Figure 7.3. Note that the tag sets are not disjoint because a tag does not necessarily correspond to only one dimension.

1	java programming Java eclipse software
2	css CSS web design webdesign
3	blog blogs design weblogs weblog
4	music mp3 audio Music copyright
5	search google web Google tools
6	python programming Python web software
7	rss RSS blog syndication blogs
8	games fun flash game Games
9	gtd productivity GTD lifehacks organization
10	programming perl development books Programming

Figure 7.3.: 10 out of 40 conceptual dimensions, each represented by 5 tags [WZY06].

Ramage et al. examine how social tags can improve the clustering of web resources [Ra09]. They evaluate two approaches by comparing the results to clusters they obtained from the Open Directory Project[3]. A k-means clustering algorithm and a generative clustering algorithm being based on latent Dirichlet-allocation are both run with only the document content as input as well as with input that contains social tagging data as well. In both cases the results were improved by additionally taking the tags into account [Ra09].

Finally, Au Yeung et al. apply clustering techniques to identify the different contexts in which an ambiguous tag is used [AYGS09]. They could show that for this purpose it is valuable to take into account which users assigned the tags.

7.1.2. Hierarchy construction

In the following, we will present an overview of approaches that aim to construct hierarchical structures in order to facilitate navigation in folksonomies. More precisely, the structures are

[3] http://www.dmoz.org/ (accessed 4th of February 2013).

not necessarily hierarchical in the strict sense (cf. Section 2.2.1) but sometimes just represent trees or even more general graphs. However, common to all approaches that they detect broader/narrower relationships among concepts (not always tags) which are characteristic for hierarchies. Here, we will only cover approaches that derive tag relations from the folksonomy itself. In Section 7.1.3 we cover some additional algorithms that include external knowledge bases but also construct tree structures.

Schmitz demonstrates in [Sc06b] how an existing approach for constructing concept hierarchies from text can be applied to tagging systems. The original approach has been presented by Sanderson and Croft [SC99]. In addition to text documents it has also been applied to the images in large photo collections [CJS05]. The basic idea is that *subsumption* of two terms is reflected in the cooccurrence of terms in the following way:

$$P(x|y) >= t \quad \text{and} \quad P(y|x) < 1$$

where $P(x|y)$ denotes the probability that tag x is assigned to a resource given that tag y is assigned. The probability is given by the respective fraction of resources in the folksonomy. The variable t is a threshold value that can be tuned. If the above conditions apply, we say that x subsumes y. The threshold compensates for the fuzziness of folksonomies and it distinguishes the approach from the subsumption in TACKO. As in TACKO, the subsumption simply represents a broader/narrower relationship with no particular additional semantics such as a "part-of" or an "is-a" relationship. Schmitz applies the approach to a Flickr dataset and demonstrates that meaningful relations are extracted. However, he also observed that for instance "certain country names were rarely specified and so were placed under cities" [Sc06b].

Although they do not explicitly construct tree structures from tags, we mention the work of Schmitz et al. (C. Schmitz, not P. Schmitz as in [Sc06b]) here because it can be seen as a generalization of the way P. Schmitz detects tag relations in folksonomies [Sc06a]. They propose to use association rule mining to detect tag relationships, whereby a single rule has the form $A \rightarrow B$ meaning that when all the tags in set A are assigned to a resource, it is likely that all tags in B are assigned as well. In addition to generalizing to sets of tags, they also suggest to apply association rule mining to different projections of the three-partite folksonomy graph (similar to Mika who also points out in [Mi07] that the common folksonomy model can be transformed into bipartite graphs in several ways). This means for instance that association rules among users or resources can be found as well.

Heymann and Garcia-Molina present a different approach for automatically creating "a navigable hierarchical taxonomy of tags" [HGM06]. They first create a weighted similarity graph of tags based on their cooccurrences at resources. All edges below a certain threshold are removed and subsequently the weights are removed from the edges. Then, the centrality of each tag is determined and tags are ordered accordingly, starting with the most central tag. In this order, the tags are then added to a tree that initially only consists of a root node. Each tag is assigned as a child to the most similar tag in the tree. If no sufficiently similar child exists, it is added as a child of the root node. The algorithm was applied to datasets from delicious and *CiteULike*[4], a platform for sharing scientific references [HGM06].

Heymann and Garcia-Molina give examples of the resulting tree structures to demonstrate

[4] http://http://www.citeulike.org/ (accessed 6th of February 2013)

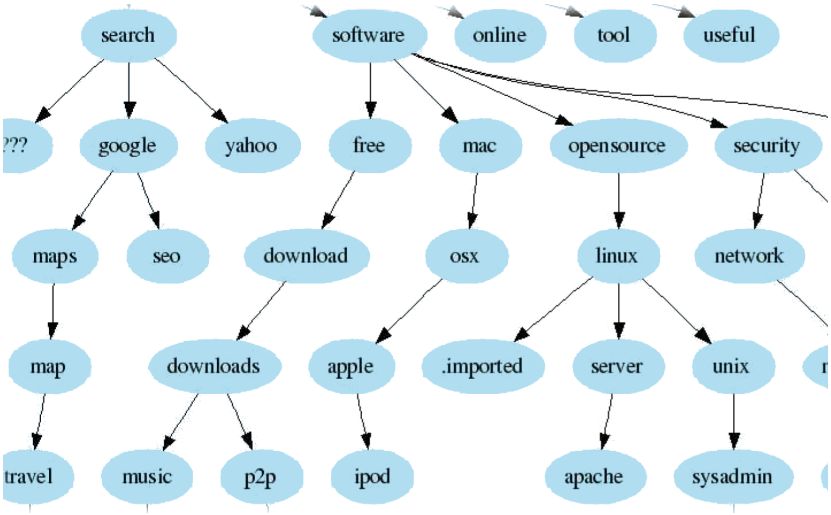


Figure 7.4.: Excerpt of the illustration in [HGM06] showing part of a hierarchical arrangement obtained based on similarity and centrality of tags.

the applicability of the approach [HGM06]. One is shown in Figure 7.4. It can be seen that meaningful tag relations could be obtained, such as "search"→"google". However, this path can be continued via "maps" and "map" up to "map"→"travel". This means that to access the "travel" category one would have know that it is located below search which suggests that the practical utility of such a tree is rather limited for the purpose of navigation. Other examples are the tags "music", "ipod", and "server" which are all located below "software".

Benz et al. enhance the algorithm of Heymann and Garcia-Molina, mainly based on an additional preprocessing of the tags [BHS10]. They create so-called *synsets* of very similar, "synonym" tags. Tags in the folksonomy are then replaced by these synsets. Additionally, they detect ambiguous tags by applying a clustering algorithm to the tags cooccurring with a given tag. The original algorithm is adapted to take this additional information into account. Benz et al. compare the results of the original algorithm and their extended version to the lexical database *WordNet* [Fe10] and Wikipedia and find that their "proposed extended algorithm yields ontologies which resemble more closely the gold-standard" [BHS10], i.e., the manually created reference "ontologies" mentioned before (although Benz et al. use the term "ontology", the result of their algorithm is always a tree structure).

Among other techniques for enhancing the browsing of folksonomies, Li et al. propose to combine several metrics to determine whether a candidate tag t_c is a subtag of a tag t [Li07]. For the sake of compactness, we describe them here in accordance with Sanderson and Croft [SC99], i.e., $P(x)$ denotes the relative frequency of tag x in the collection (the probability that randomly picked resource is tagged x) and $P(x|y)$ the relative frequency of x within the set of resources tagged y:

- **Coverage:** $P(t_c)/P(t)$
- **IR (intersection rate):** $P(t|t_c)$
- **ICR (inverse coverage rate):** $P(t_c \wedge \neg t)/P(t)$

- **IR':** $P(t_c|t)$
- **IRR:** "[...]is set to a set of discrete values of 1, 2 and 3 according to their rank by IR. If a tag ranks top 30 among all the tags, its IRR value is set to 1, if it ranks between 30th and 60th, its IRR value is 2, otherwise it will have an IRR value of 3" [Li07].

The expression $P(t_c \wedge \neg t)$ thereby denotes the relative frequency of resources being tagged t_c but not t. Li et al. constructed a decision tree to decide whether a subtag relationships exists for a given pair of tags. To obtain the respective parameters for the metrics, they were applied to a sample of tag pairs from delicious and each pair was manually assessed.

Lalwani and Huhns propose another indicator for a hierarchical relationship between two tags: They suggest, that if two tags have only little overlap but there is another tag that occurs frequently with both tags, then it is likely, that this other tag is a common supertag [LH09].

Trattner et al. argue that a problem of tag hierarchies is that the users can only browse up to a limited specificity, i.e., a tag that represents a leaf in the tree can still be assigned to many resources [TKH11]. In consequence, the user has to page through a long resource list in order to find the desired resource. To solve this problem, they propose so-called *tag-resource-taxonomies*. Such taxonomies are obtained by first creating a resource tree based on a centrality measure. Subsequently the tree, which has a fixed branching factor, is labeled with tags. Thereby, the same tag may be assigned to several nodes [TKH11]. The resulting taxonomic relationships among tags were compared to those of a thesaurus and it was found that the performance of the algorithm was similar to that of other algorithms including those proposed in [HGM06] and [BHS10].

There is a whole family of methods for hierarchy construction, referred to as *hierarchical clustering* [TKH11]. It represents the repeated application of clustering techniques whereby subsequent runs build on the results of previous ones. One distinguishes *agglomerative* and *divisive* methods [Ga07]. The former represents a bottom-up approach, this means that small clusters are gradually merged to larger ones, the latter refers to recursively clustering a set of entities in a top-down manner. One algorithm that applies hierarchical clustering to tags is briefly sketched in [WZM06].

Helic et al. compare the performance of four algorithms for constructing tag hierarchies including two hierarchical clustering algorithms [He11]. They compare hierarchical k-means clustering based on the algorithm presented in [Zh05] and recursive clustering using affinity propagation (cf. [FD07]), to the approaches proposed by [BHS10] and [HGM06]. The quality of the algorithms is compared by simulating decentralized search in a network of tags induced by tag cooccurrences. Thereby the tag hierarchies serve as background knowledge for the simulated agent which navigates from a tag to a related tag in each navigation step. Results showed that the latter two approaches outperform the hierarchical clustering algorithms [He11].

In [HS11], Helic and Strohmaier point out that since cooccurrence based tag networks are highly connected, it is not practical to offer a user all related tags for navigation. They found that the utility of tag hierarchies produced by existing algorithms suffers when only a limited number of navigation options can be presented to users: It is simply not practical to offer the user several hundred navigation options in each step. They propose an adaptation

of the algorithm presented in [BHS10] that takes into account user interface constraints and constructs hierarchies being significantly narrower than those produced by other algorithms.

A completely different approach is followed by Plangprasopchok et alii [PL09]. They suggest an approach that builds on the shallow hierarchies users can or could [5] create in the popular web-based platforms Flickr and delicious with the limited means provided there. In Flickr users can assign photos to named sets and the sets can be grouped to named collections[6], in delicious users could group tags to so-called *tag-bundles*. Plangprasopchok et al. assume that terms used in Flickr collection names are broader than that of sets. They present an algorithm that aggregates all these relationships into a large taxonomic structure that does not necessarily have to be a tree, i.e., one node can have several parents or superordinate nodes. If terms are used in a conflicting way by different users, frequencies are compared for both directions to resolve the conflict. The applicability of the approach is demonstrated with examples obtained from Flickr datasets [PL09] and by comparing them to the hierarchy of the Open Directory Project [PLG10].

Finally, Zhang et al. propose an approach to generate a binary tree based on their model of latent semantic concepts underlying a folksonomy ([WZY06], covered in Section 7.1.1) [ZWY06]. Each node in the tree is not represented by a single tag but a set of tags corresponding to one of the latent abstract concepts.

7.1.3. Mapping tags to external knowledge representations

When folksonomies gained popularity around the middle of the last decade, the tag-based organization of knowledge was perceived as being fundamentally opposed to the application of ontologies [Sh05]. However, Gruber points out that they are mostly contrasted with hierarchical organization structures and not ontologies as they are understood in the semantic web community [Gr07]. Since the lightweight nature of tags facilitates the annotation of resources in a certain way but at the same time it impedes the structured access to these resources, it was soon attempted to harness the knowledge captured in external knowledge representations by combining them with the folksonomy tags. The approaches being covered below are not limited to those involving semantic web ontologies but they have in common that they comprise at least one external knowledge base. These knowledge bases are independently maintained an not directly related to a folksonomy.

It shall be noted here, that there are two popular sources of knowledge that are utilized by several of the following approaches. The first is *WordNet*, a large semantic network of English words [Fe10]. It groups words with the same meaning to so-called *synsets* and connects these synsets — representing *concepts* — with semantic relations. A single word can be contained in several such synsets. An important kind of relations among synsets is hypernymy — and hyponymy respectively (A is a hypernymy of B if every instance of B is also an A, such as "mammal" and "horse"). Together they form two large, separate hierarchies for nouns and verbs. Additionally, the semantic web search engine *Swoogle* can be used to retrieve ontologies that contain certain terms [Di04].

Van Damme et al. propose to combine several knowledge bases and techniques to integrate

[5] In delicious this feature has been discontinued.
[6] http://www.flickr.com/help/with/sets/ (accessed 6th of February 2013).

folksonomies and more structured sources of knowledge [VDHS07]. In addition to statistical analysis of the folksonomy tags, they advocate the application of "online lexical resources like dictionaries, Wordnet, Google and Wikipedia" as well as "ontologies and Semantic Web resources". They give a structured overview of appropriate resources and techniques, for instance social network analysis and ontology matching, but do not implement and evaluate the approach.

Specia and Motta combine a generative clustering algorithm, that generates groups of related tags based on tag similarity (cf. Section 7.1.1), with semantic web ontologies [SM07]. Pairs of tags of the same cluster are looked up in semantic web ontologies and thesauri. If concepts for both tags are found in one ontology and a relation between the tags exists, this relation is included in the cluster. However, this process is not completely automated. Although an ontology is only taken into account if it contains at least two tags of the cluster, the problem of tag ambiguity is not solved. The algorithm was tested with Flickr and delicious datasets. Specia and Motta describe the result as "*faceted ontologies*, that is, partial ontologies conceptualizing specific facets of knowledge" [SM07]. An example of the tag relations found in one cluster is shown in Figure 7.5. It is argued that the resulting mapping offers potential for query extension or disambiguation, for improving the visualization of tags and for suggesting tags during indexing [SM07].

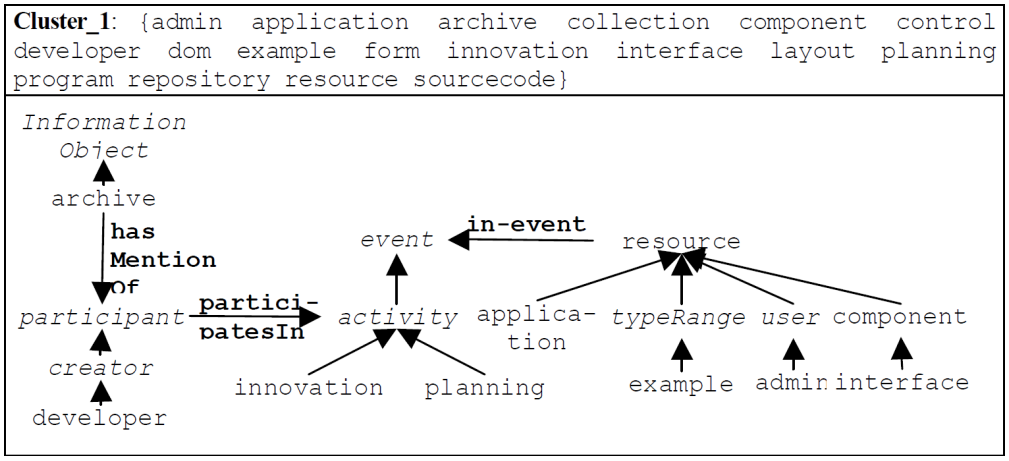


Figure 7.5.: A "partial" ontology based on a cluster of tags presented by Specia and Motta in [SM07].

In [An07], the authors further elaborate and automate the approach presented in [SM07]. The semantic enrichment of a tag cluster was divided into two phases: concept identification and relation discovery. In the first phase, the algorithm identifies ontology concepts for individual tags. Subsequently, it is searched for subsumption and disjointness relations, generic relations as well as sibling and instance-of relations in the ontologies. A relation can also be obtained by combining relations of different ontologies via a linking concept. The algorithm has been partly implemented and tested with tag clusters obtained from certain Flickr photos. The fraction of tags that had matches in an ontology is reported to lie between 18% and 29%. This is due to several problems related to the tag vocabulary. Among novel terminology and names of instances — such as people or places — the authors identify the usage of

multiple languages and folksonomy jargon as the causes for this low rate [An07]. But also the sparseness of semantic web ontologies contributed to the problem.

The FLOR (FoLksonomy Ontology enRichment) system can be considered a further development of the aforementioned approach [ASM08, An08b, An08a, ASM09]. The authors describe it as follows

> "Intuitively, FLOR performs three basic steps [...]. First, during the **Lexical Processing** the input tagset is cleaned and all potentially meaningless tags are excluded. We rely on a set of heuristics to decide which tags are likely to be meaningless. Second, during the **Sense Definition and Semantic Expansion** we attempt to assign a WordNet sense to each tag based on its context (i.e., the other tags in its cluster) and to extract all relevant synonyms and hypernyms so that we migrate to a richer representation of the tag. Finally, during the **Semantic Enrichment** step each tag is associated to the appropriate [Semantic Web Entity]" [ASM08].

In [ASM09], the authors mention an additional "semantic aggregation" phase. Applying the algorithm to a Flickr tag set revealed that again many tags could not be related to semantic web entities, however if it was possible, the accuracy was very high, i.e., in the most cases the correct semantic web entities were selected for the tags [ASM09].

In contrast to finding ontologies containing concepts for tags, the *T-KNOW* algorithm that is an integral part of the *T-KNOW* system being described by Abbasi et al. starts with a set of ontology concepts [ASC07]. These concepts are manually selected and represent tag categories to which the folksonomy tags are assigned. Central in this classification process is the Google API[7] that is used to determine which category a tag belongs to. To determine for instance whether the tag "paris" should be classified as a "city", the algorithm issues queries of the form "city such as paris" [ASC07]. The text snippets in the search result are compared to the *context* of the tag being derived from the folksonomy.

The approach of Laniado et al. relies on WordNet as an external knowledge representation. It is "based on the idea of integrating an ontology in the navigation interface of a folksonomy" [LEC07]. The authors aim to organize the related tags in the delicious user interface based on the WordNet noun hierarchy. Therefore, tags are considered on a resource level, i.e., the disambiguation of tags is performed on the individual tag assignments and based on the tag context being represented by other tags frequently occurring at the same resource. This way, a set of tag assignments can be mapped to WordNet. For each matching entry (a WordNet synset), the path to the root of the WordNet hierarchy is selected and all paths are eventually merged to a tree that effectively represents an excerpt of the WordNet noun hierarchy. The resulting tree is compressed, for instance by eliminating intermediate nodes with only one child.

Although not explicitly designed for folksonomy tags, the approach of Stoica et al. [SH04, SHR07] is closely related to that of Laniado et alii [LEC07]. It is also based on the mapping of terms to synsets of the WordNet hierarchy in order to organize the terms in a tree structure. However, another disambiguation strategy is applied. The mapping is performed in two

[7] API stands for application programming interface. The Google API allows, among other things, to programmatically perform web searches.

phases. In the first phase only the unambiguous terms are mapped which results in a "core-tree". Weights are assigned to the nodes of the tree depending on the number of related documents. Subsequently, the set of possible mappings is determined for each ambiguous term together with the respective paths to the root of the tree. For each such path the weight of the first node common with the core tree is determined and the path with the highest such weight is selected [SHR07]. In other words, from the several possible meanings for an ambiguous terms, the one belonging to the field of knowledge, that is most popular in the collection, is chosen. Eventually, the approach involves a little manual effort: The top level categories are pruned, so the tree is divided into a set of smaller trees representing hierarchically organized facets [SHR07].

Tomuro and Shepitsen present an approach that relies on Wikipedia as an external knowledge base [TS09]. However, no explicit knowledge is involved. In very simplified terms the algorithm first generates clusters of the Wikipedia articles and then represents tags by vectors expressing the similarity to each of these clusters. This similarity is measured by comparing the contents of the respectively tagged documents to the contents of the Wikipedia articles and it additionally takes into account the link structure of the respective hypertext documents. Subsequently, the tags are clustered in two phases. The first phase results in a set of tag clusters called *committees*. These clusters are not disjoint so that ambiguous tags occur in several committees. Finally, an agglomerative hierarchical clustering algorithm is applied to construct the hierarchy. The authors could show in an experiment that such a hierarchy is beneficial for the personalization of search [TS09].

Finally, Kiu and Tsui combine several heuristics and data-mining techniques — including k-means clustering and formal concept analysis — in the TaxoFolk algorithm [KT10, KT11]. The algorithm integrates a given hierarchical taxonomy with folksonomy tags. Rather than generating a tag mapping, the algorithm creates a hybrid structure of tags and the taxonomy concepts.

7.2. Manual structuring of folksonomies

In the following, we will cover approaches in which the automatic structuring of folksonomies plays at most a subordinate role. Instead, folksonomies are structured manually, for instance by establishing explicit tag relations — similar to TACKO. Sometimes semantic web technologies are applied to express relations among the concepts represented by tags but also to express facts about tag assignments as well. The benefits of the combination of the social web with the semantic web are well illustrated by Gruber [Gr08]. He sees the potential to transform the "*collected* intelligence" ,which manifests itself for instance in folksonomies, into "*collective* intelligence". As one promising approach that brings us further towards this goal he identifies capturing "structured data on the way into the system" [Gr08]. At the example of tagging this could mean, among other things, that the system makes it easy to turn for instance a keyword into a concept of a taxonomy at the very moment it is entered.

In [Gr07], Gruber lays the groundwork for ontologies of concepts related to the tagging domain, such as tags, users and tag assignments. They can be considered a prerequisite for a further integration of the social and the semantic web. Several such ontologies have been developed. An overview of is given in [Ki08].

The approach of Passant and Laublet builds on one of these ontologies, – the *Tag Ontology* [Ne05] – and allows to add the meaning of a tag to the resource-tag-user triples of the standard folksonomy model as an additional component [PL08, Pa09]. The authors present *MOAT* (Meaning Of A Tag), a "lightweight Semantic Web framework that provides a collaborative way to let Web 2.0 content producers give meanings to their tags in a machinereadable way" [PL08]. The framework aims to solve the problem of tag ambiguity and the lack of structure in folksonomies. The tag meaning is represented by a URI that denotes a concept in a knowledge base, such as for instance *DBpedia* [Au07, Bi09].

With MOAT, an ambiguous tag can be linked to different meanings in different contexts and as well can synonym tags refer to the same meaning. A MOAT server maintains a list of all global meanings of a tag. The meaning of a tag is manually selected by the users during the tagging process. If no appropriate meaning is offered by the MOAT server, a new URI that better represents the intended meaning can be associated to the tag [PL08].

Faviki[8] is a social bookmarking tool that implements a comparable approach. Users can tag web pages with Wikipedia terms. Figure 7.6 shows a screenshot of the tagging dialog. The autocomplete menu shows suggestions as the user types. Additional details are displayed for each entry when hovered with the mouse.

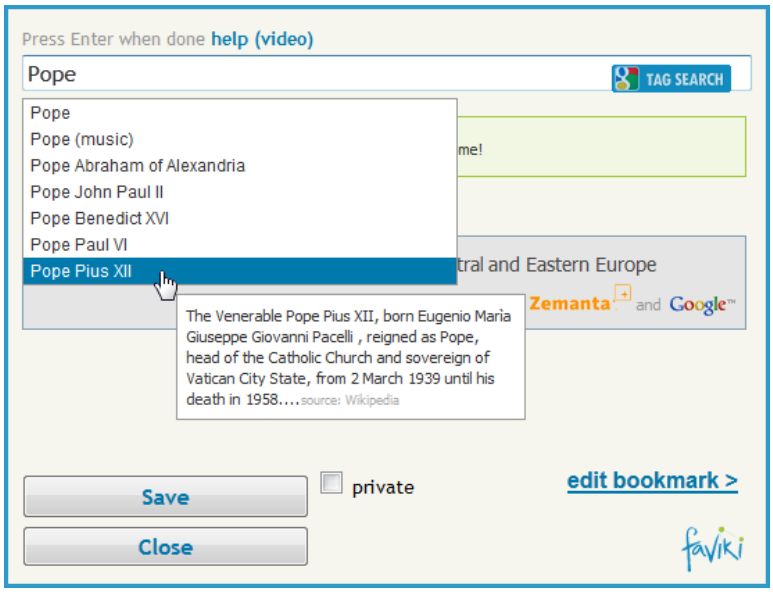


Figure 7.6.: Screenshot of the Faviki tagging dialog.

Peters and Weller discuss "tag gardening", the manual maintenance and refinement of folksonomy tags [WP08, PW08]. They pick up the metaphor from Governor [Go06] and describe several concrete activities that can improve the quality of a folksonomy. For instance, they suggest to eliminate spam and misspelled tags (*weeding*), introduce specific tags where necessary (*seeding*), or to "[combine] Folksonomies with other, more complex Knowledge Organization Systems which then act as fertilizers" [WP08]. They further suggest to group tags according to their purposes, i.e., to form facets. Golov, Weller, and Peters very briefly describe TagCare, a repository for the personal tags of a single user that allows to "create a

[8] http://www.faviki.com (accessed 11th of February 2013).

consistent cross-platform personal tagging vocabulary" [GWP08]. The web-based tool further allows to create hierarchical tag relations and link synonym tags. However, in [GWP08] the user interface is hardly covered and the service has meanwhile been discontinued.

Limpens et al. propose a bookmarking system that automatically analyses folksonomy tags and detects semantic relationships but that at the same time allows users "to validate or correct the automatically suggested tags and semantic relationships" [LGB09]. They describe a model that allows to link tags to semantic web concepts but also to express relations among the tags without reference to external knowledge bases. Thereby the relations are limited to "thesaurus-like relations as modeled in SKOS" [LGB10]. There may be even disagreement regarding tag relations. This means that contradictory statements about tags can — at least temporarily – exist in the system. Users can reject or approve semantic statements and the administrator of the system can decide whether contradictions are made visible or if they are resolved by taking into account the community to which a user belongs [LGB09].

The goals pursued by the creators of the *SOBOLEO* (Social Bookmarking and Lightweight Engineering of Ontologies) system [ZB07, ZBS09, Br11] are very similar to those that led us to the development of TACKO. The system serves as a tool for the collaborative annotation of web resources with concepts of a lightweight ontology. The kinds of concept relations being specified in the ontology are a subset of SKOS (cf. Section 2.2): For each concept, a list of broader, narrower and related concepts is maintained. The emergence of ontologies is facilitated by the possibility to assign arbitrary tags to resources as well. They are treated as "prototypical concepts" that can be integrated in the ontology later.

Zacharias and Braun stress in [ZB07] that the SOBOLEO system shall allow all users to extend the ontology according to their needs and there is no such role as an ontology engineer. While this is also similar to TACKO, the SOBOLEO systems separates the ontology editor from the browsing interface and the dialog used for the annotation of resources. All three user interface components are integrated in TACKO. Figure 7.7 shows the SOBOLEO taxonomy editor.

From a conceptual point of view, the TACKO organization structures is more complex since SOBOLEO has no notion of facets. Instead, the structure is mainly represented by a single tree of concepts. However, in SOBOLEO a concept can point to related concepts and it can have additional labels, comparable to snynonyms, which is not possible in TACKO. Another difference is that SOBOLEO focuses on the collaborative maintenance of the repository and the ontology. For this reason it also features a chat component that proved to be very beneficial for this purpose in several case studies [Br08, Br11]. In contrast, TACKO emphasizes functionality to improve the indexing quality of the collection, for instance with the "none of these" option in facets and by providing batch tagging operations.

Quintarelli et al. present the *FaceTag* system that allows to organize tags in a hierarchical faceted structure [QRR07, QRR08]. Figure 7.8 shows the browsing interface that allows to combine tag filters with filters for other metadata such as the publication date. In contrast to TACKO, the set of facets is fixed and thus it is not possible to create new facets in specific contexts. Additionally, a tag is always assigned to at most one place in a single hierarchical facet. Another difference is that the browsing mode is strictly separated from the administrative or editing mode.

Tanasescu and Streibel suggest *Extreme Tagging Systems* as an extension of common collab-

Figure 7.7.: Screenshot of the ontology editor of the SOBOLEO system [Br11].

orative tagging systems [TS07]. The central feature of these systems is that they allow to "tag the tags themselves, as well as relations between tags" [TS07]. Thereby tagging the tags can help to disambiguate different meanings while the annotation of tag relations results into triples that are comparable to RDF [9] triples: Annotating the relation between "dog" and "mammal" with the tag "is-a" results in the triple ("dog", "is-a" , "mammal"). However, the paper remains very vague with regard to how such annotations can be exploited.

Similarly, García-Castro et al. propose *TagSorting*, an approach to collaboratively derive an ontological model from a set of tags [GCHG10]. It encompasses the annotation of actors and tags themselves. However, the authors describe rather a guided modeling process that is based on a certain meta-model than an extension to a tagging system.

Another approach for structuring tag assignments is proposed by Gassler et alii [Ga12]. They introduce the *SnoopyTagging* concept which extends common tags with a *context* that is added as a prefix followed by a colon. Thus, the resulting tags have the form `context:tag`. This way the ambiguity of tags is reduced to some extent. If the tag stands for instance for the name of a person, the context can specify how the person relates to the resource. For instance in case

[9] http://www.w3.org/RDF/ (accessed 11th of February 2013).

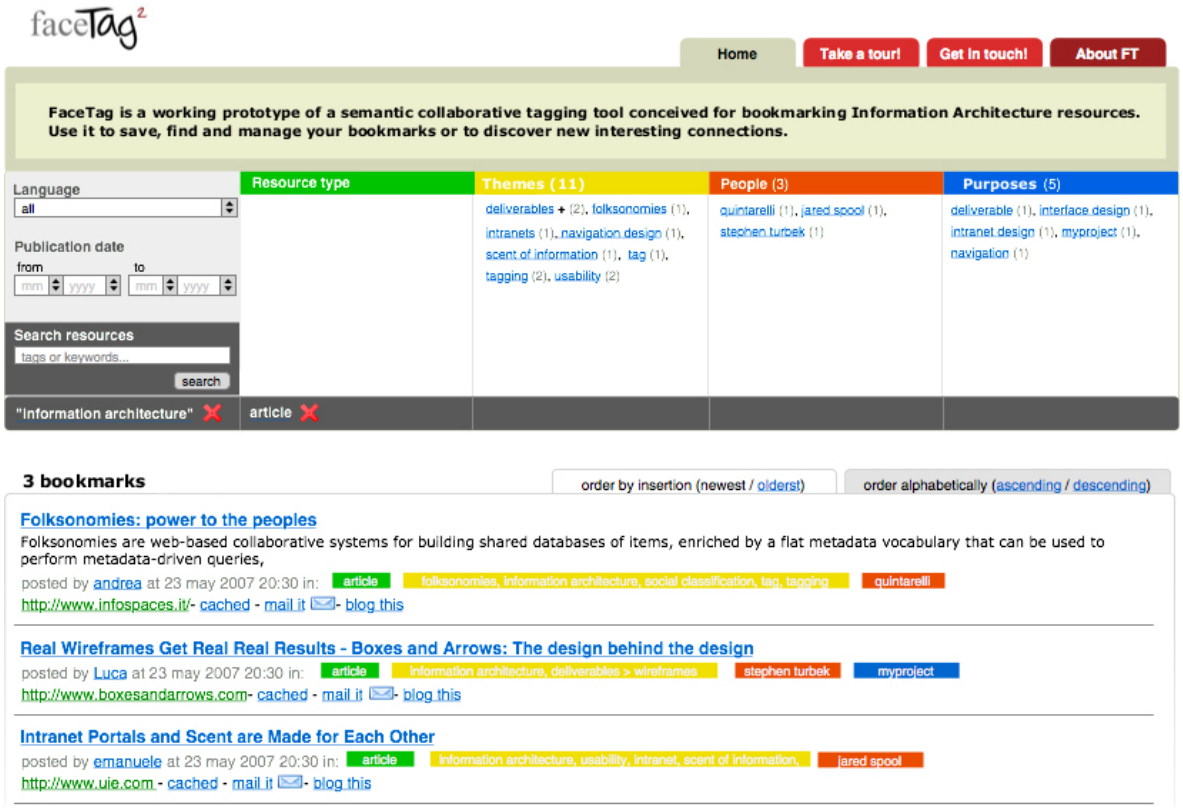


Figure 7.8.: Screenshot of the faceted browsing interface of the FaceTag system [QRR07].

of photos, the "photographer" context can be used [Ga12]. Specifying a context can further contribute to the completeness of tag assignments: The authors combine their approach with a tag recommendation algorithm that suggests contexts and tags depending on the given folksonomy and the tags already assigned to a resource. When the "photographer"-context is used, the system might for example further suggest to add a tag with context "camera" since this combination of contexts is used for many other resources [Ga12]. However, contexts and tags cannot be explicitly related.

7.3. Personal information management systems

First, it has to be noted that there are far too many applications for personal information management to mention all of them in this thesis. Instead, we focus on scientific approaches and commercial systems that are based on tags, combine tags with other structures or have other related properties that make them relevant for a comparison with TACKO. We include seminal research on such systems as well as the software applications being similar to TACKO or having interesting properties that informed the design of TACKO.

It is not easy to draw a line between the management of information resources and the organization of the categories to which the information resources are assigned. However, we will not include systems that focus only on the latter aspect. This means in particular, that

we will neither cover mind maps or concept maps nor tools for the management of more formal ontologies.

An early approach to overcome the limitations of traditional file systems was that of *semantic file systems* proposed by Gifford et alii [Gi91]. It is insofar comparable to tagging, that it gives users associative access to documents based on extracted attributes. Documents are accessed through *virtual directories* that dynamically reflect the set of documents matching a certain attribute filters. However, the authors suggest to extract the attributes automatically and they do not provide a graphical user interface. Instead, a command line interface is assumed [Gi91].

Although not directly related to tagging, the *lifestream* model developed by Freeman and Gelernter was developed with the goal to allow users a more flexible access to documents than with common directories [FG96]. Their model is based on a chronologically ordered stream of documents that can be dynamically filtered with search queries. The application of a filter creates a so-called *substream*: "[...] a substream is a temporary collection of documents that already exist on the main stream. Substreams may overlap and can be created and destroyed on the fly without affecting the main stream or other substreams" [FG96]. However, no structure can be defined on substreams.

As the approaches above, the *Presto* system was designed with the problems related to hierarchical file organization in mind [Do99]. Dourish et al. address the problem that documents are used by different people for different purposes by allowing to assign name-value pairs, i.e., "meaningful, user-level document attributes", expressing certain features of these documents, or the roles the documents play for the users [Do99]. Again, these attributes can then be used to define dynamic collections of documents. The approach can be seen as a special form of tagging, since the annotations are not just single terms but key-value pairs. The keys play a similar role as the facets in TACKO. However, neither are the annotations structured, i.e., there are no relations among annotations, nor did Dourish et al. develop a faceted user interface.

Cutrell et al. aim to harness the power of tagging for personal information management with the *Phlat* system [Cu06]. The system features an integrated interface for tagging and for filtering a user's personal content, such as files and emails. Documents can not only be filtered by tags but also by properties such as the file extension or the sender of an email. Similar to TACKO, the system also integrates search and browsing functionality. This means that users can issue queries with a text input and they can select filters from the navigation bar in the left part of the screen (see Figure 7.9). The tagging user interface is similar to that of TACKO because it also allows to drag and drop tags on information resources to assign them. However, an important difference is that tags cannot be grouped to facets but only hierarchical relations can be established. Faceted search is only supported for the built-in facets.

In the *Haystack* project, a powerful tool for the management of personal information was developed [AKS99, QHK03, Ka05]. The goal was to develop a customizable system that can be adapted to match each user's personal requirements. The haystack system "stores (by reference) arbitrary objects of interest to the user". It further "records arbitrary properties of and relationships between the stored information that the user considers important" [Ka05]. These properties are stored as RDF triples. Additionally, information objects are organized

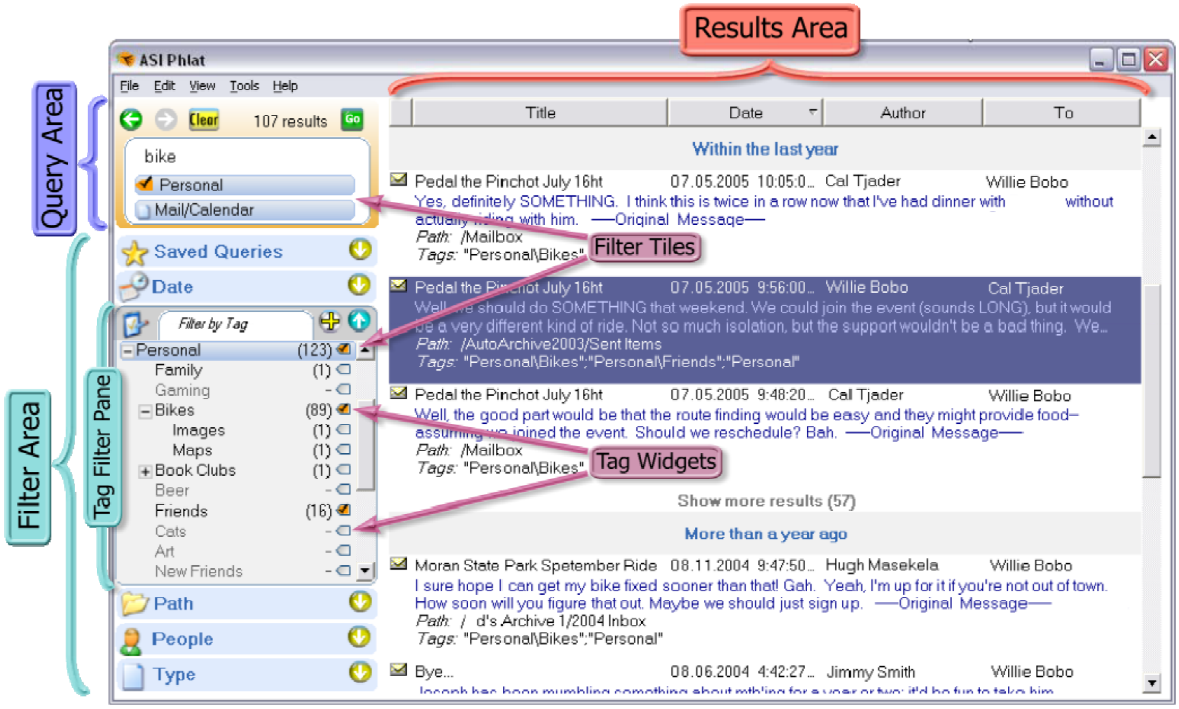


Figure 7.9.: The Phlat interface [Cu06].

in named *collections*. These collections can be compared to tags insofar as each object can be part of many collections and users can create new collections with arbitrary names. However, collections are not organized with subsumption or facet relationships. Compared to TACKO, haystack is rather heavyweight. It builds on semantic web technologies to express relations between information objects and uses separate concepts for organizing objects in collections and for expressing their attributes.

The web-based email service *Gmail*[10] provides a combination of tags and folders for the organization of email [RL10]. From a technical point of view, the solution is based on tagging (tags are called *labels* in Gmail). However, tags can be used like folders to some extent. Incoming mail always has the label "inbox", and the default view shows all messages with this label. Now, there are two ways to assign labels to a message: Users can *label* a message or *move* the message *to* a label. In the latter case, the current label – such as "inbox" – is removed, so it is comparable to moving a message to another folder. Both actions are also available via drag and drop operations. When the user drops a label on a message, the message is labeled. When the message is dropped on a label, the message is moved [RL10].

Gmail further features so-called "nested labels" or "sub-labels". A nested label can be created by including a slash in the label name. For instance, if there is a label "super", a new label "super/sub" will be displayed as a subordinate tag of "super" in a tree view. From a technical point of view, there is no real structure among the tags but the feature can rather be considered a convention regarding how the tags are displayed in the navigation bar. It is for example possible to create another label "sub" which is not related to "super". It is also possible to use

[10] http://gmail.com (accessed 21st of February 2013).

the labels “super” and “super/sub” independent from each other, so messages can be labeled “super” and “super/sub” at the same time.

The personal information management software *TheBrain*[11] — formerly called *PersonalBrain* — allows to attach information resources to so-called *thoughts*. A thought is a named concept that is related to other thoughts. Thoughts and their relationships form the basis for knowledge organization in TheBrain. However, the purpose of thoughts is not only the organization of knowledge but also its representation. Although several files and a textual description can be attached to a thought, it often makes sense to use thoughts that just have a name and connections to other thoughts. Figure 7.10 shows a screenshot of the desktop software. The focus is always on one thought which is displayed in the middle of the screen. Superordinate thoughts or “parent thoughts” are displayed above and children are shown below. Sibling thoughts, i.e., thoughts sharing a parent with the current thought are displayed to the right. It is also possible to create so-called “jumps” which are crosslinks to other related thoughts. These thoughts are displayed to the left of the current thought. Furthermore, labels and types can be assigned to thoughts.

Although the basic organization structure of TheBrain that is mostly determined by parent-child relationships among thoughts is similar to the structure of subsumption relationships in TACKO, there are several differences: In TACKO, no cycles are possible in the subsumption graph. Additionally, a resource can only be assigned to one thought in TheBrain while it can be tagged with several tags in TACKO. However, on can create a new thought for a single resource and assign several other thoughts as parents to circumvent this restriction. Finally, there is no notion of facets in TheBrain.

With the *TagTree* approach, Voit et al. address the problem that files can only reside in one place within a folder hierarchy [VAS11]. Their implementation called *tagstore* allows the user to assign tags to files and it dynamically generates hierarchical navigation structures from these tags in order to facilitate file access. Figure 7.11a shows an example of a folder structure generated for a single resource with three tags. There are no explicit relations defined on the tags, so at the bottom level of the generated hierarchy, the user has to select a tag out of all the tags used in the system. It is also not possible to group tags in facets. As a consequence the (separate) tagging interface is not as structured as in TACKO (see Figure 7.11b). A user experiment showed that fewer mouse clicks are required to locate in documents stored in tagstore compared to classical folder structures [VAS12]. However, more time was required for storing the documents.

The TagTree approach is similar to *TagFS* presented by Bloehdorn et alii [Bl06]. Like TagTree, TagFS gives users access to tagged documents through a virtual hierarchical folder structure. TagFS has no separate tagging interface but the operations performed in the folder structure (such as moving and copying documents) are mapped to the documents’ tags [Bl06].

As already noted above, we only focus on systems and scientific approaches that are related to tagging or in which tagging plays a major role. However, for the interested reader we want to at least mentioned two more research projects on personal information management here. The *MyLifeBits* platform was developed as a system to store “web pages, phone calls, meetings, room conversations, keystrokes and mouse clicks” of its users[GBL06]. One of the design goals was to avoid that users are “constrained by a strict hierarchy” [Ge02]. Similarly,

[11] http://www.thebrain.com (accessed 26th of February 2013).

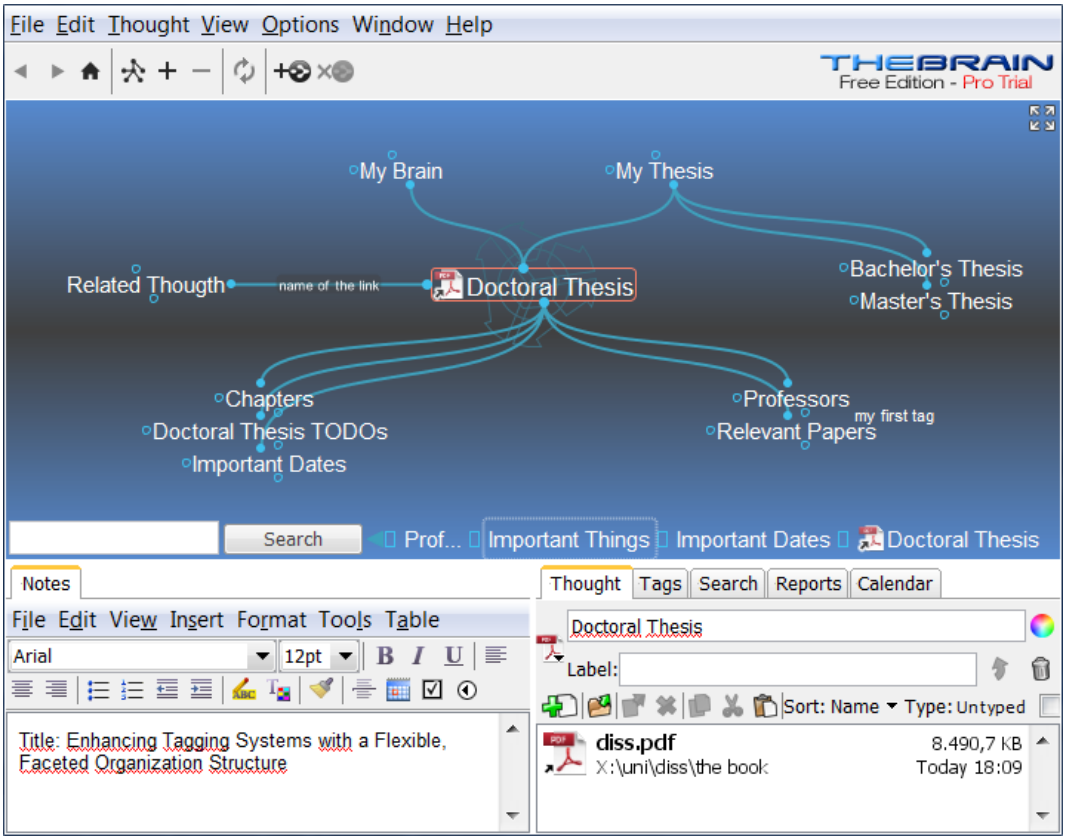
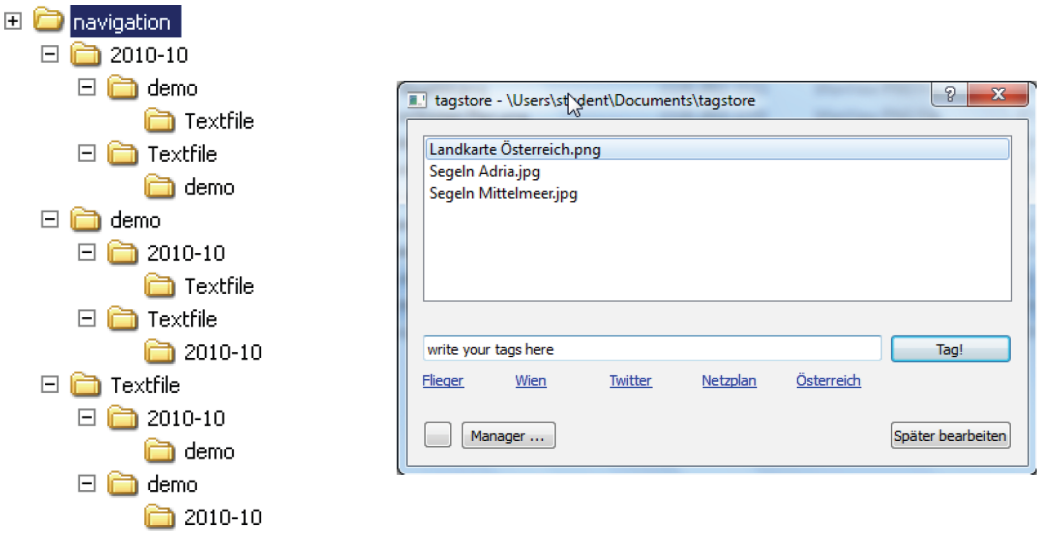


Figure 7.10.: Screenshot of the personal information management software *TheBrain*.



(a) Generated folder structure

(b) Tagging interface

Figure 7.11.: Illustration of the browsing and tagging interface of the *tagstore* system [VAS12].

the *Stuff I've seen* system focuses on refinding files and web pages a user has visited [Du03]. In contrast to TACKO, static facets such as document type, time of last access, and other document attributes are applied.

7.4. Hybrid Wikis

As already stated above, the *Hybrid Wikis* concept was also realized as part of the Tricia platform (cf. Section 5.1.1). It was developed in parallel to TACKO and serves a complementary purpose: While TACKO is applied to categorize information resources — such as wiki pages —, their contents can be structured with Hybrid Wikis. Nevertheless, both approaches are similar in various aspects. Before we discuss how Hybrid Wikis relate to tagging, we will first introduce their elementary structural concepts. A much more detailed description of Hybrid Wikis and related research can be found in [MNS11] and [Ne12].

As the name suggests, Hybrid Wikis were first applied to structure the contents of wiki pages. Traditional wikis are mostly text-based and do not allow to access their contents in a structured way. It is for instance not possible to issue a query for wiki pages about persons born in a certain year but only full-text search is provided. In contrast, so-called *semantic wikis* use semantic annotations of wiki pages to formalize the meaning of their contents. However, to take advantage of these tools, the user usually requires some understanding of semantic web technologies. Hybrid Wikis aim to provide a more lightweight way of structuring than semantic wikis while still allowing to access the wiki contents in a structured way. Two basic constructs are applied for this purpose: *attributes* and *type tags* or just *types*.

Attributes are simple key-value pairs that can be added to a wiki page. The key of an attribute is an arbitrary text label and an attribute can have more than one value. Each value is either a literal or a reference to another wiki page. Attributes can be dynamically added to a page as required. There is no need to first extend a schema or data-model. They are displayed in a table in the right half of the wiki page as shown in the screenshot in Figure 7.12. Below the attributes, the table contains references from other pages. If for instance a page about a person "worker" has an attribute "boss" pointing to another wiki page, this other page will show the reference with the key "boss of" and value "worker". However, an attribute is always contained in only one of the pages, in this case in the "worker" page.

Type tags can be used to express the type of the object being described on the wiki page. They are displayed above the attributes (see Figure 7.12). As for normal tags, an arbitrary number of type tags can be assigned to a page and type tags are simple text labels. Type tags make it easier to structure wiki pages because they are loosely connected to attributes. When a type tag is assigned to a page, the system generates attribute suggestions for this page based on the most frequent attributes of other pages with the respective type tag. The suggestions are shown below the attribute list. Type tags can also be used for the retrieval of pages or to generate overviews of pages of a certain type. Figure 7.13 shows an overview table of wiki pages with the type tag "research project". Each row represents a wiki page and each column corresponds to a common attribute. The columns are sortable and attributes can be edited directly in this table. It was attempted to design the tables in a way that they resemble spreadsheets since users are familiar with this form of structured information management.

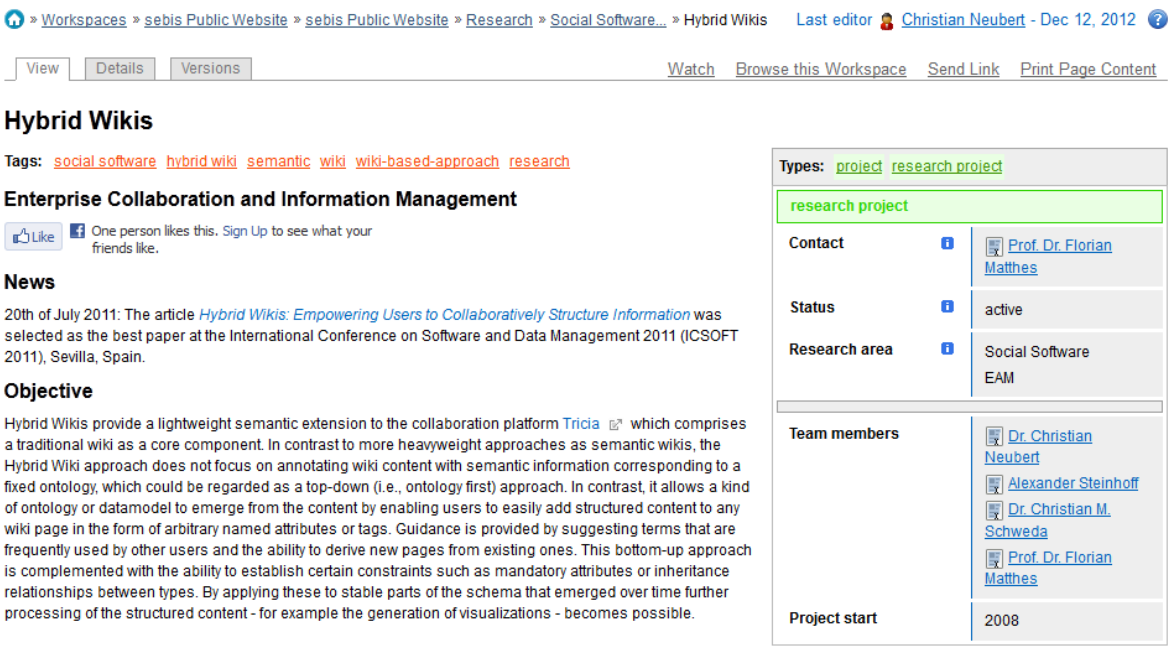


Figure 7.12.: Type tags and attributes displayed in a wiki page.

Search for

Name	Research area	Status	Project start	Contact	Project end	Acronym
[CALM3] Complexity of Application Landscapes - Models, Measures, Management	EAM	active	2013			CALM3
Agile Enterprise Architecture Management Survey	EAM	active	2012	Matheus Hauder Sascha Roth		
Automated Enterprise Architecture Documentation Survey	EAM	active	2012	Matheus Hauder Sascha Roth Matthias Farwick		
BEAMS - Building Blocks for EAM Solutions	EAM	active	2009	Dr. Sabine Buckl Alexander W. Schneider		BEAMS
Business and IT transformation in the automotive industry	EAM	active	2008	Christopher Schulz		
Content Factory	EAM	completed	2008	Thomas Büchner	2008	
EAM for the German Public Sector	EAM	active	2010	Sascha Roth		
EAM in the context of M&A	EAM	active	2009	Christopher Schulz Andreas Freitag		
EAM KPI Catalog	EAM	active	2011	Ivan Monahov	01.06.2014	EAM KPI Catalog
EAM Pattern Catalog	EAM	completed	2008	Dr. Sabine Buckl	2010	

Total results: 25 Show 10 entries Page 1 of 3

Figure 7.13.: Overview table of wiki pages with the common type tags “research project”.

Attributes and type tags can be combined with a full-text search to filter out particular sets of pages. Additionally, attributes and type tags can be explicitly connected in so-called *type definitions*. The latter also allow to define constraints on attribute values and multiplicity. In later development stages, attributes and types could also be applied to other resources, such as files and user profiles.

Apart from the possibility to define types and attribute constraints, the structuring concepts of Hybrid Wikis and TACKO are closely related. Obviously, the type tags are a special kind of tags with additional semantics. While a normal tag can just express that a resource is related to a certain topic or concept, a type tag represents a more precise statement. A resource being tagged "software" can be a blog post about software development, a list of software products or a particular software product. If a type tag is used instead, it is clear that the tagged resource represents a piece of software.

There is also an analogy between attributes and TACKO tags. Like tags, many attributes can also be considered metadata of the resource. Although they structure part of the resource's content, they are also used for accessing resources and serve as descriptive metadata (we refer to [TJ09, Chapter 4] for an overview of the different kinds of metadata). This is best illustrated with a small example: Among many other things, Hybrid Wikis were used for the management of bug reports during the development of Tricia. Each bug report was recorded on a wiki page with a respective type tag and attributes. For some properties of the bug reports it was not always clear whether they should be better expressed as tags or as attributes. The status for example can also be easily expressed with the tags "unconfirmed, "confirmed", or "fixed". Since these tags can be grouped to a facet, tags are in this case as powerful as attributes, at least with regard to systematically retrieving bug reports. Facets can also provide guidance during data entry.

However, an attribute value can be a reference to another wiki page but this is not possible with tags. Furthermore, the attribute name contains additional information that cannot be expressed in a simple tag. If a bug report is just tagged with the name of a person, it is not clear whether the person reported the bug or whether the person is supposed to fix it. There are workarounds such as prefixing the tag with the role of the person but this limits querying capabilities: To find all resources related to the person, one would have to search for all possible prefixes as well. This makes retrieval more complex and error prone.

Finally, one advantage of TACKO tags shall not be left unmentioned here: In contrast to attribute values, tags can be organized with subsumption relations. Assume for instance that for each bug report it shall be recorded which software component is probably affected. Furthermore, it shall be possible to search for all bug reports related to server components and client components respectively. With TACKO, subsumption relations can be established among tags for the components and the tags "client" and "server". A bug report then only has to be tagged with the component name and the respective subsuming tag will be added automatically. With attributes this is not possible. One would have to maintain two separate attributes and assert that they are used consistently.

Hybrid Wikis have been applied for various tasks ranging from the management of the bugs and feature requests during their development to technology surveillance in a university context [IAMS11] or enterprise architecture management [MN12]. Starting as a research project, they have now become an essential feature of the commercial Tricia platform.

CHAPTER 8

Conclusion

This chapter concludes the thesis with a summary and an outlook on future work. Section 8.1 summarizes the findings of this thesis with respect to the research questions posed in Section 1.3. Finally, Section 8.2 points out opportunities for future research.

8.1. Summary

This thesis primarily examined the general question whether tagging systems can be enhanced in a way that makes them suitable for the management and structured organization of collections of information resources and, if yes, how this can be established. In order to answer this question, a design science approach was followed. We developed a prototype of a tagging system extension as part of a web-based enterprise collaboration platform. This allowed the author and his colleagues to continuously evaluate the prototype during their everyday work.

In the first development phase, it was attempted to structure tagging systems by means of so-called implicit tag relations. The relations among tags were not made explicit but were instead embodied in the entirety of tag assignments. Three different kinds of implicit tag relations were identified and supported by the prototype: tag equivalence, subsumption and mutual exclusion. While subsumption is a sign of a broader/narrower relationship with regard to the semantics of the tags, mutual exclusion indicates that two tags belong to the same dimension of classification, i.e., the same facet.

Although the first experiences with implicit tag relations were promising, it was found that they could not be reliably detected within small resource sets. For this reason, it was decided to make tag relations explicit in the second development phase. This led to the development of the TACKO organization structure. Its prototypical implementation includes an integrated

user interface for searching and navigating the tagged contents as well as for the management of the tag relations.

Research question 1, the primary research question — whether and how an enhancement of tagging systems is possible — was broken down into more specific research questions. Research question 2 asks whether information can be accessed more reliably in the enhanced tagging system and whether users are faster in finding information. To answer this question, we conducted an experiment in which 152 participants performed different information retrieval tasks in a collection of tagged photos. We compared their performance for three different variants of the tagging system. In the first variant, the raw folksonomy was used as it had emerged on a public photo sharing platform on the web and the participants worked with a standard user interface. The second variant employed the same standard user interface but the tag assignments had been harmonized and augmented with TACKO. Finally, the third version combined this organized collection and the TACKO user interface. For this combination a significant increase in the success for photo searches was observed compared to the first variant. While the results suggest that this increase is to a large extent due to the refinement of the tag assignments, this would not have been possible without an appropriate tool, such as TACKO. Our results further indicate that TACKO improves the perceived organization quality.

Research question 3 is concerned with the transformation of existing folksonomies into a more structured form. As described above, a real-world folksonomy that had been retroactively organized served as the basis for the experiment. The reorganization required a user interface that allows to modify the tag assignments of specific sets of resources. Furthermore, the tag assignments of a large number of resources had to be updated efficiently. It has been demonstrated that a viable solution is to temporarily record the update operations in the form of rules until they are materialized as changes of the actual tag assignments. Until then, the rules are transparently applied during access to the tagged information resources.

In how far non-expert users are able to understand the structuring concepts that were developed in this thesis is the focus of research question 4. User tests and interviews revealed that instructions are required if users have to organize information resources themselves. This is probably due to the fact that the TACKO organization structure is different from most other structures that users are familiar with. However, if the users were just browsing or searching for a particular resource, they could use the system effectively with no or only very little additional information. This was also confirmed by the results of the experiment.

Research question 5 addresses the interesting aspect in how far knowledge can be expressed in the organization structure itself. Similar to other organization structures, such as trees, the organization of information resources with TACKO goes along with the expression of knowledge that is represented by the relationships among categories. This is not the primary purpose of the structure, but it is a valuable complement to the organization of information resources. In TACKO, tags are related by means of subsumption relations and facets. Although the semantics of a particular relation is rather unspecific, e.g., it is not possible to specify a part-of relationship, we demonstrated in several examples, that the combination of both kinds of relationships — subsumption relations and facets — allows to represent complex networks of relations. Thereby, it is advantageous that a tag can be subsumed by several other tags. For instance can a tag that represents an event be subsumed not only by the tag “events” but also by tags representing years or locations. We did not employ semantic

web technologies, so with regard to the concepts of the organization structure, there is no interoperability with other knowledge organization systems or external ontologies.

Research question 6 addresses the trade-off between the flexibility of tagging systems and the utility of an organization structure which requires a certain degree of rigidity with regard to the consistent usage of tags. The initial approach, i.e., only organizing resources with implicit tag relations, was driven by the attempt to minimize the loss of flexibility that goes along with the introduction of structure. The idea was that tags can be freely used as required and the implicit structure reflects the usage of tags. By automatically deriving the structure from the folksonomy, users can be guided towards a consistent categorization, but the structure follows the tag assignments and not the other way around. Later, when tag relations were explicitly represented in the second development phase, it was tried to preserve this flexibility by only making explicit the kinds of relationships that were already implicitly present before.

Nevertheless, the existence of an organization structure that is separate from the tag assignments implies some restrictions. Primarily the explicit subsumption relationships constrain the usage of tags to some extent. While implicit tag relations allowed to derive the meaning of a tag from the usage context, global subsumption relations relate a tag to other tags regardless of the context. For instance could the same tag "apple" be used in the context "fruits" and "consumer electronics" but explicit relations require that distinct tags, such as "apple (fruit)" and "Apple Inc." are used. In contrast to subsumptions, the facets have no implications on the tag assignments. Thus, they do not limit flexibility but they rather make visible the inconsistent usage of tags.

Subject of research question 7 is the technical realization of a system for structuring and systematically browsing tagged information resources. We demonstrated how such a system can be implemented based on a full-text search index and a common relational data store. A particular challenge was to efficiently apply changes of the tag assignments of many resources to the data store. It was shown how this can be accomplished without a notable delay for the user. We further demonstrated how to facilitate the tagging process with an autocomplete functionality that takes users' access rights into account .

In summary, we can conclude that we could enhance tagging systems with TACKO, an extension that allows to establish a lightweight tag-based organization structure. Existing tags can be taken as a basis for such a structure and TACKO can be smoothly used in combination with tags that have not been organized yet. An experiment showed that users were more successful when searching for particular resources in a collection that had been organized with TACKO. Furthermore, the participants perceived the collection as better organized and they changed their behavior: Rather than searching for specific tags, they navigated along the categories of the structure.

8.2. Outlook

In this thesis, we could show that there is potential to improve existing tagging systems by allowing a more structured access to its contents. However, this research can only be considered a first step towards a new generation of tagging systems that do not only facilitate serendipitous findings but that are able to replace conventional tools for the organization of

information, for instance in work environments. The following sections briefly cover unsolved problems as well as challenges and opportunities with regard to further research.

8.2.1. Further simplification

During the development of TACKO, our goal was to preserve the flexibility of tagging systems as far as possible. This led to a powerful organization structure. However, we also found that users had problems to apply the structure for the organization of resources without an explanation. We also observed that the users rarely took advantage of all the capabilities of the organization structure. This suggests that we have not yet reached the optimum with regard to the tradeoff between ease of use on the one hand and flexibility on the other hand.

More user studies with variations of the prototype are required to explore opportunities for improvement. A possible simplification that could be investigated would for instance be the introduction of the constraint that all tags in a facet have to be subsumed by the tags in the facet context. Currently it is for example possible that a facet defined in the context {“cars”} contains a tag “bargains” that is not subsumed by “cars” although it stands next to other tags, such as “sports cars”, that are subsumed. While this can be practical during the process of organizing the collection, it makes the organization structure more complex and it is hard to explain to users.

Apart from a simplification of the organization structure, we also see potential in the further improvement of the user interface.

8.2.2. Standardized experiments for the evaluation of knowledge organization systems

In contrast to the information retrieval (IR) community that regularly compares different approaches and algorithms for information retrieval based on benchmark datasets and tasks (cf. [VH05]), there are no such widely accepted standards for the evaluation of knowledge organization systems. To some extent this is certainly due to the fact that comparing algorithms for text based information retrieval is simpler than comparing different kinds of knowledge organization systems:

- An IR dataset is usually fixed and does not have to be organized.
- Apart from assessing the relevance of each information resource with regard to a certain query, human participants are not required when IR algorithms are compared. Once the relevance of resources in a given dataset has been determined, different algorithms can be objectively evaluated.
- There are undisputed metrics, such as recall and precision, for the evaluation of IR algorithms. A textual query yields a result that can be evaluated based on these metrics. On the contrary, there are many ways to use a knowledge organization system and thus it is not clear how to evaluate it. There are different ways to enter and to access information.

Nevertheless, a certain degree of standardization is required for an objective evaluation of knowledge organization systems. In our experiments, the tasks users had to complete followed those employed by other researchers (e.g., [Tr12]). However, without standards for the process of organizing the resources, the results can hardly be compared.

For future research, we see an opportunity in the availability of web-based crowdsourcing services such as the one we used for our experiment. Such services make it possible to conduct experiments with hundreds of participants at very low cost. Thereby, the large number of participants can compensate for the fact that the experimental conditions, such as the used hardware and the level of distraction during the experiment, may vary.

8.2.3. Application in practice

While experiments can help to estimate the utility of a system, they are no substitute for an evaluation in practice. Only over time and with users working on real problems can we answer questions like the following:

- Are there different classes of users with regard to their role in the collaborative organization of resources?
- For which purposes is the system applied? Do users apply the system in ways that were not anticipated?
- Is the system superior to conventional knowledge organization systems in everyday use?

With regard to the second question, we hypothesize that some users will apply TACKO as a tool for designing taxonomies and not primarily for the organization of resources.

8.2.4. Generalization of the approach

Finally, we like to point out another direction for the further development of TACKO that challenges a fundamental assumption regarding the structure of a tagging system. Apart from users, the fundamental entities in a tagging system are tags and information resources. However, during our work with the TACKO prototype, we observed that there is often a strong correspondence between a certain tag and an information resource. For instance the name of a person can occur in form of a tag and it can as well be the title of a document. This increases complexity and is a potential source of inconsistencies. Therefore, we propose to examine in how far it is possible to blur the distinction of tags and information resources and thereby allow to organize the tags themselves in the same way as resources are organized with TACKO.

APPENDIX A

Experimental Setup

In the following sections it is described how the participants of the experiment, that has been described in Section 6.2, were guided through the different tasks. A description of the initial task indipendent briefing is included as well as the final short survey after task completion.

A.1. Task independent briefing

The persons participating in the experiment were hired using the crowdsourcing platform *MobileWorks*[1]. They accessed the platform via their web browser and could see short descriptions before accepting the tasks. They started a task by following a certain task-specific URL defined by the task providers. In our case they were forwarded to a certain page on a web server running a Tricia installation that has been set up for the experiment. In addition to the TACKO extension, further functionality has been implemented that realizes all features required for the execution of the experiment.

The task description on the MobileWorks platform was as follows:

> You will have to use and evaluate a tool for browsing photos. Your task will either be to find a specific photo or to select several photos of a certain kind. More detailed instructions follow when you access the URL.
>
> Technical requirements:
>
> - a modern web browser with cookies enabled
> - a desktop or laptop computer (no mobile devices supported!)
> - a pointing device such as a mouse or a touchpad (not a touchscreen)

[1] http://www.mobileworks.com, (accessed 3rd of January 2013).

> Please abort the task when you have the feeling that either your browser is not supported (the website looks "broken") or your computer is too slow to display the interface in a proper way!
>
> Please read the instructions carefully before you start. If you do this task more than twice, the instructions will be much shorter. Also pay close attention to changes in the user interface and the photo categories when you do several tasks. You will have to rate them afterwards.
>
> When you finished a task, you will see a four digit number that will verify that you completed the task. Please enter it in the respective answer field.

After accepting a task, the participants were sent to the web page primarily containing the blue box shown in Figure A.1. After filling out the small form, they were forwarded to the explanation of a particular task.

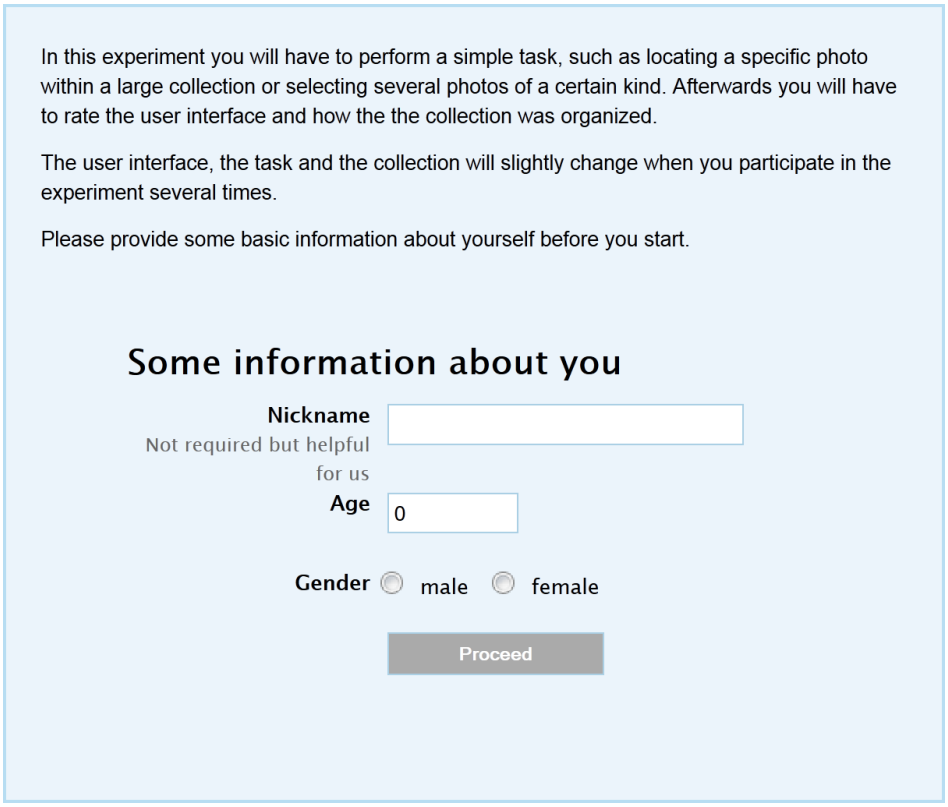


Figure A.1.: Main content of the screen shown to participants after they accepted a task.

A.2. Task explanations

Participants accepting a task were either assigned a search or a collection task (see Section 6.2.5). They were shown a brief explanation of the TACKO user interface and the task-specific controls before they could start.

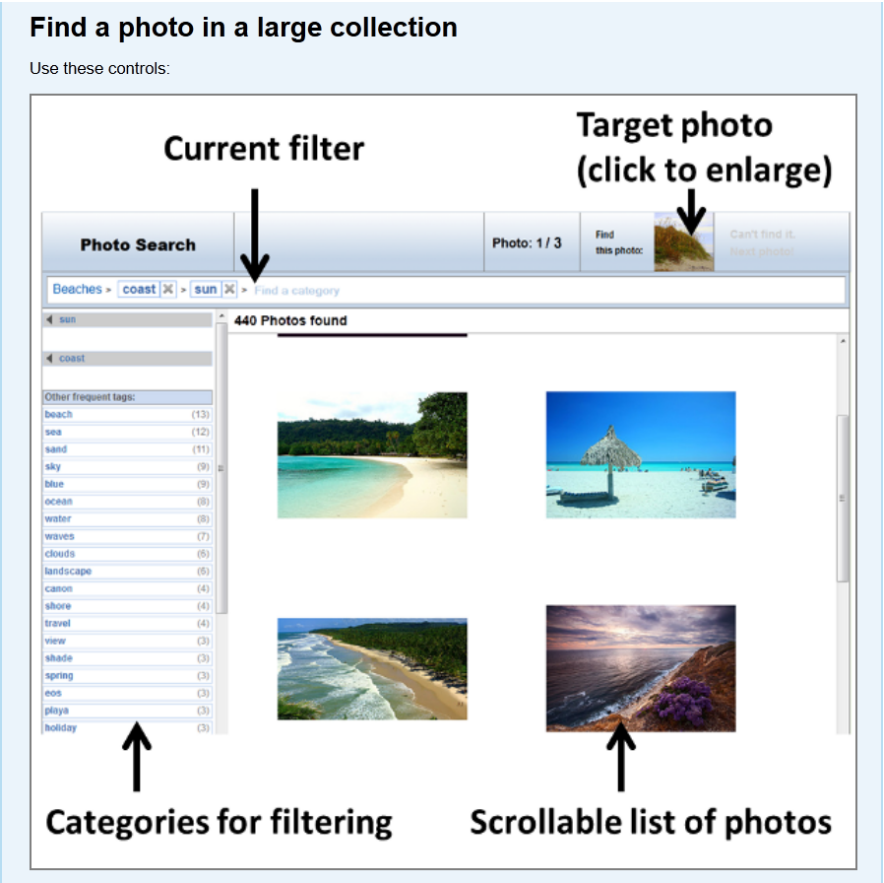


Figure A.2.: First part of the briefing page the participants were shown before starting a search task.

A.2.1. Search task

Figure A.2 shows the upper part of the web page the participants saw just before starting a search task. Below the image, some additional textual explanations were shown:

> When you have found the photo, click it. The next photo will be shown at the top.
>
> Choose categories from the left or enter arbitrary categories into the filter at the top to narrow down the collection. Click the "X" next to a category to remove it from the filter.
>
> **Important:**It happens often that the target photo is not categorized properly. If you cannot find it immediately, try other combinations of categories or remove categories from the filter again.
>
> If you tried for some time and it is impossible to locate the photo, click the button at the top right of the screen ("Can't find it").

If the participant had completed a search task before, the explanation was shorter. Neither the image nor the first two paragraphs of the explanation were displayed but they could be accessed by clicking a link labeled "Review basic user interface instructions". Instead, the following hint was shown:

> **The user interface and/or the categories have changed compared to the last time you searched a photo. Note that the categories on the left may be helpful for finding the photo faster.**

A.2.2. Collection task

The explanation of the collection tasks started with the particular task instructions and was followed by a graphical illustration of the user interface as shown in Figure A.3.

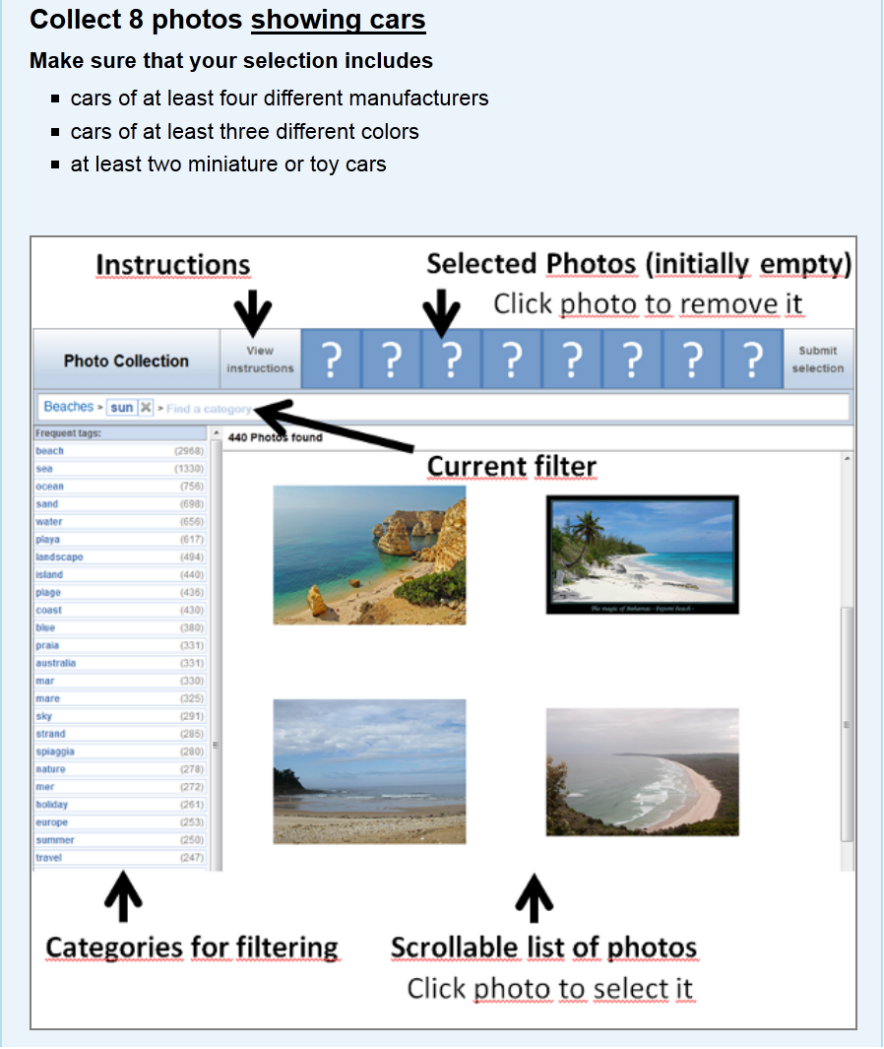


Figure A.3.: First part of the briefing page the participants were shown before starting a collection task.

Below the image, the following text was shown:

> To select a photo, just click on it. It will appear at the top where you see the question marks. Remove a photo with a single click.
>
> Choose categories from the left or enter arbitrary categories into the filter at the top to narrow down the collection. Click the "X" next to a category to remove it from the filter.
>
> **Important:** Do not just rely on the categories! Look at the photo before you select it.
>
> Note that you can review the instructions by clicking the button at the top of the screen (left of the questions marks).

As for the search task, the basic user interface instructions were hidden when the participant had already completed a collection task before. Instead, the following text was shown.

> **The user interface and/or the categories have changed compared to the last time you collected photos. Note that the categories on the left may be helpful for finding certain photos faster.**

After starting the task, the user could review the specific task instructions by clicking a button at the top of the screen. The instructions were then displayed in a dialog window as shown in Figure A.4.

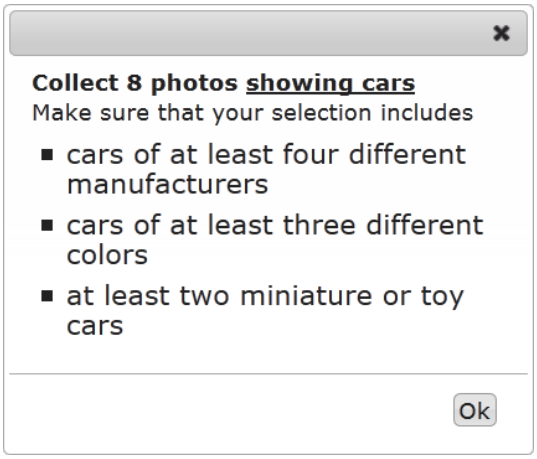


Figure A.4.: Review of collection instructions after the task has started.

There were three different collection tasks. The respective instructions are given in the following sections.

A.2.2.1. Car photos

Collect 8 photos showing cars

Make sure that your selection includes

- cars of at least four different manufacturers
- cars of at least three different colors
- at least two miniature or toy cars.

A.2.2.2. Food and drinks

Collect 8 photos showing something to drink AND something to eat

Make sure that

- four of the photos clearly show a drink that typically contains alcohol.
- The other four photos show drinks that generally do not contain alcohol.

A.2.2.3. Things you can wear

Collect 8 photos showing things you can wear

Make sure that

- four of the photos show **only** things you wear on the upper half of your body
- the other four show **only** things you wear on your feet or legs
- at least three photos contain mainly purple colored things.

A.3. Post task survey

After completing a task, the participants answered the very short survey displayed in Figure A.5. A search task was completed after three photos had been searched. A collection task ended after eight photos had been collected.

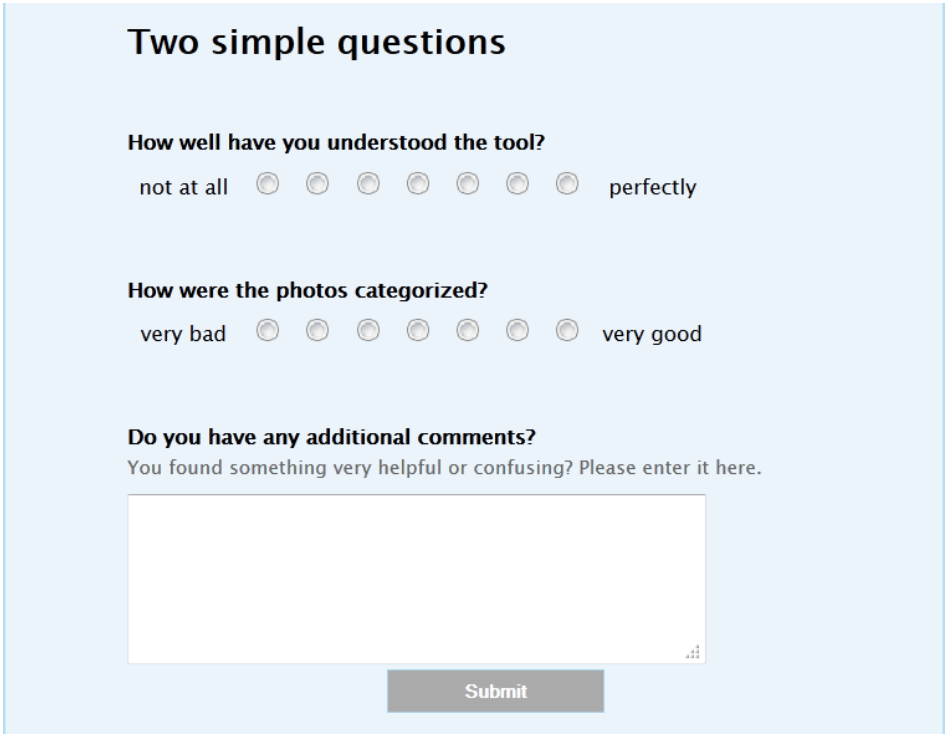


Figure A.5.: Short survey answered after task completion.

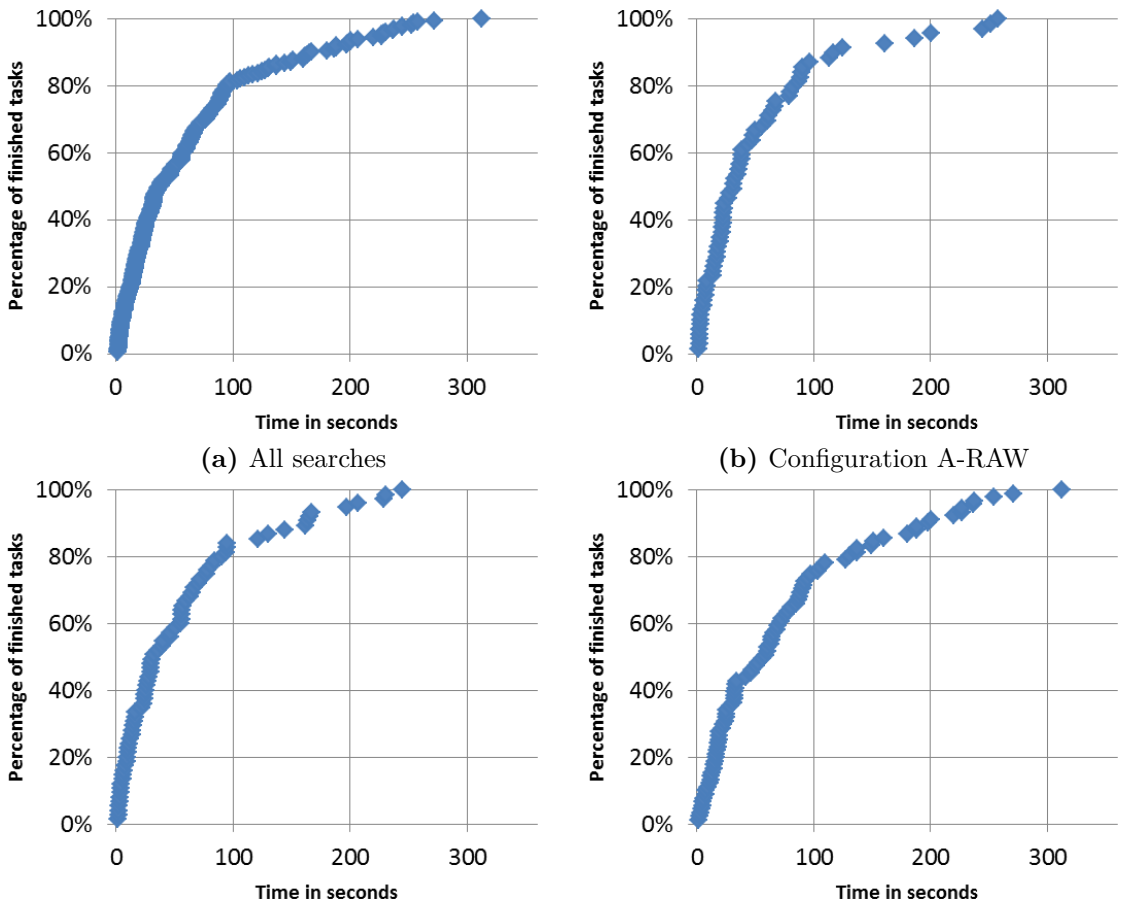
APPENDIX B

Experiment Measures

On the following pages, we illustrate the results of different measures taken during the experiment that have not yet been shown in Section 6.2. Figure B.1 shows the distribution of the time required to find a photo in the different configurations. It can be seen that after 100 seconds nearly 90% of successful photo searches were finished in configuration A-RAW while less than 80% were finished in configuration C-TACKO.

Figure B.2 shows the respective plots of time required for collection tasks. It can be clearly seen that the distributions are very different, particularly if one compares A-RAW and C-TACKO. The plots for A-RAW and C-TACKO both resemble straight lines when the single outlier in C-TACKO is ignored. However, the gradient is much higher for C-TACKO.

Finally, Figure B.3 shows the distribution of answers to the question how well the tool was understood and Figure B.4 visualizes how the answers to the question regarding the organization of the photos are distributed. In both charts, no significant differences among the configurations can be observed.



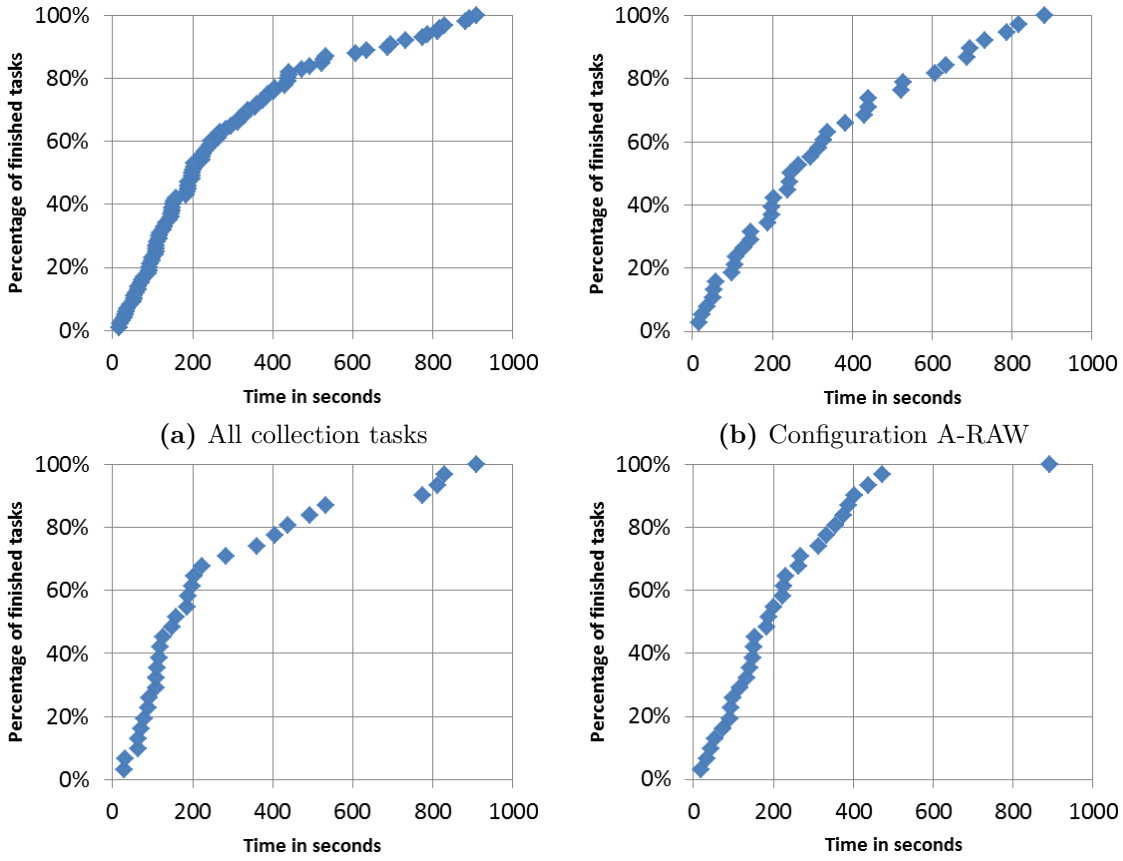
(a) All searches

(b) Configuration A-RAW

(c) Configuration B-ORGANIZED

(d) Configuration C-TACKO

Figure B.1.: Percentage of photos found (y-axis) found in less than x in seconds.



(a) All collection tasks

(b) Configuration A-RAW

(c) Configuration B-ORGANIZED

(d) Configuration C-TACKO

Figure B.2.: Percentage of completed collection tasks (y-axis) finished in less than x seconds.

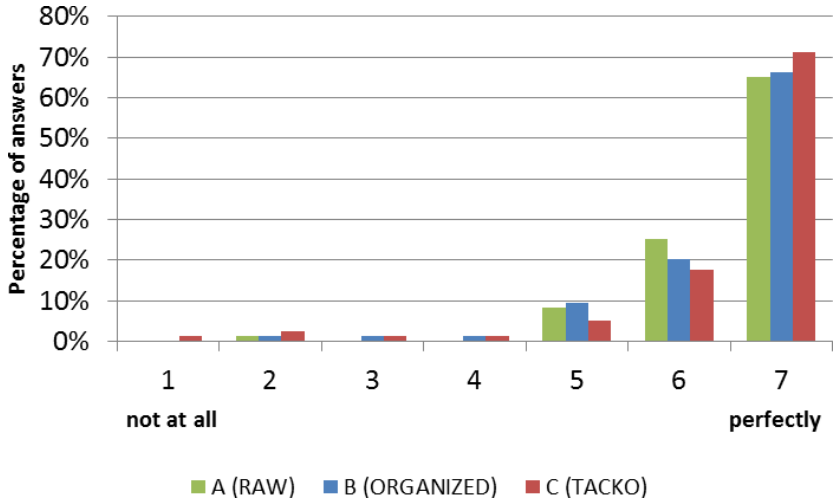


Figure B.3.: Distribution of answers to the question "How well have you understood the tool?".

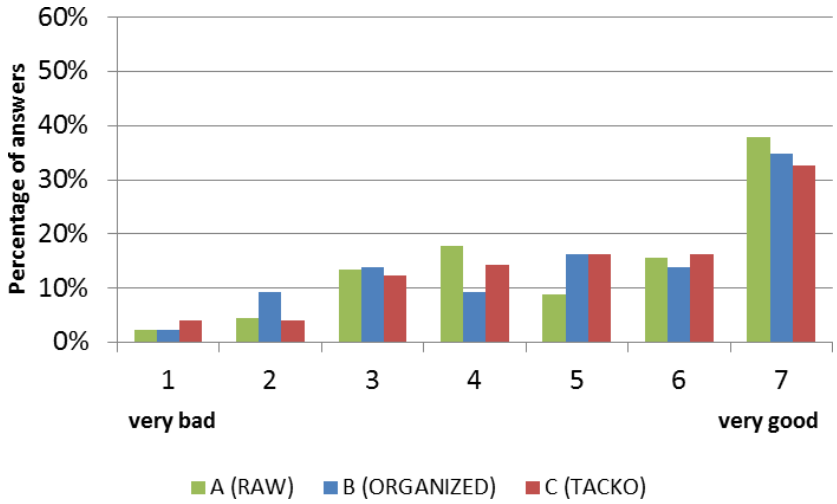


Figure B.4.: Distribution of answers to the question "How were the photos categorized?" after search tasks.

Bibliography

[Ab10] Abbas, J.: *Structures for Organizing Knowledge: Exploring Taxonomies, Ontologies, and Other Schema.* Neal Schuman Publishers. 2010.

[AD04] von Ahn, L.; Dabbish, L.: *Labeling images with a computer game.* In *Proceedings of the 2004 Conference on Human Factors in Computing Systems, CHI 2004, Vienna, Austria, April 24 - 29, 2004.* pages 319–326. 2004.

[AK05] Aula, A.; Käki, M.: *Less is more in Web search interfaces for older adults. First Monday.* 10(7). 2005.

[AKS99] Adar, E.; Karger, D.; Stein, L. A.: *Haystack: per-user information environments.* In *Proceedings of the eighth international conference on Information and knowledge management.* CIKM '99. pages 413–422. New York, NY, USA. 1999. ACM.

[An04] Anderson, C.: *The Long Tail. Wired.* October 2004.

[An07] Angeletou, S.; Sabou, M.; Specia, L.; Motta, E.: *Bridging the Gap Between Folksonomies and the Semantic Web: An Experience Report.* In *The 4th European Semantic Web Conference 2007 (ESWC 2007).* Innsbruck, Austria. June 2007.

[An08a] Angeletou, S.: *Semantic Enrichment of Folksonomies.* Technical report. Knowledge Management Institute. oct 2008.

[An08b] Angeletou, S.: *Semantic Enrichment of Folksonomy Tagspaces.* In (Sheth, A.; Staab, S.; Dean, M.; Paolucci, M.; Maynard, D.; Finin, T.; Thirunarayan, K., Ed.): *The Semantic Web - ISWC 2008.* volume 5318 of *Lecture Notes in Computer Science.* pages 889–894. Springer Berlin Heidelberg. 2008.

[ASC07] Abbasi, R.; Staab, S.; Cimiano, P.: *Organizing resources on tagging systems using T-ORG.* In *The 4th European Semantic Web Conference 2007 (ESWC 2007).* pages 97–110. Innsbruck, Austria. June 2007.

[ASM08] Angeletou, S.; Sabou, M.; Motta, E.: *Semantically enriching folksonomies with FLOR.* In *Proceedings of the CISWeb Workshop, located at the 5th European*

Semantic Web Conference ESWC 2008. Tenerife, Spain. June 2008.

[ASM09] Angeletou, S.; Sabou, M.; Motta, E.: *Folksonomy Enrichment and Search*. In (Aroyo, L.; Traverso, P.; Ciravegna, F.; Cimiano, P.; Heath, T.; Hyvönen, E.; Mizoguchi, R. et al., Ed.): *The Semantic Web: Research and Applications*. volume 5554 of *Lecture Notes in Computer Science*. pages 801–805. Springer Berlin Heidelberg. 2009.

[Au07] Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; Ives, Z.: *DBpedia: A Nucleus for a Web of Open Data*. In (Aberer, K.; Choi, K.-S.; Noy, N.; Allemang, D.; Lee, K.-I.; Nixon, L.; Golbeck, J. et al., Ed.): *The Semantic Web*. volume 4825 of *Lecture Notes in Computer Science*. pages 722–735. Springer Berlin Heidelberg. 2007.

[AYGS09] Au Yeung, C.-m.; Gibbins, N.; Shadbolt, N.: *Contextualising tags in collaborative tagging systems*. In *Proceedings of the 20th ACM conference on Hypertext and hypermedia*. HT '09. pages 251–260. New York, NY, USA. 2009. ACM.

[Ba89] Bates, M. J.: *The design of browsing and berrypicking techniques for the online search interface. Online review*. 13:407–431. 1989.

[Ba08] Bailey, R.: *Design of Comparative Experiments*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press. 2008.

[Be08] Bergman, O.; Beyth-Marom, R.; Nachmias, R.; Gradovitch, N.; Whittaker, S.: *Improved search engines and navigation preference in personal information management. ACM Trans. Inf. Syst.* 26(4). 2008.

[BHS10] Benz, D.; Hotho, A.; Stumme, G.: *Semantics made by you and me: Self-emerging ontologies can capture the diversity of shared knowledge*. In *Proceedings of the 2nd Web Science Conference (WebSci10)*. Raleigh, NC, USA. 2010.

[Bi08] Bischoff, K.; Firan, C. S.; Nejdl, W.; Paiu, R.: *Can all tags be used for search?* In *Proceedings of the 17th ACM conference on Information and knowledge management*. CIKM '08. pages 193–202. New York, NY, USA. 2008. ACM.

[Bi09] Bizer, C.; Lehmann, J.; Kobilarov, G.; Auer, S.; Becker, C.; Cyganiak, R.; Hellmann, S.: *DBpedia – A crystallization point for the Web of Data. Web Semantics: Science, Services and Agents on the World Wide Web*. 7(3):154–165. 2009.

[BKS06] Begelman, G.; Keller, P.; Smadja, F.: *Automated tag clustering: Improving search and exploration in the tag space*. In *Proceedings of the Collaborative Web Tagging Workshop at the WWW 2006*. pages 15–33. Edinburgh, Scotland. May 2006.

[Bl06] Bloehdorn, S.; Görlitz, O.; Schenk, S.; Völkel, M.; Al., E.: *TagFS - Tag semantics for hierarchical file systems*. In *Proceedings of the 6th International Conference on Knowledge Management (I-KNOW 06)*. Graz, Austria. 2006.

[BMM07] Bisgaard Munk, T.; Mørk, K.: *Folksonomy, the power law & the significance of the least effort. Knowledge organization*. 34(1):16–33. 2007.

[BMN09] Büchner, T.; Matthes, F.; Neubert, C.: *A Concept and Service based Analysis of Commercial and Open Source Enterprise 2.0 Tools*. In *International Conference*

on Knowledge Management and Information Sharing. pages 37–45. Madeira, Portugal. 2009.

[BMN10] Büchner, T.; Matthes, F.; Neubert, C.: *Data Model Driven Implementation of Web Cooperation Systems with Tricia*. In (Dearle, A.; Zicari, R., Ed.): *Objects and Databases*. Lecture Notes in Computer Science. pages 70–84. Springer. Berlin, Heidelberg, Germany. 2010.

[BN95] Barreau, D.; Nardi, B. A.: *Finding and reminding: file organization from the desktop*. *SIGCHI Bull.* 27(3):39–43. July 1995.

[Bo12] Boixadós, J.: *Evaluating the Usability of a Tag-based, Multi-faceted Knowledge Organization System*. Master's thesis. Technische Universität München. September 2012.

[Br08] Braun, S.; Schmidt, A.; Walter, A.; Zacharias, V.: *Using the Ontology Maturing Process Model for Searching, Managing and Retrieving Resources with Semantic Technologies*. In (Meersman, R.; Tari, Z., Ed.): *On the Move to Meaningful Internet Systems: OTM 2008*. volume 5332 of *Lecture Notes in Computer Science*. pages 1568–1578. Springer Berlin Heidelberg. 2008.

[Br11] Braun, S.: *Community-driven & Work-integrated Creation, Use and Evolution of Ontological Knowledge Structures*. PhD thesis. Karlsruhe Institute of Technology. Karlsruhe, Germany. November 2011.

[BYRN11] Baeza-Yates, R.; Ribeiro-Neto, B.: *Modern Information Retrieval: The Concepts and Technology behind Search*. Addison-Wesley Professional. 2 edition. February 2011. 0321416910.

[Ca08] Cattuto, C.; Benz, D.; Hotho, A.; Stumme, G.: *Semantic Grounding of Tag Relatedness in Social Bookmarking Systems*. In *Proceedings of the 7th International Conference on The Semantic Web*. ISWC '08. pages 615–631. Berlin, Heidelberg. 2008. Springer-Verlag.

[Ci08] Civan, A.; Jones, W.; Klasnja, P.; Bruce, H.: *Better to organize personal information by folders or by tags?: The devil is in the details. Proceedings of the American Society for Information Science and Technology*. 45(1):1–13. 2008.

[CJS05] Clough, P.; Joho, H.; Sanderson, M.: *Automatically organising images using concept hierarchies*. In *Multimedia Information Retrieval Workshop 2005 in conjunction with the 28th annual ACM SIGIR conference on Information Retrieval*. August 2005.

[CL99] Constantine, L. L.; Lockwood, L. A. D.: *Software for use: a practical guide to the models and methods of usage-centered design*. ACM Press/Addison-Wesley Publishing Co. New York, NY, USA. 1999.

[Co98] Conover, W. J.: *Practical Nonparametric Statistics*. John Wiley & Sons. dec 1998. 0471160687.

[Cu04] Cutting, D.: *Lucene*. `http://lucene.sourceforge.net/talks/pisa/`. November 2004. Accessed February 27th, 2013.

[Cu06] Cutrell, E.; Robbins, D.; Dumais, S.; Sarin, R.: *Fast, flexible filtering with phlat.* In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* CHI '06. pages 261–270. New York, NY, USA. 2006. ACM.

[DC94] Dourish, P.; Chalmers, M.: *Running out of Space: Models of Information Navigation.* In *Proceedings of HCI '94.* Glasgow, Scotland. 1994. ACM Press.

[Di04] Ding, L.; Finin, T.; Joshi, A.; Pan, R.; Cost, R. S.; Peng, Y.; Reddivari, P. et al.: *Swoogle: a search and metadata engine for the semantic web.* In *Proceedings of the thirteenth ACM international conference on Information and knowledge management.* CIKM '04. pages 652–659. New York, NY, USA. 2004. ACM.

[DL83] Dumais, S. T.; Landauer, T. K.: *Using examples to describe categories.* In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems.* CHI '83. pages 112–115. New York, NY, USA. 1983. ACM.

[Do99] Dourish, P.; Edwards, W. K.; LaMarca, A.; Salisbury, M.: *Presto: an experimental architecture for fluid interactive document spaces. ACM Transactions on Computer-Human Interaction.* 6(2):133–161. 1999.

[Du03] Dumais, S.; Cutrell, E.; Cadiz, J.; Jancke, G.; Sarin, R.; Robbins, D. C.: *Stuff I've seen: a system for personal information retrieval and re-use.* In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval.* SIGIR '03. pages 72–79. New York, NY, USA. 2003. ACM.

[FD07] Frey, B. J.; Dueck, D.: *Clustering by Passing Messages Between Data Points. Science.* 315(5814):972–976. 2007.

[Fe10] Fellbaum, C.: *WordNet.* In (Poli, R.; Healy, M.; Kameas, A., Ed.): *Theory and Applications of Ontology: Computer Applications.* pages 231–243. Springer Netherlands. 2010.

[FG96] Freeman, E.; Gelernter, D.: *Lifestreams: a storage model for personal data. SIGMOD Rec.* 25(1):80–86. March 1996.

[Fu87] Furnas, G. W.; Landauer, T. K.; Gomez, L. M.; Dumais, S. T.: *The vocabulary problem in human-system communication. Commun. ACM.* 30(11):964–971. November 1987.

[Fu06] Furnas, G. W.; Fake, C.; von Ahn, L.; Schachter, J.; Golder, S.; Fox, K.; Davis, M. et al.: *Why do tagging systems work?* In *CHI '06 Extended Abstracts on Human Factors in Computing Systems.* CHI EA '06. pages 36–39. New York, NY, USA. 2006. ACM.

[Ga94] Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J.: *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley. November 1994.

[Ga07] Gan, G.; Ma, C.; Ma, C.; Wu, J.: *Data clustering: theory, algorithms, and applications.* ASA-SIAM series on statistics and applied probability. SIAM, Society for Industrial and Applied Mathematics. 2007.

[Ga08] Gantz, J. F.; Chute, C.; Manfrediz, A.; Minton, S.; Reinsel, D.; Schlichting, W.;

Toncheva, A.: *The Diverse and Exploding Digital Universe: An Updated Forecast of Worldwide Information Growth Through 2011.* Technical report. International Data Corporation. March 2008.

[Ga12] Gassler, W.; Zangerle, E.; Bürgler, M.; Specht, G.: *SnoopyTagging: recommending contextualized tags to increase the quality and quantity of meta-information.* In *Proceedings of the 21st international conference companion on World Wide Web.* WWW '12 Companion. pages 511–512. New York, NY, USA. 2012. ACM.

[GBL06] Gemmell, J.; Bell, G.; Lueder, R.: *MyLifeBits: a personal database for everything. Commun. ACM.* 49(1):88–95. January 2006.

[GCHG10] García-Castro, L.; Hepp, M.; García, A.: *TagSorting: A Tagging Environment for Collaboratively Building Ontologies.* In (Cimiano, P.; Pinto, H., Ed.): *Knowledge Engineering and Management by the Masses.* volume 6317 of *Lecture Notes in Computer Science.* pages 462–472. Springer Berlin Heidelberg. 2010.

[Ge02] Gemmell, J.; Bell, G.; Lueder, R.; Drucker, S.; Wong, C.: *MyLifeBits: fulfilling the Memex vision.* In *Proceedings of the tenth ACM international conference on Multimedia.* MULTIMEDIA '02. pages 235–238. New York, NY, USA. 2002. ACM.

[GH05] Golder, S.; Huberman, B. A.: *The Structure of Collaborative Tagging Systems.* Technical report. Information Dynamics Lab, HP Labs. August 2005.

[GH06] Golder, S. A.; Huberman, B. A.: *Usage patterns of collaborative tagging systems. Journal of Information Science.* 32(2):192–208. 2006.

[Gi91] Gifford, D. K.; Jouvelot, P.; Sheldon, M. A.; O'Toole, Jr., J. W.: *Semantic file systems.* In *Proceedings of the thirteenth ACM symposium on Operating systems principles.* SOSP '91. pages 16–25. New York, NY, USA. 1991. ACM.

[Go06] Governor, J.: *On The Emergence of Professional Tag Gardeners.* `http://redmonk.com/jgovernor/2006/01/10/on-the-emergence-of-professional-tag-gardeners/`. October 2006. Accessed January 14th, 2013.

[Gr07] Gruber, T.: *Ontology of Folksonomy: A Mash-up of Apples and Oranges. International Journal on Semantic Web & Information Systems (IJSWIS).* 3:1–11. 2007.

[Gr08] Gruber, T.: *Collective knowledge systems: Where the Social Web meets the Semantic Web. Web Semantics: Science, Services and Agents on the World Wide Web.* 6(1):4–13. 2008.

[GT06] Guy, M.; Tonkin, E.: *Folksonomies: Tidying up tags. DLib Magazine.* 12(1). January 2006.

[GWP08] Golov, E.; Weller, K.; Peters, I.: *TagCare: A Personal Portable Tag Repository.* In (Bizer, C.; Joshi, A., Ed.): *Proceedings of the Poster and Demonstration Session at ISWC (2008).* CEUR Workshop Proceedings. CEUR-WS.org. 2008.

[He04] Hevner, A. R.; March, S. T.; Park, J.; Ram, S.: *Design Science in Information*

Systems Research. MIS Quarterly. 28(1):75–105. 2004.

[He06] Hearst, M. A.: *Design recommendations for hierarchical faceted search interfaces.* In *Proc. SIGIR 2006, Workshop on Faceted Search.* pages 26–30. August 2006.

[He09] Hearst, M. A.: *Search User Interfaces.* Cambridge University Press. 1 edition. 2009.

[He11] Helic, D.; Strohmaier, M.; Trattner, C.; Muhr, M.; Lerman, K.: *Pragmatic evaluation of folksonomies.* In *Proceedings of the 20th international conference on World wide web.* WWW '11. pages 417–426. New York, NY, USA. 2011. ACM.

[HGM06] Heymann, P.; Garcia-Molina, H.: *Collaborative Creation of Communal Hierarchical Taxonomies in Social Tagging Systems.* Technical Report 2006-10. Stanford InfoLab. April 2006.

[HK07] Halvey, M. J.; Keane, M. T.: *An assessment of tag presentation techniques.* In *WWW '07: Proceedings of the 16th international conference on World Wide Web.* pages 1313–1314. New York, NY, USA. 2007. ACM.

[HMHS06] Hassan-Montero, Y.; Herrero-Solana, V.: *Improving Tag-Clouds as Visual Information Retrieval Interfaces.* In *InScit2006: International Conference on Multidisciplinary Information Sciences and Technologies.* 2006.

[Ho06] Hotho, A.; Jäschke, R.; Schmitz, C.; Stumme, G.: *BibSonomy: A Social Bookmark and Publication Sharing System.* In *Proceedings of the Conceptual Structures Tool Interoperability Workshop at the 14th International Conference on Conceptual Structures.* pages 87–102. Aalborg University Press. Februar 2006.

[HR08] Hearst, M. A.; Rosner, D.: *Tag Clouds: Data Analysis Tool or Social Signaller?* In *HICSS '08: Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences.* pages 160–169. Washington, DC, USA. 2008. IEEE Computer Society.

[HRS07] Halpin, H.; Robu, V.; Shepherd, H.: *The complex dynamics of collaborative tagging.* In *Proceedings of the 16th international conference on World Wide Web.* WWW '07. pages 211–220. New York, NY, USA. 2007. ACM.

[HS11] Helic, D.; Strohmaier, M.: *Building directories for social tagging systems.* In *Proceedings of the 20th ACM international conference on Information and knowledge management CIKM 11.* pages 525–534. ACM Press. 2011.

[IAMS11] Infante Abreu, M.; Matthes, F.; Steinhoff, A.: *Using Web 2.0 Technologies to Support Technology Surveillance in a University Context.* In *Proceedings of the 12th European Conference on Knowledge Management.* Passau, Germany. 2011.

[IS01] ISO/IEC: *Software engineering – Product quality – Part 1: Quality model.* 2001.

[Ja06] Jaffe, A.; Naaman, M.; Tassa, T.; Davis, M.: *Generating summaries and visualization for large collections of geo-referenced photographs.* In (Wang, J. Z.; Boujemaa, N.; Chen, Y., Ed.): *Multimedia Information Retrieval.* pages 89–98. ACM. 2006.

[Jo05] Jones, W.; Phuwanartnurak, A. J.; Gill, R.; Bruce, H.: *Don't take my folders away! Organizing personal information to get things done.* In *CHI 05 extended*

abstracts on human factors in computing systems CHI 05. pages 1505–1508. ACM Press. 2005.

[Ka05] Karger, D. R.; Bakshi, K.; Huynh, D.; Quan, D.; Sinha, V.: *Haystack: A Customizable General-Purpose Information Management Tool for End Users of Semistructured Data*. In *Proceedings of of CIDR 2005*. 2005.

[Ki08] Kim, H. L.; Passant, A.; Breslin, J.; Scerri, S.; Decker, S.: *Review and Alignment of Tag Ontologies for Semantically-Linked Data in Collaborative Tagging Spaces*. In *Proceedings of the 2nd IEEE International Conference on Semantic Computing (ICSC 2008)*. pages 315–322. Santa Clara, California, USA. August 2008. IEEE Computer Society.

[Ko01] Kohonen, T.: *Self-Organizing Maps*. Springer Series in Information Sciences. Springer. 2001. 9783540679219.

[KSS10] Knautz, K.; Soubusta, S.; Stock, W. G.: *Tag Clusters as Information Retrieval Interfaces*. In *Proceedings of the 43rd Hawaii International Conference on System Sciences*. pages 1–10. 2010.

[KT10] Kiu, C.; Tsui, E.: *TaxoFolk: A hybrid taxonomy–folksonomy classification for enhanced knowledge navigation. Knowledge Management Research & Practice*. 8(1):24–32. 2010.

[KT11] Kiu, C.; Tsui, E.: *TaxoFolk: A hybrid taxonomy–folksonomy structure for knowledge classification and navigation. Expert Systems with Applications*. 38(5):6049–6058. 2011.

[Kw99] Kwasnik, B. H.: *The role of classification in knowledge representation and discovery. Library Trends*. 48(1):22–47. 1999.

[La87] Lakoff, G.: *Women, Fire and Dangerous Things: What Categories Reveal About the Mind*. University of Chicago Press. 1987.

[LA06] Lambiotte, R.; Ausloos, M.: *Collaborative tagging as a tripartite network*. 2006.

[La07] Lambe, P.: *Organising Knowledge: Taxonomies, Knowledge and Organisational Effectiveness: Taxonomies, Knowledge and Organization Effectiveness*. Chandos Publishing (Oxford) Ltd. 1st edition edition. 2007.

[LEC07] Laniado, D.; Eynard, D.; Colombetti, M.: *Using WordNet to turn a folksonomy into a hierarchy of concepts*. In *Semantic Web Application and Perspectives - Fourth Italian Semantic Web Workshop*. pages 192–201. December 2007.

[LGB09] Limpens, F.; Gandon, F. L.; Buffa, M.: *Collaborative Semantic Structuring of Folksonomies*. In *IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies, 2009 (WI-IAT '09)*. pages 132–135. IEEE. 2009.

[LGB10] Limpens, F.; Gandon, F. L.; Buffa, M.: *Helping online communities to semantically enrich folksonomies*. In *Proceedings of the WebSci10: Extending the Frontiers of Society On-Line*. pages 132–135. Raleigh, NC, US. apr 2010. IEEE.

[LH09] Lalwani, S.; Huhns, M. N.: *Deriving ontological structure from a folksonomy*. In

Proceedings of the 47th Annual Southeast Regional Conference. ACM-SE 47. pages 49:1–49:2. New York, NY, USA. 2009. ACM.

[Li07] Li, R.; Bao, S.; Yu, Y.; Fei, B.; Su, Z.: *Towards effective browsing of large scale social annotations*. In *Proceedings of the 16th international conference on World Wide Web WWW 07*. volume 23. pages 943–952. ACM Press. 2007.

[LZT09] Lohmann, S.; Ziegler, J.; Tetzlaff, L.: *Comparison of Tag Cloud Layouts: Task-Related Performance and Visual Exploration*. In *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part I*. INTERACT '09. pages 392–404. Berlin, Heidelberg. 2009. Springer-Verlag.

[Ma91] Mayhew, D. J.: *Principles and guidelines in software user interface design*. Prentice Hall. 1991.

[Ma04] Mathes, A.: *Folksonomies — cooperative classification and communication through shared metadata*. Technical report. Graduate school of library and information science, University of Illinois Urbana Champaign. 2004.

[Ma06] Marlow, C.; Naaman, M.; Boyd, D.; Davis, M.: *HT06, tagging paper, taxonomy, Flickr, academic article, to read*. In *Proceedings of the seventeenth conference on Hypertext and hypermedia*. HYPERTEXT '06. pages 31–40. New York, NY, USA. 2006. ACM.

[Ma10] Ma, S.: *Using Hierarchical Folders and Tags for File Management*. PhD thesis. Drexel University. March 2010.

[MB09] Miles, A.; Bechhofer, S.: *SKOS Simple Knowledge Organization System Reference*. `http://www.w3.org/TR/2009/REC-skos-reference-20090818/`. aug 2009.

[Me04] Merholz, P.: *Metadata for the Masses*. `http://www.adaptivepath.com/ideas/e000361`. October 2004.

[MF06] Millen, D.; Feinberg, J.: *Using social tagging to improve social navigation*. In *Workshop on the Social Navigation and Community based Adaptation Technologies*. 2006.

[MFK06] Millen, D. R.; Feinberg, J.; Kerr, B.: *Dogear: Social bookmarking in the enterprise*. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*. pages 111–120. New York, NY, USA. 2006. ACM Press.

[MHG10] McCandless, M.; Hatcher, E.; Gospodnetić, O.: *Lucene in Action*. Manning Publications Co. 2010.

[Mi07] Mika, P.: *Ontologies are us: A unified model of social networks and semantics. Web Semantics: Science, Services and Agents on the World Wide Web*. 5(1):5–15. March 2007.

[Mi12] Michel, F.: *Coupling tag-based and hierarchical information organization*. Bachelor's thesis. Technische Universität München. July 2012.

[MN12] Matthes, F.; Neubert, C.: *Hybride Wikis als Repository für die IT-Unternehmensarchitektur*. chapter 5.4.2, pages 174–182. Dpunkt. 2012.

[MNS11] Matthes, F.; Neubert, C.; Steinhoff, A.: *Hybrid Wikis: Empowering Users to Collaboratively Structure Information.* In (Cuaresma, M. J. E.; Shishkov, B.; Cordeiro, J., Ed.): *ICSOFT 2011 - Proceedings of the 6th International Conference on Software and Data Technologies.* volume 1. pages 250–259. Seville, Spain. jul 2011. SciTePress.

[MNS12a] Matthes, F.; Neubert, C.; Steinhoff, A.: *Multi-faceted context-dependent knowledge organisation with TACKO.* In *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies.* i-KNOW '12. pages 9:1–9:8. New York, NY, USA. 2012. ACM.

[MNS12b] Matthes, F.; Neubert, C.; Steinhoff, A.: *Structuring folksonomies with implicit tag relations.* In *Proceedings of the 23rd ACM conference on Hypertext and social media.* HT '12. pages 315–316. New York, NY, USA. 2012. ACM.

[Mo08] Montgomery, D.: *Design and Analysis of Experiments.* John Wiley & Sons. seventh edition. 2008. 9780470128664.

[MR06] Morville, P.; Rosenfeld, L.: *Information Architecture for the world wide web: designing large-scale web sites.* O'Reilly Media, Incorporated. 3rd edition edition. 2006.

[Ne05] Newman, R.: *Tag ontology design.* `http://www.holygoat.co.uk/projects/tags/`. March 2005.

[Ne07] Neal, D.: *Folksonomies and image tagging: Seeing the future. Bulletin of the American Society for Information Science and Technology.* 34(1):7–11. 2007.

[Ne12] Neubert, C.: *Facilitating Emergent and Adaptive Information Structures in Enterprise 2.0 Platforms.* PhD thesis. Technische Universität München. Munich, Germany. September 2012.

[Ni93] Nielsen, J.: *Usability Engineering.* Morgan Kaufmann Publishers Inc. San Francisco, CA, USA. 1993.

[Ni95] Nielsen, J.: *Usability inspection methods.* In *Conference Companion on Human Factors in Computing Systems.* CHI '95. pages 377–378. New York, NY, USA. 1995. ACM.

[Ni12] Nielsen, J.: *Thinking Aloud: The #1 Usability Tool.* `http://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/`. jan 2012. Accessed January 14th, 2013.

[O'05] O'Reilly, T.: *What Is Web 2.0.* `http://www.oreilly.de/artikel/web20.html`. September 2005. Accessed January 14th, 2013.

[Pa09] Passant, A.; Laublet, P.; Breslin, J. G.; Decker, S.: *A URI is Worth a Thousand Tags: From Tagging to Linked Data with MOAT. International Journal on Semantic Web and Information Systems (IJSWIS).* 5(3):71–94. 2009.

[Pe09] Peters, I.: *Folksonomies. Indexing and Retrieval in Web 2.0.* De Gruyter. Berlin. 2009.

[PG08] Prasad, A.; Guha, N.: *Concept naming vs. concept categorisation: a faceted ap-*

proach to semantic annotation. Online Information Review. 32(4):500–510. 2008.

[PL08] Passant, A.; Laublet, P.: *Meaning Of A Tag: A collaborative approach to bridge the gap between tagging and Linked Data.* In *Proceedings of the WWW 2008 Workshop Linked Data on the Web (LDOW2008).* CEUR Workshop Proceedings. Beijing, China. April 2008.

[PL09] Plangprasopchok, A.; Lerman, K.: *Constructing folksonomies from user-specified relations on flickr.* In *Proceedings of the 18th international conference on World wide web.* WWW '09. pages 781–790. New York, NY, USA. 2009. ACM.

[PLG10] Plangprasopchok, A.; Lerman, K.; Getoor, L.: *Growing a tree in the forest: constructing folksonomies by integrating structured metadata.* In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining.* KDD '10. pages 949–958. New York, NY, USA. 2010. ACM.

[PPI07] Pak, R.; Pautz, S.; Iden, R.: *Information organization and retrieval: An assessment of taxonomical and tagging systems. Cognitive Technology.* 12(1):31–44. 2007.

[PW08] Peters, I.; Weller, K.: *Tag gardening for folksonomy enrichment and maintenance. Webology.* 5(3). 2008.

[QHK03] Quan, D.; Huynh, D.; Karger, D.: *Haystack: A Platform for Authoring End User Semantic Web Applications.* In (Fensel, D.; Sycara, K.; Mylopoulos, J., Ed.): *The Semantic Web — ISWC 2003.* volume 2870 of *Lecture Notes in Computer Science.* pages 738–753. Springer Berlin Heidelberg. 2003.

[QRR07] Quintarelli, E.; Resmini, A.; Rosati, L.: *Information Architecture: FaceTag: Integrating bottom-up and top-down classification in a social tagging system. Bulletin of the American Society for Information Science and Technology.* 33(5):10–15. 2007.

[QRR08] Quintarelli, E.; Resmini, A.; Rosati, L.: *Browsing architecture : metadata and beyond.* chapter 17 – The FaceTag Engine: A Semantic Collaborative Tagging Tool, pages 204–217. EAAE transactions on architectural education. Fraunhofer IRB Verlag. Stuttgart, Germany. 2008.

[Ra33] Ranganathan, S.: *Colon Classification.* Madras Library Association. 1933.

[Ra09] Ramage, D.; Heymann, P.; Manning, C. D.; Garcia-Molina, H.: *Clustering the tagged web.* In *Proceedings of the Second ACM International Conference on Web Search and Data Mining.* WSDM '09. pages 54–63. New York, NY, USA. 2009. ACM.

[Ri07] Rivadeneira, A. W.; Gruen, D. M.; Muller, M. J.; Millen, D. R.: *Getting our head in the clouds: toward evaluation studies of tagclouds.* In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* CHI '07. pages 995–998. New York, NY, USA. 2007. ACM.

[RL10] Rodden, K.; Leggett, M.: *Best of both worlds: improving gmail labels with the affordances of folders.* In *CHI '10 Extended Abstracts on Human Factors in Computing Systems.* CHI EA '10. pages 4587–4596. New York, NY, USA. 2010.

ACM.

[Ro76] Rosch, E.; Mervis, C. B.; Gray, W. D.; Johnson, D. M.; Boyes-Braem, P.: *Basic objects in natural categories. Cognitive Psychology.* 8(3):382–439. 1976.

[Ro78] Rosch, E.: *Principles of Categorization.* In (Rosch, E.; Lloyd, B. B., Ed.): *Cognition and Categorization.* pages 27–48. Lawrence Erlbaum Associates. Hillsdale (NJ), USA. 1978. Reprinted in *Readings in Cognitive Science. A Perspective from Psychology and Artificial Intelligence*, A. Collins and E.E. Smith, editors, Morgan Kaufmann Publishers, Los Altos (CA), USA, 1991.

[RW08] Rader, E.; Wash, R.: *Influences on tag choices in del.icio.us.* In *Proceedings of the 2008 ACM conference on Computer supported cooperative work.* CSCW '08. pages 239–248. New York, NY, USA. 2008. ACM.

[Sa09] Sacco, G. M.: *Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience.* volume 25 of *The Information Retrieval Series.* chapter 1 – The Model, pages 1–17. Springer. 2009.

[SBC97] Shneiderman, B.; Byrd, D.; Croft, W. B.: *Clarifying Search: A User-Interface Framework for Text Searches. D-Lib Magazine.* 3(1). 1997.

[SC99] Sanderson, M.; Croft, B.: *Deriving concept hierarchies from text.* In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval.* SIGIR '99. pages 206–213. New York, NY, USA. 1999. ACM.

[Sc06a] Schmitz, C.; Hotho, A.; Jäschke, R.; Stumme, G.: *Mining Association Rules in Folksonomies.* In *Data Science and Classification.* pages 261–270. Springer. 2006.

[Sc06b] Schmitz, P.: *Inducing Ontology from Flickr Tags.* In *Proceedings of the Workshop on Collaborative Tagging at WWW2006.* Edinburgh, Scotland. May 2006.

[SC08] Sinclair, J.; Cardew-Hall, M.: *The folksonomy tag cloud: when is it useful? Journal of Information Science.* 34(1):15 –29. February 2008.

[Se06] Sen, S.; Lam, S. K.; Rashid, A. M.; Cosley, D.; Frankowski, D.; Osterhouse, J.; Harper, F. M. et al.: *tagging, communities, vocabulary, evolution.* In *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work.* CSCW '06. pages 181–190. New York, NY, USA. 2006. ACM.

[SFT09] Sacco, G. M.; Ferré, S.; Tzitzikas, Y.: *Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience.* volume 25 of *The Information Retrieval Series.* chapter 3 – Comparison with Other Techniques, pages 35–74. Springer. 2009.

[Sh92] Shneiderman, B.: *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA. 2nd edition. 1992.

[SH04] Stoica, E.; Hearst, M. A.: *Nearly-Automated Metadata Hierarchy Creation.* In *Companion Proceedings of HLT-NAACL'04.* pages 117–120. Boston. may 2004.

[Sh05] Shirky, C.: *Ontology is Overrated : Categories, Links, and Tags.* `http://www.shirky.com/writings/ontology_overrated.html`. 2005.

[SHR07] Stoica, E.; Hearst, M. A.; Richardson, M.: *Automating Creation of Hierarchical Faceted Metadata Structures.* In (Sidner, C. L.; Schultz, T.; Stone, M.; Zhai, C., Ed.): *Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007).* pages 244–251. The Association for Computational Linguistics. 2007.

[SM07] Specia, L.; Motta, E.: *Integrating Folksonomies with the Semantic Web.* In (Franconi, E.; Kifer, M.; May, W., Ed.): *The Semantic Web: Research and Applications.* volume 4519 of *Lecture Notes in Computer Science.* pages 624–639. Springer Berlin Heidelberg. 2007.

[Sp11] Sprinthall, R.: *Basic Statistical Analysis.* Prentice Hall. 9th edition. 2011.

[ST09a] Sacco, G. M.; Tzitzikas, Y., Ed.*Dynamic Taxonomies and Faceted Search.* Springer. 1st edition edition. 2009.

[St09b] Stefaner, M.; Ferré, S.; Perugini, S.; Koren, J.; Zhang, Y.: *Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience.* volume 25 of *The Information Retrieval Series.* chapter 4 – User Interface Design, pages 75–112. Springer. 2009.

[SZ08] Sigurbjörnsson, B.; van Zwol, R.: *Flickr Tag Recommendation based on Collective Knowledge.* In *Proceedings of the 17th International World Wide Web Conference, WWW 2008.* pages 327–336. New York, NY, USA. April 2008. ACM.

[Te04] Teevan, J.; Alvarado, C.; Ackerman, M. S.; Karger, D. R.: *The perfect search engine is not enough: a study of orienteering behavior in directed search.* In *CHI '04: Proc. of the SIGCHI conf. on Human factors in computing systems.* pages 415–422. ACM Press. 2004.

[TJ09] Taylor, A.; Joudrey, D.: *The Organization of Information.* Library and Information Science Text Series. Libraries Unlimited. 3rd edition edition. 2009.

[TKH11] Trattner, C.; Körner, C.; Helic, D.: *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.

[Tr12] Trattner, C.; Lin, Y.-l.; Parra, D.; Yue, Z.; Real, W.; Brusilovsky, P.: *Evaluating tag-based information access in image collections.* In *Proceedings of the 23rd ACM conference on Hypertext and social media.* HT '12. pages 113–122. New York, NY, USA. 2012. ACM.

[TS07] Tanasescu, V.; Streibel, O.: *Extreme Tagging: Emergent Semantics through the Tagging of Tags.* In (Haase, P.; Hotho, A.; Chen, L.; Ong, E.; Mauroux, P. C., Ed.): *Proceedings of the International Workshop on Emergent Semantics and Ontology Evolution (ESOE2007) at ISWC/ASWC2007, Busan, South Korea.* November 2007.

[TS09] Tomuro, N.; Shepitsen, A.: *Construction of disambiguated Folksonomy ontologies using Wikipedia.* In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources.* People's Web '09. pages

42–50. Stroudsburg, PA, USA. 2009. Association for Computational Linguistics.

[TT91] Tanaka, J. W.; Taylor, M.: *Object categories and expertise: Is the basic level in the eye of the beholder? Cognitive Psychology.* 23(3):457–482. 1991.

[VAS09] Voit, K.; Andrews, K.; Slany, W.: *Why Personal Information Management (PIM) Technologies Are Not Widespread. Human-Computer Interaction.* 2009.

[VAS11] Voit, K.; Andrews, K.; Slany, W.: *TagTree: Storing and Re-finding Files Using Tags.* In (Holzinger, A.; Simonic, K.-M., Ed.): *Information Quality in e-Health.* volume 7058 of *Lecture Notes in Computer Science.* pages 471–481. Springer Berlin / Heidelberg. 2011.

[VAS12] Voit, K.; Andrews, K.; Slany, W.: *Tagging might not be slower than filing in folders.* In *Proceedings of the 2012 ACM annual conference extended abstracts on Human Factors in Computing Systems Extended Abstracts.* CHI EA '12. pages 2063–2068. New York, NY, USA. 2012. ACM.

[VDHS07] Van Damme, C.; Hepp, M.; Siorpaes, K.: *FolksOntology: An Integrated Approach for Turning Folksonomies into Ontologies.* In *Proceedings of the ESWC Workshop "Bridging the Gap between Semantic Web and Web 2.0".* 2007.

[VH05] Voorhees, E. M.; Harman, D. K.: *TREC: Experiment and evaluation in information retrieval.* The MIT Press. September 2005.

[VW05] Vander Wal, T.: *Explaining and Showing Broad and Narrow Folksonomies.* `http://www.vanderwal.net/random/entrysel.php?blog=1635`. February 2005. Accessed January 25th, 2013.

[VW07] Vander Wal, T.: *Folksonomy Coinage and Definition.* `http://vanderwal.net/folksonomy.html`. February 2007. Accessed January 14th, 2013.

[Wa12] Waltl, B.: *Partitioning instead of Clustering: An alternative approach for mining structured content in folksonomies.* Guided Research Project, Technische Universität München. October 2012.

[We05] Weinberger, D.: *Tagging and why it matters.* `http://cyber.law.harvard.edu/sites/cyber.law.harvard.edu/files/07-WhyTaggingMatters.pdf`. May 2005. Accessed January 25th, 2013.

[WP08] Weller, K.; Peters, I.: *Seeding, weeding, fertilizing. Different tag gardening activities for folksonomy maintenance and enrichment.* In *Proceedings of I-Semantics, International Conference on Semantic Systems.* pages 110–117. Graz, Austria. 2008.

[WR06] Wash, R.; Rader, E.: *Collaborative Filtering with del.icio.us.* `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.79.5371&rep=rep1&type=pdf`. 2006. Accessed January 26th, 2013.

[WR09] White, R. W.; Roth, R. A.: *Exploratory Search: Beyond the Query-Response Paradigm.* Morgan & Claypool Publishers. [San Rafael, Calif.]. 2009. 978-1-59829-783-6.

[WZM06] Wu, H.; Zubair, M.; Maly, K.: *Harvesting social knowledge from folksonomies.* In

Proceedings of the seventeenth conference on Hypertext and hypermedia. HYPERTEXT '06. pages 111–114. New York, NY, USA. 2006. ACM.

[WZY06] Wu, X.; Zhang, L.; Yu, Y.: *Exploring social annotations for the semantic web*. In *Proceedings of the 15th international conference on World Wide Web*. WWW '06. pages 417–426. New York, NY, USA. 2006. ACM.

[ZB07] Zacharias, V.; Braun, S.: *SOBOLEO - Social Bookmarking and Lightweight Ontology Engineering*. In *Workshop on Social and Collaborative Construction of Structured Knowledge (CKC), 16th International World Wide Web Conference (WWW 2007)*. May 2007.

[ZBS09] Zacharias, V.; Braun, S.; Schmidt, A.: *Social Semantic Bookmarking with SOBOLEO*. In (Murugesan, S., Ed.): *Handbook of Research on Web 2.0, 3.0 and X.0: Technologies, Business, and Social Applications*. pages 225–241. IGI Global. 2009.

[ZE98] Zamir, O.; Etzioni, O.: *Web document clustering: a feasibility demonstration*. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '98. pages 46–54. New York, NY, USA. 1998. ACM.

[Zh05] Zhong, S.: *Efficient online spherical k-means clustering*. In *Proceedings of the IEEE International Joint Conference on Neural Networks IJCNN '05*. volume 5. pages 3180–3185. IEEE. 2005.

[Zu09] Zubiaga, A.; García-Plaza, A. P.; Fresno, V.; Martínez, R.: *Content-Based Clustering for Tag Cloud Visualization*. In (Memon, N.; Alhajj, R., Ed.): *2009 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2009)*. pages 316–319. IEEE Computer Society. 2009.

[ZWY06] Zhang, L.; Wu, X.; Yu, Y.: *Emergent Semantics from Folksonomies: A Quantitative Study*. In (Spaccapietra, S.; Aberer, K.; Cudré-Mauroux, P., Ed.): *Journal on Data Semantics VI*. volume 4090 of *Lecture Notes in Computer Science*. pages 168–186. Springer Berlin Heidelberg. 2006.