PhD-FSTC-33-2010
The Faculty of Sciences, Technology and Communication

DISSERTATION

Defense held on 10/12/2010 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN INFORMATIQUE

by

SASCHA KAUFMANN
Born on 10 August 1969 in Offenbach/Main (Germany)

CUBA: ARTIFICIAL CONVIVIALITY AND USER-BEHAVIOUR ANALYSIS IN WEB-FEEDS

Dissertation defense committee

Dr Christoph Schommer, dissertation supervisor
Professor, Université du Luxembourg

Dr Denis Zampuniéris, Chairman
Professor, Université du Luxembourg

Dr Karsten Tolle, Vice Chairman
Goethe-Universität Frankfurt/Main Germany

Dr Vincent Koenig
Université du Luxembourg

Dr Monique Janneck
Professor, Universität Hamburg, Germany

Abstract

Conviviality is a concept of great depth that plays an important role in any social interaction. A convivial relation between individuals is one that allows the participating individuals to behave and interact with each other following a set of conventions that are shared, commonly agreed upon, or at least understood. This presupposes an implicit or an explicit regulation mechanism based on consensus or social contracts and applies to the behaviours and interactions of participating individuals. With respect to an intelligent web-based system, an applicable social contribution is the give of assistance to other users in situations that are unclear and in guiding him to find the right decision whenever a conflict arises. Such a convivial social biotope deeply depends on both implicit and explicit co-operation and collaboration of natural users inside a community. Here, the individual conviviality may benefit from *"The Wisdom of Crowds"*, which fosters a dynamic understanding of the user's behaviour and a strong influence of an individual's well being to another person(s). The web-based system CUBA focus on such a behavioural analysis through profiling and demonstrates a convivial stay within a web-based feed system.

Acknowledgements

Writing a thesis is not an easy thing. There are a lot of people being involved in the process. Sometimes they help you with suggestions or advices, sometimes they inspire you within discussions and sometimes they are just there.

In the following I would like to thank some people and institutions, without which this work would not have been possible. First of all to the supervisor of my PhD thesis, Christoph Schommer. We know each other now for several years. Thank you very much for all the time of your inspiration, discussion, support and fun. To Christoph Schommer and Ralph Weires in Luxembourg and Thomas Ambrosi, Natascha Hoebel, Karsten Tolle, Peter Werner and Roberto Zicari, who inspired my scientific work and helped to bring it on papers. I would not have been able to write this work without your help. To the members of my Supervisor Committee, Christoph Schommer, Karsten Tolle and Denis Zampunieris. And to the members of my PhD Committee Monique Janneck and Vincent Koenig. To the Fonds National de la Recherche Luxembourg and especially to Mr. Alvez for his support. To the Univerité du Luxembourg that affords me to write this work. To all my friends and Colleges, especially Maria Biryukov, Ronny Heinz, Michael Hilker, Yayanta Pouray, "Maison Dommeldange" and all the rest. To my parents that loved, demanded and supported me during all the time. Thank you very much for all. And to Julie, for your love, your support and your patience. You are my star that guided me during all the countless hours of work.

Luxembourg, 12 November 2010

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Conviviality is still a widely unexplored research field and an artificial simulation is far away from a standardised concept.

We concern with the problem of simulating natural conviviality in an artificial, web-based environment. We believe that such a convivial environment directly influences the acceptance of the web-based system and, consequently, offers a competitive advantage.

However, an exact definition of what conviviality is does not really exist in the literature and there is neither a clear model nor a singular vision of how it can be realised. A usage of the word in a communication environment like the World-Wide Web is often understood as a layout problem. Moreover, the relationship of conviviality to terms like amicability or comfort ability remains fluent: does conviviality refer to a place or a situation where someone is welcomed and/or feels well? Can conviviality be computed by algorithmic parameters, being adjusted and adapted? May external signal be considered and internal rules be activated such that we can obtain conviviality?

We believe that artificial conviviality is a concept of greater depth that is significant in a social interaction and in social systems in general. Convivial relations between individuals allow them to behave and interact among each other following a set of conventions shared, commonly agreed upon or at least understood. This presupposes implicit or explicit mechanisms that are based on consensus or social contracts and applies to the behaviour and interactions of participating individuals [1]. Following Merriam-Webster [2], a natural conviviality distinguishes between a technological and a sociological meaning of the term. The technological meaning addresses the *quality pertaining to a software or hardware easy and pleasant to use and understand even for a beginner.* The sociological meaning stresses the importance of *positive relations between the people and the groups that form a society, with an emphasis on community life and equality rather than hierarchical functions.*

In Computer Science, a reliable and convincing definition of *artificial conviviality* has been less considered. As mentioned above, it is commonly understood as a user-friendly design or event, being often equated with GUIs (Graphical User Interfaces). *Artificial conviviality* also occurs in conjunction with digital cities and normative agents [3], design processes [4], or more generally in the context of sharing and enjoying a "good time"

with other users. However, what a *good time* is remains unclear, since *good* is strictly subjective and, because of his previous experiences, almost each user has a different understanding of it. There are computer-related sources that may influence a "good time" in many ways, which makes it nearly impossible to introduce a single definition of *artificial conviviality* that covers all aspects.

An interesting concept, however, is proposed by Illich [5], where the term *conviviality* is associated to (software) tools as the result of a conscious user-centric decision: "I am aware that in English *convivial* now seeks the company of tipsy jolliness, which is distinct from that indicated by the Old English Dictionary and opposite to the austere meaning of modern *eutrapelia*, which I intend. By applying the term *convivial* to tools rather than to people, I hope to forestall confusion." And, in fact, Illich intends to bring the technology to the level of "common" people making it accessible (and hence usable) to everybody. The idea is to enable (potentially all) users to use technique in a better/smoother way. Instead of certain specifications on how convivial (software) tools should look like or should be used, Illich proposes some characteristics of convivial (software) tools, for example, no users must be excluded, no education should be necessary to use such a software tool, or avoid too complex features, even if they are implementable, must exist. Unfortunately, these guidelines have not been intended to the World Wide Web.

A promising approach is to foster the principles of "The Wisdom of Crowds". This principle is not new and there had been a lot of experiments and research work in the past. The term has been populated by Surowiecki [6]. He argues that certain situations exist where a group of humans (i.e. crowd) come up with a better solution (regarding a given problem) than the group's smartest individual if a certain number of conditions exist:

- a *diversity of the existing points of view*
- a *decentralisation* and the *independence* of the participants
- a form of *aggregation*

The main idea behind the *diversity of points of view* is that knowledge – being unavailable for experts (the so called private or local knowledge) – must be collected when it is used for the final solution. To ensure that other group members do not influence such knowledge, there exist certain requirements of *decentralisation* and *independence*. At the end, an independent instance *aggregates* the different knowledge to *the wisdom of crowd*. This can be obtained in different ways, which means that there is no well-defined way of coming up with the perfect solution. Generally, there are several types of problems where wisdom of crowd is very beneficial and applicable. For example, *cognition problems* require a particular solution. This can be a judgment of a market or optimisation problems. *Co-operation problems* involve trust between the members for their interaction. And *co-ordination problems* occurs when there is a need to arrange resources in relation with time.

In practice, it may happen that experts are smarter than the "crowd". Generally, this may become crucial in case that one or more experts continuously dominate such

a group. An aggregation of the wisdom may not be guaranteed in an acceptable way. However, there are also situations, where the crowds can fail with their solution, for example, when the composition of a group is too centralised, emotional, homogeneous, and/or imitative. Some of these reasons are only the opposite characteristics of the conditions discussed above (e. g. *low vs. high centralised*, *diversity vs. homogeneity or imitation*). In any case, the most important thing is that the members of the group are independent. The goal is to let everybody in the group share their knowledge and experience; a little bit cheating wont do any harm. The most dangerous situations are that groups are too small – group leaders can grow up and influence the opinion of other members – or those groups with nearly the same background knowledge exist (no diversity). In a small group, the avoidance of an imitative behaviour may lead to a cascading effect over the whole group. On the other hand, this does not heavily influence the result as long as it happens in a small environment.

Structure of the Document

The remaining part of this document is organized as follows: in Chapter 2 we examine several definitions of conviviality and present our understanding of it by introducing *Artificial Conviviality*; Chapter 3 introduces the concept of "The Wisdom of Crowds" and how it can be used to support conviviality on Web sites; next, we introduce the Non-Obvious Profile algorithm, an algorithm that builds interest profiles of Web site users; in Chapter 5, we discuss several techniques to aggregate the other users' knowledge and use it for a recommendation system. Chapter 6 describes the concept and architecture of our implemented prototype CUBA; in Chapter 7, we depict the most important implementation aspects of the CUBA prototype; Chapter 8 presents results of our experiments while in Chapter 9 we summarise the work, present the conclusions and discuss future research options.

Part I

Theoretic

Chapter 2

Of the Different Kinds of Conviviality

In this chapter, we investigate the term conviviality in detail and regard several existing definitions. We discuss general definitions from amongst others the Merriam Webster dictionary and the Grand Dictionaire Terminologique. Afterwards we have a closer look to context specific definitions of conviviality. We regard to Illich's approach of conviviality, which considered the concept in a social environment. We also look at Fischer's definition, which takes Illichs approach and transforms it into a computer science environment. Then we examine Caire's approach that settles conviviality in the environment of digital cities. We close this section by introducing a new definition of conviviality, which we call *Artificial Conviviality*

2.1 Definitions of Conviviality

In this chapter we present and examine existing definitions of *conviviality*. We start with common definitions, which take part in the English and French languages. Afterwards we introduce definitions of conviviality, which are more specific and domain depended.

2.1.1 Common Definitions

The adverb *convivial* respectively the noun *conviviality* takes place in the active vocabularies of the English and French languages. In the following, we have a closer look at different definitions of conviviality to get an overview and a common understanding of its usage.

Merriam-Webster

Merriam-Webster [2] defines *convivial* and its original meaning as follows:

> Main Entry: con · viv · ial
> Pronunciation: \kən-ˈviv-yəl, -ˈvi-vē-əl\
> Function: *adjective*
> Etymology: Late Latin *convivialis*, from Latin *convivium* banquet, from *com-* + *vivere* to live – more at **quick**
> Date: circa 1668
> : relating to, occupied with, or fond of feasting, drinking, and good company <a convivial host><a convivial gathering>
> con·viv·i·al·i·ty \-vi-vē-ˈa-lə-tē\ *noun*
> con·viv·ial·ly \-ˈviv-yə-lē, -ˈvi-vē-ə-lē\ *adverb*

As we see, *convivial* in its original sense corresponds to individuals, which form a social group (company) and interact with each other (feasting). The term "good company" indicates that an individual enjoys some kind of activity within a social group. It indicates further a positive apperception and evaluation by the individual. It implies that conviviality is a subjective sensation that depends on an individual. Since every individual senses a given situation differently, there could not be an objective measurement for conviviality. The reasons may differ and be caused by many reasons. Fromm [7] gives on possible explanation. He says that there could not be the same perceptions for two different individuals in the same situation, because every one has its unique experiences and therefore will gather the same situation differently. Nevertheless, deeper discussions of theses reasons are out of the focus of this work. Let us maintain at this point that the original meaning of convivial (for an individual) is to share a good time in company.

We continue with other definitions, and present more modern meanings of convivial that are slightly different. While Morris [8] follows the former definition in general, it is also less restrictive and expands conviviality by mention the more broaden terms sociable and jovial:

1. Fond or feasting, drinking, and good company; sociable; jovial
2. Relating to or of the nature of a feast; festive. - See Synonyms at jolly

Another definition, given by Summers [9], goes a step further and covers the original meaning only fractionally:

> convivial: pleasantly merry and friendly: convivial companions | a very convivial atmosphere

In this definition, the (social) group and interaction context is putting back. In contrary, it seems that the meaning of the individual's feelings (i.e. the result) shifts into focus and becomes more important. It is more perceived in the individual's feelings and enjoyment, which is not necessarily attached to social interactions any more.

There is also a counterpart of convivial in the French language. We present two definitions to show the equivalent usage of convivial in both languages and introduce different domains where convivial has its special meaning. The French-English dictionary [10] defines convivial as:

> convivial, e : friendly, convivial; user-friendly (Computer)
> convivialité : social interaction; (= jovialité) friendliness, conviviality; user-friendliness (Computer)

When we compare the translation with the previous definitions, we notice that in French the meaning of conviviality is similar to the modern English definitions we seen before and is also been used in the same context. One interesting thing is that this definition assigns convivial to two different domains. While it still has a common meaning, it can now also be related to the domain of computers where it means *user-friendly*. In this context, the part of social interaction usually fully vanishes and tends to relate to a Graphical User Interface (GUI), the interaction with computer systems by itself, or the design and usability of applications in common.

Nevertheless, the term convivial is not only related to two domains. There are many more. "Le Grand dictionnaire teminologique" [11] for example lists the following meanings:

> français convivial adj.
>
> Domaine(s) :
> - géneralité
> Définition : Qui a rapport aux festins, aux repas.[1]
>
> - informatique, télécommunication
> Définition : Relatif à un logiciel ou à un matériel facile et agréable à utiliser, même pour un utilisateur novice.[2]
> Équivalent(s): English user-friendly
>
> - sociologie
> Définition : Où l'accent est mis sur une vie communautaire et égalitaire, plutôt que sur une vie fondée sur la hiérarchie des fonctions.[3]
>
> - philosophie, sociologie
> Définition: Rendu « convivial », c'est-à-dire utilisable directement, sans relais.
> Note(s): Courant dans un des domaines techniques ou scientifiques.[4]

[1] Related to feasts or meals [2] Related to software or hardware, easy and friendly to use, even for a novice user. [3] Where the focus is put on a life in community and equal for everyone, instead of a live base on a hierarchy of functions. [4] Made convivial, it means immediate usable without any delay. Note: Frequently used in technical and scientific domains.

Here the term conviviality is also related to the sociological and philosophical domain with its own meaning. We are especially interested in the philosophical sense on conviviality, because it leads us to the following approach from Illich, which connects several aspects of above's definition.

2.1.2 Illich's Tools of Conviviality

Illich looks at conviviality from a philosophical and sociological point of view. In his work "Tools for Conviviality" [5] he shares his thoughts on convivial societies and on convivial tools in particular. He discusses the meaning of conviviality in relation to the society and its tools, where he usually has their impact on the society in mind. He starts with societies structures in the western world and criticize the "artificial" barriers in their systems. As an example, he points out artificial barriers for choosing a profession or getting permission to perform a task, like building a house for example. In his opinion, most of these acts are limited by artificial barriers for different reasons, like seal or protect a social group (and let them take advantage of this separation). In this context he makes a connection to tools and introduces the term *convivial tools*.

> "Tools are intrinsic to social relationships. An individual relates himself in action to his society through the use of tools that he actively masters, or by which he is passively acted upon. To the degree that he masters his tools, he can invest the world with his meaning; to the degree that he is mastered by his tools, the shape of the tool determines his own self-image. Convivial tools are those, which give each person who uses them the greatest opportunity to enrich the environment with the fruits of his or her vision. Industrial tools deny this possibility to those who use them and they allow their designers to determine the meaning and expectations of others." [5]

Illich argues that only convivial tools allow an individual to express personal visions and, in the best case, enrich the environment and influence social relationships. These kind of tools are quite related to the critics he does to the barriers. Later he points out the characteristics of convivial tools when he says:

> "Tools foster conviviality to the extent to which they can be easily used, by anybody, as often or as seldom as desired, for the accomplishment of a purpose chosen by the user. The use of such tools by one person does not restrain another from using them equally. They do not require previous certification of the user. Their existence does not impose any obligation to use them. They allow the user to express his meaning in action." [5]

The main character of *convivial tools* is, that there are no restrictions to use them. For Illich, for example, the telephone represents all attributes of a convivial tool. This means on the one hand, that there should also be no administrative and social restrictions to use these tools. On the other hand, and here he goes a step further, there should also be no (specific) instructions how to use these tools. In other words, anybody should be free

to use them in an individual way. This perspective is strong related to the philosophical definition of conviviality, which we presented in section 2.1.1.

Nevertheless, his concept of *convivial tools* is inspiring for research in computer related fields, especially in Human-Computer Interaction (HCI).

2.1.3 Fischer

Fischer [4] seizes Illich's suggestions and transferred them in the environment of HCI, where he used it especially in software construction environments. In his work, he identifies a gap between convivial, or "simple" tools like Turn-Key Systems on the one side and "powerful" tools like programming languages on the other side. He notices that the most powerful tools require additional learning and training time to benefit from their full capability. Therefore he uses "The spectrum of conviviality" (see Figure 2.1) to depict this gap in a graphical way.

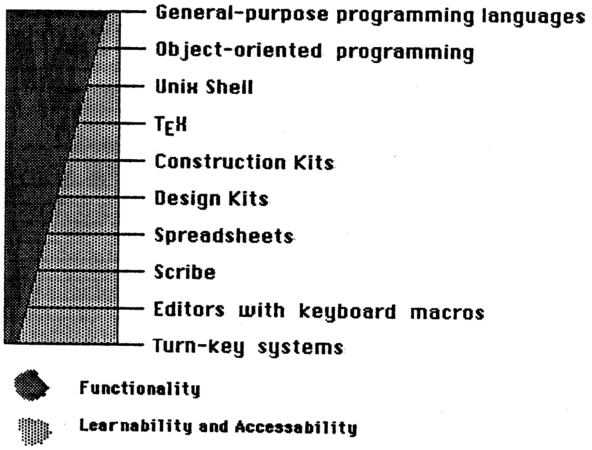

Figure 2.1: The spectrum of conviviality [4]

To be more concrete of the difference between simple and powerful tools, he describes the characteristics of both ends of the spectrum and their pitfalls as follow:

> "[General purpose programming languages] ... are powerful, but they are hard to learn, they are often too far away from the conceptual structure of the problem and it takes too long to get a task done or a problem solved. This class of systems can be adequately described by the Turing tar-pit: Beware of the Turing tar-pit, in which everything is possible but nothing of interest is easy." [4]

On the other hand we can describe convivial tools as common "ready to go" systems, which offer a simple handling, but usually lack in the power of there functions.

> "[Turn-key systems] ... are easy to use, no special training is required, but they can not be modified by the user. This class of systems can be adequately described by the converse of the Turing tar-pit: Beware the over-specialized system where operations are easy, but little of interest is possible." [4]

Fischer points further out that

> "Despite our goal of making computer systems more convivial, i.e., giving more control to the user, we do not believe that more control is always better." [4]

We share Fischer's opinion for several reasons: First, Fisher wrote this text in 1988. Today's computer systems are more powerful and advanced then these in 1988. But while the development in hardware went on, there were only little progress in the software development towards convivial computing in Fisher's sense. Even today, there are no "simple" tools that allow anybody to use a computer in sense of Illich's definitions; they are still industrial tools. Creating computer programs requires still a comprehensive studying of (at least) a computer language and its libraries, not mention the operating system or additional hardware issues. This ends up in a situation where most users still depend on software created by other persons. This is really not convivial in Illich's sense, because of the lack of opportunities to configure them to satisfy an individual's real requirements. Computer program providers usually try to solve this problem by either introduce more features ("more is always better") or implement some kind of "intelligent" tools. Both solutions can be very awkward and might include many pitfalls for both sides. As Schwartz [12] shows, a wide range of options can lead to an uncomfortable individual, because options require choosing. In general, the opportunity of having options is good, but at some level, they become to an opposite perception on the user's side. This could lead a user to feelings of uncertainty, indecisiveness or procrastination. Intelligent tools can also be tricky, because of their way to interpret "intelligence". Fischer describes it as follow:

> "Intelligent tools can have problems because many of them fail to give any indication of how they operate and what they are doing; the user feels like an observer, watching while unexplained operations take place. This mode of operation results in a lack of control over events and does not achieve any conviviality." [4]

This may be true for a wide range of applications. On the other hand, there are applications and Web sites that follow these critics and explains the user the reasons for their behaviour. Finally, we consider the programmers. Every program is very complex and usually contains errors. In practice the amount of errors rises with the number of coding lines. Putting more options or "intelligent behaviour" to a program adds more complexity and leads often to more errors in the program. If these errors occur during the usage of the application, the user may become unsatisfied.

As a conclusion, we can say that today's computer systems are technically still on the complex side of "Fisher's spectrum of conviviality". They are powerful but they are

still too complex and require some level of education to use their power. Fischer himself gives a hint to the right direction when he says:

> "The development of convivial tools will break down an old distinction: there will be no sharp border line between programming and using programs – a distinction which has been a major obstacle for the usefulness of computers." [4]

2.1.4 Caire

Another approach of defining conviviality is made by Caire [13]. It is motivated by sociological topics and placed in context of user agents and digital cities.

Her approach is partly based on "Convivial Masks" and "Convivial Norms". While "Convivial Masks" allows an individual to adopt different personalities, depending on the particular environment, "Convivial Norms" becomes important to coordinate or organise the interactions between individuals to create conviviality.

One of the works main ideas is to describe relations between individuals with a mathematical model (i.e. graph) to compute so-called "levels of conviviality". Therefore the interaction between members of a group is examined and modelled as a directed graph. Any directed connection between two individuals implies that an individual gives information to another one, who takes and uses it. The algorithm analyses the graph and computes the level of conviviality. For example, a cycle in a graph is interpreted as more convivial (everybody gives and takes), while "death-ends" or "stars" are interpreted as less convivial, because it implies that there are "takers" that receives information, but do not want to give or share their own. This is only a simple explanation of Caire's approach. For further information we refer to [13].

2.2 Introducing Artificial Conviviality

In the previous part of this chapter, we have presented several definitions and concepts of conviviality. We have seen that there is a colloquial meaning of conviviality and that domain specific definitions also exist. However, during our research we could not find an adequate definition of conviviality that describes conviviality in respect to the World Wide Web (WWW), its Web sites and Web applications. In our opinion this is a lack, because today the Internet is used by millions of people that are concerned by it. It is without question that the Internet and its related technologies have been enhanced since its start in the 1990s. But we think there is still something missing that takes advantage of its possibilities. Frankly speaking, the majority of Web sites or Web applications are still some kind of "normal applications". In the core, most of the Web sites are still acting the way like newspapers: There is someone (an individual, a group of individuals, or an institution) who provides content in a given style that other individuals have to consume in the given form. There is no possibility provided to include the user in a convivial experience. On the other side there are also first signs of convivial concepts

(in the original sense) on some Web sites. For example, many blogs or newspaper Web sites allow their visitors to comment or discuss their articles. This is one way to allow visitors to become an active part of the Web site's community and benefit from a convivial experience. Other Web sites' concepts are based on social and communication aspects only. They allow to communicating and connecting to friends directly. Services like Facebook[14], Twitter[15], Skype [16] and many others, allow people to share time together and feel convivial in the original sense. Of course, these Web sites have been designed to support that purpose (and often that purpose only). But both approaches are adopted only for a fraction of all Web sites.

Web applications, on the other hand, are in principle often a 1:1 copy of an existing software application. They exist to perform jobs and they are relative inflexible. Some of them allow to share documents or tasks to provide a collective work environment. This can also be regarded as an approach to support a convivial experience.

Taking this situation into account, the main question, this work deals with, is:

> Does it make sense to define conviviality in conjunction of the World Wide Web? And if so, what would be a reasonable definition of conviviality?

We try to answer these questions in the following.

Why should we come up with the concept on conviviality in relation to Web sites? In our opinion the sociological definition in this case seems to narrow, because it covers only a minor part of Web sites, namely social communities or other sites where people can interact with each other and exclude other ones. The other definition, where conviviality is reduced to the computer application's point of view, the human-computer interaction or evaluation and optimisation of the user interface, is also too narrow. In our sense, these approaches are a promising start for a better user experience, but usually they are regarded separately and there is a lack of a broader concept what conviviality could be in combination with Web applications.

In our opinion the World Wide Web could be a place where the user is able to enjoy a convivial experience.

Therefore we define *Artificial Conviviality* as follows:

> To enable a Web site to support a convivial experience for a user, the following two things must be fulfilled:
>
> 1. The user should be aware that he is part of a (social) group and be able to interact with it.
> 2. The system should be user-centric designed.

This definition is consciously lean towards the original definition of conviviality that involves a group, and (in a wider range) Illich's concept of "Convival Tools". We belief that for a real convivial experience the user should be aware to be part of a community. And in reality every Web site has a community, the community of their visitors. In our opinion a Web site should offer possibilities to connect these individuals, let them know that they are part of this community and enable them to become an active member in

this community, if they want to. Offering the user the opportunity to become part of the community may also influence his mental attitude towards the Web site. It is really important in this context to make it clear, that a convivial approach should never force an individual to become an active member that involves additional responsibilities.

The second part of the definition points out the importance to design a system with the primary focus on the user *and* the systems functionality. This kind of system should allow the individual to use a Web site or Web system in a comfortable way. In reality it could be very difficult to satisfy this requirement, because there are a lot of parameters that influence users' experiences and attitudes during a Web site visit. There are attributes such as interface or graphic design, the kind of Human-Computer Interaction, the architecture of the Web site, the response time and many more. Figure 2.2 gives a broad overview over disciplines that may involve for a good user experience.

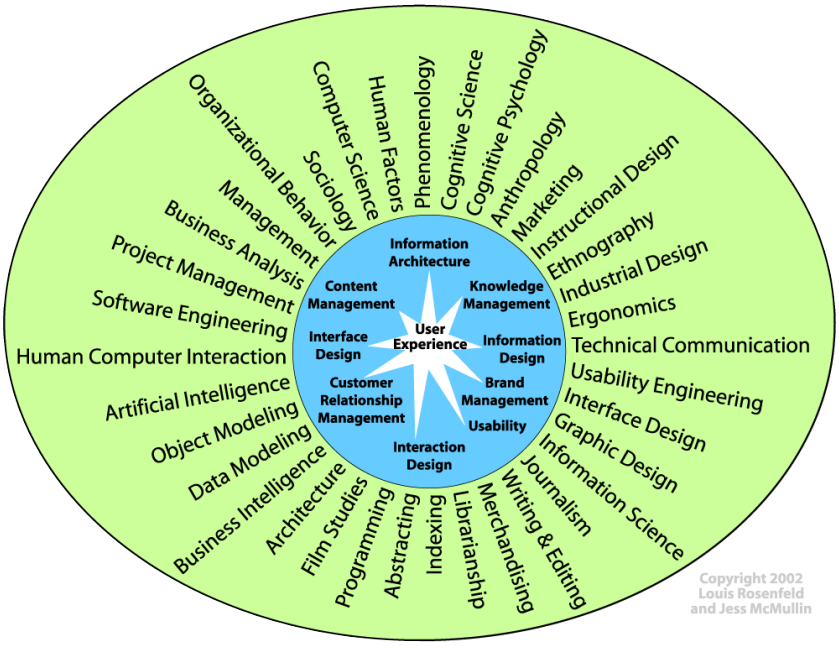

Figure 2.2: The User Experience Disciplines Radial [17]

Finally let us notice that the definition does not apply to use the User-Centred Design concept of the Usability Professionals' Association [18]. Nevertheless it might be a part of the process that results in a user-centred design that meets our requirements for *Artificial Conviviality*.

Chapter 3

The Wisdom of Crowds

Beside artificial conviviality, there is another concept that has high relevance for the given work: that is the concept of "The Wisdom of Crowds". In the following, we present the concept that was firstly introduced by Surowiecki [6]. We begin this chapter by introducing the idea behind "The Wisdom of Crowds".

In short, if an individual cannot solve a problem, a typical approach would be to consult one or more experts and follow their advice. The concept of "The Wisdom of Crowds" takes a different approach. Its main idea is to ask many different people (i.e. "crowds") about their knowledge related to this problem and combine their answers to get an appropriate solution. Surowiecki describes the idea as follows:

> "If you put together a big enough and diverse enough group of people and ask them to 'make decisions affecting matters of general interest', that group's decisions will, over time, be 'intellectually [superior] to the isolated individual,' no matter how smart or well informed he is." [6]

We discuss the details of the approach later in the chapter. To get an idea, how "The Wisdom of Crowds" approach works, we present two examples.

We begin with the "Jelly-Beans in the jar" experiment. Treynor [19] did this experiment in his class, where he presented his students a jar filled with 810 beans. The students' mean estimate was 841 beans. Only two of the 46 group members were closer to the right result. When he repeated the experiment with 850 beans, the result was quite similar. The mean result was 871 beans and only one person out of 56 was closer to the right value.

Another example of group intelligence is given by the Iowa Electronic Markets (IEM) project [20]. It is an online future market, which allows people to buy and sell futures on the outcome of elections, i.e. they bet on the outcome of elections. Over the years, it figured out, that this project predicts the outcome of elections with a high accuracy, even of the (relative) small number of participants [21].

There are many other examples (e.g. Google's Page-Rank Algorithm [22], etc.) that proof the concept of "The Wisdom of Crowds".

For a working instance of this concept, the following requirements must be satisfied: Diversity of Opinion, Independence, Decentralisation and Aggregation. We especially stress the points of gather knowledge and aggregate it to resolve a given problem. Note that these are very important points, because they connect the concept of Artificial Conviviality with "The Wisdom of Crowds" and motivate the later approach. The chapter includes also a short overview and comparison of closely related strategies like collaborative filtering, collective intelligence and motivates the decision for Surowiecki's approach. A discussion about "The Wisdom of Crowds" strengths, weaknesses and application areas finishes this chapter.

3.1 Surowiecki's Concept of "The Wisdom of Crowds"

"The Wisdom of Crowds" approach helps to find good solutions for some kind of problems. It is based on the idea to collect as much independent opinions as possible, aggregate and evaluate them, and finally come up with an appropriate solution. While these obtained results can be quite good, there is no guarantee for a successful outcome. This process of knowledge aggregation is nothing new at all by itself. Everyday, people ask others about their opinion to support them on a decision making process. However, today, because of e.g. the Internet, it is much easier to reach a number of people in a fast way. Of course, the approach is not suitable for every kind of problem. Surowiecki limits the concept on three different kinds of problems, which are *cognition* problems, *coordination* problems and *cooperation* problems. Nevertheless, experts are still needed (Nobody would like the idea of a doctor, doing a survey on the Internet during an operation, to decide what he should do next,...). However, for some kind of problems, Surowiecki's approach is more than appropriate.

Surowiecki specifies four requirements that must be satisfied to fulfil "The Wisdom of Crowds" approach. We present these requirements in the following.

3.1.1 Diversity of Opinion

The first requirement for a proper work of "The Wisdom of Crowds" is that there must exist a sufficient *Diversity of Opinion* inside the crowd. The reason is to get as much different perspectives as possible into account, to solve the given problem.
Under ideal circumstances, each person should have (and share) some private information, even if they are unusual or eccentric interpretation of the known facts that could help to solve the problem.

The demand for an adequate diversity may lead to different kinds of problems in practice:

- Unwanted group dynamic processes may arise, that distorts the result.
  For example, members will not come up with their own opinion, because they fear personal consequences, etc.

- Pattern of group composition.
  A group of experts may fail to solve a problem, because they share the same points of views. Most of the time, they might not be aware of this fact and therefore they are unable to change their positions. With a more open group there is a better chance to find appropriate solutions, because there will be more opinions, that means more points of view and therefore more options. This could make it also more difficult in situations where small groups are involved. In this case, it could be more difficult to get a "real" diversity of opinion inside the group.

On the other side, it is also important to take care, not to bury a potential solution in unimportant noises.
As an example how the rule of *Diversity of Opinion* works in practice, Surowiecki cites bee's foraging. During this process bees swarm out in different directions, hoping that at least one bee will find food (and share this private information) for the benefit of the swarm.

As a first conclusion, we maintain that Diversity of Opinions helps in the process to find a good solution, because it actually adds perspectives that would otherwise be absent. It also takes away, or at least weakens, some of the destructive characteristics of group decision-making. Fostering diversity is actually more important in small groups and in small organizations than in the kinds of larger collectives.

3.1.2 Independency

The *Independency* of the crowd members gives another requirement. At a first glance, it looks quite like the previous requirement, just in other words, but it is different.

Following this rule makes it easier to ensure the crowd's *Diversity of Opinion*. The motivation is to avoid mistakes people make when their behaviour becomes correlated. One side effect is that there is a better chance that independent individuals come up with new information instead of the same facts, everybody knows already. In fact, an individual's opinion should not be determined by the opinions of those around him. This independency is important to intelligent decision making for two reasons:
First, it keeps away the mistakes that people make from becoming correlated. Second, independent individuals are more likely to have new information.

This is a simplified assumption. In practice, individuals are embedded into social systems and, as we will see later, they are influence by other individuals.
Another non-neglected aspect that influences individuals is known as *information cascading* or *herding*. This problem may occur, when people's decisions are made in a sequence, instead of simultaneously. Therefore, it could happen, that some people start sharing their knowledge and the rest just follows. In this case, people might try to imitate others or they do not act on their own, but on what they think other people know. As Surowiecki says:

> "Collective decisions are most likely to be good ones when they are made by people with diverse opinions reaching independent conclusions, relying

> primarily on their private information. In cascades, none of these things are true. Effectively speaking, a few influential people – either because they happened to go first, or because they have particular skills and fill particular holes in people's social networks – determine the course of the cascade. In a cascade people's decisions are not made independently, but are profoundly influenced – in some cases, even determined – by those around them." [6]

Therefore, one approach to support Independency is to force people to make decisions simultaneously. We show an example that illustrates this, later in the chapter.

3.1.3 Decentralisation

The motivation for claim of *Decentralisation* is to allow people to specialise and draw on local knowledge. It should set a crowd of self-interested, independent people to work in a decentralised way on the same problem, instead of trying to direct their efforts from the top down. Their collective solution is likely to be better than any other solution a single individual could come up. This is a kind of bottom-up approach, where the result is build upon a set of partial solutions.

A good example might be the comparison between the development strategies from Linux and Microsoft Windows. While Microsoft is build on a command structure, and therefore it takes time to modify and improve Windows, Linux is build as a network of users, where every user is doing what he can do at best.

A drawback of the principle of decentralisation is that there is no guarantee that a satisfactory solution will be found, even if the required "special" knowledge exists inside the crowd.

3.1.4 Aggregation

The last condition to complete Surowiecki's requirements concerns the existence of an *Aggregation Instance*. He arrogates that there exists some mechanism for turning the private judgements into a collective decision. There is neither a specific mechanism mentioned, nor an explicit rule given, how to solve a given problem. In practice, the complexity of this mechanism might vary from simple to very complex approaches. In other words, they can vary from taking a simple average, over a weighted average or revert to a few (specialised) members only.

For example, to determine the number of jellybeans in a jar, it is sufficient to compute the average of the given opinions, i.e. estimations. A more complex example would be Google's page rank algorithm [22] that examine the relation between Web sites and turn them into a result set.

The general idea is that an instance collects and evaluates the input from the crowd members. Afterwards it decides which approach will be taken to solve the problem

3.2 Solvable kind of Problems

The concept of "The Wisdom of Crowds" is quite applicable for much kind of problems. As mentioned before, Surowiecki identifies three different kinds of problems, where the concept of "The Wisdom of Crowds" is suitable [6]:

- *cognition* problems,
  which have a definite solution.

- *coordination* problems,
  which requires members of a group to figure out how to coordinate their behaviour with each other, knowing/assuming that everyone else is trying to do the same.

- *cooperation* problems,
  which have the aim to get self-interested, distrustful people to work together, even when narrow self-interest would seem to dictate that no individual should take part

It should be obvious, as we touched upon, that there are also other kinds of problems, where the approach would not fit. In practice, this involves problems with a deep knowledge of the domain, the problem belongs to, like medicine, mathematics, etc.

On the other hand, there are problems that looks like to be solvable by a "Wisdom of Crowds" approach, but they are not. Tammet [23] gives an example, by mention Wikipedia and the discussion that it uses a "Wisdom of Crowds" approach to create accurate entries. He does not agree with this opinion and executes the following reasons: First, he cited an analysis of the University of Minnesota in 2007 that says that 10% of the editors are responsible for 86% and even 0.1% for 44% of the whole edits. In other words, these "super editors" control a huge part of Wikipedia. As a second argument, he argues that the process of creating and editing an entry is too complicated as to be handled by this approach.

3.3 Related Approaches

The concept of "The Wisdom of Crowds" should not be considered as an isolated approach. On the contrary, other approaches are related to it. In the following, we present two other concepts that, in our opinion, are in a strong relation to "The Wisdom of Crowds" concept.

Collective Intelligence

While there exists no common or formal understanding of what Collective Intelligence is, we want to present two definitions. Malone [24] and the "Centre of Collective Intelligence" at the MIT [25] define that collective intelligence is given, when:

- Group(s) of individuals doing things collectively that seem intelligent
- Groups addressing new or trying situations
- Groups applying knowledge to adapt to a changing environment [25]

We also cite Allag [26], who defines Collective Intelligence as follows:

> "When a group of individuals collaborate or compete with each other, intelligence or behaviour that otherwise did not exist suddenly emerges; this is commonly known as collective intelligence. The actions or influence of a few individuals slowly spread across the community until the actions become the norm for the community." [26]

Given these definitions and the application area where "The Wisdom of Crowds" techniques are being used, the "Wisdom of Crowds" method can be seen as a member of the Collective Intelligence family, in our opinion.

Collaborative Filtering

Since there is also no general definition of "Collaborative Filtering", we present the following one from O'Reilly [27]

> "Collaborative Filtering is one of many examples of internet-enabled collective intelligence". [27]

As we see, Collaborative Filtering techniques are also belonging to Collective Intelligence. It usually focuses on how to "filter" promising result from a set of data, using "Collective Intelligence" as a technique to achieve this goal. A common approach is based on statistical techniques, which uses concepts of similarities to produce recommendations. As an example, we want to refer to Google's PageRank algorithm [22], where the recommendations are based on the links between Web sites. In our opinion, this qualifies it as a promising candidate for an *Aggregation process* in conjunction with "The Wisdom of Crowds" approach.

3.4 Summary

In this chapter, we presented the principle of "The Wisdom of Crowds", a useful approach to find solutions for different kind of problems. We also presented the four principles that must be required to apply the "Wisdom of Crowds" method.
"The Wisdom of Crowds" strong sides are in finding solutions for cognition, coordination, and cooperation problems.

In the next chapter, we examine the concept of profiles and how they support to collect knowledge from people's behaviour. In Chapter 5, we deal with the aggregation process and discuss several approaches to perform it in conjunction with a profile approach.

Chapter 4

Introducing Non-Obvious Profiles

The topic of this chapter is profiles and how they may be helpful in the process of knowledge collection. We examine the meaning of profiles and show examples where and how they are used. We then introduce an algorithm to create the so-called Non-Obvious Profile (NOP). This algorithm uses a hybrid approach of explicit and implicit feedback to describe a Web site visitor's interests, by using a profile.

4.1 Profiles

Before we present the Non-Obvious Profile algorithm, we take a closer look at profiles in general. A common definition of a profile can be found in Merriam-Webster [28]:

> Main Entry: **pro·file**
> Pronunciation: \prō,fī(-ə)l\
> Etymology: Italian *profilo*, from *profilare* to draw in outline, from *pro-* forward (from Latin) + *filare* to spin, from Late Latin
> Date: 1645
> 1 : a representation of something in outline; *especially* : a human head or face represented or seen in a side view
> 2 : an outline seen or represented in sharp relief : contour
> 3 : a set of data often in graphic form portraying the significant features of something <a corporation's earnings profile>; *especially* : a graph representing the extent to which an individual exhibits traits or abilities as determined by tests or ratings
> ... [28]

In this work, we mainly focus on the third definition. While we consider the first and second entry in this context in a figurative sense. We might also interpret a profile as a kind of characteristic that describes a subject's most important properties. Rather than describe every single attribute and its value in detail, a profile only describes distinctive and noticeable attributes and passes on the less important ones.

4.1.1 What kinds of profiles exist?

As seen, profiles are not limited to specific domains. In practice, there exists a wide range of profiles and their applications. For example, if somebody wants to do a bicycle tour in the mountains, an *elevation profile* of the route can be very helpful (in this case it characterises the altitudes). Figure 4.1 shows an example of a stage from the Tour de France 2010. Profiles also exist for example on the job market, where the participants deal with company's profiles and job profiles of the offered jobs. There exist many more kinds of profiles and the examples can be continued endless.

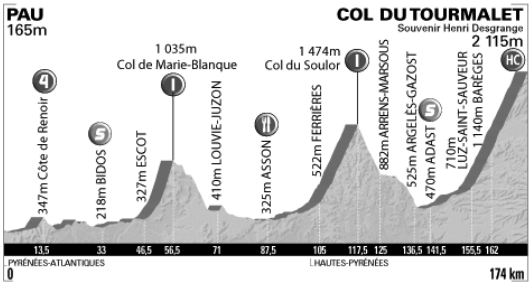

Figure 4.1: Example of a profile [29]

We get a little bit more specific and turn our focus on Web sites. In this context, there also exist different kinds of profiles, depending on the purpose. On the technical point of view, load-profiles of servers are widely used. These profiles are based on Web server log files and can be used to predict when a Web site gets high access rates. If an administrator knows, when he could expect a high access rate on the Web site, he then can optimise the server to reduce the general response time. Another thing is a Web page access profile, which allows someone to determine the probability of a user's next requested page, etc. The tools and techniques for doing this kind of analysis are commonly known as *Web metrics*. In its beginning, these techniques were heavily based on log files from Web servers. However, by the time they became increasingly smarter, by using additional techniques like cookies, hidden images or JavaScript. One reason of this was that people find out that profiles could help them to optimise their business. Another reason could just be to make it more comfortable for the user if the Web site provider knows something about the user's preferences.

So, how do they work and how can we build a user's profile? It is very simple. Usually every action that involves the Web server can be protocoled. As a result, the Web site operator knows (more or less accurately) which pages a user has visited or what he has done. This kind of profile is called an *explicit profile*. If other variables are coming into account, like assuming the "reading time", we speak of an *implicit profile*, because we make assumptions that may not be true.

From the technical point of view, there exists no general rule. Normally, an *explicit feedback* is more accurate then an *implicit* one. The reason is that the user normally decides consciously and knows what he is doing. The explicit profiles are built on

assumptions, models and guesses, how the world should be and what are the reasons that caused the users' actions. One solution could be to ask the user explicitly why he did an action and consider this when building the profile. The disadvantage could be that we ask the user too many times so that he will leave the site.

4.2 Non-Obvious Profiles

The Non-Obvious Profile Algorithm (NOP-Algorithm) was first introduced by Mushtaq et al. [30] and later extended by Hoebel et al. in [31] and [32]. It creates Non-Obvious Profiles (NOPs), which is one approach to model a Web site visitor's interests. Knowing visitors' interests can be used in many ways, such as for personalisation of a Web site or in recommendation systems to point the user to (unknown) content, that he is probably looking for. In the best case, the user appreciates the support given by the system.

Before we explain the algorithm in detail, we give a short overview. To build a user's NOP the algorithm uses two different techniques, which are *implicit* and *explicit feedback*. Roughly spoken, the system perceives the content of the visited pages and the user's behaviour during his stay to affiliate an *implicit feedback* profile. It then merges this information with an *explicit feedback* profile, which was given by the user before. The motivation is, to get an accurate interest profile while disturb the user as less as possible in his work (which would be the case, if we ask the user too often about his current interests). The name *Non-Obvious Profile* is derived from the fact that the algorithm takes the content and kind of actions into account to build a profile, in contrast to algorithms that are based on click-stream processing only. Figure 4.2 shows the principle function of the algorithm.

One of the advantages of this approach is that the algorithm can build meaningful profiles immediately after the system's initialisation and therefore avoid the so-called "cold start" problem. It trades off results in a more complex initialisation process. We will discuss these topics later in section 5.1.6

4.3 The Non-Obvious Profile Algorithm

After this short overview, we now present the algorithm in detail. In the following, we partition the algorithm in several pieces to make it easier to understand.

1. Preparation and initialisation of the system.
2. Compute a user's implicit interest profile and, if existing, update the old one.
3. Compute a Non-Obvious Profile for a visitor.
4. Merge an explicit profile with the NOP. Use the result for future computations.

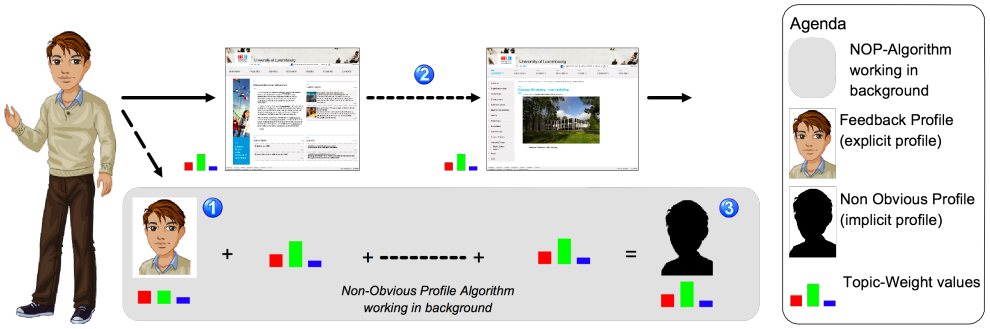

Figure 4.2: Principle of the NOP-Algorithm.
The visitor stores his explicit profile in the system ❶. He visits some pages ❷. After he finished the session, the system starts to build the NOP, by taking into account the content of the visited pages and performed actions ❸.

4.3.1 Preparation and initialisation of the system

To apply the algorithm, the first thing to do is to prepare and initialise the system. Here the Web site owner decides on which set of topics an interest profile should be based. These topics usually depend on the Web site's contents. Next, the topics are assigned to the related content together with a weight that describes the topic relevance in conjunction with the content. Finally the system must be 'taught', how to handle this information to build Non-Obvious Profiles.
The preparation has to be done only once at the beginning and then every time the content or actions are modified.

Let be more specific and describe the necessary steps for an initialisation in more detail: [1]

Define a set of Topics $\{Tp_1, Tp_2, ..., Tp_n\}$

These topics should cover the Web site's content and describe the future interest profiles. They have two functions: The topics describe the content and compose the attributes for the Non-Obvious Profile. As a by-product the Web site administrator gets a classification of the Web site. This is a very important decision because later computations (and hence interpretations) are based on these assignments. To do this task someone needs a basic understanding of the Web site's domain logic, which means there should exist a basic understanding of the contents and how it is represented to the user. Note that there is no specific instruction on how to do this classification.

In practice, this could be a very difficult process and should not be underestimated. The definition can be done manually or by taking advantage of automatic approaches,

[1] The initialisation process differs from these, given by Hoebel et al. in [31] and [32], but the results are equivalent.

such as document classification techniques like tf-idf [33], Naive Bayes [34], Latent Semantic Indexing (LSI) [35] or Latent Dirichlet Allocation [36], [37].

Example

Let us imagine that we want to use the NOP approach on our fictive Web site. Since we know the content of the site, we decide to define three topics that describe the meaning of the Web site's content. For reasons of visualisation, we also assign colours to the topics. In the example, they are *red* for $Topic_1$, *green* for $Topic_2$ and *blue* for $Topic_3$ We are going to continue the example in conjunction with each of the following steps.

Figure 4.3: Definition of three Topics: Tp_1 = red, Tp_2 = green Tp_3 = blue

Divide each Web site's page into zones

Divide each page P_j, with $(0 \leq j \leq n)$, of a Web site into zones, Z_k, where the *number of zones* ≥ 1. Afterwards assign to each zone Z_k one or more Tp_i that are related to Z_k's content. To precise this relation, assign a weight $v_j(Tp_i)$, where $0 \leq v_j(Tp_i) \leq 1$. The weight $v_j(Tp_i)$ describes how well the topic Tp_i covers Z_k's content, where the value 0 has the meaning of no relation and 1 of a perfect coverage or description.
This is probably one of the most difficult parts of the algorithm. The accuracy of the Non-Obvious Profiles depends very strongly on this step. Like in the previous step, there exists no "right" method to define zones and assign topics and weights to them. The task can be done manually or with support of document classification methods, already mentioned in the previous step.

In practice, there are two different techniques to build a Web site and its pages. If a page's content is fixed and if it is stored in html code on the Web server, the Web site is called a *static* Web site. We speak of *dynamic* Web sites, when pages are generated every time a request is made. The advantage of this approach is, that the Web site could be implemented with a modular approach, where the developer can create "templates" that could be used for many pages. An appropriate structure for the Web pages can be used to support the zone identifying process.

Example (continued)

After we defined the relevant topics for our Web site, we are now dealing with its single pages. Figure 4.4 shows three typical Web pages as an example. *Page 1* (on the left), contains only one zone, this means its content is only one block. A typical example for this kind of partitioning could be a page that contains a single article.
Page 2 (in the middle), contains two zones. They are arranged like a newspaper layout.
Page n finally represents a page with three zones.
These are only some examples; other layouts or partitions are possible too.

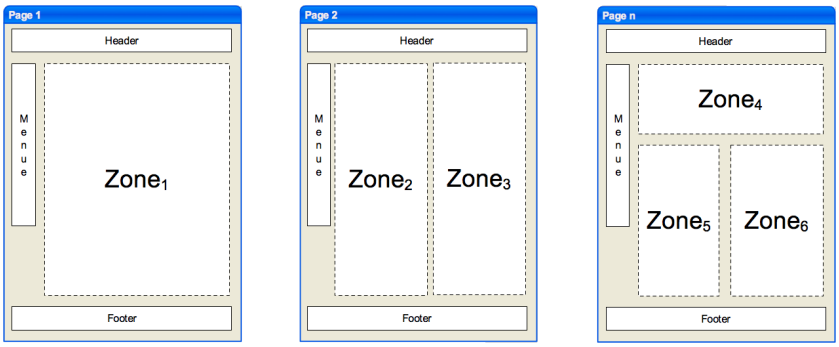

Figure 4.4: Example of Pages and their partition into Zones

Determine for each page its Topic/Weight values

Determine for each page P_j the page's Topic/Weight values $v(Tp_i, P_j)$. The idea behind this step is, to determine a $v(Tp_i, P_j)$, that describes a page's content in respect to Tp_i. Therefore, the algorithm follows a "bottom-up" approach, where it first determines the Topic/Weights for the zones and then for the page. The reason is that the algorithm later deals with these overall values, because it cannot detect in which zone the visitor is particularly interested.
In case a page contains only one zone, its Topic/Weights values are usually the same as the zone's ones. If there is more than one zone, we have to define these values explicitly.

To automate the process, Hoebel et al. [31] and [32] proposes three different rules, each based on a different assumption:

- $MIN - Rule$:
  for each topic i, scan all zones on a page P_j and take the minimum topic weight.

$$x_i = min(v(Tp_i, Z_q))\forall Z_q \in P_j \tag{4.1}$$

- $MAX - Rule$:
  for each topic i, scan all zones on a page P_j and take the maximum topic weight.

$$x_i = max(v(Tp_i, Z_q))\forall Z_q \in P_j \tag{4.2}$$

- $AVG - Rule$:
  for each topic i, scan all zones on a page P_j and take the average topic weight.

$$x_i = avg(v(Tp_i, Z_q))\forall Z_q \in P_j \tag{4.3}$$

Other approaches are possible and depend on the given circumstances.

Example (continued)

We continue the example and examine the impact of the three rules on a page's Topic/Weight value. The example is valid for static and dynamic Web sites. While, for a static Web site, the examples are fixed for each page, on a dynamic Web site, it shows the general principle of the strategy, because the zones act as placeholders only and the result depends on the content.
Figure 4.5 shows three examples. *Page 1* includes one Zone. In this case, all three strategies produce the same result. *Page 2* contains two Zones. In this case, the Page's Topic/Weight values depend on the chosen strategy. Here, the values for $Zone_2 = (0.8, 0.0, 0.6)$ and $Zone_3 = (0.4, 0.8, 0.2)$. In this case, applying the MIN-Rule would result in $Page_2 = (0.4, 0.0, 0.2)$. The MAX-Rule would result in $Page_2 = (0.8, 0.8, 0.6)$ and the AVG-Rule in $Page_2 = (0.6, 0.4, 0.4)$. *Page 3* follows the same approach.

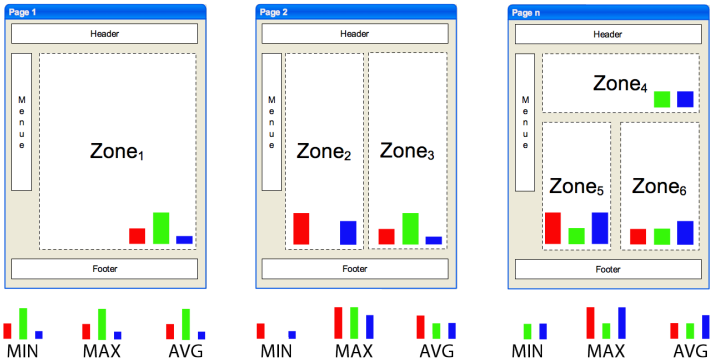

Figure 4.5: Example of Zones and the influence of the Page/Weight Strategy
The Zones contain the Topic/Weights regarding the Zones' content. Below each Page there are the resulting Topic/Weight values for the Page, taking the *MIN*-, *MAX*- or *AVG-rule* into account.

Identify relevant actions on the Web site.

In this context, relevant actions are all actions that imply an interest (or counter interest) in one or more of the previous defined topics. This has to be seen under action's affect in the real world and must be distinguished from the technical point of view.

While, technically spoken, an action is nothing more than sending a request to the Web server, its meaning in the real world could be quite different. For example, pressing the "buy"-button on a Web site is technically nothing more than a request, while it implies in the real world that an individual changes money for a product. Most of these actions directly imply an interest in one or more topics. For example, buying something could be interpreted as an interest in the product.

After having identified the relevant actions, we assign topic and weights to them in the sense that they describe the interest of the visitor. Most of the time it will be a good approach to stay related to the Topic assignments of the zone, the action could occur. For example, we assume the user wants to print the zone's content. Usually this indicates a (strong) interest in the specific content and therefore in the content's topics.

The reason for this step is to raise the NOP's accuracy. While, in the previous steps, we prepared the system to capture the user's interests by the requested pages, we now prepare the system to characterise the user's interests by considering his actions.

4.3.2 Determine Implicit Interest Profiles

After the preparation, the system is now able to create Non-Obvious Profiles. To do so, it records every page request and action a visitor performs during the stay on the Web site. After the user has left the site, the system starts to create the interest profile related to this session. Therefore, it uses the following algorithm:

1. Determine the duration the user has been stayed.
   This is done by sum up the duration the visitor stayed on each visited page P_k (Equation 4.4).

$$session\_duration = \sum_k duration(P_k) \tag{4.4}$$

   In practice this might be not a trivial task and can be very tricky, because it exists no exact method to determine the accurate time a user has stayed on a page. Stayed in this context means being active on a page (e.g. reading or interact with the page). This is difficult, because the system can only determine the time, a user requests a resource from the Web server. A common approach is to suppose a user stayed the whole time on a page between two page requests. Nevertheless, in practice, there is no possibility (by now) to proof this assumption. For example it might be possible, that the user switches to another application/browser tab or leaves the computer. A common practice is it therefore to define an upper time limit that will take place in case a user stayed longer on a page, to prevent the system to use wrong assumptions. For a deeper discussion of this problems and how they can be handled, we refer to Sterne [38], Eisenberg et al. [39] or Peterson [40].

2. Calculate for each topic Tp_i its interest value $DurP(i)$, based on the duration a user stayed on a page during the session.
   The system calculates the part of the NOP, which is based on the duration, a user stayed on a page P_j. For every Web page P_j that contains content related to Tp_i and that has been visited in the observed session, it takes the associated value $v(Tp_i, P_j)$. These $v(Tp_i, P_j)$ are multiplied by the duration, stayed on the page, and added to $DurP(i)$. Finally $DurP(i)$ is divided by the sessions total duration to compute the topic ratio for the session. The whole step is summarized in 4.5.

$$DurP(i) = \frac{\sum_j duration(P_j) * v(Tp_i, P_j)}{session\_duration} \tag{4.5}$$

   The underlying motivation in this step is, as already touched, to build an Interest Profile; regarding the duration a user "reads" a page.

3. Determine the actions the user performed in calculating a cumulative value.
   Analogue to the first two steps, we first calculate the session's cumulative action weight for a further usage 4.6.

$$sum\_actionweights = \sum_s aw_s \tag{4.6}$$

4. Calculate for each topic Tp_i its interest value $ActP(i)$, based on the actions a user has performed during the session. The system calculates the part of the NOP, which is based on the actions a user has performed during the session. For every Web page P_j that has been visited and where an action has been performed that is related to Tp_i, the algorithm takes the associated value aw_t and multiplies it with $v(Tp_i, P_j)$. The results are added to $ActP(i)$. Finally $ActP(i)$ is divided by the sessions $sum\_actionweights$, computed in the last step, to compute the topic's action ratio for the session. The whole step is summarized in 4.7.

$$ActP(i) = \frac{\sum_q (\sum_t aw_t * v(Tp_i, Z_q))}{sum\_actionweights} \tag{4.7}$$

5. Calculate a topic's session value by combining the Interest Profiles based on the Interests and the Actions. To determine the sessions' interest value x_i in Topic Tp_i, the algorithm combines the previous calculated $ActP(i)$ and $DurP(i)$. This is done by performing a "weighted" addition (4.8) where $(a + b) = 1$. This allows the algorithm to weight a factor stronger than the other one. If $a = b = 0.5$ it means that both parameters have the same influence on the result.

$$x_i = a * ActP(i) + b * DurP(i), \text{where a + b = 0} \tag{4.8}$$

6. If there exists already a value from a previous session, update this value with the calculated one. In the final step, we may update a previous build NOP. We do this by taking the old NOP value w_i and multiply it with the number of previous sessions *scount*. We also take the computed x_i, multiply it by a factor f, that determines how strong the new NOP will be influenced by the session's NOP, and add it to the previous result. We divide the result by the sum of $scount + f$ (4.9).

$$w_i = \frac{scount \cdot w_i + f \cdot x_i}{scount + f} \tag{4.9}$$

Listing (1) shows another way to illustrate the algorithms functionality.

4.3.3 Explicit feedback

As mentioned above, an explicit feedback implies the system to use data that have been entered by the user explicitly. The user tells the system about his current level of interest in the given topics. The user then enters a value that describes his level in a good sense. We assume the user interprets a topic in the same sense as the system and gives an honest answer. After this step, the system knows the user's true level of interest in each topic.

At this point, the system has to deal with the user's *implicit* and *explicit interest profile*. This is an important moment in the algorithm, because most of the time both profiles will not match against each other. In other words, there will be a difference between "what the system knows" about the user and his self-evaluation. One solution is to continue with the average of both profiles.

Useful indicators

Before we continue, let us examine the algorithm's state at time t_i (Figure 4.6).

The system contains the values of the current and last *implicit* interest profile (i.e. NOP) and *explicit interest profile*. It uses this information to compute the following indicators:

1. The **NOP-difference (ND)**,
   which is the difference between two NOPs. To determine **ND** from two NOPs, we subtract the older NOP (at time t_{i-1}) from the newer one (at time t_i). (4.10)

$$NOP(Tp_j, t_i) = NOP(Tp_j, t_i) - NOP(Tp_j, t_{i-1}) \tag{4.10}$$

   ND represents the change of interest between two sessions. A positive result indicates an increased interest in Tp_i, while a negative value indicates a lower interest.

/* Determine the duration for the whole session */
$session\_duration = \sum_k duration(P_k)$;

/* Determine the sum of all action weights */
$sum\_actionweights = \sum_s aw_s$;

foreach *Topic* i **do**

/* Determine the Duration Profile, where */
/* $duration(P_j) :=$ time stayed on Page j (in seconds) */
/* $v(Tp_i, P_j) :=$ value for Topic i on Page j */

$$DurP(i) = \frac{\sum_j duration(P_j) * v(Tp_i, P_j)}{session\_duration};$$

/* Determine the Action Profile, where */
/* $aw_t :=$ action weight for action t (in seconds) */
/* $v(Tp_i, Z_q) :=$ value for Topic i in Zone q */

$$ActP(i) = \frac{\sum_q (\sum_t aw_t * v(Tp_i, Z_q))}{sum\_actionweights};$$

/* Compute Interest Value in Topic i, where */
/* a := proportion of Action Profile on session Interest Value */
/* b := proportion of Duration Profile on session Interest Val */
/* and $(a + b) = 1$ */

$x_i = a * ActP(i) + b * DurP(i)$;

/* Update overall interest value in Topic i, where */
/* $scount$:= number of sessions */
/* w_i := last NOP value for topic i */
/* f := function to control value */

$$w_i = \frac{scount \cdot w_i + f \cdot x_i}{scount + f};$$

end

Algorithm 1: Compute a user's interest profile for a session

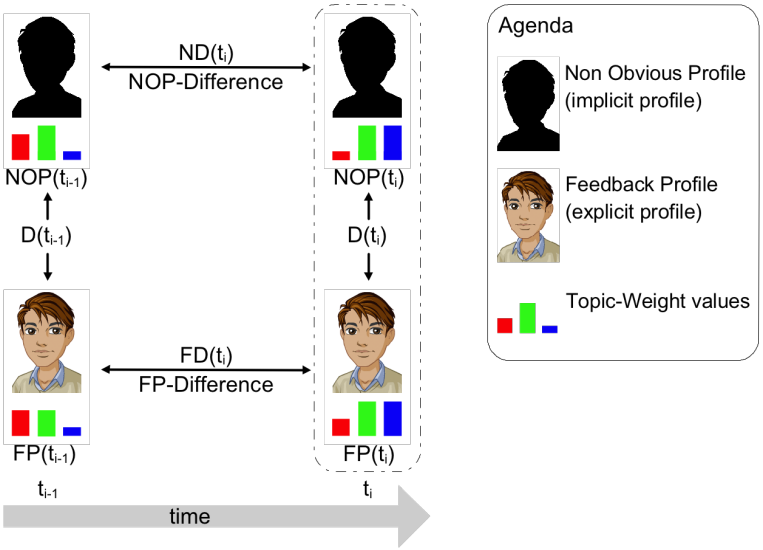

Figure 4.6: Dependencies between Indicators [30]

2. The **FD-difference (FD)**
   That is the difference between two contiguous explicit feedbacks. To achieve the FP-difference at time t_i we also subtract for each topic Tp_i the value $FP(t_{i-1})$ from $FP(t_i)$, as seen in (4.11)

$$FD(Tp_j, t_i) = FP(Tp_j, t_i) - FP(Tp_j, t_{i-1}) \tag{4.11}$$

   The indication follows the same principle as before. *FD* represents the change of interest between two sessions. A positive number indicates an increased interest, while a negative value indicates a decreased interest in Topic Tp_j.

3. **Difference (D)**
   This is the difference between a NOP and an FP at time t_i (*implicit* and *explicit* feedback). To compute the difference between a calculated NOP and a given feedback profile it follows the usual pattern. For each Topic Tp_j we subtract the NOP value from the FP value (4.12).

$$D(Tp_j, t_i) = FP(Tp_j, t_i) - NOP(Tp_j, t_i) \tag{4.12}$$

   This value represents the discrepancy between the NOP and the feedback profile. In other words, it values the difference between how the system assesses the user and how he assesses himself.

How can the system take advantage of these indicators? As mentioned before, the most important element is the difference between the explicit and the implicit value,

$D(t_i)$. In an ideal case, there would be a perfect match between a NOP and a feedback ($\pm$ a small delta δ that indicates when two values could be considered as "equal"). However, in reality two profiles match in rare cases only. There could be many reasons, why both profiles differ: for example, the algorithm computes the NOP very wrong. In this case, we should use the feedback profile **FP**, given by the user. But even if the algorithm is correct, other sources of inaccurate profiles could be wrong assignments of topics and their weights, different topic interpretations by the user and the Web site owner or wrong feedback profiles FP, entered by intention.

To compute a new profile and avoid possible sources of errors, Mushtaq et al. [30] suggests the following approach (Algorithm 2) :

if $|FD(Tp_j, t_i)| > y \; where \; 0 < y < 1$ **then**
  $DP(Tp_j, t_i) := FP(Tp_j, t_i)$ ❶
else
  if $|D(Tp_j, t_i)| < x \wedge 0 < x < 1$ **then**
    $DP(Tp_j, t_i) := avg(FP(Tp_j, t_i), NOP(Tp_j, t_i))$ ❷
  else
    $DP(Tp_j, t_i) := NOP(Tp_j, t_i)$ ❸
  end
end

Algorithm 2: Compute session interest profile

This approach can be described as follows:

❶ If there is a change in the FP which is greater than a given value y, we assume that there is a fundamental change in the user's interests. We consider this by taking the FP values as the new origin for future NOP calculations.
Because the *FP* is based on an explicit feedback, it reflects the users interests in a very good sense. This could be of advantage by calculating further NOPs.
On the other hand, there is a risk, the user did not enter his real interests (on purpose or because of a different understanding); then the NOP algorithm gets a wrong direction and computes meaningless results, based on wrong assumptions. This may also happen, e.g. if the user changed his interest in the recent past and the NOP-algorithm is not fast enough to reflect this change.

❷ Otherwise, if there is only a small difference between FP and NP (smaller than y), we interpret this, that the NOP reflects the user's current interest profile very accurately. In this case, we use the average between FP and NOP as the new starting point for future NOP calculations.
Choosing this option is without any risk to use wrong start values for future NOPs, but we may throw away the chance to improve future NOP values

❸ If none of the above rules applies, we assume a stable pattern of interest (since $FD < y$). However, there is also significant difference between the FD and the calculated NOP ($D > x$). There can be different reasons for this: e.g. a wrong feedback, a different understanding of the topic or a wrong topic mapping to some content. Here we can do several things, depending on the underlying assumption. If we assume a user has given a wrong feedback, we do not have to take care at the *FD* and could continue to use our NOP values for future computations. Otherwise, we can compute the difference D and use it for further computations.
We think that in both case the administrator should use some additional information to explore the reason for this gap. For example, he could try to detect suspicious pattern in the feedback. An indicator for some different understanding could be given, if a significant number of users show the same symptoms.

Chapter 5

The Aggregation Process and NOPs

In the last three chapters, we introduced the concept of *artificial-conviviality*, discussed the principles of "The Wisdom of Crowds" and introduced the Non-Obvious Profile Algorithm. When we remember "The Wisdom of Crowds" principles, we find that we can use the NOP-Algorithm to collect knowledge and represent it in form of profiles. In the following, we are going to present the missing part, the aggregation process.

As we have seen in chapter 3, "The Wisdom of Crowds" approach allows us to solve some kinds of problems, which are cognition, coordination and cooperation problems. As already mentioned in the definition of *artificial-conviviality*, we primary want to support users during their web experience to influence their conviviality. For this we consider a visitor's request for support as a problem that can perhaps be solved by the community. In other words, we try to take advantage of the wisdom of crowds so, that the system is able to offer the user a solution. In this work we assume without loss of generality, that the conditions of "Diversity of opinion", "Independency" and "Decentralisation" are satisfied. (In reality, there is no real guarantee that these assumptions are satisfied. Especially on web sites with a small user community one or more of the conditions may be not satisfied.)

In general, we can assume that visitors of a common web site do not know each other personally, which means there is usually no dependency between the users. (Nevertheless, there might be situations, where people know each other, e.g. private web sites, communities, etc. but in general this is an exception.) We can also assume that there is a diversity of opinion between the users and that the individuals are decentralised.

On the other side there might be no real diversity of opinion, because the users share their interest in the website. Then there are "outside sources" that may influence a "monotony of opinions", like the mass media or social interactions for example.

Since it is impossible to ask individuals about their opinions, every time a problem occurs, we have to find another way to perform this task. One approach would be to store the knowledge of each person at a central place that we can use to perform the

aggregation process. This is where the NOP becomes a matter of interest. We use the NOPs in this work to store the individuals' knowledge and use it for the aggregation process. The aggregation process will be the main subject of the following section.

5.1 Knowledge Aggregation

The aim of the aggregation process is to generate a solution that is based on the collected and available data (knowledge). We are now at a point, where we use the collected data (i.e. NOPs) to offer solutions based on the crowds' wisdom. Since the topics of this work are *artificial-conviviality* and Web sites, our intention is to use the visitors' profiles to support a recommendation system.
To give a short example, we examine a situation, where a visitor wants a recommendation from the system that covers his interests. In our model the recommendation system is feed by the interest profiles of other users and their behaviour during their visits. There, the knowledge is reflected in the interests, which are also related to the actions, the users performed during their web site visits. The underlying assumption is that people with similar interests, like and prefer similar thinks. A technique, that supports this task, is known as "collaborative filtering".

There are several approaches to perform collaborative filtering in practice. One method to distinguish collaborative filtering techniques is to divide them into approaches that are based either on similarities or on descriptive models, which in our case are built on the collected NOPs. In the following, we discuss some approaches that enable us to perform an aggregation of knowledge.

5.1.1 Similarity based Approaches

The idea behind similarity-based approaches is to describe a user with a set of parameters. Additionally there is also one rule that describes how to determine and measure similarity, or considered vice versa, dissimilarity.

If there is the need to predict a user's future behaviour, a possible approach could be to compare the user with similar users and create a prediction that is based on these similar users' behaviour in the past.

Alternatively, it is also possible to determine similarities between users' profiles and the documents, provided by the system. This could be the case in situations where there are not enough data or information available. However, to apply this approach, it is necessary that the documents be classified before they get used (which is the case with the NOP approach).

The following represents a small excerpt of techniques to determine and measure similarities. To use these techniques it is necessary to "put" the profiles into an *n-dimensional vector space* that allows us to relate a profile to another one. In our case, this is no problem, because we can use the NOP and regard them as vectors. In order to simplify things we speak in the following of *profiles* when we mean *vectors*.

Distances

One way to determine similarities is to regard the distance between two profiles. If we are able to put a profile into a *normed vector space*, we are able to measure the distance between two profiles. The distance then can be interpreted as a degree of similarity: the closer the profiles, the more similar they are.

If we consider two profiles, where a profile consist of two attributes x and y, then $profile_1 = (x_1, y_1)$ and $profile_2 = (x_2, y_2)$.

A typical measurement of a distance can be done by using the *Euclidean Distance* (5.1):

$$Euclidean\ Distance(profile_1, profile_2) = \sqrt{|x_1 - x_2|^2 + |y_1 - y_2|^2} \tag{5.1}$$

Another distance that can be used is the *Manhattan-Distance* (5.2):

$$Manhattan\ Distance(profile_1, profile_2) = |x_1 - x_2| + |y_1 - y_2| \tag{5.2}$$

In general, these distances are known as $L_n - distances$ and they are defined as (5.3):

$$L_n\text{-}distance(profile_1, profile_2) = (|x_1 - x_2|^n + |y_1 - y_2|^n)^{\frac{1}{n}} \tag{5.3}$$

One characteristic of the $L_n - distances$ is that it preserves the proportion of distance, even if n is changing. It means that the absolute value of a distance may change, but the ratio will be the same. If, for example, $profile_b$ is more similar to $profile_a$, than $profile_c$ in a specific $L_n - distance$, then this will be valid within all $L_n - distances$.

Angle based Similarities

Another way to define similarity is by calculate the angle between two profiles. We then can measure the angle between two vectors instead of using the distance between two points. The idea behind this approach is, to describe similarity via the proportions between the attributes. A common approach is to compute the cosine of the angle formed by two profile vectors (5.5). We compute it by using the *dot-product* (5.4).

$$profile_1 \cdot profile_2 = x_1 * x_2 + y_1 * y_2 * \ldots z_1 * z_2 \tag{5.4}$$

$$\begin{aligned} Sim(profile_1, profile_2) &= cosine\theta \\ &= \frac{profile_1 \cdot profile_2}{|profile_1||profile_2|} \\ &= \frac{x_1 * x_2 + y_1 * y_2}{\sqrt{x_1^2 + y_1^2} * \sqrt{x_2^2 + y_2^2}} \end{aligned}$$

In practice, this technique is applied in Information Retrieval, where the similarity of two documents is often measured this way (Salton et al. [41]).

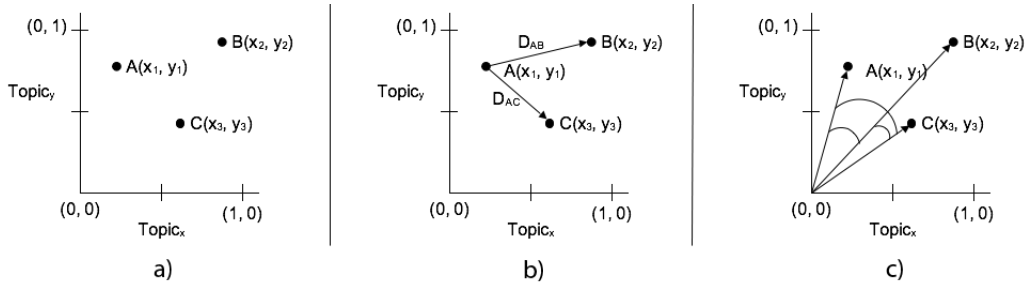

Figure 5.1: Example of Similarities
a) shows profiles of three users A, B and C and their interests in Topics x and y;
b) shows there similarities based on the distance; c) shows their similarities based on the cosine-vector concept

Correlations between Attributes

Another possibility to define similarity is to concentrate at the correlations between the profile's attributes. If two profiles share the same correlations, they will be called similar, regardless of the absolute distance between them. A method to measure this kind of similarity is given by the *Pearson Correlation Coefficient* or *Pearson Score* (5.8).

$$\sum P = x + y + \ldots + z \tag{5.5}$$

$$\sum P^2 = x^2 + y^2 + \ldots + z^2 \tag{5.6}$$

$$\sum P_1 P_2 = x_1 * x_2 + y_1 * y_2 * \ldots z_1 * z_2 \tag{5.7}$$

$$Pearson\_Score(P_1, P_2) = \frac{\sum P_1 P_2 - \frac{\sum P_1 \sum P_2}{N}}{\sqrt{(\sum P_1^2 - \frac{(\sum P_1)^2}{N} \sum P_2^2 - \frac{(\sum P_2)^2}{N})}}$$
where N is the number of Topics

(5.8)

Figure 5.2 illustrates how the *Pearson Score* works. We assume that all spots represent profiles. If samples of spots are "lying at a line" it means their attributes share the same correlation among each other. This is the case where the values are 1.0 and -1.0. As we can also see, the algebraic sign indicates the kind of correlation. If we allow a less strict correlation, the line becomes "broader".

This approach is similar to the cosine-vector concept, but not equal.

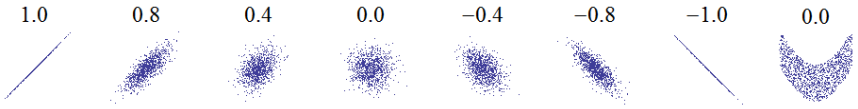

Figure 5.2: Example of Correlations ([42])

5.1.2 Model based Approaches

In contradiction to the previous methods, the model-based approaches are based on a different concept. The primary idea is to take all available data and create mathematical models that describe these data and their relations as accurate as possible. For the outside user it looks like a black-box. The main work in this approach has to be done by create the "right" model, while the output is usually based on the cheap execution of the model. This differs from the previous methods, where there is not that much preparation to do, but the computations are expensive, because they are related on the number of comparisons between user profiles. In the following, we present two typical approaches that could be classified as model based.

5.1.3 Data Mining / Associative Rules (machine learning)

Data Mining describes the task to extract new knowledge from existing data. It is not a unique method; in fact, it is more a concept and includes a set of different methods from different kind of disciplines like statistics, neural networks or machine learning to perform this task. More formally, Cabena et al.[43] define Data Mining as:

> the process of extracting previously *unknown*, *valid* and *actionable* information from large databases and then using the information to make crucial *business decisions*. [43]

It is out of the scope of this work to introduce the reader in Data Mining, or to go deeper and explain some techniques in detail.

5.1.4 Neural Networks

Neural Networks can also be appropriate tools in the process of aggregation. We also do not want to give an introduction into neural networks at this place. In short, neural networks are models to simulate the human brain. They consist of nodes (neurons), which are connected in one way. The connections are usually weighted. If a connection is active, it "fires" the weights value to its output. The neuron at the output adds all its active incoming weights. It "fires" too, if its internal function is true.

There are many application areas for neural networks. They are commonly used for function approximation, time series predictions, classification, pattern recognition and data processing, like filtering and clustering. There are different approaches to train a

neural network, like supervised or unsupervised learning, etc. A common approach is to train a neural network with data of a so-called "training set" and verify the results with an independent set of data.

5.1.5 Cluster Algorithms

As we mentioned earlier, another approach, to determine content that could be of interest for the user is to deal with the documents directly. The aim of cluster algorithms is to partition data into a set of clusters or segments. A cluster, roughly spoken, has the characteristic that it contains "similar data" only. This has the advantage, in our case, that we have to cluster the documents only once. Afterwards we can use the result and recommend documents from clusters, where we know the user is interested in. Problems that may occur during the process of clustering may concern the number of clusters. If there are too few clusters, the results may be too vague. On the other hand, to many clusters may result in a small number of recommendations. Well-known cluster algorithms are K-means Clustering or hierarchical cluster algorithms like Agglomerative Clustering, Divisive Clustering (all [44]) or Self-Organizing Maps [45].

Bayesian Network Model

Instead of measure similarities between users and pages, which requires pages classifications, we could also use probabilistic approaches. In this case, we can change our focus on the question, to calculate expected values for a user to read a given page or perform a specific action, based on the information we already know about the user. This might be a difficult task, because we also have take pages into account that were not read by now.

One approach to determine the probability that a user would like a specific page is to use a Bayesian Network. In this case, the network consists of n nodes, where n is the number of pages in the system. The states of each node are equal to the probability that the user would be interested in the page, related to the node. An additional state exists for the "non read" event, which are used during the training process to handle these data. During the training process, the algorithm searches over some model structures to detect dependencies between the pages. In the final network, each page will have a set of parent pages, that indicates the best predictions based on the stored information.

5.1.6 Discussion

Breese et al. [46] has examined similarity based and model based approaches. In there work, they distinguish between two types of algorithms.

The main characteristic of "Memory based Algorithms" is that they have to check every profile, before they are able to produce a result. Our similarity-based approaches belong to this category, because they require an examination of all profiles, to identify similar ones.

"Model based Approaches" on the other side, have the characteristic, that they are based on models. These models are normally extracted from observations of available data. Suitable tools for constructing these models may be coming from Data Mining for example. In case the user wants to filter something, the model then produces a result.

They ran several tests to characterise both kind of approaches in practice. The results were not always clear. It figures out, that the advantage of the "Model based Approaches" can be found in their performance. They usually are more accurate in their predictions and they are faster and less memory consuming. On the other side, their learning phase could take a long time, not to mention that they need data to learn.

This in turn, is the advantage of the "Memory based Algorithms". They do not depend on this "cold start problem" and they are ready to go from the beginning.

The results seem to indicate, that a hybrid-approach might be a good solution. In the "beginning" phase of a project, the "Memory based"/"Similarity based" algorithms could be used to provide suitable results right from the start. After a while, if enough data are available to start constructing models, the web site may switch to a "Model based Approach" to come up with better predictions. A disadvantage of this approach might be the switch between the methods. In practice, it would consume many resources and might be expensive, not to mentioned the enormous demand of time.

While Breese examines only some "simple" approaches, in reality, it seems quite more difficult to construct good, and here we mean really well, algorithms for recommendation systems. Most of the e-commerce web sites depend on recommendation systems, and a slightest advantage might be the difference for the customer to buy an item or not. Because of this reason, and to illustrate how difficult it is to construct a good recommendation system, we present the story of the Netflix-Prize in the next section.

5.1.7 The Netflix Prize

After we discussed several techniques to aggregate knowledge from data, we now present a real example for a recommendation system to demonstrate its complexity. We do this by presenting the case of the "Netflix Prize".

Netflix is a company located in California, USA that offers movies to their clients. A customer can either borrow DVDs by mail or watch streamed movies directly on their computer or TV. Customers choose their favourite movie, put it into their queue and decide about the kind of delivery. To support the customer during the selection process, Netflix provides a recommendation system, called *Cinematch*. This system has access

to the customer's history and to users movie ratings. [1]

In October 2006, Netflix announced an open contest [49] to raise the accuracy of *Cinematch*. They announced a grand prize of 1 million US Dollar for an algorithm that improves *Cinematch* by 10% or more. They further announced an annual progress price of 50.000 US Dollar for the team with the best algorithm within the last 12 month. A condition was that the winners have to publish their algorithm. To support interested developers, Netflix also released a data sample with an anonymised extract of their real data for working purposes. After a rapid improvement at the beginning, it figures out that there were only small steps of improvements afterwards. An interesting insight from the 2008 progress price winning team is given by Bell et al.[48]. After two years of disbursement the progress price, Netflix announced a Grand Prize winner on September 21, 2009. Overall, there were 51051 contestants organized in 41305 teams from 186 countries. During the run of the contest, Netflix has 44014 valid submissions from 5169 different teams [50].

After three years of improvements, which were not constantly, a consortium of three teams finally won the prize. It is very interesting to analyse the competition's process. At the beginning, there were fast improvements of the original *Cinematch* algorithm. After this period, the algorithms improved slower. Then the competitors noticed that they might improve faster, if they combine their different algorithms. The 2008's Progress Prize, for example, was won by a cooperation of two former independent teams. A successor of this team (another team joined in) finally won the competition [51]. The final algorithm is composed from three different approaches, each for one group, which can be found at Töscher et al. [52], Piotte et al. [53] and Koren [54]

---

[1] To get an idea of Netflix's business, here are some facts, taken from Netflix's Press Kit [47]:
- Netflix has 12.3 million subscribers at the end of 2009.
- Netflix has more than 3 billion movie ratings from members.
- The average member has rated more than 200 movies.
- Approximately 60% of Netflix members select their movies based on movie recommendations tailored to their individual tastes.
- Netflix members add 2 million movies to their Queues every day.
- More than 90% of Netflix members say they are so satisfied with the Netflix service that they recommend the service to family and friends.

The last point is very important for Netflix's business model. Since there are many factors that influence a customer's satisfaction like quality and speed of delivery, etc., but there is one aspect that is very important in our context: the fact that approximately 60% of all customers use the recommendation system to find interesting movies. Unfortunately, there is no further explanation how Netflix measures the usage of the recommendation system. For example, it could include all users that used the system at least one time and then dropped it. However, it seems to be obvious that the recommendation system is one important, maybe the most important, feature for Netflix to "sell" movies to the customers. On the other hand, is it very important that the recommendation system comes up with an accurate recommendation for some grade. Otherwise, nobody will use it. For now, let us assume the users are satisfied with the recommendations. Another important thing is that Netflix has a large clientèle and customers that are willing to rate movies and take this process serious. This is a good result taking into account that there is no constraint to rate movies. Nevertheless, how accurate are the ratings? [48] argues that the ratings are very accurate, because the users know that their rating will influence the underlying system and will gain from it in the future. In other words they are afraid of getting suggested movies they are not interested in by the recommendation system.

We can learn three interesting things from this example:

1. It seems that there is no "simple" statistical approach that could generate the best results for a recommendation system. Even with a huge amount of data in the background.

2. However, this data could make it easier to design improved algorithms with a higher accuracy in recommendations.

3. Finally, this approach leads us back to the Wisdom of Crowds approach, where the aggregated recommendation is better then an expert's one [48].

5.2 Discussion about Recommendation Systems

As we have seen there are many techniques, which could be helpful in our aggregation process. Which of one should we use for our recommendation system? Honestly, we do not know. In our opinion none of these approaches are helpful, because they violate a simple fact: They are not based on independent decisions. The first time we use a result of any of these recommendation-algorithms, future results will depend on each other. We are in a real dilemma. We want to serve a visitor at our best and come up with recommendations, but the trade off is that we (probably) influence future users in addition, because of the decision the current visitor will do.

In this context, we want to present an experiment, performed by Salaganik [55]. In this experiment 14341 people were partitioned into nine groups. The participants spent their time on a music portal, which was created for this project, where they could listen to unknown songs or download them. One group got no additional information about the music, while the other groups where also able to see the statistics for their own group. We may expect an equal or normal distribution in the groups, but showing the statistics changed it all. While the first group's behaviour was as expected, the rankings in the other groups shown an interesting phenomena: Titles that where successful where more successful as expected (the same applies in the other direction). There were even no two groups that share the same successful interprets. It looks like, that people are influenced by the ranking, which at the beginning were randomly. However, it seems, most people look at "successful" songs and keep concentrate on them. Salaganik explanation is that people do not want belong to losers, or do something "wrong" and therefore try to choose the winners, or songs that other already have chosen.

This experiment shows impressively our dilemma (and it also might hurts the condition of in-dependency of the Wisdom of Crowd approach).

Part II

Prototype Implementation

Chapter 6

CUBA - Concepts and Architecture

After presented the theoretical basics of the work, this part introduces the implementation of these concepts. In this chapter, we present CUBA's concept and its application architecture. The CUBA prototype itself is available at `http://cuba.lu`.

6.1 Concepts

CUBA is a prototypical implementation to prove the, previous developed, concepts of artificial conviviality. It is important to note, that the following concept and prototype is not the only way to achieve artificial conviviality. The reader will be encouraged to realize his own implementation(s) in another way.

The heart of our concept is a demonstration of how a Web site may support the concept of artificial conviviality (and Illich's approach). For this reason we designed and built a prototype called CUBA, that is an abbreviation for Conviviality and User Behaviour Analysis (CUBA). CUBA implements a News-feed reader that allows visitors to manage News-feeds. They can read News-feeds on CUBA, subscribe to feeds and cancel feed's subscriptions. In case they subscribe to a feed, CUBA offers a recommendation system to find suitable feeds. This system is based on the NOP approach combined with a collective filtering system that takes other users' actions and preferences into account to recommend adequate feeds. One distinctive feature of CUBA is a "dynamic" Home page for every visitor. It allows the user to modify the page to his personal comfort, by putting the subscribed feeds to different places.

Figure 6.1 shows the general functionality of CUBA. While a visitor uses CUBA, the system tries to extract the visitor's "knowledge" and stores it in the system. In 6.1 individuals (at the outside) use a Web site that includes the CUBA system. The portraits

in the inside represent the visitors “knowledge”, i.e. their interest profiles.

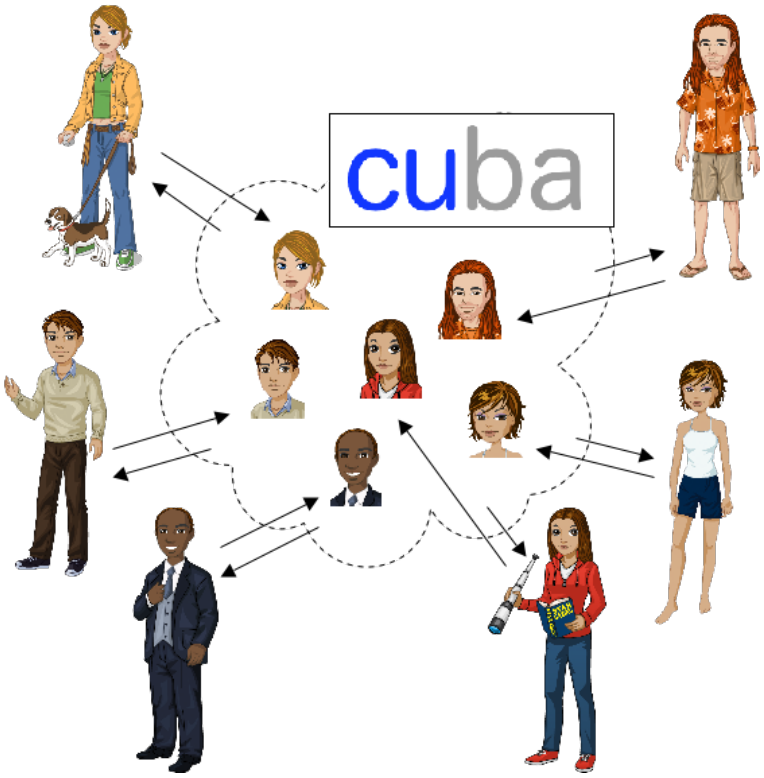

Figure 6.1: Visualisation of CUBA’s Concept

Next, we give a general idea how the system implements the convivial aspects, shown in Figure (6.2)
What happens if a user looks for a new feed? ❶ In “The Wisdom of Crowds” context, this means the user is looking for a solution (the right feed), to his problem (is there a feed?). In this case, the CUBA system performs an aggregation process, which is also part of the “Wisdom of Crowds” concept. In a first step, it identifies “similar” users ❷. It then “asks” these users which feeds they would suggest to solve the problem ❸. In the next step, it prepares the received suggestions and presents them to the user ❹.

In our opinion, this satisfies the definition of artificial conviviality, because every person is automatically part of the group, while its “knowledge” is modelled by the system. On the other hand, there are other people involved to find a solution for the problem. In other words, this is a kind of common experience with the aim to support a user and enable him to have a convivial experience.

After shown the main concept, we also present another important concept of CUBA. CUBA takes heavily advantage of the underlying NOP-algorithm, introduced in section 4.2. This algorithm builds interest profile by using implicit and explicit feedback. Since

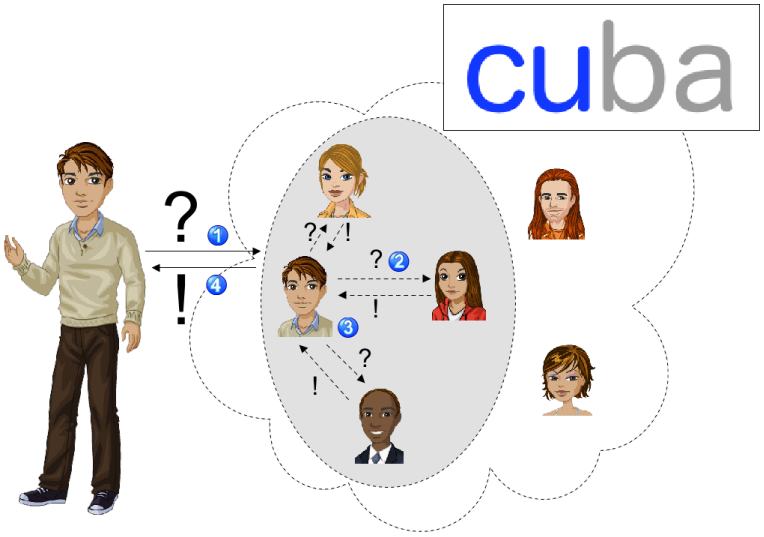

Figure 6.2: Graphical Representation of CUBA's Concept

CUBA is based on News-feeds, it allows many actions a user can perform. It is convenient to use these actions to improve the implicit feedback and therefore the NOP. We identify several actions that imply an interest or non-interest in a specific feed. At this point, we only mention a feed's state or the layout for the user's homepage. We describe this topic in more detail in section 7.6.

6.2 Application Architecture

After the main concepts of CUBA are described, we now turn towards CUBA's application architecture. In this section, we describe the most important modules of the CUBA prototype.

Note: The following architecture should be seen as an abstract model. The implementation is more complex then the described architecture, since we used the Model-View-Controller (MVC) concept for the implementation. For example: the *Input-Engine* consists of more then one class. The application is partitioned into several parts (like a public and private part). These parts usually have their own logic and therefore they are implemented in separate classes. A deeper discussion is given in section 7.6.

Figure 6.3 visualises CUBA's architecture that we describe in more detail in the following.

We begin with the *Input-Engine*. This module takes requests from the user, interprets them and decides how to respond to them. It contains the general application logic and controls the rest of the application, by trigger subroutines, further computations or just forwards to another page.

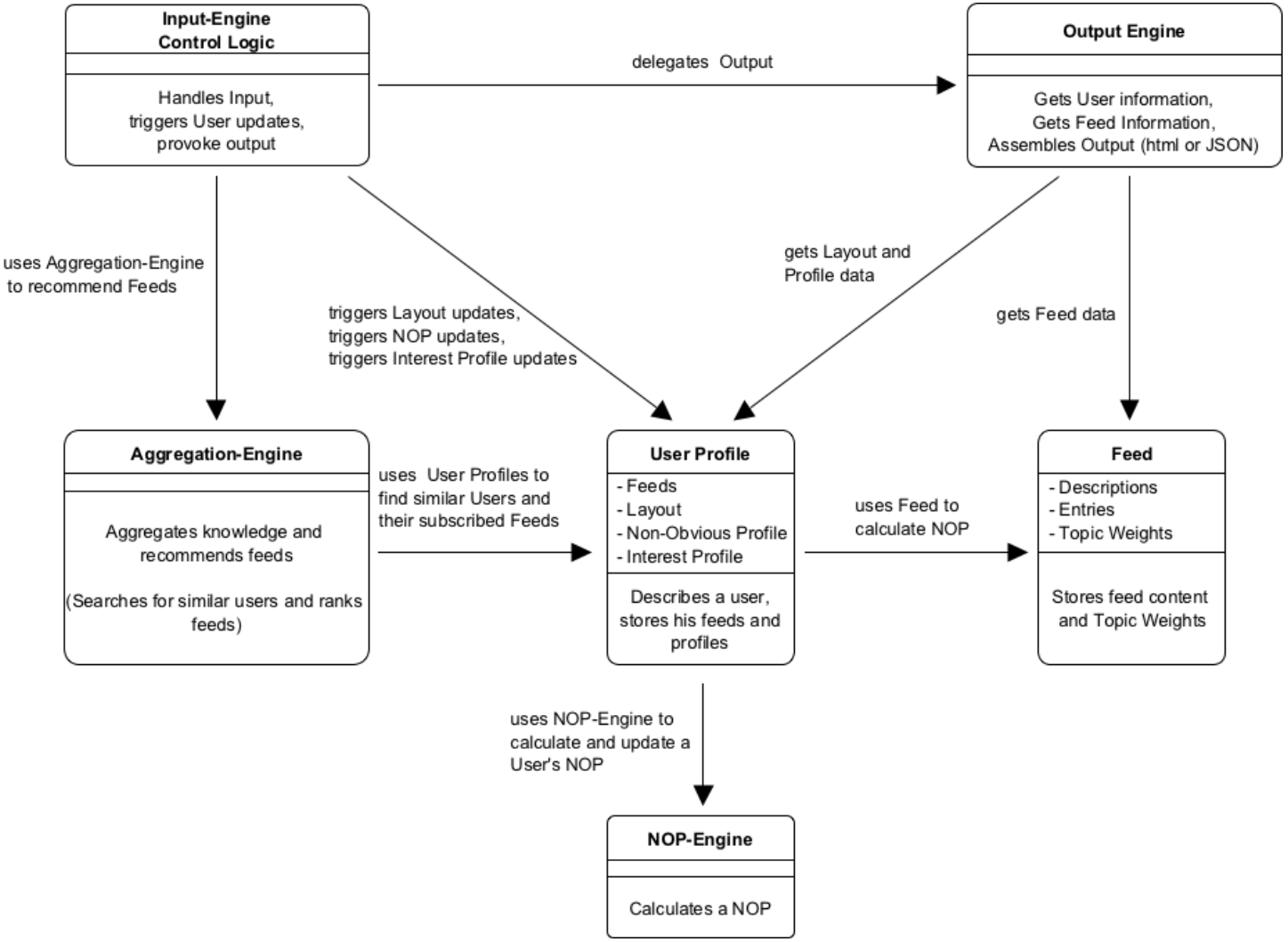

Figure 6.3: Visualisation of CUBA's Application Architecture

In case the user's request requires a response, the *Input-Engine* delegates the output to the **Output-Engine**. The *Output-Engine* prepares the output into the right format and gets, if necessary additional information from the *User Profiles-* or the *Feed* module. Afterwards it sends its output back to the browser, where the user will see the results.

Since CUBA implements a News-feed reader, the *Feed* module handles the news-feeds. It takes care to provide the output with the feed's current content and is responsible to transmit a feed's topic weights to the *NOP-Engine* module.

The *User Profile* module is responsible for all operations that belong to a user. It handles information like the current layout, which feeds the user has subscribed or the user's action history. It also includes the NOP and Interest Profile.

If there is a need to update a user's NOP, the *User Profile* module triggers the *NOP-Engine*, which computes a new NOP.

Finally, there is the *Aggregation-Engine* module. It is triggered from the *Input-Engine* in case a user searches for a new feed. It uses the *User Profile* module to get information about the users. The module aggregates this information and prepares the result, which then is send back to the user.

Chapter 7

CUBA: Implementation Aspects

In this chapter, we present some implementation details of the CUBA prototype that is available at `http://cuba.lu`. We start with the technical details that include the feed fetching and management issues. In addition, a part deals with the database and data structure topics and other important insights about the prototype application itself.

In the second part, we introduce the CUBA prototype from the user's perspective. We show the appearance of the Web site explains the functionality and discuss the chosen design decisions.

7.1 Newsfeeds Formats and Publishing Protocols

Since CUBA's primary focus (for the user) lies in the ability to present the contents of News feeds, one important point is to collect and store them in the system. In the following, we present the basic concepts of News feeds and how we deal with them in CUBA.

7.1.1 The History of Newsfeeds

To understand the current feed formats it might be helpful to begin with the history of News feeds. It is very interesting and explains why we have today three different feed formats.

RSS-Feeds

The history of News-feeds starts in 1999 when Netscape released RSS 0.90 [57]. It is an application of Resource Description Framework (RDF) and means RDF Site Summary (RSS). Netscape invented and used it to provide a personal portal for their users where they could read News. Soon after the release, the format was adopted and supported by many Web sites, which was later updated to RSS 0.91 [58].
In the background, the community differed about the direction of future releases of RDF. There was a controversy about the use of RDF. While a part of the community wanted

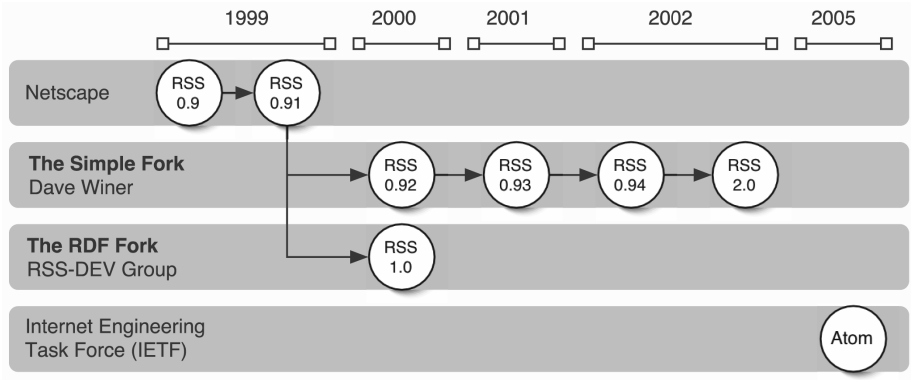

Figure 7.1: Releases of RSS and Atom specifications and their relations[56]

a better link to RDF, another part argued to simplify RSS and abandon RDF out of the specification. In June 2000, Winer released RSS 0.91 [59], which is a simpler version of Netscape's RSS 0.91 and gets on without RDF. While the abbreviation is still RSS, it now stands for Really Simple Syndication (RSS).

In the same year the RSS-DEV group [60] published their specification of RSS, an extension of Netscape's RSS 0.91, and gave it the name 'RSS 1.0' [61]. The RSS format was split up into two incompatible branches.

Winer continued to release successors of his RSS 0.9 specification over the next years. In 2002, he released the final specification, which he named RSS 2.0 and a year later RSS 2.0.1 [62], which is the current version.
Overall, today we are in an odd position as respects RSS feeds. We have two "valid" specifications (RSS 1.0 and RSS 2.0), which share the same name, but have nothing else in common.

Atom-Feeds

After the confusion about the different RSS versions, a group of bloggers created a new standard in 2003 and give it later the name Atom. They began from the scratch and with the aim to create a protocol that improves the different RSS versions without taking care of compatibility issues. In 2004, they joined the Internet Engineering Task Force (IETF) and created the Atom Syndication Format [63] that has been released in 2005. The Atom Publishing Protocol [64] has been released in 2007. Today it is one of the de-facto standards for News-feeds beside RSS.

7.1.2 Technical Specifications

In the previous section, we have given an overview of the history of News-feeds. In the following, we introduce the specifications of today's three most important feed formats:

RSS 1.0, RSS 2.0 and Atom. The introduction is only rudimentary and its purpose is to get a general understanding of the different feeds. For the specifications, we refer the official sources.

General issues

Technically, a News feed is nothing more than a file in eXtensible Markup Language (XML) [65] format. Beside the properties that must be fulfilled by a valid XML-file, there are some rules that are specific for each type of feed. A feed is logically partitioned into two parts. One part contains meta-data about the feed and describes the feed by itself. Usually we will find there its name, a description of the feed, a link to the publisher's homepage, or the release date of the feed. The second part contains the News and information about them. For each News, common attributes are the *headline*, *description*, a *link* to the full article or the *release date*.

RSS 1.0

An RSS 1.0 file is unique in sense that it is based on a RDF definition (RSS 2.0 and Atom 1.0 are ordinary XML files). Another thing is that it has a small number of attributes to describe a feed and its item(s), comparing to the other two approaches. The only mandatory fields are the information about the channel and the *title*, *link*, *description* fields. For the items, these are also the item's title and link. Another thing is that it provides no separated area for the items by themselves. They are "equal" to the description part of the feed. The most commonly used elements in RSS feeds (regardless of version) are *title*, *link*, *description*, *modified date*, and *entry ID*. The modified date is provided by the *pubDate* element, while the *entry ID* comes from the *guid* element. See the appendix for an RSS 1.0 example.

RSS 2.0

As mentioned in the previous section, there are differences between RSS 1.0 and RSS 2.0. The first difference regards to the kind of document, a feed is. While an RSS 1.0 feed is an RDF document, an RSS 2.0 is a simple XML file. A channel represents a feed, and according to the specification, it is possible to put more than one channel into a single file. In the first level of a channel, we find describing elements like *title*, *description*, *language* or *image*. There is also an embedded list of items, which have attributes like *title*, *link*, *pubDate* and *author*. In comparison with RSS 1.0, RSS 2.0 provides a broader set of attributes to describe a channel and an item. Figure 7.2 shows the skeleton of a valid RSS 2.0 feed. A valid feed requires at least a title, a description and a link specification. Items are organized under an item directory. An item requires a title or a description. An example of a RSS 2.0 feed can be found in the appendix.

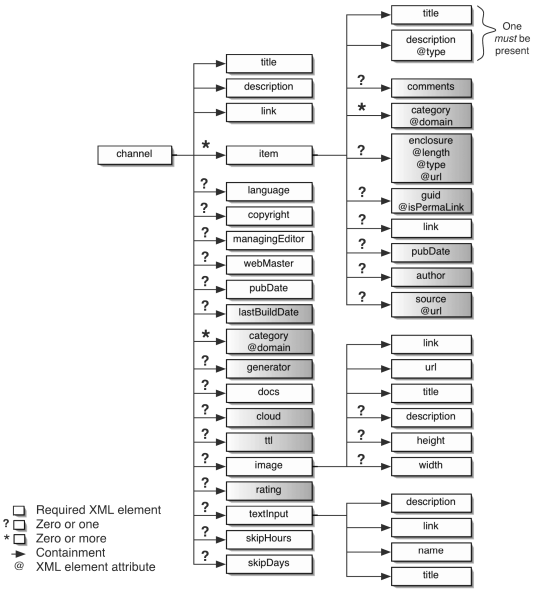

Figure 7.2: Schema of a RSS 2.0 XML-file [56]

Atom

Atom feeds generally contain more information than RSS feeds, because more elements are required. As we can see in Figure 7.3 the specification is a mixture of the RSS 1.0 and RSS 2.0 specifications. The basic structure is equal to RSS 1.0; the entries (items) are placed at the same level as the information about the feed. The level of detail (number of attributes) to describe a feed or an entry is more like in RSS 2.0. Mandatory fields are the feeds *title*, the *updated date* and an *id*. For the items, these are also the *item's title* and *updated date*. An example of an Atom-Feed file can be found in the appendix.

7.1.3 Handling Feeds in CUBA

The fact, that there exist several different formats for News-feeds, made it very difficult to develop a reasonable parser by ourselves. When we started the project, there were only a limited number of reasonable open source parsers available. These were the Apache Commons Feed parser [66], the Universal Feed parser [67] and ROME [68]. While Apache's feed parser stopped their development, at the time we started, the Universal Feed parser is written in Python, while ROME is written in Java. They both share the same kind of functionality to satisfy our purposes. Finally, we decided in favour of ROME, because of some subjective reasons (It supports a modular design and we already have had a basic programming experience in Java).

In short: "ROME is a set of open source Java tools for parsing, generating and publishing RSS and Atom feeds." [68] It covers all of the popular feed formats, while the

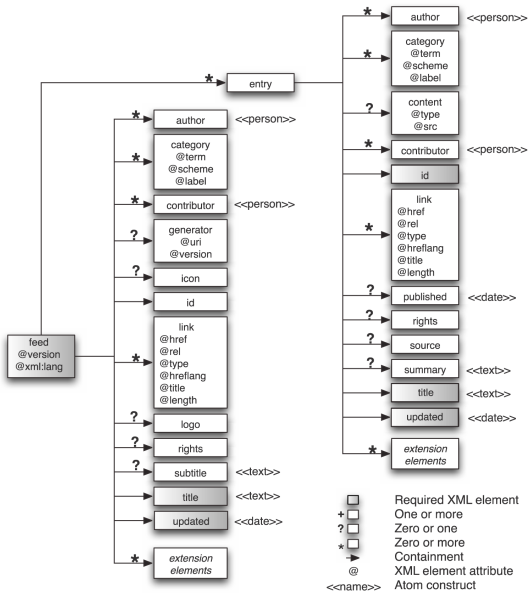

Figure 7.3: Schema of an ATOM 1.0 XML-file (taken from [56])

core ROME library uses the JDOM XML parser only. It is highly flexible and applicable on different platforms. On the other hand, its parser is DOM-based and not liberal, which sometime leads to problems. During our development, the ROME team released version 1.0. Further information can be found on the project's homepage [68].

Using ROME has some advantages. One of them is that third parties are able to develop additional modules. One of these projects it the ROME-Fetcher project [69]. We employ this module to catch feeds. Afterwards the feed is parsed by ROME. If applicable, the feeds information is updated and new entries are stored into the database.

To deal with feeds we developed a separate program that retrieves a feed, parses it and store the contents in a database. Each feed is checked three times per hour (every 20 minutes). By September 9th, 2010 the database contains 586 feeds with approximately 400000 News entries.

7.1.4 Problems in Practice

Even with support of ROME, we have had to struggle some problems regarding the News-feeds during the project. While dealing with feeds, it seems there are two major problems that occur several times:

- Validity of feeds
  Sometimes it occurs, that a feed moves to another location, is redirected temporarily to another location or is just suspended. The first case is no problem in reality.

However, sometimes the feeds are just redirected to an "ordinary" HTML-page. In this case it can happen that the feed parser has some problems to detect the right file-type. In the worst case, this can influence the output of CUBA during the recommendation process.

- Different understandings of feed attributes / bad mapping of attributes
  An important information is the News publishing time. Be it because of a different understanding of the feed's definitions or, because of a bad mapping from the ROME-feed reader, it seems that the publishers use the *published* and *update* fields interchangeably. This makes it on our side nearly impossible to decide, which entry is a new one and which is an updated one. As a result, some feeds cannot be right displayed, which is not good for the user.

These two topics cause a lot of trouble during the experiments and it was really time consuming to fix them (a thing that still needs some adjustment). It figured out during the tests, that the users' willing to use the service decreased immediately, in case some technical problems occurred.

7.2 The Model-View-Controller Design Pattern

In this section, we want to introduce the MVC Design Pattern and discuss its characteristics. Nevertheless, before we start doing this, we examine what a Design Pattern is.

7.2.1 Design Pattern

Patterns occur anywhere and any time. That is the core of a pattern. Patterns are also not bound to software (design). Alexander [70] says:

> "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice". [70]

Larman [71] describes a pattern in general as a structured way to find solutions for problems, based on principles and idioms. In object-oriented (OO) design, a pattern is a named description of a problem and solution that can be applied in new contexts. He defines it as:

> "... a good pattern is a named and well-known problem/solution pair that can be applied in new contexts, with advice on how to apply it in novel situations and discussion of its trade-offs, implementations, variations, ad so forth." [71]

Therefore, in general, Design Patterns are useful. One idea behind the MVC pattern is the so-called "Model-View Separation Principle" [71]

Bruegge et al. [72] describes the MVC architecture as follows:

> "Model/View/Controller (MVC) architectural style [is]
> A three-tier architectural style in which domain knowledge is maintained by model objects, displayed by view objects and manipulated by control objects."
> [72]

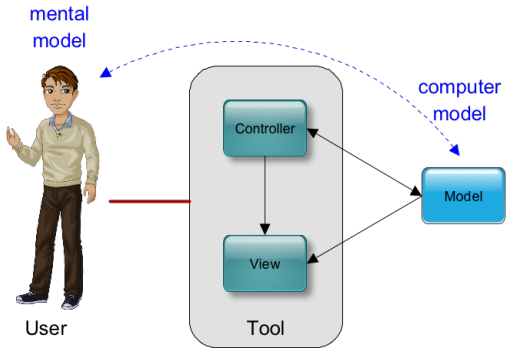

Figure 7.4: Schema of the Model-View-Controller Concept (taken from [73])

The Model-View-Controller pattern allows partitioning an application into logical pieces. The user only interacts with the Controller or a View. The Controller usually gets a request from the user, interprets it and disposes further operations. It usually contains the application logic that controls the behaviour of the application. For further operations, the Controller might use a Model. The Model is responsible to deal with data. It is a kind of data abstraction layer, which could use external data sources (e.g. a database, etc.). It also usually models objects and use them to perform computations if necessary. The View-Module is responsible for the output. It gets the computed data, prepare them and send them back to the user. Figure 7.4 shows how the Model-View-Controller interacts with each other.

7.3 Ajax

Today the term Asynchronous JavaScript and XML (AJAX) is commonly used in relation to Web 2.0 and Web applications. Nearly everybody who travels along the Internet gets in touch with it, probably without knowing that he uses it. We can say that AJAX has changed our expectations to and behaviours in the Internet. So, what is AJAX?

The term AJAX was first introduced by Garrett [74] in an article of his blog. The time Garrett published his article, all involved techniques already existed.

AJAX is a concept that combines the best of several techniques and creates new possibilities. So, what is new on this approach, if each technique has already existed

before? It is the combination, it is seeing the big picture, where every technique has its place and is used, what it can do the best. In the following, we give a short overview, what each techniques is responsible for.

XHTML and CSS

Extensible HyperText Markup Language (XHTML) and Cascading Style Sheet (CSS) are responsible for displaying the content inside a web browser. While XHTML is used to print the content, CSS is responsible for the Style of a page e.g. font, size, colour, etc.

The Document Object Model (DOM)

Every web browser represents a page in an internal Document Object Model (DOM) structure. If the browser gets an HTML- and CSS-file, it parses it and translates it into a DOM. This also works in the other direction: a change in the DOM results in a change of the browser output.

JavaScript

JavaScript, or ECMA-Script, is an object-oriented scripting language designed by Eich in 1995 and Developed by Netscape Communications Corporation and the Mozilla Foundation. With the support of JavaScript, it is possible to manipulate the DOM systematically, which could be used to make a Web site more dynamic.

XMLHttpRequest

All modern browsers provide an XMLHttpRequest object. This makes it possible to communicate with the server without doing a page request explicitly. Given this opportunity it is possible to request data from the server.

XML, JSON and XSLT

Usually the browser gets this data in an XML [65] or JavaScript Object Notation (JSON) [75] format. A common approach is to let JavaScript get and handle the data and manipulate the DOM. To transform XML documents in another format, eXtensible Stylesheet Language Transformations (XSLT) [76] is commonly used.

7.3.1 The Ajax Web Application Model

The Ajax web application model follows and extents the classical web application model. In contradiction to the classical approach, it is necessary that an Ajax-Engine in form of one or more JavaScript scripts be uploaded to the client. A typical scenario is given as follows:

1. The browser client performs a HTTP-request to the server.
2. The web server consults the back-end (databases, etc.).
3. The web and or XML-server return XML data.
4. The Ajax engine interprets the XML data, transforms it into HTML and CSS data and manipulates the DOM.
5. Finally, the browser client displays the web page.

In contradiction to the classical model, where the web server returns HTML and CSS Data, the web server returns XML data that is interpreted by the Ajax engine. Figure 7.5 shows the functionality of the Ajax Web Application Model in a graphical way.

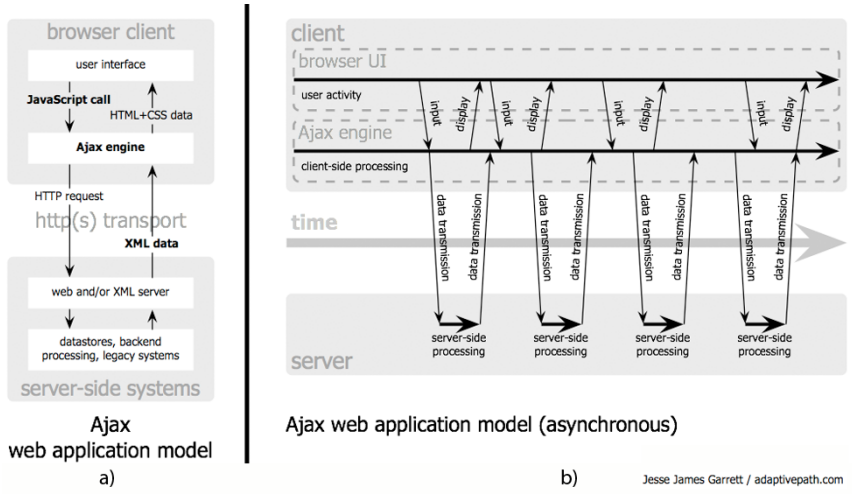

Figure 7.5: The Ajax Web-Application Model [74]

What are the consequences for the user? In the classical Web Application Model, the user has to wait after each request until the server responds and the browser displays the new content. In the Ajax Web Application Model, the user sends a request to the server and continues with his work. When the Ajax engine gets the result, it manipulates the DOM and the browser displays the new content (Figure 7.5).

The advantage of this approach is that the response time usually shrinks dramatically. Instead of loading a completely new web page, it is now possible to download only the part, which is needed. The Ajax engine then manipulates the current Web page and the user will see the result without having to wait until a new page is downloaded and created. This also allows the effect of an immediate reaction from the application. Another usual technique is to download the most important data first and the rest in the background while the user is using the page.

However, there are also disadvantages. The most important thing when we talk about Ajax is the existence of a JavaScript Engine. While these engines are part of all web browsers, the user has the possibility to switch-off the JavaScript Engine. In this case, an Ajax application will not work. Therefore, a Web site that uses the Ajax technology should always make sure to provide its service, even with a switched off JavaScript Engine at the client side. In the worst case, this requires to create a second application that is running with HTML only.
Another disadvantage is the working of the browser's "back"-button. This button reloads the last page in history. In case an Ajax Application uses only one page that is modified the whole time, this can end up in unwanted results.

These were only some aspects that become to determine, if a web application is built on the Ajax concept. For further discussions, we refer to Crane et al.[77], Johnson et al. [78] or Mahemoff [79].

Overall, the Ajax approach is very powerful, but it should also be used very carefully. As Garret says:

> "... the more power we have, the more caution we must use in exercising it. We must be careful to use Ajax to enhance the user experience of our applications, not degrade it." [74]

7.4 Database Architecture

In this section, we present CUBA's database architecture. The architecture consists of several parts, where each is responsible for a different purpose. It follows the MVC concept, which makes it not very difficult to understand.

One main part of the architecture is used for the feeds. It is in principle a one to one translation from the objects, which are defined by the ROME feed reader.

In our project, we use a PostgreSQL database server [80] for data storage. The database design and implementation take advantage of PostgreSQL schemas to partition and store data into logical pieces. There are no special tricks or anything else about the used database techniques. For example, relevant fields involved in speed critical operations are indexed, etc.

The database design for storing feeds was originally very close related to the capabilities of ROME, trying to store any available information. While this database still exists, only a small part (the core elements) is used by CUBA. Mainly these are the Feed, Entry and Tag tables. It also contains tables to store the feed's assigned Topic/Weights in the database. Figure 7.6 shows the implemented database schema for the feeds part.

The core of CUBA's database architecture is taken for the Non-Obvious Profile storage. When a visitor begins a session, CUBA generates it and stores the session data in the database (i.e. the feeds and the layout). Every time a user performs an action, it is also stored in the database. Usually each action has its own table (i.e. feed refresh, link hit and feed preview). Other actions like open/close a feed or delete feeds were automatically taken into account in the layout. They will be used, if a user wants to add a feed

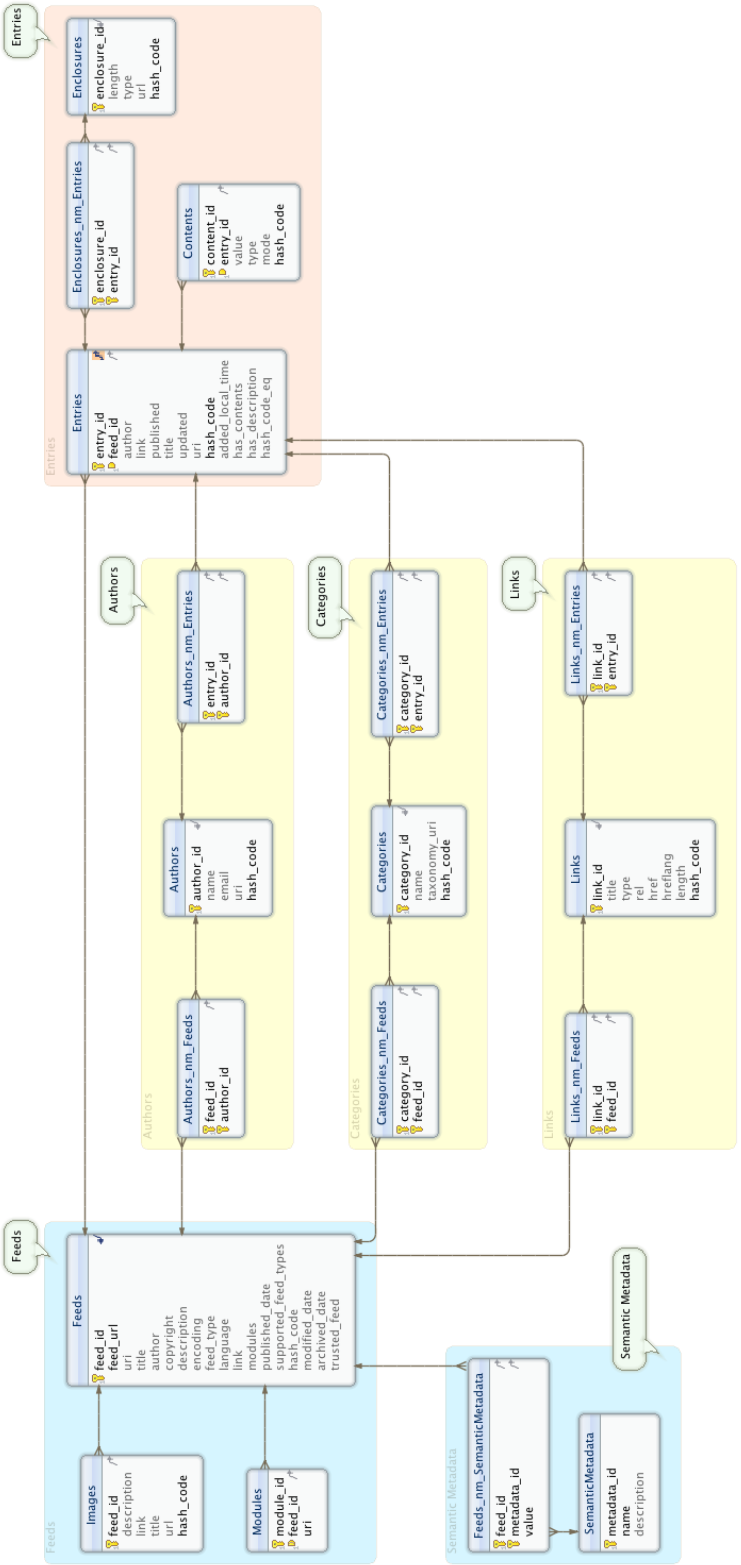

Figure 7.6: Database Design concerning Newsfeeds

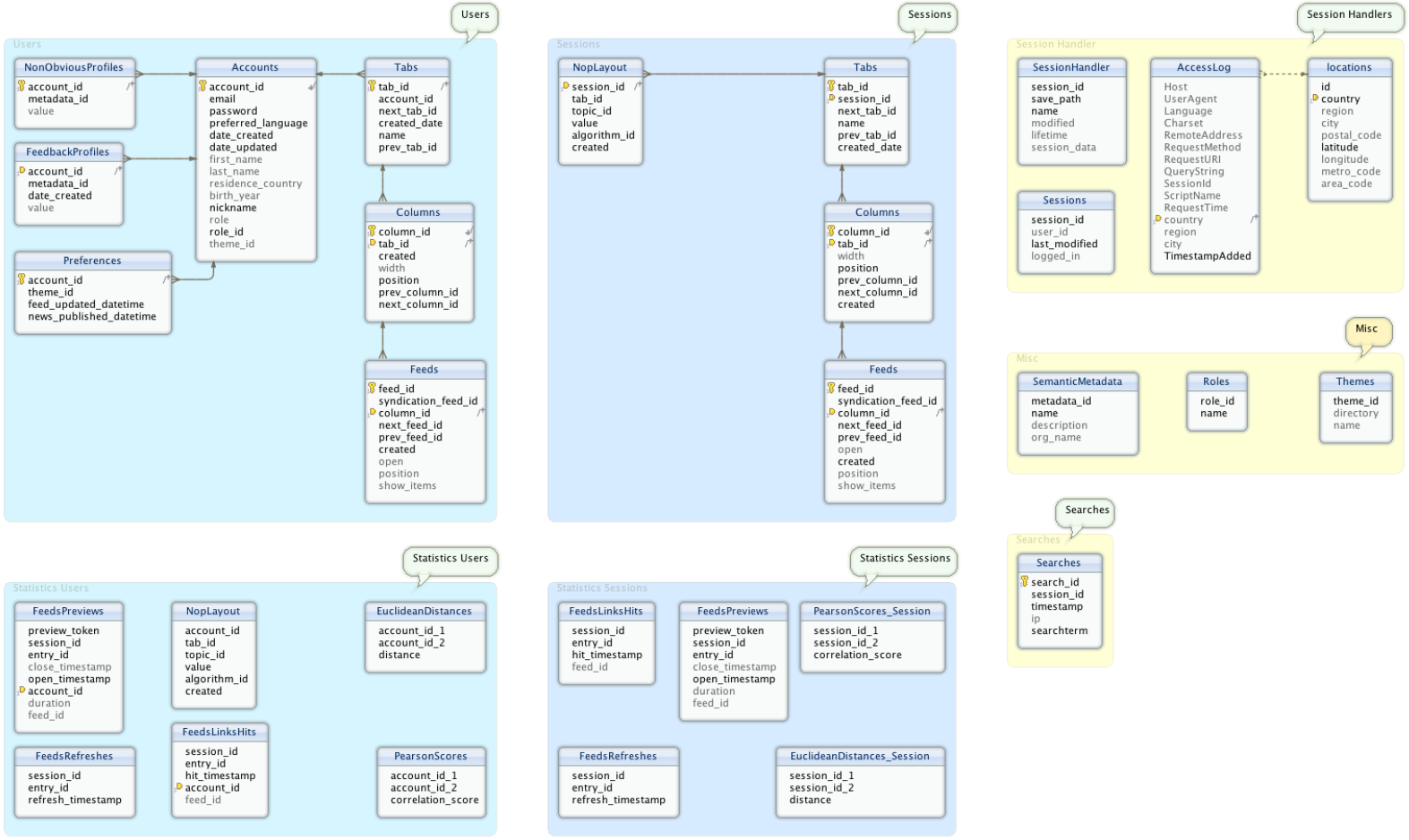

Figure 7.7: Database Design concerning User Data and Non-Obvious Profile storage

to his homepage. He then could enter a name of the feed, a description or key words that describe the desired information: In a first step, CUBA searches for a feed that contains the search terms in its title or description. In a second step, CUBA calculates the user's current NOP. In case there are not enough profiles stored in the system, it uses the feed profiles and ranks the feeds against the requester's profile (the shorter the better). If the database consists enough user profiles, CUBA determines similar users by computing the distances between the user's NOP and all other NOPs. After CUBA has determined similar users, it ranks a feed from the result set higher, if one of these similar users subscribed to this feed.

7.5 Other Technical Aspects

7.5.1 Web Server

As for the Web server, we have chosen the lighttpd Web server [81] over the Apache http Web server [82]. We tested both web servers and decided in favour for the lighttpd Web server. The reasons were that lighttpd is less heavy and uses fewer resources than the Apache Web server. These are important features, because the whole project is running on one server only (retrieving and processing feeds, database server, and web server). For the project's script language that is used by the output module PHP version 5.2 was chosen.

7.5.2 Application issues

This is the part where we use the database to produce the content for our CUBA Web application.

As we came along the decision to implement the back end engine for CUBA , we had to decide to build it from scratch by ourselves or with support of a framework. Building it by ourselves gives us the freedom in design, but on the other hand it is very time consuming and error-prone. Because of this we decided to use a framework. We examined and compared different frameworks and decided in favour of ZendFramework [83]. The reasons were the following:

- ZendFramework supports the MVC architecture paradigm
  This forces the programmer to use a clear design for the application. After a short introduction time this saves a lot of time.

- It uses a modularized architecture.

- It allows using some modules independently.

- It provides a powerful database abstraction layer.
  This makes it simple to retrieve data from the database.

- AJAX is supported through JSON.
  This is a very important point, because one of the key features of CUBA is based on AJAX and data transmission via JSON.

- It provides forms and validators. This feature is used in the application to make it more robust against faulty insertions.

- the framework is expanded continuously.
  This was very helpful during the project. It appears two times that new features were introduced when there were needs of them.

- There is an excellent support.
  This is a very important point. Usually framework errors are fixed quickly.

- The existence of many useful basic modules.
  There exist useful basic modules that allow the user to concentrate on the important tasks. For example, there are existing modules for authentication or database operations, which usually are used in a controller or module.

Originally, one strong advantage that influenced the decision among ZendFramework concerned the support of consuming News-feeds through the `Zend_Feed` module. Unfortunately, it becomes clearer through the developing process, that it was very slow and embryonic. Thus, we had to replace the feed reading and parsing process to another, independent solution that leads us to ROME. [1]

7.5.3 JavaScript Frameworks

One main feature of CUBA is to enable the user to personalize the home page to some degree. This task includes modifying the page's layout by dragging feeds around the page, which is impossible by using HTML only. For this reason we implement some parts in JavaScript. This part is handled by the JavaScript Engine, which we mentioned earlier in section 7.3. This has some advantages; the user is enabled to do more things. However there is also one serious disadvantage: If the user turns JavaScript off, the site must still be able to work. Like in the case for the application (and because of the same reasons), we also got some help from a framework. In this case, our decision felt on jQuery [84], which is a lightweight JavaScript Framework. According to buildwith.com [85], it is the most used Framework in the World Wide Web (29.2%). The framework itself is split into two parts, jQuery Core [84], which allows us to perform DOM and JSON operations, and jQuery UI [86], which provides the developer with some pre-defined user interface operations like drag and drop, etc. As we will see later, there are some actions in CUBA, which take heavily advantage of jQuery such as "Drag and Drop" or modify the layout of News-feed, etc.
CUBA's underlying JavaScript engine was totally built using jQuery to perform all front end and DOM modifications. There is also an additional JavaScript library used in the

[1] With ZendFramework 1.9.0 a revised and extended version of this module was released

project, which is Raphaël [87]. This is a library for creating graphics on web sites via JavaScript. In CUBA Raphaël is used to display a graphical interpretation of a user's interest profile (see next section).

7.5.4 Privacy and Cookies

Let us share some words about privacy. During the project, we keep and protect the users' data and privacy. We put only one cookie at the user's computer, containing his session identification.

Another issue concerns the handling of passwords. CUBA encrypts a password immediately and stores it into the database. There is no possibility to get a password, even for the administrator. Furthermore, we collected only data that were necessary for the project. As a side effect, this also kept the amount of collected data small. No user data was used in other terms then the one defined in the project.

7.6 The CUBA Frontend

The following section introduces the CUBA Web site from the visitor's perspective.

Figure 7.8: Homepage of the CUBA Web Site

7.6.1 CUBA in practice

One of the primary concepts of CUBA is to allow the visitor to design a web page for his own needs. Because we are dealing with News, which means with texts and (some) pictures, we decided to base on the grid-system approach that is described by Müller-Brockmann [88] and Ambrose et al. [89]. This approach is widely used in the print media sector. The idea is to arrange the content in a grid system. The result will be a clean layout that appears calm and allows reading the context comfortably. Figure 7.9 visualises the grid-system principle.

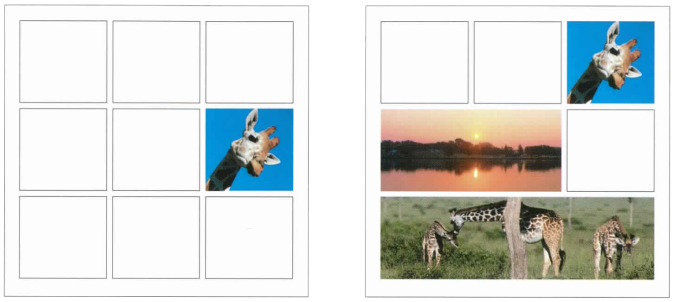

Figure 7.9: Two examples of a grid-layout system [90]

The working area in CUBA is divided into three columns. Each feed is displayed in its own box. The user is able to arrange the subscribed feeds (i.e. boxes) in one of the three columns, and arrange them within the columns, too. Overall, it has a little bit the appearance of a newspaper. This is a coincidence only (dealing with News). There are no restrictions for other uses.

While the intention is clear, in practice there occurs a problem with the display of the feeds. While it is no problem to set the column's margins, it is quite a problem to keep the grid-system intact. The reason therefore is the way browsers threat HTML. CUBA uses a proportional character set to display feeds. Since it allows only a fixed number of messages to be displayed inside a feed's box, the layout depends heavily on: the number of messages, the length of the messages, the font and the font size. Alternatively, it might be possible to support vertical scrollbars, in case a feed will not fit in a box. Nevertheless, in worst case, every feed has a scrollbar, which results in a "bumpy" look.

Use Layout Information to improve Implicit Feedback

As mentioned above, the user is able to move feeds (content) inside the area to create a suitable layout. We can use the layout information to get additional implicit feedback concerning the users' interests.

A user will add feeds that he is interested in and remove feeds, he is not interested in. To continue this thought, we assume that most users also show a tendency to rearrange the layout to satisfy their needs. Therefore, we assume that they will move these feeds on

top of their page, where they are most interested in. These feeds are the first ones they will see (and probably read), in case they request their page. Based on this assumption, we expand the original MIN-Rule, MAX-Rule and AVG-Rule from section 4.3.1. We introduce new strategies that enable us to estimate the users interests based on the current layout of the News feed page. These strategies are visualised Figure 7.14 and described in the following:

- *Dovetailing*: this strategy follows the psychological assumption that the more a user is interested in content the higher the assigned value will be.

- *Coating*: here we argue that the left-most/top-most content receives the highest weight again, but that – in contrast to *Dovetailing* – each following inverse coat, which is identified by the diagonal, is assigned to the same weight. We therefore assume that content with the same distance to the top content (left-most, top-most) should have the same weight.

- *Waving*: with this strategy, we perform a weighting following the radius around the top content.

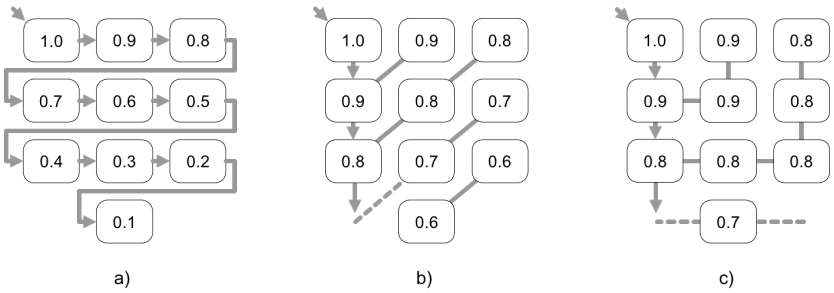

Figure 7.10: Example of possible value assignment strategies for a layout with 3 rows (a) *dovetailing*, b) *coating*, and c) *waving*). The boxes represent content (here: feeds) with their (relative) importance for the user. Arrows represents the way of calculation for the importance.

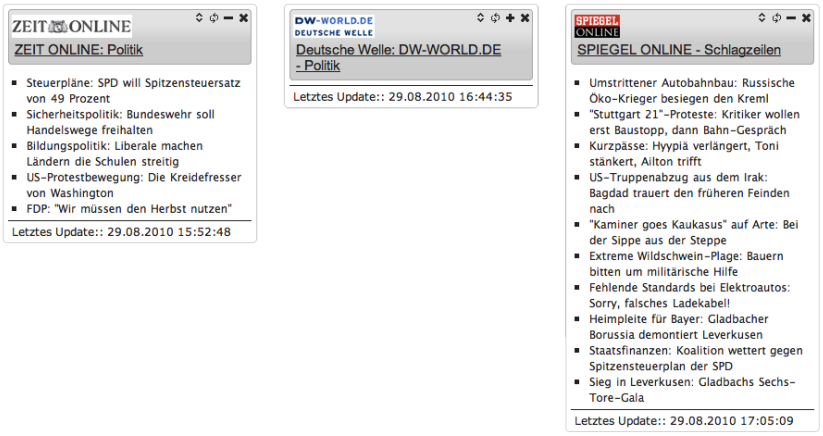

Figure 7.11: Typical states of News-feeds in CUBA . a) is the initial state, where five headlines are displayed. b) represents a "closed" News-feed, which was done by using the Toggle-Action. c) shows an "expanded" feed, displaying ten headlines. This was reached by the Expand/Reduce-Action.

7.7 Operations and Actions in CUBA

Beside the possibility to re-arrange News feeds, CUBA offer also other feed related actions, which we describe in the following. Figure 7.12 points out CUBA's most important feed actions.

Expand/Reduce feed

This action allows the user to watch more headlines at a time. Initially CUBA displayed the latest five headlines only. Press this button forces CUBA to show the latest ten headlines. Pressing it again forces it to show fifteen headlines. Another pushes forces CUBA to show the latest five headlines again. An example is given in Figure 7.11.

Toggle feed

This action is related to the Expand/Reduce-Action. It is a "binary" action that allows the user to close or open a feed. Historically, this action was introduced before the Expand/Reduce-Action. There is also an example given in Figure 7.11.

Refresh feed

Pressing this button forces CUBA to retrieve the contents of the feed and show the latest News. This action affects this feed only. On the one hand, it reduces traffic between the client and the server and responds faster than reloading the whole page. On the other

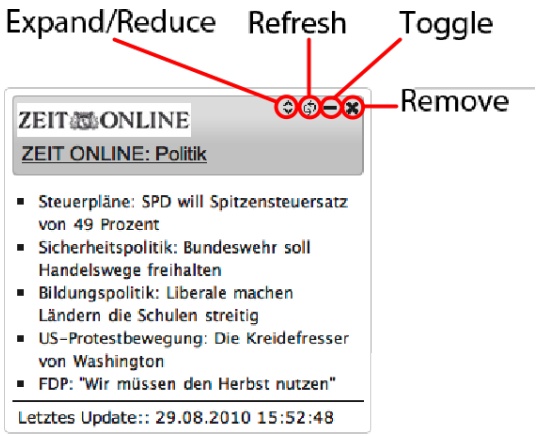

Figure 7.12: Available News Feed Actions in Cuba. From Left to Right: Expand/Reduce a Feed, Refresh of a feed, Toggle of a feed and Remove a feed

hand, we use this information as part of the process to calculate the user's NOP. Forcing the system to refresh only one feed indicates, in our opinion, a high interest in the News feed's assigned topics.

Remove the feed from subscription

This action removes the feed from the web site and from the user's subscription list. It is an indicator, that the user is not interested in the News feed any more.

Figure 7.13: Preview of a Newsfeed Article

Preview of the Article

If the user moves the pointer over a headline, the article's summary appeared to be read (Figure 7.13) This action occurs only if a summary exists. There may be feeds, which do not deliver these summaries. In this case, nothing will happen. By now, there is no visual sign that indicates the existence of a summary. One possibility would be to use

different kind of items in the list. (e.g. SQUARE if there is a preview and otherwise a BULLET).

Open an article

A click on a headline opens the corresponding article in a new browser tab or in a new window.

Other Pages

Beside of the main page there are other pages, which we introduce next.

A real import page is the "add fee"-page, where the user has the opportunity to subscribe to new feeds. The mask follows the commonly known behaviour of other search systems. At the top centre, there is a field where the user can enter search terms. The system then presents a result in form of a list. It contains all feeds that are known by the system and match to the search terms. The list is divided into "pages"; each of it contains five entries. Below the list, there is an opportunity to navigate in the list. This approach has been chosen for readability reasons. We got some feedback that a list, presenting all recommendations could become to long and therefore difficult to read. The user is invited to pick up as many recommended feeds, as he wants, which will then appear on his personal main page. Figure 7.14 shows an example of the subscription page.

Figure 7.14: Example of a News-feed search in CUBA

The statistic Part

Next, we present another important part of CUBA . It is about informing the user of his profile, calculated and stored in the system. There are three reasons we implemented this:

1. We want to let the user know, what CUBA computes.
2. We want to let the user know how CUBA works.
3. We want to sensitise the user.

Business u. Finanzen	schwach
Entertainment u. Kultur	schwach
Environment	schwach
Gesundheit u. Medizin	schwach
Erholung	keine
Gesetz u. Verbrechen	schwach
Andere	schwach
Politik	hoch
Sport	schwach
Technologie u. Internet	schwach
Wetter	schwach

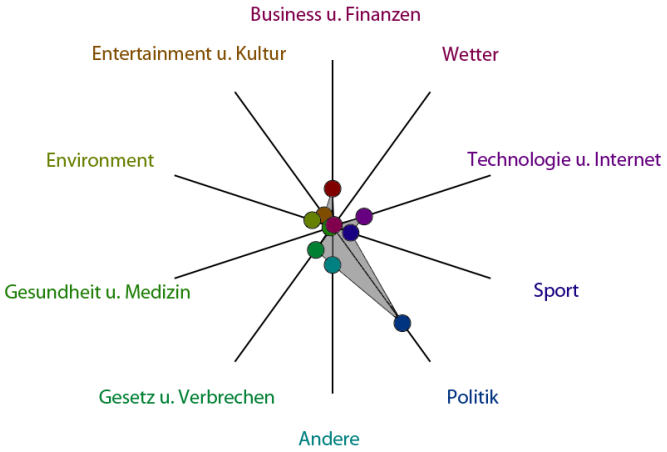

Figure 7.15: Representation of an interest profile, which describes the level of interest in a topic, derived by the subscribed feeds and the current layout

There are different pages, which are available via tabulators. They have the same layout but show different kind of profiles (Figure 7.15). These are:

- the “page profile”[2]
  this is the user’s current interest profile, derived by the subscribed feeds and the current layout.
- the “session profile”[3]
  this describes the interest profile, computed by the information given by the current session. It involves all page layouts, all actions performed and all requested previews and articles in this session. We can consider it as the Non Obvious Profile of the session, if it would ends now.
- the “clicks profile”[4]
  this shows the user’s interest profile, based on the requested articles within the session. At the bottom there is also a note that tells the user how many articles requests he did in the session.

[2] Mein Seitenprofil [3] Mein Sessionprofil [4] Meine Session Klicks

- the "preview profile"[5]
  this shows the user's interest profile, based on the called previews during the session. At the bottom there is also a note that tells the user how many previews she requested.

CUBA's interest profiles are based on the following topics:
Business & Finance[6], *Environment* [7], *Entertainment & Culture* [8], *Health, Medical & Pharma* [9], *Hospitality & Recreation* [10], *Law & Crime* [11], *Politics* [12], *Sport* [13], *Technology & Internet* [14], *Weather* [15], *Other* [16]

These items are based on the *NewsCodes* used by OpenCalais [91], a service that tags and weighting text. This in turn is based on the News Codes, defined by the International Press Telecommunications Council (IPTC) [92] The reasons why we chose them were to a) build on a standard, and b) make it extendable by using e.g. OpenCalais at a later point in the project. The advantage, by using such a service, would be able to get individual tags and weights for each article instead of relaying on an average weight for each feed.

On the upper left corner of each page is a legend that describes the current level of interest in a topic. The values are displayed as text, where the text covers an interval. This interval has the following mapping.

- 0 None[17]
- $0 < value < 0.3$ little[18]
- $0.3 <= value < 0.5$ moderate[19]
- $0.5 <= value < 0.7$ curious[20]
- $0.7 <= value < 0.9$ high[21]
- $0.9 <= value <= 1.0$ addicted[22]

In case JavaScript is being activated, there will be a graphical interpretation of the profile in the centre of the page. This is done by a "star". It is build by a line (with the same length) for each topic, where the interest > 0. The lines join each other in the centre. This is the 0-point for each topic. The outer side of each line represents the value 1. Since all interest values are in the range between 0 and 1, we just mark each line with a coloured dot, at the position that represents the level of interest. At the bottom of each page, the user finds a short description of the displayed profile and the parameters that influences the profile.

In case the user is registered, he has also to opportunity to give an explicit feedback and informs the system about his current interests (Figure 7.16). For each topic he can chose a number of stars that corresponds to his level of interest, where 0 stars means no interest and 5 stars represents an "addiction" in a topic).

[5] Meine Vorschauen [6] Business u. Finanzen [7] Environment [8] Entertainment u. Kultur [9] Gesundheit u. Medizin [10] Erholung [11] Gesetz u. Verbrechen [12] Politik [13] Sport [14] Technologie u. Internet [15] Wetter [16] Andere [17] keine [18] wenig [19] moderat [20] neugierig [21] hoch [22] abhänging

Figure 7.16: Input Mask for User's Interest Profile in CUBA

Chapter 8

Research Results

8.1 Usability of CUBA

In summer 2009, the then current prototype was subject of two studies at the University of Hamburg, concerning CUBA's usability and emotional perceptions.

8.1.1 Usability under the aspect of the Interface Design

One study examined CUBA under the perspective of usability [93]. In this context, CUBA was evaluated using Shneiderman's "Eight Golden Rules of Interface Design" [94]. These rules apply to development and design processes of computer systems and their interfaces. They constitute an important basis for a good human-computer interaction.

In the following, we present each rule separately and present the work's results.

Strive for consistency

> "Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent commands should be employed throughout." [94]

The study lists some of CUBA's behaviours that violated this rule. For example, the meaning of the ⊕ and ⊖ signs for toggling a feed have had the opposite meaning, i.e. ⊕ for close an open feed and ⊖ for open a closed feed. Other points concern the mixture of English and German on the user interface, the inconsistent display of News-feeds' headlines and some difficulties in the searching process.

Enable frequent users to use short cuts

> "As the frequency of use increases, so do the user's desires to reduce the number of interactions and to increase the pace of interaction. Abbreviations, function keys, hidden commands, and macro facilities are very helpful to an expert user." [94]

The study's report neither identified the existence of short cuts, nor found a need for short cuts at this time, because CUBA supplies only one level or services.

Offer informative feedback

> "For every operator action, there should be some system feedback. For frequent and minor actions, the response can be modest, while for infrequent and major actions, the response should be more substantial." [94]

The study detects an inconsistent strategy to offer informative feedback to the user. While CUBA gives feedback under some circumstances, there is also a total lack of feedback in some situations or it occurs in a mixture of English and German.

Design dialogue to yield closure

> " Sequences of actions should be organized into groups with a beginning, middle, and end. The informative feedback at the completion of a group of actions gives the operators the satisfaction of accomplishment, a sense of relief, the signal to drop contingency plans and options from their minds, and an indication that the way is clear to prepare for the next group of actions." [94]

This subject is almost non-existent in CUBA. The report criticise that only one appropriate message, that an action finishes (successfully), was detected. It gives also an example, where a change in the interest profile did not result in a change at the user's personalised homepage. The report states that actions in CUBA have neither intermediate steps, nor a true end with some feedback.

Offer simple error handling

> "As much as possible, design the system so the user cannot make a serious error. If an error is made, the system should be able to detect the error and offer simple, comprehensible mechanisms for handling the error." [94]

Regarding the study, this point does not belong to CUBA, because there are no possibilities to make serious errors. However, it criticises the way CUBA handles deleted feeds and recommends to keep deleted feeds in a separate list that offers the user an "undo" option in case the feed was deleted accidentally.

Permit easy reversal of actions

> "This feature relieves anxiety, since the user knows that errors can be undone; it thus encourages exploration of unfamiliar options. The units of reversibility may be a single action, a data entry, or a complete group of actions." [94]

Here the results refer to the previous point, because in CUBA's case they are quite similar.

Support internal locus of control

> "Experienced operators strongly desire the sense that they are in charge of the system and that the system responds to their actions. Design the system to make users the initiators of actions rather than the responders." [94]

The study states that the user has only few options to adjust parameters that show an immediate result. They criticise the option where the user is able to enter his interests, which results in the same recommendations (e.g. the term "Weather"). Another point is CUBA's insufficient work with different browsers (which is traced back to the different browser-engines)

Reduce short-term memory load

> "The limitation of human information processing in short-term memory requires that displays be kept simple, multiple page displays be consolidated, window-motion frequency be reduced, and sufficient training time be allotted for codes, mnemonics, and sequences of actions." [94]

Here the report says:

> "This program runs little risk of the user to overwhelming his short-term memory. Its learning steps are short, not complex and easy to learn."

Miscellaneous

During their evaluation the group members identified also other points that reduces CUBA's conviviality, but could not classify these to one of Shneiderman's rules. Because of this, the report lists other critical points in a separate section. These points were, e.g. some aesthetic points (chosen colour for visited links) and a careful choice of descriptive terms (unambiguous). Furthermore, they suggested implementing a welcome-page, a tutorial and a help-system.

The report's conclusions

The study identifies some inconsistency inside CUBA as well as some serious errors. It suggests correcting the mistakes before a productive appliance of CUBA could start.

The influence to CUBA

Some of the complained problems where already known and at the time of the presentation already corrected (e.g. completed German translation, fixed buttons, etc.) Others were caused by a different understanding of CUBA's concepts. For example, there is no intention to modify a user's homepage when modifying the interest profile. This should only be happen by adding a feed explicitly. Finally, there are problems that belong to browsers, like different interpretation of CSS commands.

In general we followed most of the advices given in the report, corrected many errors and improved the prototype. On the other side, the implementation of some suggestions where neglected, because they were not important in relation to our concept, or there implementation would be too time consuming (e.g. implementation of a help-system, a tutorial or a Frequently Asked Question-system).

8.1.2 The CUBA prototype under emotional perceptions

The second study focus on the user's perception of CUBA and which emotions are released from CUBA [95]. The goal of the study remains to the question, how products like CUBA and Google news are perceived by a user.

Experimental setup

The study compares CUBA and "Google news" [96], where the "Joy of Use" is in the centre of attention. Therefore, it takes advantage of the AttrakDiff™ system [97]. The AttrakDiff system contains 28 pairs of adjectives, where each pair describes a contradiction (e.g. simple - complicated). A subject has to decide which attribute describes the product at its best on a seven level scale (Figure 8.1).

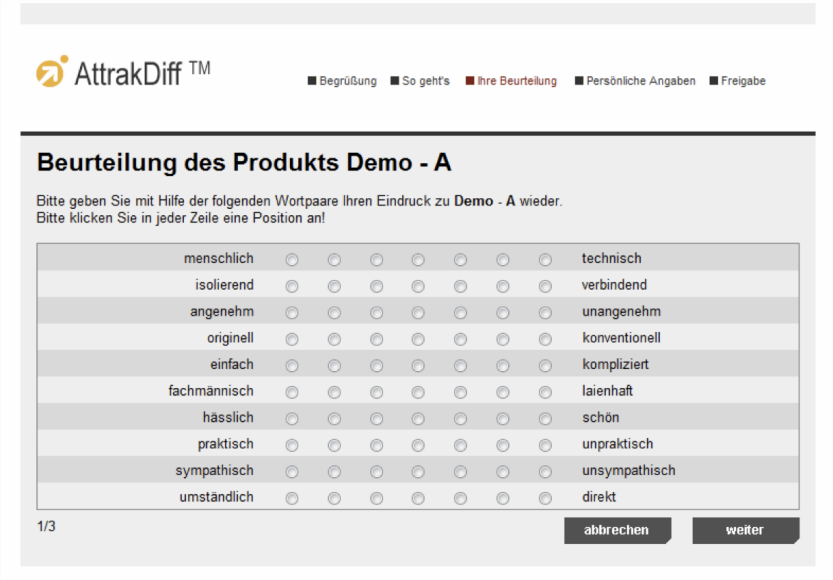

Figure 8.1: Screenshot of AttrakDiff's Questionnaire [95]

The 28 pairs of adjective are partitioned into four groups, where each group includes seven pairs. These groups are introduced in the following:

1. **Pragmatic Quality (PQ)**
   This group describes the usability of the product and how good it supports the user to reach his goals.
   Typical pairs are “complicated - simple” or “confusing - clear”

2. **Hedonistic Quality Stimulation (HQ-S)**
   This group is based on the assumption that an individual usually tends to develop himself. It measures how good this aim is reached by evaluating, if the product includes something new or interesting. Typical entries are “conservative - innovative” and “lame - excited”

3. **Hedonistic Quality Identity (HQ-I)**
   This tries to describe how far the user is able to identify himself with the product. Entries include “unprofessional - professional” and “bad style - good style”

4. **Attractiveness (ATT)**
   This group tries to figure out the overall rating of the product, based on the perceived quality. Typical entries amongst others are “ugly - beautiful” and “repugnant - attractive”

The study is based on 22 participants. They were divided into two groups where a group worked with CUBA (12 subjects) and the other with Google news (10 subjects). To become familiar with the Web sites, each group got some identical assignments, which were to solve. Afterwards the participants evaluated the Web sites with AttrakDiff.

Results of the experiment

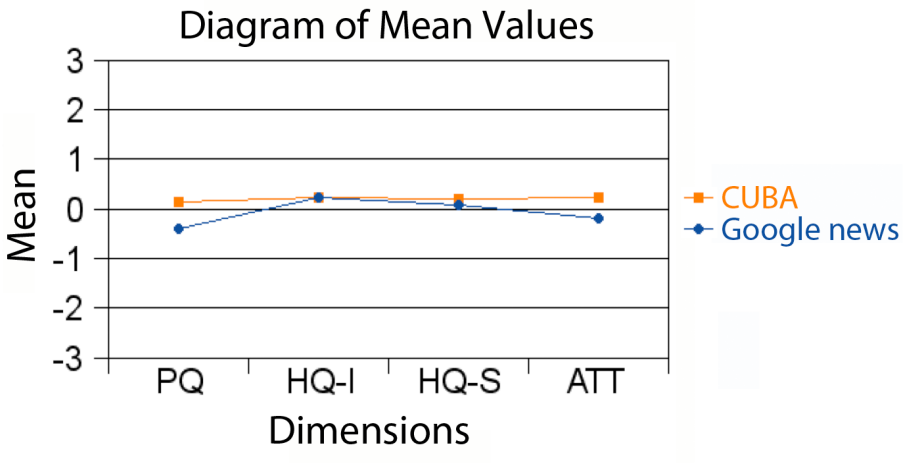

Figure 8.2: The Experiment’s Arithmetic Mean Results [95]

The evaluation of the GUI shows that CUBA and "Google news" are considered similar (Figure 8.2). Both are considered as "neutral". The result attests both Web sites an existing *Pragmatic Quality*. This means the users are already attracted by the interface. Nevertheless, given the results, there is still some space for improvements in respect of the *Hedonistic Quality*.

The study also measures the users perception of CUBA and "Google news" in general. The study reasons, that CUBA get slightly better reviews than "Google news". Indeed the differences are not statistical significant, but CUBA has a smaller confidence interval. That means CUBA's reviews are more similar. Figure 8.3 shows the results for every pair.

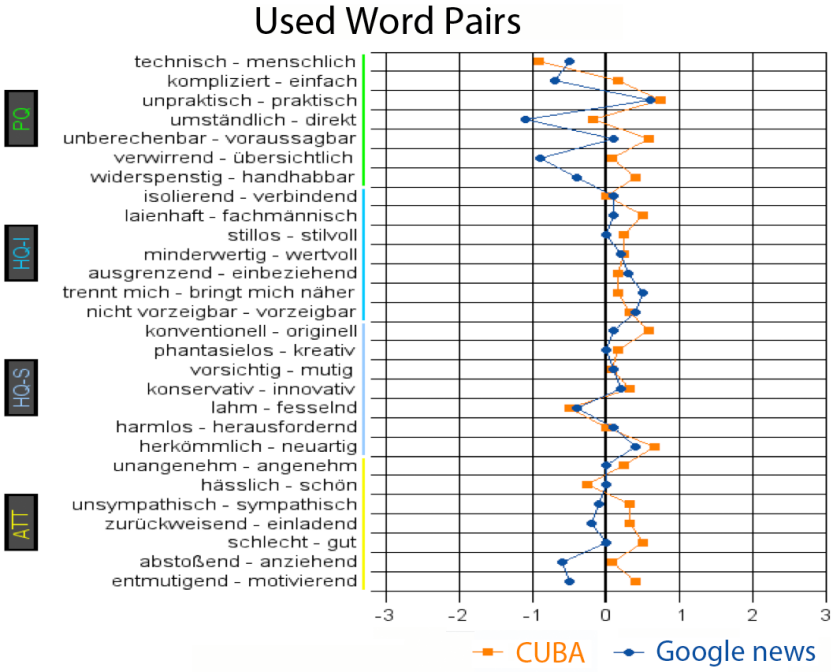

Figure 8.3: Result of AttrakDiff-Experiment [95]

The report concludes also that CUBA is slightly better than "Google news", even if the subjects perceive both only as "neutral". It suggests that both products should improve the *Pragmatic Quality* to tie the users in a better way to the product. The authors also suggest improving the *Hedonistic Quality*, especially *HQ-I*. To improve *HQ-I* they propose to modify the application, to target on usual users instead of experts.

The study also recommends adding new features to CUBA. Namely it mention a separate start page, a "Frequently Asked Question" area, a tutorial, which describes CUBA's behaviours and more opportunities to personalize a page, like changing the font or colours.

8.2 Own Survey's Results

In *November 2010*, we performed a small survey with *seven* participants by our own.[1] This means the survey is not very meaningful from a statistical point of view, but it offered some interesting aspects about the usage of CUBA.

The numbers of usages per person of CUBA were specified between 5 times and up to 40 times. All liked the concept of CUBA, citing various reasons. While some of them liked the *News feed concept* and the opportunity to *have an overall view over the subscripted News feeds*, others mentioned the *concept of interaction* and its possibilities.

On the contrary, all participants located CUBA's weakest point in the task to subscribe to a News feed. During the process it could happen, that CUBA shows a Home page of a feed, within the search results. This leads to an ugly and unreadable layout, which is caused by CUBA's feed handling process. We are aware of this problem and worked on it continuously during the whole development process, but we could not manage to get it under full control by now. Every person mentioned this problem and said that it turned down their motivation to use CUBA constantly.
There were no other common problems and disadvantages mentioned by the group. Logically, this would be the first thing they would improve on CUBA.

An interesting fact concerns the usage of CUBA's statistics. While everybody knows that CUBA provides statistics, only a minority is interested in it and use it. This is in contrary to our assumption and motivation to support statistics.

CUBA's got an average rating of 3.32 (within a range for 0 to 5) from the participants.

8.3 CUBA Statistics

After discussing cognitive topics like usability and perception of CUBA , we finally show some statistical data about the usage of CUBA . For this, we examined the data that are collected during an individual's visit on the CUBA Web site.

In addition to the usual log-data, like IP-address, time stamp, browser type or request, CUBA also stores data, which are necessary to run the application. This includes data such like NOPs, Interest Profiles or the visitor's current layout of his personal home page. Within the scope of this work, we analysed the available data and present the results in the following.

Basic Data

To understand the context of the analysis, we present some basic data first.
The examined data were collected between *10th December 2009* and *3rd November 2010*. Data between *27th May 2010* and *12th July 2010* are missing due to a crash of the Web

[1] All participants were personally known. They were chosen, because they could be identified as users of **CUBA**. There were six other registered users, but we were not able to contact them, because they used faked e-mail addresses.

server and an erroneous backup copy. Until *3rd November 2010*, CUBA has *12 registered users.*

After we have cleaned the data from unwanted sessions caused by search engines, or attacking attempts, we started with *1296* different Session IDs. We then removed to the best of knowledge all Sessions that were used for testing purposes and all sessions that consisted of only one request.[2] Overall, we have data based on *301 Sessions* on our disposal. Table 8.1 shows the sessions' distribution over the observed period. The sessions partitioned according their month/year. The first row displays the absolute number of sessions per month, while the second row displays the same information in percentage.

mm/yy	12/09	01/10	02/10	03/10	04/10	05/10	07/10	08/10	09/10	10/10	11/10
#	6	50	19	32	10	29	21	51	23	56	8
%	2	17	6	11	3	10	7	17	8	19	3

Table 8.1: Sessions per Month

Session Characteristics

The *shortest session* in our data set lasted *280ms.* On the contrary, the longest session took *7 days, 05:39.55 hours.* Usually these long sessions would be split up into several smaller ones, because nobody is staying such a long time permanently on a Web site without any breaks. We accepted these sessions, because most of the users used CUBA without taking advantage of their registration or of removing their session data that caused CUBA to consider these users as new users. In this case, long sessions allow us to follow users' behaviour over a longer time. Table 8.2 shows the partition of the sessions' durations.

Interval	< 0.5 min	< 1 min	< 5 min	< 10 min	< 30 min	< 60 min	< 24 h	≥ 24 h
#	51	35	74	35	30	24	40	12
%	17	12	25	12	10	8	13	4

Table 8.2: Statistics of Session Durations

The *number of requests* in a session varies from *minimum 2 requests* to *maximum 716 requests.*[3]

[2] The reasons of these sessions were primarily visitors' attempts to use CUBA with a browser that rejected cookies. In this case, a request generates a new cookie, every time there is a new request.

[3] To put the high number of maximum requests into perspective, we explain how CUBA collects requests: Beside ordinary requests like page requests, CUBA also deals with AJAX requests. In case of CUBA , this means that actions such moving a feed around results in a request, because CUBA has to know the feed's new position. The reason for the high number of requests lies in the "news preview" implementation. Every time a user moves over a news-link and releases the link a request is send to the server. We consider this behaviour as a feature and not as a bug, because it allows us to determine, if a user is still active on a CUBA Web page.

	# max occurrence	# avg occurence
Open News Preview	122	5.97
Adjustments	77	3.90
Open Feeds Home Page	33	1.68
Remove Feed	20	0.97
Open Article	20	0.91
Expand Feed	58	0.73
Re-arrange Feed	16	0.67
Session Statistics	35	0.58
Add Feed	20	0.47
Preview Statistics	11	0.43
Refresh Feed	12	0.26
Toggle Feed	11	0.33

Table 8.3: Statistic of Performed Actions

User Behaviour

The collected data enables us to analyse the user's behaviours, too.
In a first step, we broke the data down towards their corresponding actions' meaning (Table 8.3).
The first column describes the kind of activity. Here we list all traceable actions such as: a user opened a preview (*Open Preview*) or he removed a feed from his personal page (*Remove Feed*). The entry *Adjustments* has a special meaning in this context. It indicates a recalculation of the interest profile; build upon the layout of the personal Home page (see chapter 7.6.1). This happens when one of the following actions occurred: *Remove Feed*, *Expand Feed*, *Re-arrange Feed*, *Add Feed* or *Toggle Feed*.
Therefore, it is more an indicator then an action. The second column shows the maximal number of times the action occurred within a session.
The third column represents the average number of actions per session.

For example, if we look at the first row. It says that the *Open Preview* action happened 122 times at most inside a session, with an average of 5.97 times per session.

An analysis of this data does not reveal some big surprises. Since, we know from our own survey, nearly all users tend to scan their initial personal page for something interesting at first. If the user finds something interesting, he usually sneaks onto the interesting article and opens its preview. The most interesting fact was, in our opinion, that it twice as much people removing a feed instead of adding a new feed to their page.

In the next step, we take a closer look on the action/session dependency (Table 8.4). Therefore, we include the number of sessions where an action happened to the statistic (and its corresponding percentage, too). Based on this numbers we computed the average number of times the action was performed respective the sessions.

	# max occurence	#Sessions	% Sessions	# avg occurence
Adjustments	77	203	69.52	5.62
Open Preview	122	183	60.80	9.83
Open Feeds HP	33	109	36.21	4.64
Open Article	20	96	31.89	2.84
Remove Feed	20	64	21.26	4.96
Session Statistic	35	63	20.93	2.76
Preview Statistic	11	57	18.94	2.26
Re-arrange Feed	16	55	18.27	3.63
Add Feed	20	51	16.94	2.75
Expand Feed	58	28	9.30	7.89
Refresh Feed	12	28	9.30	2.96
Toggle Feed	11	21	6.98	4.67

Table 8.4: Statistic of Performed Actions II

We will show with the help of an example how to read the table. The second row can be read as follows: "The *Open Preview* action occurred in 183 sessions, that is in 60.8% of all 301 sessions. If the action occurred, we can expect in average an occurrence of 9.83 times."

One interesting thing concerns the number of feed previews, indicated by "Open Preview". It indicates that users like this feature, if they know about its existence. This was confirmed by our own survey, too. The same is true for CUBA's feed expanding feature, which has also a high average usage.

It could be an interesting approach in future works to focus more on information based on the number of actions.

Analysis of the Layout-Interest Profile Model

Finally we examined the consequences regarding the choice of the algorithm for computing the NOP based on the layout. For this reason, CUBA computes such a new interest profile, every time there is a change concerning the layout. Actually, it computes four different interest profiles in parallel. Therefore, it uses the *Average*-, *Coating*, *Dovetailing* and *Waving* algorithms (see section 7.6.1) to perform the computation.

This allows us to compare the different algorithms among each other. In Table 8.5, we list the minimal, maximal and average distance between profiles, which are based on the same layout, but computed with a different strategy. The listed values are absolute values and determined by computing the Euclidean-Distance between two profiles.

It is no surprise that the minimum-values of any combinations are 0. There are a

	A vs C	A vs D	A vs W	C vs D	C vs W	D vs W
Minimum	0	0	0	0	0	0
Maximum	0.0410	0.0410	0.0410	0.0147	0.0227	0.0227
Average	0.0085	0.0148	0.0047	0.0064	0.0040	0.010

Table 8.5: Comparison between different Layout Strategies
(A = average strategy; C = coating strategy; D = dove strategie; W = Wave strategy
X vs Y means, we compare strategy X with strategy Y.

few combinations, where this can happen. For example, if there is only one feed on a page. More surprisingly is the fact that in average, the distances are very small. Even the maximum numbers are a little bit higher.

Nevertheless, while it seemed an interesting idea in theory to offer different strategies to improve the implicit part of a NOP, there is no advantage of one algorithm in the current implementation. This leaves only two conclusions: either the current parameters for the computation of the interest profile lead to no distinction or the approach itself is not suitable. The evaluation of this problem could be an interesting point for further explorations.

Chapter 9

Summary and Outlook

In the following, we summarise the previous chapters and provide an outlook for some possible future work, based on our research.

9.1 Summary and Conclusions

The work began with an introduction into different concepts of *conviviality* and their domain specific definitions and meanings. We argued that these definitions do no meet requirements concerning the usage of Web sites and the interaction with them. Therefore, we defined *artificial conviviality* that is our own domain specific definition of conviviality concerning Web systems in general. The rest of the work's purpose is to present a suitable approach to support *artificial conviviality* on Web sites and evaluates its applicability.

First, we introduced Surowiecki's concept of "The Wisdom of Crowds" . Its purpose is to aggregate or extract valuable knowledge from a group of individuals, the crowd. To apply this concept in practice, the following four requirements must be fulfilled: There must be a *Diversity of Opinion* inside the crowd, an *Independency* of the members and a *Decentralisation* of them; furthermore there must also exist an *Aggregation* instance. Therefore, we motivated the usage of profiles and introduced the *Non-Obvious Profile algorithm*. The purpose of this algorithm is to determine a user's interest profile based on a user's explicit and implicit interest profiles. These profiles are building during the user's interaction with the Web system. Next, we introduced different common techniques for collaborative filtering. We argued that these techniques could be used inside an Aggregation instance to extract valuable knowledge. Thus, we used the NOP algorithm in combination with a similarity based approach in this work to gain from "The Wisdom of Crowds" and create a recommendation system.

At the beginning of the practical part, we introduced our CUBA prototype, which allows the user to read News-feeds, and motivated the purpose of its existence. We first discussed CUBA's underlying architecture. Afterwards we introduced the concepts of *News feeds* and *AJAX* technology and described where it is applied in the CUBA prototype. Then we presented the CUBA prototype implementation. Based on screen shots we show its appearance, its provided functions and its usage. We finished the work

by presenting some research results about CUBA's usability and CUBA's perceptions. We also show several usage statistics and their related conclusions.

In retrospect, the work has its some limitations and some strong sides. There are also some interesting questions that raised by the work, but which are still open. Regarding the first part that deals with conviviality and especially with *artificial conviviality*, we think that the work misses a better motivation why there is a need of *artificial conviviality* in principle and why its definition was chosen. While the motivation and definition was clear in the author's mind and the different definitions of conviviality leads towards the topic, we realised during writing the work that it is difficult to describe these ideas in clear and precise words. Nevertheless, there are still open questions that could be subject of future work (9.2). The rest of the first part deals with a possible way implementing our concept of artificial conviviality inside a Web site. Here, too, there might be some flaws in the motivation of choosing this approach, but in our opinion, the work points out the way our concept works. On the other hand, the description of the important design and architecture aspects of CUBA should left no answers open. The same applies to the description of implementation details and supported functions of CUBA. The few results of CUBA's usage and their interpretations becloud this chapter. Normally these results could be the basic for a better understanding of users or building of usage models. Nevertheless, since the Web site could not motivate enough individuals to use its service, there were not enough data generated. This did not enable us to come up with a measurable statement of the suitability of our approach. We identified trouble with catching and displaying feeds as one reason for this problem. Every time there were some problems, the users stopped the usage directly. However, since these problems occurred during most time of the project, we have had to questions the usage of News-feeds for such a project. On the other hand, if they are working, they provide interesting content that could be use to motivate users to use the system and participate to experiments.

9.2 Future Work

Considering this work, there are many directions and options for future work.

One direction should be, as mentioned in the previous section, to continue to work on the concept of *artificial conviviality*. The main effort should be put in to improve the concept of *artificial conviviality* and make the current definition more precise. We also highly recommend turning into a more interdisciplinary approach to do so. As mentioned in the work, there are other research fields that deal with conviviality. It would be interesting to gain more from their experiences.

It could also be interesting to explore other approaches, beside the one, chosen in this work that will support the concept of *artificial conviviality*. We could imagine, for example, a system that includes a chatbot, which enables the user to communicate with the system in a "human"-way. Another approach could be to develop a new concept of collaborative work, which enables the user to work on (and solve) problems together.

Another direction concerns the Non-Obvious Profile approach. Since its preparation and update processes are not trivial and require a lot of work, we suggest finding a way to make these processes easier for the user. One way to do this is to examine a "hybrid"-approach to determine users' interests. In this case, the NOP algorithm would still be used for an initial setup of a Web site and calculation of the NOPs. However, we would suggest switching to a model-based approach as soon as there are enough data for a promising deployment of this approach to avoid further content classifications. This suggestion would also include to making further experiments how accurate the NOP approach behaves in practice and an eventual modification of the adjustments and conclusions respectively the several profiles (NOP-difference (ND), FD-difference (FD) and Difference (D)).

Regarding to the preparation process it could also be interesting to examine the usage of text classification approaches and their influence to NOPs. This could be use to determine the quality of NOPs based on manual of algorithmic text classification.

Finally, there are many possibilities to improve the CUBA prototype. A first primary effort should be to improve the feed-module to catch feeds in a better way. How it has become apparent, this is the most important requirement to perform future user experiments, because it strongly influences the acceptance to use CUBA.

Since the user acceptance of the layout and the kind of work with CUBA was mostly positive, it would also be encouraging to continue the work in this direction. It could be interesting, for example, to develop new kinds of "User-Web site-Interaction" models or enable the user to "play" with new kind of layouts and how News are displayed. Related to this is the possibility to make CUBA available on mobile devices. Since CUBA is based on News-feeds, it might be interesting to adapt it to support such devices to enable to receive the latest News everywhere. In its current state, CUBA is implemented to be used on a computer's web browser only. As there are some differences of Event-handling on computer and mobile device browsers, there is the necessity to adapt CUBA on such systems. This will result in some changes on the interface, but it could also influence CUBA's underlying architecture, etc.

It would also be conceivable to open CUBA in the direction to become a social Web site, where the users are more involved. This could be done in many ways. One approach could be to enable the user to use a feed in more ways. We can imagine enabling users to build their own feeds by taking interesting News from other feeds, or writing their own News. These feeds then could be shared with other people. This would allow us to provide a social infrastructure where people can meet each other and exchange their opinions.

Another idea would belong to a method to improve the quality of feeds and recommendations. For this case, CUBA could provide a module that let the user tag feeds. This could help to improve CUBA's search functionality on the one hand side. On the other hand it would also help the user to find appropriate feeds more easily.

Finally, the probably most challenging problem in our opinion, regarding *artificial conviviality* and CUBA is to find/develop an accurate method to measure conviviality.

Appendix A

Schemas of several Newsfeed Formats

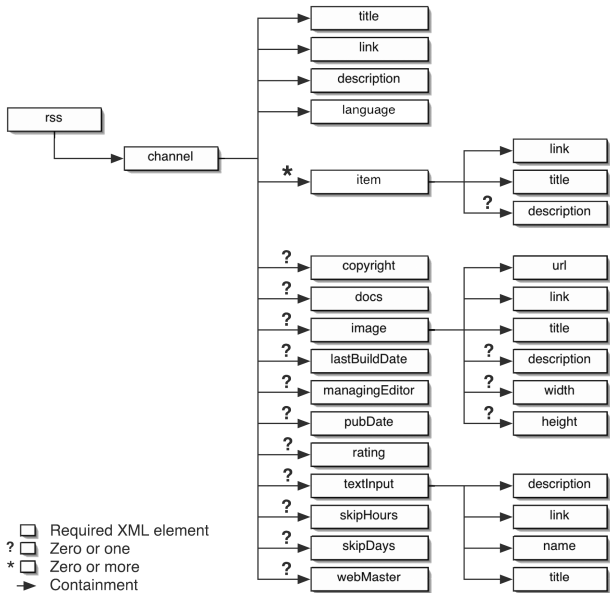

Figure A.1: Schema of a RSS 0.91 XML-file [56]

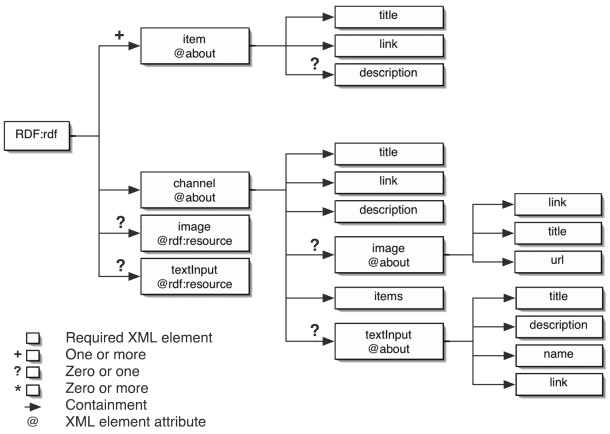

Figure A.2: Schema of a RSS 1.0 XML-file [56]

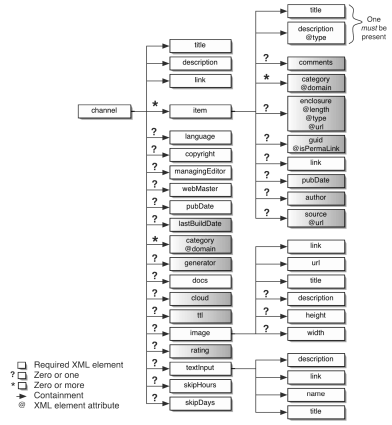

Figure A.3: Schema of a RSS 2.0 XML-file (taken from [56])

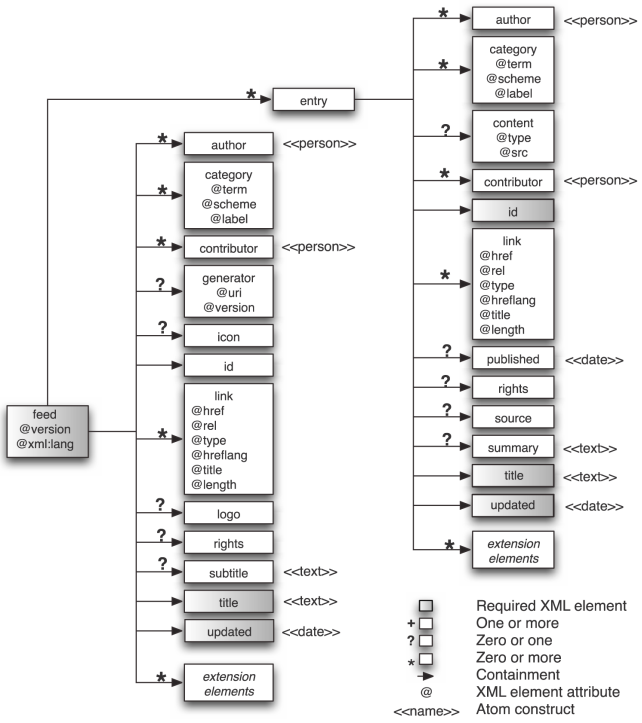

Figure A.4: Schema of an ATOM 1.0 XML-file (taken from [56])

Appendix B

Code Examples for Newsfeed Files

Sample of and Atom-Feed

```
<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xml:base="http://example.org/"
      xml:lang="en">
  <title type="text">Sample Feed</title>
  <subtitle type="html">
    For documentation &lt;em&gt;only&lt;/em&gt;
  </subtitle>
  <link rel="alternate" href="/"/>
  <link rel="self"
      type="application/atom+xml"
      href="http://www.example.org/atom10.xml"/>
  <rights type="html">
     &lt;p&gt;Copyright 2005, Mark Pilgrim&lt;/p&gt;&lt;
  </rights>
  <id>tag:feedparser.org,2005-11-09:/docs/examples/atom10.xml</id>
  <generator
      uri="http://example.org/generator/"
      version="4.0">
          Sample Toolkit
  </generator>
  <updated>2005-11-09T11:56:34Z</updated>
  <entry>
    <title>First entry title</title>
    <link rel="alternate"
        href="/entry/3"/>
    <link rel="related"
        type="text/html"
        href="http://search.example.com/"/>
    <link rel="via"
        type="text/html"
        href="http://toby.example.com/examples/atom10"/>
    <link rel="enclosure"
```

```
        type="video/mpeg4"
        href="http://www.example.com/movie.mp4"
        length="42301"/>
    <id>tag:feedparser.org,2005-11-09:/docs/examples/atom10.xml:3</id>
    <published>2005-11-09T00:23:47Z</published>
    <updated>2005-11-09T11:56:34Z</updated>
    <summary type="text/plain" mode="escaped">Watch out for nasty tricks</summary>
    <content type="application/xhtml+xml" mode="xml"
             xml:base="http://example.org/entry/3" xml:lang="en-US">
      <div xmlns="http://www.w3.org/1999/xhtml">Watch out for
      <span style="background: url(javascript:window.location='http://example.org/')">
      nasty tricks</span></div>
    </content>
  </entry>
</feed>
```

Sample of an RSS 2.0-Feed

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
    <channel>
        <title>Sample Feed</title>
        <description>For documentation &lt;em&gt;only&lt;/em&gt;</description>
        <link>http://example.org/</link>
        <pubDate>Sat, 07 Sep 2002 0:00:01 GMT</pubDate>
        <!-- other elements omitted from this example -->
        <item>
            <title>First entry title</title>
            <link>http://example.org/entry/3</link>
            <description>Watch out for &lt;span style=
                "background-image:url(javascript:window.location='http://example.org/
                &gt;nasty tricks&lt;/span&gt;</description>
            <pubDate>Sat, 07 Sep 2002 0:00:01 GMT</pubDate>
            <guid>http://example.org/entry/3</guid>
            <!-- other elements omitted from this example -->
        </item>
    </channel>
</rss>
```

Appendix C

Zend MVC-Implementation

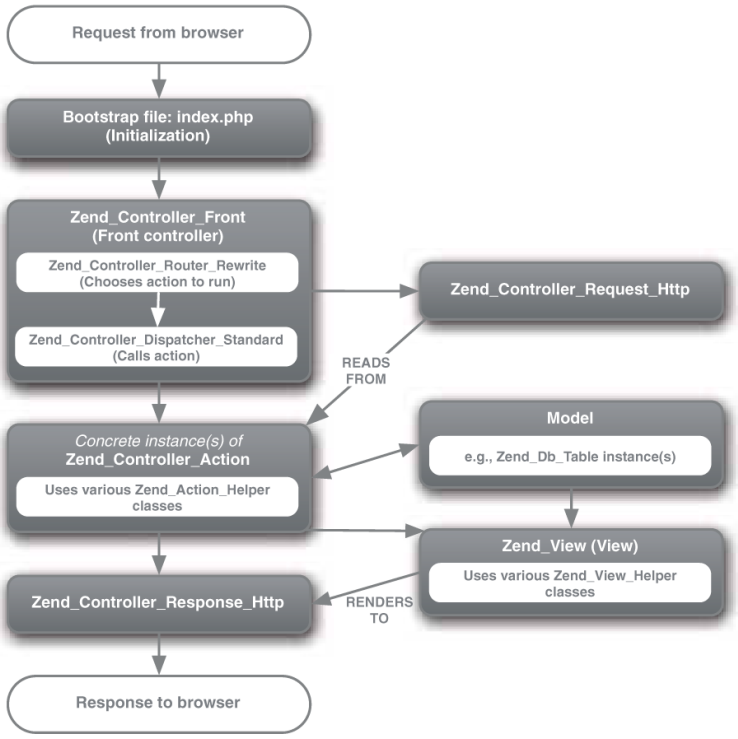

Figure C.1: Schema of the Zend MVC-Implementation (taken from [98])

Appendix D

Own Survey's Questionnaire

- Do you know CUBA? (yes/no)
- Did you use *CUBA* (yes/no)
- In case you did, how many times did you use it?
- What do you like on CUBA?
- What do you dislike on CUBA?
- What would you like to improve on CUBA?
- Are you a registered in CUBA? (yes/know)
- In case you are, do you know that you can enter your interests?
- Do you know CUBA's statistic part?
- How satisfied you are with CUBA's usability?
- How satisfied you are with CUBA's recommendations?
- Do you know any other News feed reader?
- In case you do, which ones?
- If you have to rate CUBA at a scale from 0-5, where 5 being the best score, which score would you give it?

Appendix E

Usage Statistics

To get a feeling about CUBA's data, we present in the following an excerpt of CUBA's session statistics.

The fieldnames have the following meaning:

- SessionId: Session Id
- requests: Number of requests within a session
- requests2: Cleaned number of requests after taking away unnecessary prev_closed requests
- mainpage: number of requests for the main page
- prev_open: number of request for article previews (open)
- prev_open: number of request for article previews (close), could also mean a movement over a link
- clicked: number of opened articles
- hlmo: number of opened Home pages
- moved: number of feed re-arrangements
- expand, toggle: number of times a feed was expanded, toggled
- refresh: number of feed refreshes
- removefeed: number of times a feed was removed from the Home page
- add: number of times a feed was added to the Home page
- csclicks, csession, cspreviews: concerns statistical requests
- minTimestamp, maxTimestamp: timestamp of session's first and last request
- duration: duration of session, i.e. maxTimestamp - minTimestamp

SessionId	requests	requests2	mainpage	prev_opened	prev_closed	clicked	hlmo	moved	expand	refresh	removefeed	addfeed	toggle	csclicks	csession	cspreviews	ctab	minTimestamp	maxTimestamp	duration
49a42279a5a582	88	62	9	3	23	1	1	4	0	0	8	2	0	0	5	3	0	2010-01-11 11:57:32	2010-01-15 23:45:21.315256	4 days 11:47:49.3
d38b34ae80add4	91	41	6	2	48	0	2	0	0	0	1	4	0	0	0	0	0	2010-10-09 09:33:10.369433	2010-10-09 09:49:36.4294	00:16:26.059967
8dc8ebb08f8556f	92	50	2	11	31	1	1	1	0	0	4	0	0	0	10	0	0	2010-01-11 14:22:05	2010-01-11 14:47:32	00:25:27
8e18d5c9861398	92	18	2	15	59	3	1	2	0	0	3	2	0	0	0	0	0	2010-08-21 06:38:51.135919	2010-08-21 07:09:25.129891	00:30:33.993972
c06960ef016fa01	92	27	4	17	48	2	1	0	0	0	5	1	0	0	0	0	0	2010-05-11 07:38:38.812042	2010-05-11 07:42:58.28559	00:04:19.473548
30de3ffa59e215d	94	32	15	13	49	0	1	1	0	0	0	0	0	0	0	0	0	2010-02-10 14:04:53.31656	2010-02-10 14:45:35.308898	00:40:41.992338
a783e2ca36a076(	96	44	6	7	45	2	2	2	0	3	7	5	2	0	0	0	0	2010-01-23 16:43:12.894708	2010-01-23 16:51:36.494161	00:08:23.599453
daf9b4f86177ae4	99	70	17	1	28	0	9	5	1	0	12	0	0	0	1	1	0	2010-09-28 16:09:27.558901	2010-09-28 16:57:46.532358	00:48:18.973457
77ec090bc9d006(	102	2	2	32	68	0	0	0	0	0	0	0	0	0	0	0	0	2010-07-29 13:51:14.084068	2010-07-29 13:54:31.264306	00:03:17.180238
0278282a5fd01b!	102	38	3	5	59	2	5	0	0	0	3	1	0	0	2	2	0	2010-08-14 17:47:46.722025	2010-08-14 19:00:53.89397	01:13:07.171945
b74e42cedd7925	102	68	8	5	29	0	2	5	10	3	7	1	2	0	4	4	0	2010-02-25 12:51:25.458152	2010-02-25 15:18:04.552104	02:26:39.093952
6d45556e169874	105	18	3	24	63	7	2	0	0	0	0	0	0	0	0	0	0	2010-10-11 18:03:19.461687	2010-10-11 19:09:57.516854	01:06:38.055167
d23aa56c63bf8c5	112	29	10	25	58	7	3	0	0	0	0	0	0	0	0	0	0	2010-10-05 11:51:26.025006	2010-10-05 18:47:44.172724	06:56:18.147718
f2447e1a13897b:	114	31	3	12	71	6	10	0	0	0	6	1	0	0	0	0	0	2010-08-17 07:58:00.119906	2010-08-17 08:11:40.616382	00:13:40.496476
48dea87d68bdc6(	121	51	5	10	60	0	7	1	0	0	14	3	0	0	1	1	0	2010-08-13 15:27:34.079463	2010-08-13 16:06:45.816705	00:39:11.737242
bfece2c781c8468	122	53	8	18	51	1	3	0	0	0	0	0	0	0	8	8	0	2010-05-05 17:15:06.669678	2010-05-06 12:51:12.584181	19:36:05.914503
1228dc1c0c297a(	123	37	5	17	69	0	1	2	18	3	4	0	0	0	0	0	0	2010-01-26 09:18:46.385622	2010-01-26 11:17:47.109628	01:59:00.724006
364f9760d33733!	128	32	27	16	80	0	3	0	0	1	1	0	0	0	0	0	0	2010-03-11 14:12:36.530474	2010-03-11 15:26:29.761473	01:13:53.230999
918d2c32047795	137	16	3	25	96	0	0	3	2	0	3	0	0	0	1	1	0	2010-10-01 08:09:35.392476	2010-10-01 08:28:56.552349	00:19:21.159873
943666ed0c3c96:	139	19	5	26	94	0	1	1	0	2	3	2	0	0	0	0	0	2010-03-15 12:22:02.548019	2010-03-15 14:26:38.176543	02:04:35.628524
imhjpsjrl9nktpfor:	140	59	4	9	72	2	13	6	0	0	4	1	11	0	2	0	0	2009-12-11 20:09:22	2009-12-11 22:13:54	02:04:32
d89d80a77d995c	148	40	5	20	88	8	12	2	0	0	0	2	0	0	1	1	0	2010-08-18 12:14:49.907869	2010-08-18 12:46:52.914148	00:32:03.006279
4c0e982da3ec7fe	150	31	6	6	113	5	1	2	0	0	3	2	0	0	1	1	0	2010-08-19 14:20:12.537942	2010-08-20 12:33:29.638503	22:13:17.100561
66f0be0f4e89fcb4	167	62	6	9	96	0	7	10	1	0	7	2	9	0	1	1	0	2010-02-03 12:21:52.762207	2010-02-03 12:34:38.023898	00:12:45.261691
609ad6760adccd:	168	58	6	7	103	0	0	4	1	2	5	0	2	0	3	3	0	2010-08-10 21:21:45.670606	2010-08-11 23:22:03.378368	1 day 02:00:17.7(
4e6547b4ed6abd	171	57	15	14	100	3	0	0	0	0	0	3	0	0	3	3	0	2010-09-29 22:45:20.097906	2010-09-29 23:17:42.89283	00:32:22.794924
kbjb6iafn9emdt4!	173	31	5	21	121	1	3	3	0	1	0	1	9	0	0	0	0	2009-12-14 21:19:47	2009-12-14 21:41:31	00:21:44
8935ec4f4c109ea	174	68	6	7	99	2	13	2	6	0	3	2	0	0	0	0	0	2010-10-18 14:15:50.286141	2010-10-18 14:49:16.097333	00:33:25.811192
0e822c9b235380	179	51	3	23	105	4	5	0	34	0	0	0	0	0	1	1	0	2010-05-04 08:32:09.749371	2010-05-04 08:45:54.995721	00:13:45.24635
cd2d4a908b89bf4	183	112	16	12	59	2	26	0	0	9	4	4	0	0	3	3	0	2010-10-04 20:16:12.044406	2010-10-04 20:40:45.548881	00:24:33.504475
gqrqpe74enbiqrd!	190	34	3	18	138	2	14	1	0	1	0	1	7	0	0	0	0	2009-12-11 20:23:54	2009-12-11 22:18:09	01:54:15
46186960e942da	191	46	14	35	110	1	4	0	7	0	1	1	7	0	0	0	0	2010-01-23 13:37:34.955443	2010-01-26 10:03:01.75616	2 days 20:25:26.8
3c220b18251657	200	57	11	21	122	0	3	3	0	0	12	4	0	0	3	3	0	2010-01-14 07:31:59	2010-01-18 10:46:12.804761	4 days 03:14:13.8
b6c11b7d73a9dc!	202	28	6	42	132	4	5	1	7	1	3	1	0	0	0	0	0	2010-03-12 10:42:10.923602	2010-03-12 12:53:04.88039	02:10:53.956788
965291983b9c0c	204	131	16	10	63	6	2	13	0	0	3	4	0	0	8	8	0	2010-02-25 11:38:53.708331	2010-02-25 12:46:14.795253	01:07:21.086922
8e015458d85d7e	220	39	5	16	165	4	11	1	0	1	1	2	0	0	1	1	0	2010-01-14 21:39:44.073294	2010-01-14 22:01:38.084903	00:21:54.011609
5e686a4a257462	277	170	30	15	92	2	6	8	0	0	12	20	0	0	0	0	0	2010-10-12 08:50:32.282007	2010-10-19 14:30:27.004585	7 days 05:39:54.7
7e36d6625d33f0(	279	76	12	33	170	3	10	7	0	0	2	2	10	0	1	1	0	2010-01-12 15:19:56	2010-01-15 08:54:00.753537	2 days 17:34:04.7
d9a6660c6def09(	282	91	11	18	173	2	20	6	4	4	3	3	0	0	3	3	0	2010-08-17 12:16:17.52813	2010-08-17 12:39:00.077167	00:22:42.549037
asovj7o6i9bb9aev	286	55	6	21	210	1	16	2	0	3	2	2	8	0	1	0	0	2009-12-12 09:43:28	2009-12-12 10:24:31	00:41:03
0fddff9d3b4e707(	295	101	12	32	162	3	9	4	58	5	5	1	0	0	0	0	0	2010-01-25 08:38:58.648001	2010-01-25 14:37:50.632924	05:58:51.984923
0d89e47838ef7e2	310	190	15	20	100	6	0	1	0	1	3	2	0	0	35	11	0	2010-01-11 09:15:24	2010-01-12 12:22:44	1 day 03:07:20
c46b96aaa3ab77:	312	72	13	53	187	3	22	2	3	0	6	3	0	0	1	1	0	2010-05-05 13:13:11.2503	2010-05-06 12:09:18.189879	22:56:06.939579
081059b4698378	315	23	5	65	227	0	14	0	1	3	0	0	0	0	0	0	0	2010-07-23 14:34:31.880379	2010-07-23 16:53:05.928384	02:18:34.048005
c2be0f6deabc6b2	341	97	12	12	232	2	4	7	8	12	6	7	2	0	2	2	0	2010-09-10 14:59:42.19596	2010-09-10 15:42:52.396137	00:43:10.200177
b0404a598b60bc	348	57	6	43	248	3	12	5	4	0	8	3	0	0	2	2	0	2010-07-23 16:12:18.146414	2010-07-23 16:32:33.289615	00:20:15.143201
180bb37fe17f4be	395	93	5	34	268	14	27	5	0	0	6	2	0	0	1	1	0	2010-08-13 15:57:29.838542	2010-08-14 08:37:53.60022	16:40:23.761678
da049f9c4d2464:	400	158	20	21	221	3	33	11	0	3	9	9	2	0	7	7	0	2010-01-18 21:21:05.124489	2010-01-18 21:35:13.77416	00:14:08.649671
c4efdbdb47acaa4	442	150	18	25	267	20	6	16	0	0	8	2	0	0	11	11	0	2010-02-23 12:28:54.371162	2010-02-24 11:27:31.898763	22:58:37.527601
1e409352cbddfec	475	63	10	98	314	0	2	0	17	1	5	2	2	0	1	1	0	2010-03-11 16:31:16.982233	2010-03-11 17:36:36.385423	01:05:19.40319

Figure E.1: Screenshot of User Statistics Table

Acronyms

AJAX	Asynchronous JavaScript and XML
CSS	Cascading Style Sheet
CUBA	Conviviality and User Behaviour Analysis
DOM	Document Object Model
GUI	Graphical User Interface
HCI	Human-Computer Interaction
IETF	Internet Engineering Task Force
JSON	JavaScript Object Notation
MVC	Model-View-Controller
NOP	Non-Obvious Profile
NOPs	Non-Obvious Profiles
RDF	Resource Description Framework
RSS	RDF Site Summary
RSS	Really Simple Syndication
WWW	World Wide Web
XHTML	Extensible HyperText Markup Language
XML	eXtensible Markup Language
XSLT	eXtensible Stylesheet Language Transformations

Bibliography

[1] J. J. Macionis, *Sociology*. Pearson International Edition, Upper Saddle River, New Jersey, USA: Pearson Education, 12th ed., |2008|.

[2] Merriam-Webster Online Dictionary. 2010, *Definition of convivial*, (May 31, 2010). `http://www.merriam-webster.com/dictionary/convivial`.

[3] P. Caire, "A normative multi-agent systems approach to the use of conviviality for digital cities.," in *International Conference and Research Center for Computer Science, Schloss Dagstuhl*, (Wadern, Germany), |2007|.

[4] G. Fischer and A. Lemke, *Constrained Design Processes: Steps Towards Convivial Computing*. In R. Guindon, ed., 'Cognitive Science and its Applicationfor Human-Computer Interaction', Lawrence Erlbaum Associates, Inc., Hillsdale, NJ, USA, pp. 1-58., |1988|.

[5] I. Illich, *Tools for Conviviality*. London, UK: Marion Boyars Publishers, |1974|.

[6] J. Surowiecki, *The Wisdom of Crowds - Why the many are smarter than the few*. London, UK: Abacus, |2005|.

[7] E. Fromm, *Ihr werdet sein wie Gott*. Rowohlt Taschenbuch, |1980|.

[8] W. Morris, *The American Heritage dictionary of the English language*. Boston: Houghton Mifflin, new college ed., |1979|.

[9] D. Summers, *Longman dictionary of contemporary English*. Harlow, Essex, England: Longman, new ed., |1987|.

[10] D. Day, *Le Robert & Collins senior*. Glasgow ; New York, N.Y. Paris: Collins ; Dictionnaires Le Robert, 6th ed., |2002|.

[11] G. d. terminologique, *Definition of convivialite*, 2001 (March 24, 2010). `http://www.granddictionnaire.com/BTML/FRA/r_Motclef/index800_1.asp`.

[12] B. Schwartz, *The paradox of choice : why more is less*. New York: Ecco, 1st ed., |2004|.

[13] P. Caire, *New Tools for Conviviality – Masks, Norms, Ontology, Requirements and Measures Bridging the Conviviality Gap between Policy and Informatics*. PhD thesis, Université du Luxembourg, 162a, avenue de la Faïencerie, L-1511 Luxembourg, Luxembourg, |2010|.

[14] Facebook, Inc, *Homepage of Facebook*, October 20, 2010. `http://www.facebook.com`.

[15] Twitter, Inc, *Homepage of Twitter*, October 20, 2010. `http://twitter.com`.

[16] Skype Limited, *Homepage of Skype*, October 20, 2010. `http://www.skype.com`.

[17] L. Rosenfeld and J. McMullin, *The User Experience Disciplines Radial*, (November 06, 2010). `http://www.interactionary.com/`.

[18] UPA – Usability Professionals' Association, *Definition of convivial*, (November 06, 2010). http://www.usabilityprofessionals.org/usability_resources/about_usability/what_is_ucd.html.

[19] J. Treynor, "Market efficiency and the bean jar experiment," *Financial Analysts Journal*, vol. 43 (May/June), pp. 50–53, 1987.

[20] IEM Organization, *Homepage of the Iowa Electronic Markets*, (August 30, 2010). http://tippie.uiowa.edu/iem/.

[21] J. Berg, R. Forsythe, F. Nelson, and T. Rietz, *Results from a Dozen Years of Election Futures Markets Research*, |2000| (August 30, 2010). http://tippie.uiowa.edu/iem/archive/bfnr_2000.pdf.

[22] S. Brin and L. Page, *The Anatomy of a Large-Scale Hypertextual Web Search Engine*, (September 29, 2010). http://infolab.stanford.edu/pub/papers/google.pdf.

[23] D. Tammet, *Embracing the Wide Sky*. London, UK: Hodder & Stoughton Ltd, |2009|.

[24] T. W. Malone, *What is collective intelligence and what will we do about it?*, |October 13, 2006| (August 30, 2010). http://cci.mit.edu/about/MaloneLaunchRemarks.html.

[25] M. C. for Collective Intelligence, *What is collective intelligence? - Handbook of Collective Intelligence*, |August 7, 2008| (August 30, 2010). http://scripts.mit.edu/~cci/HCI/index.php?title=What_is_collective_intelligence%3F.

[26] S. Allag, *Collective Intelligence in Action*. Greenwich, CT 06830, USA: Manning Publications, |2009|.

[27] T. O'Reilly, *What's the difference between "collective intelligence" and collaborative filtering?*, (August 30, 2010). http://getsatisfaction.com/oreilly/topics/whats_the_difference_between_collective_intelligence_and_collaborative_filtering.

[28] Merriam-Webster Online Dictionary. 2010, *Definition of convivial*, (June 07, 2010). http://www.merriam-webster.com/dictionary/profile.

[29] L. T. de France, *Tour de France 2010 - Stage by stage (Stage 17 - Pau - Col du Tourmalet)*, (August 17, 2010). http://www.letour.fr/2010/TDF/COURSE/us/1700/etape_par_etape.html.

[30] N. Mushtaq, P. Werner, K. Tolle, and R. V. Zicari, "Building and evaluating non-obvious user profiles for visitors of web sites," in *CEC 2004 - IEEE International Converence on e-Commerce Technology, San Diego, CA, USA, July 6-9, 2004*, (San Diego, CA, USA), pp. 9–15, IEEE Computer Society, |2004|.

[31] N. Hoebel, S. Kaufmann, K. Tolle, and R. V. Zicari, "The design of gugubarra 2.0: A tool for building and managing profiles of web users," in *2006 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2006), 18-22 December 2006, Hong Kong, China*, (Hong Kong, China), pp. 317–320, IEEE Computer Society, |2006|.

[32] N. Hoebel, S. Kaufmann, K. Tolle, and R. V. Zicari, "The design of gugubarra 2.0: A tool for building and managing profiles of web users," in *2006 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2006), 18-22 December 2006, Hong Kong, China*, (Hong Kong, China), pp. 317–320, IEEE Computer Society, |2006|.

[33] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, pp. 11–21, 1972.

[34] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," 1997.

[35] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, p. 2003, 2003.

[36] S. Deerwester and et al., "Improving information retrieval with latent semantic indexing," *Proceedings of the 51st Annual Meeting of the American Society for Information Science 25*, pp. 36–40, 1988.

[37] Griffiths, Thomas L. and Steyvers, Mark, "Finding scientific topics," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, pp. 5228–5235, April 2004.

[38] J. Sterne, *Web Metrics – Proven Methods for Measuring Web Site Success*. Wiley Publishing, New York, NY, USA., |2002|.

[39] B. Eisenberg, J. Eisenberg, and L. T. Davis, *Call to action : secret formulas to improve online results*. Nashville, Tenn.: Nelson Business, |2006|.

[40] E. T. Peterson, *Web site measurement hacks*. Sebastopol, Calif. ; Farnham: O'Reilly, 1st ed., |2005|.

[41] G. Salton and M. McHill, *Introduction to Modern Information Retrieval*. New York: McGraw-Hill, |1983|.

[42] Watson, James B., *Correlation examples*, (October 21, 2010). `http://upload.wikimedia.org/wikipedia/commons/0/02/Correlation_examples.png`.

[43] P. Cabena, P. Hakjinian, R. Stadler, J. Verhees, and A. Zanasi, *Discovering Data Mining – From Concept to Implementation*. New York: Prentice Hall PTR, |1998|.

[44] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)*. Heidelberg: Springer, 3rd ed., |2009|.

[45] T. Kohonen, *Self-Organizing Maps*. Heidleberg: Springer, 2nd ed., |1997|.

[46] J. S. Breese, D. Heckerman, and C. M. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *UAI*, pp. 43–52, 1998.

[47] Netflix, Inc, *Netflix Press Kit*, (September 27, 2010). `http://cdn-0.nflximg.com/us/pdf/Consumer_Press_Kit.pdf`.

[48] R. M. Bell, J. Bennet, Y. Koren, and C. Volinsky, "The million dollar programming prize," *ieee Spectrum – The Magazine of Technology Insiders*, May |2009|.

[49] Netflix, Inc, *Netflix Prize: Home*, (September 27, 2010). `http://www.netflixprize.com/`.

[50] Netflix, Inc, *Netflix Prize: Leaderboard*, (September 27, 2010). `http://www.netflixprize.com//leaderboard`.

[51] E. van Bushkirk, *How the Netflix Prize Was Won*, (September 27, 2010). `http://www.wired.com/epicenter/2009/09/how-the-netflix-prize-was-won/`.

[52] A. Töscher, M. Jahrer, and R. M. Bell, *The BigChaos Solution to the Netflix Grand Prize*, (September 27, 2010). `http://www.netflixprize.com/assets/GrandPrize2009_BPC_BigChaos.pdf`.

[53] M. Piotte and M. Chabbert, *The Pragmatic Theory solution to the Netflix Grand Prize*, (September 27, 2010). http://www.netflixprize.com/assets/GrandPrize2009_BPC_PragmaticTheory.pdf.

[54] Y. Koren, *The BellKor Solution to the Netflix Grand Prize*, (September 27, 2010). http://www.netflixprize.com/assets/GrandPrize2009_BPC_BellKor.pdf.

[55] M. Salganik, P. Dodds, and D. Watts, "Experimental study of inequality and unpredictability in an artificial cultural market," *Science*, vol. 311, pp. 854–856, |2006|.

[56] D. Johnson, *RSS and Atom in action : web 2.0 building blocks*. Greenwich, Conn.: Manning, |2006|.

[57] Libby and Netscape, "RSS 0.90," (March 23, 2010). http://www.purplepages.ie/RSS/netscape/rss0.90.html.

[58] Libby and Netscape, "RSS 0.91 (Netscape)," (March 23, 2010). http://my.netscape.com/publish/formats/rss-spec-0.91.html.

[59] D. Winer and UserLand, "RSS 0.91 (UserLand)," (March 23, 2010). http://backend.userland.com/rss091.

[60] RSS-DEV Working Group, "RSS 1.0," (March 23, 2010). http://www.web.resource.org/rss/1.0/.

[61] RSS-DEV, "RSS 1.0," (March 23, 2010). http://www.web.resource.org/rss/1.0/spec.

[62] D. Winer and Harvard, "RSS 2.0.1," (March 23, 2010). http://blogs.law.harvard.edu/tech/rss.

[63] M. Nottingham and R. Sayre, "RFC 4287 – The Atom Syndication Format," |December, 2005| (April 18, 2010). http://tools.ietf.org/html/rfc4287.

[64] J. Gregorio and B. de hOra, "RFC 5023 – The Atom Publishing Protocol," |October, 2007| (April 18, 2010). http://tools.ietf.org/html/rfc5023.

[65] World Wide Web Consortium (W3C), *Extensible Markup Language (XML) 1.0 (Fifth Edition)*, November 26, 2008 (March 22, 2010). http://www.w3.org/TR/xml/.

[66] Apache Software Foundation, *Homepage of Apache Commons Feedparser / Jakarta FeedParser*, (September 10, 2010). http://commons.apache.org/dormant/feedparser/.

[67] M. Pilgrim, *Homepage of the Universal Feedparser*, (September 10, 2010). http://feedparser.org.

[68] A. Abdelnur, P. Chanezon, and E. Chien, *rome:ROME RSS Atom syndication and publishing tool*, (September 10, 2010). https://rome.dev.java.net/.

[69] *Homepage of RomeFetcher*, (September 10, 2010). http://wiki.java.net/bin/view/Javawsxml/RomeFetcher.

[70] C. Alexander, S. Ishikawa, and M. Silverstein, *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press, |1979|.

[71] C. Larman, *Applying UML and patterns : an introduction to object-oriented analysis and design and iterative development*. Upper Saddle River, N.J.: Prentice Hall PTR, 3rd ed., |2005|.

[72] B. Bruegge and A. H. Dutoit, *Object-oriented software engineering : using UML, Patterns, and Java*. Upper Saddle River, NJ.: Pearson Education International, 2nd ed., |2004|.

[73] T. M. H. Reenskaug, "Model-View-Controller modell." `http://heim.ifi.uio.no/\~trygver/themes/mvc/mvc-index.html`, (October 18, 2010).

[74] J. J. Garrett, *Ajax: A New Approach to Web Applications*, |February 18, 2005| (March 22, 2010). `http://www.adaptivepath.com/ideas/essays/archives/000385.php`.

[75] Douglas Crockford, *Introducing JSON)*, (March 22, 2010). `http://www.json.org`.

[76] World Wide Web Consortium (W3C), *XSL Tranformations (XSLT) Version 1.0*, November 16, 1999 (March 22, 2010). `http://www.w3.org/TR/xslt/`.

[77] D. Crane, E. Pascarello, and D. James, *Ajax in Action*. Greenwich, Conn.: Manning, |2006|.

[78] D. Johnson, A. White, and A. Charland, *Enterprise AJAX : strategies for building high performance web applications*. Upper Saddle River, NJ: Prentice Hall, |2008|.

[79] M. Mahemoff, *Ajax design patterns*. Sebastopol, CA: O'Reilly, 1st ed., |2006|.

[80] PostgreSQL Global Development Group, *PostgreSQL Homepage*, (October 13, 2010). `http://www.postgresql.org`.

[81] J. Kneschke, *Homepage of the lighttpd project*, (September 10, 2010). `http://www.lighttpd.net/`.

[82] Apache Software Foundation, *Homepage of the Apache HTTP Server Project*, (September 10, 2010). `http://httpd.apache.org/`.

[83] Zend Technologies Ltd, *Homepage of Zend Framework*, (September 10, 2010). `http://www.zendframework.com`.

[84] jQuery Team, *Homepage of jQuery*, (September 11, 2010). `http://jquery.com`.

[85] BuildWith.com, Manly, NSW 1655, Australia, "JavaScript Usage Statistics," |April 01, 2010| (April 07, 2010). `http://trends.builtwith.com/javascript`.

[86] jQuery Team, *Homepage of jQuery UI*, (September 11, 2010). `http://jqueryui.com`.

[87] D. Baranovskiy, "Homepage of Raphaël." `http://raphaeljs.com/`, (September 11, 2010).

[88] J. Müller-Brockmann, *Grid systems in graphic design : a visual communication manual for graphic designers, typographers, and three dimensional designers = Raster Systeme für die visuelle Gestaltung : ein Handbuch für Grafiker, Typografen, und Ausstellungsgestalter*. Niederteufen [New York]: Verlag Arthur Niggli; Hastings House Publishers, |1981|.

[89] G. Ambrose and P. Harris, *Layout: Entwurf, Planung und Anordnung aller Elemente der Seitengestaltung*. Stiebner, |2007|.

[90] D. Dabner and A. Swann, *Design und Layout verstehen und anwenden : Prinzipien, Entscheidung, Umsetzung*. München: Stiebner, |2005|.

[91] Thompson Reuters, *Open Calais*, (September 10, 2010). `http://www.opencalais.com`.

[92] International Press Telecommunications Council (IPTC), *IPTC NewsCodes*, (September 10, 2010). `http://www.iptc.org/cms/site/index.html?channel=CH0103`.

[93] R. Martin, J. Türk, and M. Zmoreck, "Usability-Untersuchung des CUBA Prototypen," *Mensch-Computer-Interaktion (Seminar 46-02.302)*, |2009|.

[94] B. Shneiderman, *Designing the User Interface - Strategies for Effective Human-Computer Interaction*. Reading, MA: Addison-Wesley Publishing Company, 3rd ed., |1998|.

[95] I. Feldhaus, S. Nowitzke, and K. Seffzek, "Produktvergleich: "CUBA" und "Google news"," *Mensch-Computer-Interaktion (Seminar 46-02.302)*, |2009|.

[96] Google Inc., *Google news*, (March 23, 2010). `http://news.google.com/`.

[97] User Interface Design GmbH, "Was ist AttrakDiff?," (March 23, 2010). `http://www.attrakdiff.de/AttrakDiff/Was-ist-AttrakDiff/`.

[98] R. Allen, R. Lo, and S. Brown, *Zend Framework in Action*. Greenwich, CT 06830, USA: Manning Publications Co, e-book ed., |2008|.

List of Publications

Peer-reviewed publications

1. Sascha Kaufmann, Christoph Schommer (2010). "CUBA - Fostering Artificial Conviviality in a Web-based Feed System". In *International Journal for Digital Society (IJDS), Volume 1, Issue 2 (2010)*, pp. 133 - 142.

2. Sascha Kaufmann, Christoph Schommer (2010). "Towards e-Conviviality in Web-based Systems". In *Proceedings of the International Conference on Agents and Artificial Intelligence , ICAART 2010*, pp. 502 - 506.

3. Sascha Kaufmann, Christoph Schommer (2009). "e-Conviviality in Web Systems by the Wisdom of Crowds" (Extended Abstract). In *Proceedings of the 4th International Conference for Internet Technology and Secured Transactions, ICITST 2009*, pp. 444-445.

4. Thomas Ambrosi, Sascha Kaufmann (2009). "Finding Non-Obvious Profiles by Using Ant-Algorithms". In *Proceedings of the 5th International Conference on Web Information Systems and Technologies, WEBIST 2009)*, pp. 681-685.

5. Ralph Weires, Christoph Schommer, Sascha Kaufmann (2008). "SEREBIF - Search Engine Result Enhancement by Implicit Feedback". In *Proceedings of the 4th International Conference on Web Information Systems and Technologies, WEBIST 2008)*, pp. 263 - 266.

6. Natascha Hoebel, Sascha Kaufmann, Karsten Tolle, Roberto Zicari (2006). "The Design of Gugubarra 2.0: A Tool for Building and Managing Profiles of Web Users". In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, Intelligent Agent Technology and Data Mining (Web Intelligence 2006)*, pp. 317 - 320.

7. Natascha Hoebel, Sascha Kaufmann, Karsten Tolle, Roberto Zicari (2006). "The Gugubarra Project: Building and Evaluating User Profiles for Visitors of Web Sites". In *Proceedings of the First IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb 2006).*