WebTrust

Agent Platform for Identifying Trustworthy Commercial Websites

Master thesis by

W.F. Stoutjesdijk

August 2001

Delft University of Technology
Faculty of Information Technology and Systems
Department of Mediamatics, Knowledge-Based Systems

TU Delft
Prof. dr. H. Koppelaar
Drs. dr. L.J.M. Rothkrantz
Prof. dr. ir. E.J.H. Kerckhoffs

Accenture
M. Illsley, PhD.
R. Aragues, MSc.

Abstract

The vast amount of information on the Internet drives the Information Age: we want to know more about more. Unfortunately, the Internet does not guarantee the quality of this information. It is posted by newspapers, scientific institutes, salesmen and individuals. All of these people have their own interests, their own cultural and political background and their own objective for posting this information. Therefore, when gathering information from the Internet we have to be careful about which of these sources to use: we only want the *trusted* ones. And because we want to gather as much information as possible, an automated way of 'harvesting' it was called for. This application should be able to gather trusted information from all over the Web and label its quality in the sense of trustworthiness.
This thesis presents the blueprint for such an application including a prototype the can indicate trust levels for commercial websites. The prototype is based on the way people assess trust on websites. This was investigated and formalised into a model. Every user of this prototype has his own agent that has learned what its user finds important in trust. The agent examines the site and returns the trust value the user would have given. When this user wants to have the trust opinion of a friend or an expert he can use his agent too. All the agents will negotiate with each other on the outcomes and supply the user with one uniform result. In this thesis tests have been performed on the quality of these agents and the way in which they should be trained.

Contents

Preface

Part of the education to become a Master of Computer Science at Delft University of Technology is to perform, report and present a research project. This project - the thesis - is done in a complete academic year consisting of 10 months for the department in which the student wants to major. My major is in Knowledge Based Systems; the department headed by Prof. Koppelaar.

The work was done at the Accenture Technology Park, Sophia Antipolis, France. I worked in the Centre for Strategic Technology and Research (CSTaR) under the supervision of Martin Illsley.

The project - WebTrust - was done in collaboration with CSTaR and my supervisor from the university, Leon Rothkrantz.
I would like to thank my committee Prof. dr. H. Koppelaar, Prof. dr. ir. E.J.H. Kerckhoffs and most of all drs. dr. L.J.M. Rothkrantz for all their feedback. The distance made it difficult at times, but email and telephone helped overcome this problem.
At CSTaR my enthusiasm was constantly boosted by all the ideas that Martin Illsley, Neill Cameron and most of all, Ramon Aragues gave me during my stay there.
With them, I also thank the rest of the CSTaR department; I worked in a great environment with people that liked to share ideas, knowledge and a lot of fun.
Long before the start of the project, there were three people who helped me to setup the connection: Bart van Ginderen, Hans Borsboom and Pascal Pijpers. They got me there in the first place.

In the course of the project I did two surveys: I know that I took too much of their time, so the least I can do is thank all the respondents (Accenture personnel and students from Delft) for their efforts: Thanks!

Wouter Stoutjesdijk
Delft, August 2001

Chapter 1

Introduction

Every day again in life one has to make decisions on trust: do I trust this company to do my banking, do I trust this person not to spread around my phone number, do I trust the news in this newspaper to be unbiased?
Psychologists and marketing people have extensively studied this problem from their respective angles in the past. They have performed research on what aspects of the entities we take into account before (dis)trusting something or someone. With a bank, do we look at the woman behind the counter, the color of the carpet or the stock price before we open an account? When asking directions in a foreign country, tourist guides often advise to ask women instead of men, are the former better in directions or are they more likely to tell you the truth?
There are many criteria that potentially trustworthy things have to meet before we actually trust them in real-life and up to now they were all comparable (people, offices, clothing).
Recently however, a new phenomenon has entered the world of trust: the *Internet*. Here we do not see who is behind the counter. We do not know if it is a guy with a stocking over his head or one with a three piece suit. Still the world of Internet business grows every day: it seems as if we *do* trust at least some of the sites on the Web. But why? What do we mean when we say that a site looks trustworthy?

1.1 Problem description

In today's world everything can be bought on the Internet; from books to cars, from flowers to bat-detectors. Many organisations (like VeriSign and BBB Online) try to protect us from harm that could be done in doing these transactions: they make sure that information is transferred safely from your PC to the company and that this company is also the one you see on the website (and not somebody who made a look-alike). Trust as security.
However, in normal stores one does not just check for the quality of the lock on the cash register, or if the salesperson actually puts the money in it in the first place, instead of putting it in his pocket and running away with it.
Before buying one of their products you may ask a friend about certain aspects of it and look at the interior of the store. You may even go as far as to check how the shop is doing as a business, because you do not want spare parts to be unavailable later because the company has ceased to exist. Everybody does this in his own way.

On the Web these checks can be done as well, but because of its size it would be good if this could be done automatically.

The aim of WebTrust is to find an automated way to indicate how trustworthy a site is according to a certain user for a particular task and domain.

To achieve this, personalised agents will be used to investigate sites. Agents are taught what a trustworthy site looks like by their user. They have the ability to negotiate on their judgments with other agents and generate one common opinion. If the user does not know enough about the subject matter to determine its trustworthiness he can use an agent of a friend or consumer magazine to cooperate with his and together let them come up with sites that are trusted by all parties. The application that accompanies this thesis implements this agent framework and is shown in Figure 1.1, it will be referred to as the *prototype*.

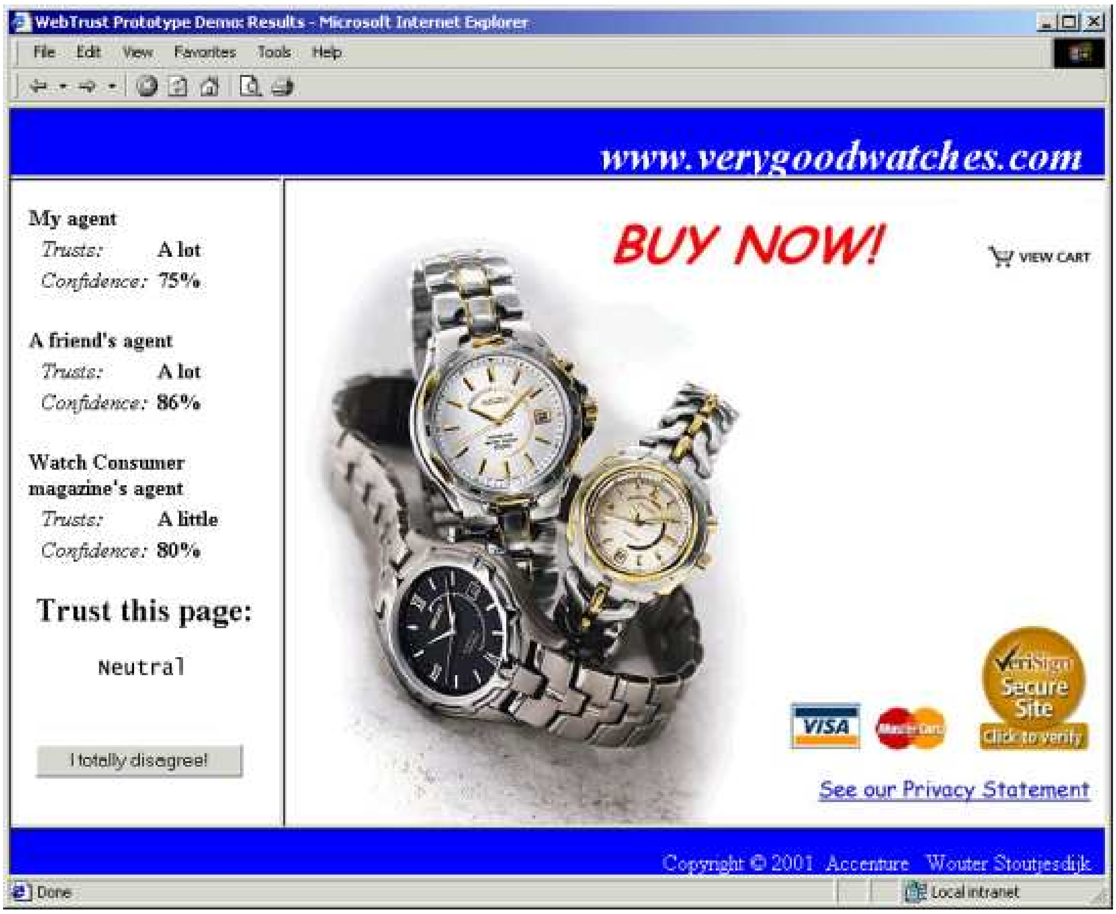

Figure 1.1: Webtrust prototype

1.2 Relevance

The great idea behind the implemented application is the fact that many applications nowadays and in the future (will) run on the Internet. They use data from this enormous resource to

generate information of all kinds. However, because of the heterogeneity of its sources, the quality of this data - and with that of the information - is questionable.
Even for the weather in one region, one can find multiple contradicting forecasts. A report generated from this data would not make a lot of sense unless an indication of its validity can be given: a measurement of trust. Some more of these applications can easily be found:

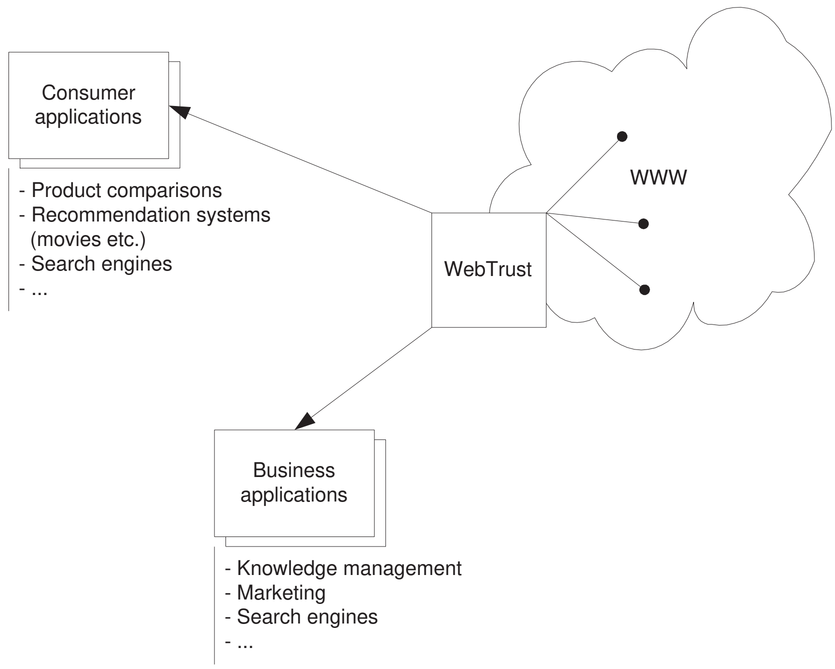

Figure 1.2: Applications of Webtrust

Knowledge management is involved in everything a company knows about everything. It manages information used for the whole business process. From design to sales, from staffing to long-term strategy. Lots of data needed to improve the quality of this information is freely available on the Web in the form of newspapers, stock-quotes etc. And with its emergence the ability to gather this data has grown exponentially and still does every day. When added to a corporate database for knowledge usage, the information needs to be trustworthy. Since there is so much information, automated solutions are called for the gathering process. These solutions need to be able to see the difference in trusted and untrusted information.

Marketing: For a product company it is important to keep track of the ideas that evolve around the products they make: what do the people think? A lot of this information is available on the Web today in the form of reviews like the ones for books on Amazon.com and can be very useful. Once again it is important to know who wrote these reviews and with what purposes to make a judgment on their validity.

Search Engines give us a lot of hits on a single search for a topic. A number of these sites is actually useful, but the majority is either unrelated or from dubious sources. An application running on top of a search engine could sort those sites in order of trustworthiness according to its user.

Product comparisons are a very popular way of using the transparency of the Web to get the best price for a product. If an application could gather all prices from online stores for a certain mobile phone, would it be wise to automatically pick the cheapest store from the list? Most people would like to know what store sells this phone before ordering: they need to trust it. This particular problem is addressed by the prototype built for this project.

In order for these applications to be automated successfully, they need to have a way of distinguishing between trusted and distrusted information. A task that can - at this moment - only be performed by humans.

This thesis presents an attempt to solve this problem by positioning an application between the Web and these applications as depicted in Figure 1.2 on the preceding page. Once this application knows what a trustworthy site is, it can give an indication on how much the person who trained it would trust the site: it is able to label sites according to trustworthiness.

The different applications mentioned above, show that trust can be seen from many angles and have different interpretations. When an online newspaper is known to give unbiased information, it does not imply that one can trust the opinions expressed in its columns to be unbiased too. The two would be measured in different ways: the country of the newspaper versus the political background of the columnist for example. To make this distinction, a model of trust angles was setup. It is shown in Figure 1.3.

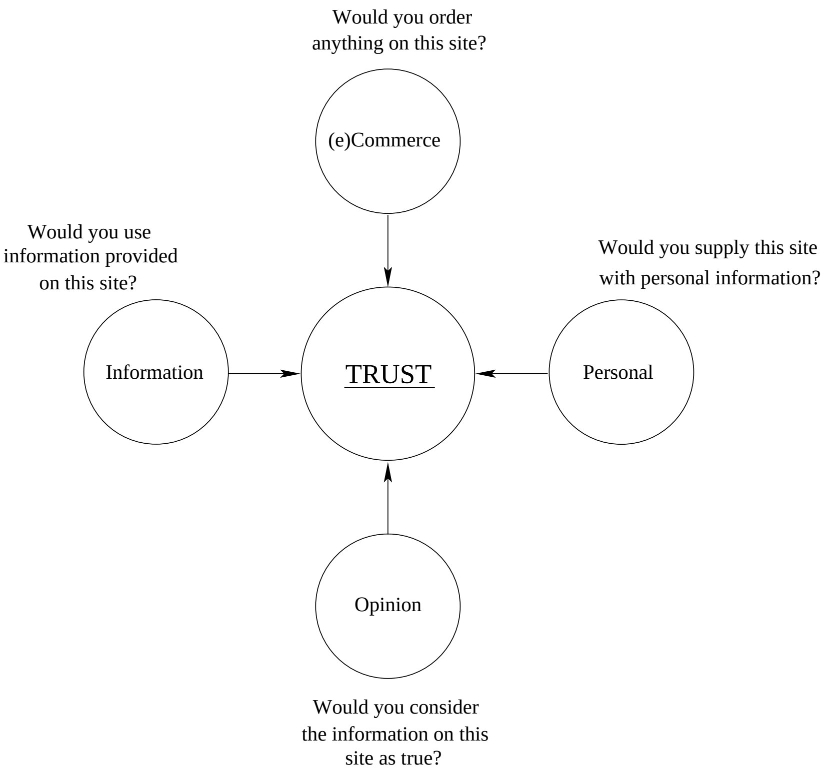

Figure 1.3: Different perspectives on trust

- eCommerce: would you buy anything on the site?
- Information: would you use information provided by the site for your purposes (e.g. research or cooking)?
- Opinion: would you consider information on the site as true (e.g. news or encyclopedia)?
- Personal: would you supply the site with personal information?

This model will be used later on to scope the project and to refer to as the different perspectives on trust.

1.3 Objectives

The objective for this project is to research trust with a focus on the Web, validate this research by surveying people and with all that, design and implement an application that can select trusted sites for its user.
The application will be a platform for agents that can give sites a trust factor on the basis of their user's behaviour. The agents can communicate with each other on these choices. This communication is a negotiation process that implements the idea of 'asking a friend for his opinion'. Before the agent can do anything it has to examine the site in the same way people would do this: by extracting certain properties from it.

More concrete, the objectives:

1. A literature research on the topics related to trust on the Web and similar research that may have been done already.
2. Validate the theory by surveying people on their trust behaviour on the Web.
3. Design a model for representing and measuring trust on the Web.
4. Identify and compare algorithms that can be used for learning user-profiles based on trust.
5. Design and implement a prototype application that incorporates this model of trust and all related aspects mentioned in this document.
6. Combine the results of the surveys and the prototype to generate test results on the quality of the approach taken.

1.4 Scope

When researching the topics related to trust, one has to keep in mind that the main angle of this work is Computer Science and not psychology. Some assumptions on this field will be made to facilitate progress, but will of course be backed up by literature whenever possible.

In Section 1.2 a point was made on the different angles of trust. For every angle different questions are asked. This is why it is useful to concentrate on only one of these angles. In this thesis only eCommerce trust will be addressed. Although the measurement of any of the angles will probably work in the same way, different world knowledge is required. The choice for eCommerce was largely based on interest from the market.

In Chapter 4 the Feature model will be introduced. It describes properties of sites that people take into account when assessing their trust. The main objective has been to identify these properties. Time constraints did not allow for a full implementation of all of them in the final *WebTrust application*. More will be explained in Chapter 5.
In Section 2.6 some comments will be made on the differences in behaviour a person may have in time: his moods. These will not be taken into account in the application nor the model.

1.5 Outline of the document

Chapter 2 will give a detailed overview of work that has been done concerning trust, trust in relation to the Web, applications that measure trust or similar phenomena and finally some theory on computer learning. The latter will provide an introduction to the 3 learning methods that are used in the application.

The theory will be tested by 2 surveys presented in Chapter 3. They deal with part of the theory, but were mainly used to build the Feature Model that is introduced in Chapter 4.
This model forms the core of the WebTrust project. It is used as the basis for the implementation of the prototype as described in Chapter 5.
Chapter 6 combines all previous chapters to gain insight in the quality of the approach. It presents the results of the tests performed. Finally in Chapter 7 conclusions will be drawn and recommendations for future work given.
This document contains two Appendices with the outcome of the two surveys. The second survey had too much data to put in this document. Its full conduct is printed in a separate report that accompanies this report.

Chapter 2

Background

A strong argument for the existence of trust in societies is the animal kingdom, in which animals that trust each other gain evolutionary advantages. A famous story is that of the bats: If a bat does not eat for a day, it dies. When returning to the cave in the morning, bats that were successful, feed the less successful bats. When feeding the other bat, it trusts *the other to do the same for him one day. Feeding each other increases the chances of survival and therefore makes the whole group stronger (Harcourt [13]).*

In this chapter the theoretical basis for WebTrust is laid out. This will be done by presenting an overview of literature and applications in the field. trust will be introduced followed by a more focused view of trusting websites. After this some applications will be highlighted that assess trust or related aspects. Finally the foundations of the 3 training methods used will be discussed.

2.1 Trust

Many people have tried to define trust; it is not easy to find one overall 'correct' definition. More people have come to this conclusion and have made an overview, like Marsh. In his 'Formalising trust as a Computational Concept' [18] he presents a broad view of trust that will be used here along with his additions and other sources.

2.1.1 Definitions of trust

Luhmann's explanation [16] for the fact that we trust in the first place, is that without trust the world would be too complex to perceive. An example illustrates this:
When I promise a friend to bring him some bread from the bakery, I trust that I can reach it: the road is not closed, I do not have a flat tire. That it is open and not closed because of a sudden illness and that he still has some bread left. If I were to consider all of these things before being able to make the promise I would have to think about too many trivial issues.
I trust that the world acts somewhat continuously and therefore like other days.
Luhmann's definition of trust brings it back to a daily issue of *reducing complexity* of life.

Luhmann continues his work on trust in [17] where he claims that "trust ... presupposes a situation of risk" (page 97). When taking a book from the shelf we do not expect the shelf to fall on top of us just like that. We trust it to stay in place if we are careful enough. Although this is a real risk (measured in the result it can have), it is just not very likely to happen and therefore we accept this risk by trusting.

Having said this, the *definition of trust* by Deutsch [7, page 303], which is commonly referred to, follows from Luhmann's almost automatically (although the work was done earlier):

> 1. *the individual is confronted with an ambiguous path, a path that can lead to an event perceived to be beneficial (Va^+) or to an event perceived to be harmful (Va^-);*
> 2. *he perceives that the occurrence of Va^+ or Va^- is contingent on the behavior of another person; and*
> 3. *he perceives the strength of Va^- to be greater than the strength of Va^+.*
>
> *If he chooses to take an ambiguous path with such properties, I shall say he makes a trusting choice; if he chooses not to take the path, he makes a distrustful choice.*

This formal definition is easily applied to trust on the Web, which gives it more clarity:

Va^- stands for one's credit-card number being abused, information being spread around, receiving spam mail or not receiving what you ordered;

Va^+ is typically the lower price one gets using the Web, or the ease of purchasing a product without having to go to the store etc.;

The other person is the person, company or organisation behind the website;

The choice one has to make is to order, give information (address, email) or to leave the site for what it is. Assuming the person finds Va^- worse than Va^+ - which is often the case - he trusts the website when he orders or leaves information.

Several properties of trust can be derived from this definition. These properties will influence the models one can build on trust; they make up some of the constraints and assumptions:

- The individual *must* have a choice to either choose the path or not. If he has to take it without having this choice there is no trust involved. It would be more like hope: "I had to order on this site, because the book was out of stock in the store and needed it for a presentation[1]".
- He cannot influence the situation other than choosing or not choosing the path: the context is a given fact and is depending on the other person (or organisation).
- Since risk is the product of probability and impact, points 1 and 3 in Deutsch's definition lead to the actual risk involved as described by Luhmann. This is assuming that there is some probability in the occurrence of the paths in 1.
- Deutsch explicitly uses the word '*perceive*', which strongly suggests that trust is subjective. This concept will come back in Marsh's model as well.

This section illustrates the fact that there are many different definitions of trust. Gerck even mentions 30 synonyms in his work [11]. Deutsch's definition will be referred to as *the* definition of trust in this document.

[1] I thank a colleague for ordering a book on a German site, while not actually understanding the German language

2.1.2 Layers of trust

The definitions mentioned in the previous section can be used as foundations to build a formal model of measuring trust. Unfortunately this was rarely done: many of these definitions merely describe trust and do not mention anything on *how much* or *how*.

Marsh however, takes trust to the next level by mathematically describing his model to support his agent society (ibid., Chapter 4). His work aims to design an agent world in which agents cooperate with each other or not, on the basis of their mutual trust. To achieve this he had to make a more unified model of trust, which was up to then 'lacking' (ibid., page 53).
His model goes into too much detail to be described in this work, but leads up to the definition of several layers of trust. They give an indication of how daily-life trust can be approached from a formal point of view. The different layers of Trust will be discussed below.

Basic trust is inherent in the trusting person. One can be an easy truster or a hard one. As in more things in life we can distinguish between optimists and pessimists:
"...optimists are general trusters whose disposition to trust is relatively inflexible in a downward direction, despite past experience. Thus their trust in others can only increase, never decrease. The opposite is true for pessimists" (page 56).
It is the way they look at life (from a trust perspective). They are gullible, suspicious or something in between.

General trust is measured from one subject to another. In other words how much person x trusts person y. It is called General trust because situational or temporal issues are not taken into account (see the next point).
In a customer-company relation this is their mutual trust. It is calculated regardless of the medium (shop, Internet or telephone). It is purely what the customer thinks of the company: how well he can trust it.

Situational trust is the most specific of the three and describes trust in a very specific context. The example Marsh gives is very clear: "while I would trust my brother to drive me to the airport, I most certainly would not trust him to fly the plane!".
In other words you have general trust, but not in every situation. For trust on the Web the site comes into the picture. Although you thought you trusted this company, their site or shop has ruined this perception.

Further on temporal issues are also discussed. Situations do not include time. This is done to be able to compare them independently of time (unfortunately, every day I go to the bakery the situation is approximately the same). Time issues will not be discussed any further, the reader is referred to Marsh for more detail.

In a real website situation one can analyse a trust decision: The Situational trust is based on the environment (the site) and the user's General trust. This General trust is - in its turn - influenced by the company and the user's Basic trust. Basic trust is inherent to the user and can be seen as the way in which he deals with past experiences. When he is pessimistic, he will only tend to 'remember' the bad experiences and the opposite for an optimist.

Marsh claims that the decision to trust is based on a threshold. If trust is higher than this threshold, the situation is 'trusted'. This threshold is influenced by (a.o.) the *perceived risk* and the *importance* of the matter. When buying a car on the Internet for example the risk is very high, people will be less likely to trust the site than with cheap products.

As for the importance, time constraints can influence your decision: when the importance is very high, the person does not actually have a choice anymore. Deutsch would no longer consider this as a trust decision, since the individual was not allowed a choice, but Marsh merely adjusts a factor in his threshold.

Risk and importance will not be taken into account in the measurements since they depend too much on the person in a specific situation.

2.2 Trusting websites

Some research has already been done in the field of trusting websites. Although most of it focuses on technical security assuming that sites using proper encryption can be trusted. In this section, some of these attempts will be presented. Some of them are not within the scope of eCommerce, but provide useful examples for a general approach.

The first category of people that took another angle were *librarians*. They needed guidelines for students to find quality background information for their research. Their work consists of questionnaires that treat subjects that add up to a trusted site.
In her 'ABC of WebSite Evaluation' [23] Schrock mentions (a.o.) the following:

- Authority: is the institute and or writer an authority in this field?
- Bias: is the researched sponsored? By whom?
- Citations: can additional information be found easily?

The answers to these questions enable the student to assess how well he can trust the information posted on the site. The bias question for example, is important when interpreting results: does the writer of this work have any benefit from producing a certain output?
With these guidelines she tries to make sure that people find trustworthy information on the Web to use as references. The university libraries of Indiana [3], UCLA [12] and Berkeley [1] university did the same:

- Are the links on the site evaluated?
- Is multimedia appropriately incorporated?
- Is there a non-Web equivalent version of this material?
- Does it have a statement telling that this is approved by the institute?
- Does the institute appear to filter information it is associated with?

Again these guides give hints on what to look for in a page that will be used for scientific purposes, to make sure that it is a valuable addition.
From a commercial angle, Cheskin provided information on commercial sites in their 'eCommerce trust Study' [5]. In this large-scale survey among Internet users from the Americas, they try to find out what elements communicate trust in eCommerce sites.
They asked participants what they found important on a site. Their main result is a list of six components that "communicate trust" (page 9):

- Seals of Approval: Symbols like VeriSign and Visa, designed to re-assure the visitor that security has been established or the transaction insured. The companies that provide these seals of approval are referred to in this document as $3rd$-parties and treated in more detail later on.

- Brand: The corporation's promise to deliver specific attributes and its credibility based on reputation and the visitor's possible previous experience.
- Ease of navigation: The ease of finding what the visitor seeks.
- Fulfillment: Clearly indicates how orders will be processed and provides information on how to seek recourse if there are problems.
- Presentation: Design attributes that connotate quality and professionalism.
- Technology: State of the art connotates professionalism, even if it is difficult to use.

There are also people who *make* trustworthy sites. They do not focus on finding them, but advise companies how to build sites that look trustworthy.
In his model of 'trust in eCommerce' [8] & [9] Egger makes companies ask their clients the following questions:

- What is our reputation?
- What information on us, would you like to see on the site?
- Which 3^{rd}-party would you see as trustworthy?
- What details would you like to have in the Privacy Policy?
- What customer service and guarantees do you expect?

In a sense, Egger makes customers of these companies rate the trustworthiness of their site and give information on how to improve it. This can be seen as unfair since the company itself does not change: it does not become more reliable. In terms of Marsh's model this means that General trust will not change, but Situational trust will. The site communicates more trust. Evaluating trust by means of these questions will be used as the basis for the *Feature Model* presented in Chapter 4.

2.3 Negotiating trust

A totally different approach to assessing trust comes from consumers themselves. They review sites or products and read each other's comments on that. From a trust perspective this makes a lot of sense, since one can consider a community like this as being 'amongst friends'. It addresses a fundamental issue in trust: advise on whether or not to trust something can only be trusted if one trusts the advisor.
The 3^{rd}-parties mentioned earlier for example, only have a positive influence on people when they trust this system of *recurrent trust*. In the consumer case it is clear that users trust each other more (although mediated through commercial sites) than reviews from companies that may benefit from this.
A good example eOpinions[2]. Users give their advice on sites and products. Every user in this system is part of the *Web of trust*: they all have a portfolio of users that they trust. These users are considered to be their friends that may share interests, but most of all have users in *their* portfolio that are trusted too. With this users have some kind reputation in the community. Their *Web of trust* is based on the following assumptions:

- Friends have a proven track-record. If a friend consistently gives you good advice, you're likely to believe that person's suggestions in the future.

[2] http://www.eopinions.com

- You know which preferences you and your friend share. If you both like the same types of films, you're more likely to trust your friend's recommendations on what to see.

Every user can build his Web of trust by adding reviewers to his portfolio. They are selected according to their history of reviews and *their* portfolio of reviewers, that you can also evaluate. The other way around people will select you as a reviewer because of your portfolio as well (if you wish to participate).
In this way, the eOpinions community gives trustworthy advise to its members. This insight is used in the prototype and will be explained in the Chapter 5.

2.4 Applications for trusting

The next step after knowing what trusted sites are, is making applications that can use this concept to automatically label sites. There are several levels of complexity of these applications. Some people have merely defined lists of sites, whereas others have made applications in another area that came very close to the prototype. In this section an overview of this work will be given. Every part discusses applications that are related to an aspect of WebTrust, starting from basic to advanced.

2.4.1 A static list

Some applications have a list incorporated in them that has all trusted sites on it. A good example of this is the Security feature in the Internet Explorer; one can put sites in a database that one considers trustworthy.
This is a lot of work, but a very solid approach: you will be sure that these sites are actually the only ones marked as trusted and all the others will not. Especially when you explicitly Distrust any sites that are not in the database. This does, of course, limit the horizon to a great extend.

2.4.2 Properties of a site

One step higher than a list, are the *3^{rd}-party seals* displayed on pages. 3^{rd}-party seals are signs from 'trusted authorities' that have guidelines for implementing security and doing business (to some extend). Security is their most well-known use.
When clicked upon, the seal should bring the user to its site and explain some details on the company where it originated from. The trusted authorities use an elaborate scheme of security measures to make sure that the seal actually authenticates.
Sites have to apply for a seal like this themselves. According to Cheskin [5] this pays off as well: many people know what these seals mean and consider many of them as trusted.
Seals like the ones from TRUSTe [24] and VeriSign [25] are mainly used in commercial and privacy applications.
The fundamental problem with this approach is the fact that we have to trust the 3^{rd}-party before we can trust its opinion. In his 'Towards Real-World Models of trust' [11, page 4] Gerck says that only the 'self' can have full trust:

> "If there is no communication channel from the outside system, then the system views itself as isolated and it is not possible for the system to have any trust besides self-trust [...] The outside world may however receive communications from the system – which can allow the outside world to have trust in regard to an isolated system"

In other words: The trusted parties may consider themselves fully trusted, but we will never know for sure.

Another approach has been, to look at the *technical quality* of the site: the quality of the HTML code behind it. This measurement is based on the strange fact that browsers allow the code of a site not to follow the exact specifications of HTML.
Applications like Linkbot [14], WebSiteGarage [10], NetMechanic [20] and the WebPage Builder [4] look at how well the code is written. They measure the quality of the HTML to infer the quality of the site. The utilities presented are most useful for owners of sites and tell something about their level of professionalism. When a site is developed in a proper program for site development, the HTML code is expected to be proper as well, whereas free-style editing can lead to inconsistencies. This does not say anything about trust directly, but contributes to it as will be shown in Chapter 4.

2.4.3 Getting to know the user

Marsh already showed that trust is very personal, it depends on the intrinsic trust a person has, on his trust towards other people and on the situation. An interesting approach is therefore presented by researchers that have tried to learn what the behaviour of a person is and to create his 'profile' based on that: *user profiling*. Some approaches to user profiling will be discussed in this section. They do not involve trust, because no work has been done in this area. In the next section these approaches will be combined with the concept of trust.

A lot of research in user-profiling has gone into *interest*: learning the interests of a user. Although this field is not the same as trust, a lot of things can be learnt from it and therefore some of its work is presented in this section.

Pazzani has performed a lot of work on agents that learn user profiles. Most of them however, are site-based: they learn from general usage of a site and adapt to a single user to refine. On his site (see the Reference chapter), Pazzani shows that this principle works very well for his collection of publications. Once the abstract of a publication is read or the whole article is downloaded, the agent will start recommending other publications that may interest the reader as well. This problem is of course very restricted to a few (although still many) publications with clearly defined relations between them, but it is a nice application. Unfortunately this agent only works on one site.

There is one publication that does it the other way around and is highly related to this work (and also by Pazzani): 'Syskill & Webert' [21] is an agent that categorises documents according to user preferences. A user indicates which kind of documents he likes by giving them a rating on an index page. This is a page with many references to the different subjects in its category, like the AI subject index list[3].
The documents referenced by this index are crunched to get the n most common and most differentiating words. These words will be used in training. Whenever a document is presented to the agent it will look for the keywords mentioned before and categorize the document accordingly.
The paper itself is a study on which technique to use for categorising. They conclude that a Bayesian classifier performs best. For the WebTrust project different methods have been tried as well. They are presented later on in this Chapter. Syskill & Webert could easily be used for other kinds of user-profiling tasks where only content was used to select. The question is now if

[3]http://ai.iit.nrc.ca/subjects/ai_subjects.html

content is a major consideration for trust. Reality will have to prove this in Chapter 3. If this is the case Syskill & Webert may also be suitable for doing the job.
Some work has been done on improving these results. Billsus and Pazzani [2] enhance their learning by adding prior knowledge to their agent. Before the agent will be used, the user has to type in words that he finds 'hot' and words that he finds 'cold'. In this way, the system has an initial idea on the user's interest. After this, learning will proceed as normal, but it will be a better start.

Krulwich does the same in his 'LifeStyle Finder' [15]. He tries to find interesting products for a user based on his interests. Their application elicits the user on some basic questions and categorises him. This categorisation uses data on interests of Americans. This data is abundant and easily accessible, as opposed to trust preferences where no data is available.

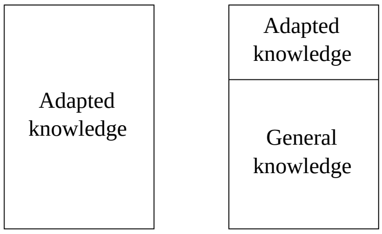

Figure 2.1: Learning about a user from scratch or using predefined information can make a difference in learning rate.

These basic start profiles would be filled with 'general' information like statistics (in Krulwich's case) or by explicit elicitation from the user (Pazzani) as depicted in 2.1. For trust this information is not available. The survey presented in Chapter 3 can be used as a basis, but a large-scale investigation with more participants would be required to obtain full knowledge.

2.5 Learning what a user trusts

People regard trust differently from one person to another. Cheskin suggests in the follow-up of their '99 studies [6] that examining features is culturally and geographically different. People from the Latin American countries regarded different issues important than people from the North: In Latin America, people looked for 3^{rd}-party seals like VeriSign, whereas in the north they looked more for credit card seals. A similar difference could be found for Spanish speakers and English ones: the word "Veri" in VeriSign made so much sense to Spanish ("truth"), that it was more successful in their countries than in the US. Cheskin's report also mentions a generation gap in taking risk. Younger people seemed to take more risk on the Web to gain their advantages than the elderly. In other words: one needs to get to know the user before his trust assessment can be predicted.

The prototype presented in Chapter 5 will therefore incorporate user profiling to learn about the preferences of a specific user. To learn, it will present the user with a site, ask him for his judgment and store it. The pattern of sites and judgments will be the profile of this user.

After this profile has been sufficiently filled with examples, it can be used to *infer* ratings for new sites presented to the system. The field of Artificial Intelligence provides many ways for doing this and it is important to choose the right one for the task: every method has different properties that influence the speed and accuracy of the final implementation. Three different methods will be discussed below along with their properties. After this a comparison will be made.

2.5.1 Artificial Neural Network (ANN)

ANNs are supposed to be modeled after - a simplification of - the workings of the human brain. This is also the reason why they are often used in 'human behaviour' applications.

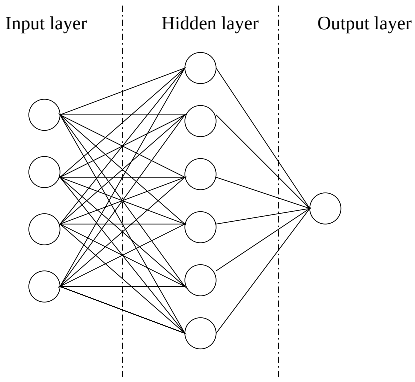

Figure 2.2: A feed-forward neural network

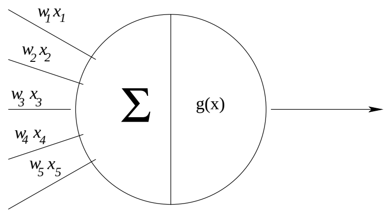

Figure 2.3: A Neuron

An ANN consists of a network of *neurons* that have 1 or more inputs and outputs. Each input is assigned a *weight*. In feed-forward Neural Networks the neurons are placed in *layers* as depicted in Figure 2.2. The first layer - the input layer - has the same number of neurons as the dimension of the problem. On the other end, the output layer has the same number of neurons as the dimension of the answer.
Each neuron adds up all the incoming inputs multiplied by their weight, the summation is passed through the *activation function* which will determine the output value of the neuron (see Figure 2.3). In the training phase, input data is presented to the network. It progresses through its neurons and finally there is an outcome in the outer layer. This outcome is compared with the correct (true) output, this gives the *error*. When using *Backpropagation*[4] the weights of all the neurons will - in reverse direction - be adjusted according to this error.
After this, more train-data is put into the network and so on, until the error has reached a suitable level: the network has been trained. This same network can now be used to predict the outcome of new data: it *generalizes* its knowledge. ANNs are - in this case - used as a regression method. The input is mapped onto a real number, which will then be rounded to the nearest label. See Russel & Norvig [22, p. 567-587] for a more complete overview.

Properties of ANNs

ANNs provide a good method for learning from *large amounts* of data by examples. With many applications this is not a negative property: there is a lot of data available. With trust however, no such data is available since it is a subjective matter: it is different for everyone. The question is now how fast the network can learn from the examples it sees.

[4]There are also other methods for training the network, but they will not be described here.

Another problem with too little data is that the phenomenon *overtraining* can easily occur. When this happens the network has 'concentrated' too much on the data it was presented with and is unable to generalize. ANNs do not require any *knowledge* of the internal workings of the process that is modeled.

2.5.2 k-Nearest Neighbour (k-NN)

This method looks at the *distance* between data points in its data-set. The answer given to a new point will depend upon the answers of its *neighbouring* points. The distance between these points can be measured in many ways. An Euclidean measurement will be used here.
The number of neighbours taken into account depends on the size of the dataset. A method for choosing this number is presented below.

In the example shown in Figure 2.4 a data point can either be black or white. When a new point is entered (the cross in the picture), the system looks at the colour of the surrounding dots.

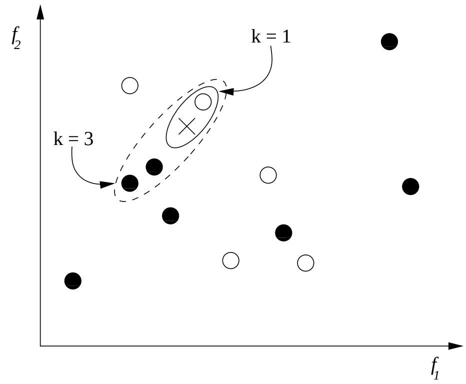

Figure 2.4: The k-NN method decides based upon datapoints in its 'neighbourhood'

In the simplest case, when $k = 1$ the *nearest* answer will be given (white). Although it is not obvious from the picture what the 'good' answer should be, it looks sensible to compare more sites. When k is increased to 3, it looks at 3 sites and will pick the answer that is most frequent (black). With more 'most frequent' groups, the one with its center of gravity closest will be selected.

For a proper choice of k the size of the data-set should be taken into account. The larger the set, the higher k. Heuristics (taken from Mitchell [19]) advise it to be $\lfloor c\sqrt{N} \rfloor$, where N is the site of the set and c is a constant (which is set to 1 in the implementation). For this example the right k would therefore be $\lfloor \sqrt{12} \rfloor = 3$.

Properties of k-NNs

The k-NN uses a different paradigm for inferring results from its data than ANN. It is a *lazy* learning algorithm (ibid., p. 244): it gradually builds up knowledge and can take decisions based on any number of examples. It does not try to find patterns like the ANN does, which is *eager*: it generalises from its data and gives output according to this internal 'model'.
This difference plays an important role in the amount of data both of them need to make a 'proper' judgment. The k-NN can be relatively fast in learning, but the ANNs generally tend

to perform better with a lot of data. This means that with only a few sites in its set, k-NN is supposed to come up with a reasonable answer already, whereas ANN does not yet have a clue.

2.5.3 Decision Tree

The third approach that was investigated was the Decision Tree. Different from the former two, this method simulates the way people make decisions by reasoning at every step. When buying a car for example, we can take 3 things into account: if it is affordable, the right type and then what colour it has. Seven cars are presented in Table 2.1 together with the decision the customer made (buy or leave it).

Table 2.1: Cars in the data set of the Decision Tree

Name	*Expensive?*	*5-Door?*	*Available in Red?*	*Buy it?*
Audi TT	YES	NO	NO	Leave
VW Golf	YES	YES	NO	Leave
BMW 5-series	YES	YES	YES	Leave
Renault Espace	YES	YES	YES	Leave
Renault Twingo	NO	NO	NO	Buy
Fiat Panda	NO	YES	YES	Buy
Peugeot 205	NO	NO	YES	Leave

From this data set, rules are extracted for the assumed way in which the customer works. From the 'Expensive' column, it is clear that when a car is too expensive, the client does not buy. When it is not, no clear decision can be made yet. This column is very *distinguishing* for the final answer. The algorithm to decide which column is the most distinguishing works by counting the number of measurement-answer pairs (taken from Russel & Norvig [22, p. 534-537]). The number of times Expensive was YES and the customer did NOT buy is 4. The same can be done for all other combinations and columns. The assumption of the Decision Tree is that the most distinguishing properties should be first in the tree followed by the second and the third. The tree of this decision is depicted in Figure 2.5 on the following page.

The process of advising on whether or not to buy a certain car will now be based on this tree. The algorithm for doing this is depicted in the flowchart in Figure 2.6.
In it a 'decide' function is mentioned that will give one answer based on a set of answers. It uses the following rules (in this order):

1. if all the answers in the set are the same, return that answer; else
2. if there is one answer that was most frequently given (largest subset), return it; else
3. return the average answer.

If the algorithm has to travel all the way down the tree and still finds answers, it means that there are cars that are *similar*. They are not necessarily the same, but have the same labels. In the car example this is true for the BMW and the Renault Espace. When nothing is found, backtracking needs to be done to decide on a less detailed level. Again it is good that the tree follows the person in his thinking. When the car is cheap and 5-door, the colour may not matter all that much anymore. The other way around however, does not make sense: when we see that it is red and has five doors, we still want to know the price before deciding.

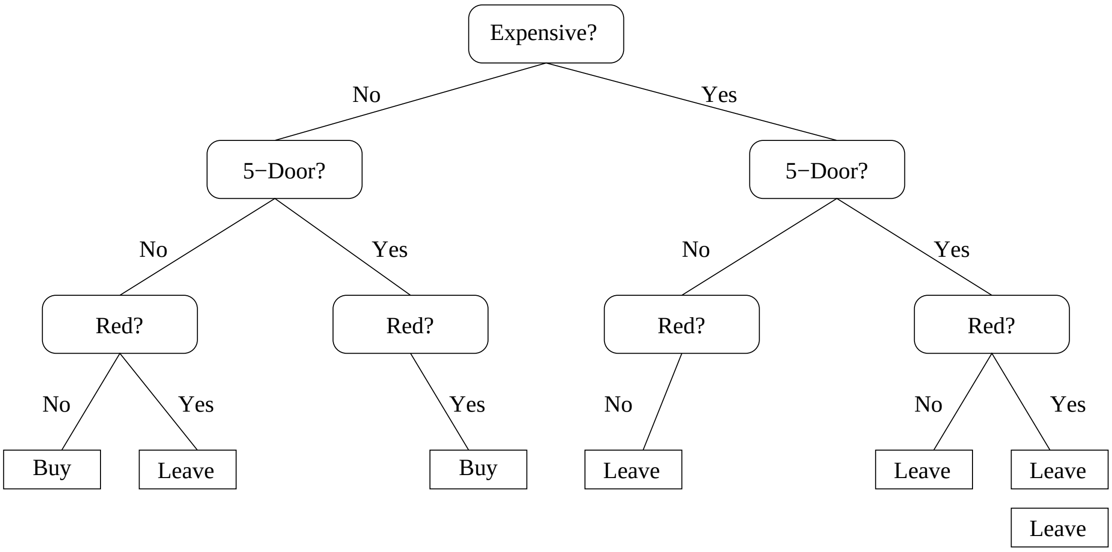

Figure 2.5: A decision tree that models buying a car

Properties of Decision Trees

Decision Trees use a rule-based approach that they extract from the data set. This can be very good if rules are really used in the system that is modeled. Unfortunately this is also a weakness in the algorithm: it can easily mistake coincidences in the set for rules of the system. When one of the cars, for example would be expensive and the customer bought it, the whole tree would be different. It could be coincidence that the price coincides with the decision. Only with a larger data set will these things become more obvious.

This was a boolean example with limited measurements, but when more complex systems are measured the size of the tree grows rapidly. This means that it will often be very sparse because there are not enough data points to fill every branch. This could mean that answers are suggested based on only a few leafs of the tree.

Finally it can be said that this method gives an insight in how the decisions are made. The others (ANN and k-NN) take a mathematical approach, while this one really tries to model the decision process.

2.5.4 Comparison of the methods

The 3 methods discussed above all have their good and bad characteristics. One may work in a situation where the others do not perform at all while the same method may fail to work later on in a different situation.
To summarise this overview a table is given that compares the methods from different angles.

All the ratings given here are for generalised cases, not for any of the examples mentioned earlier or for the application of trust. These comparisons will be made later on in the Results chapter.

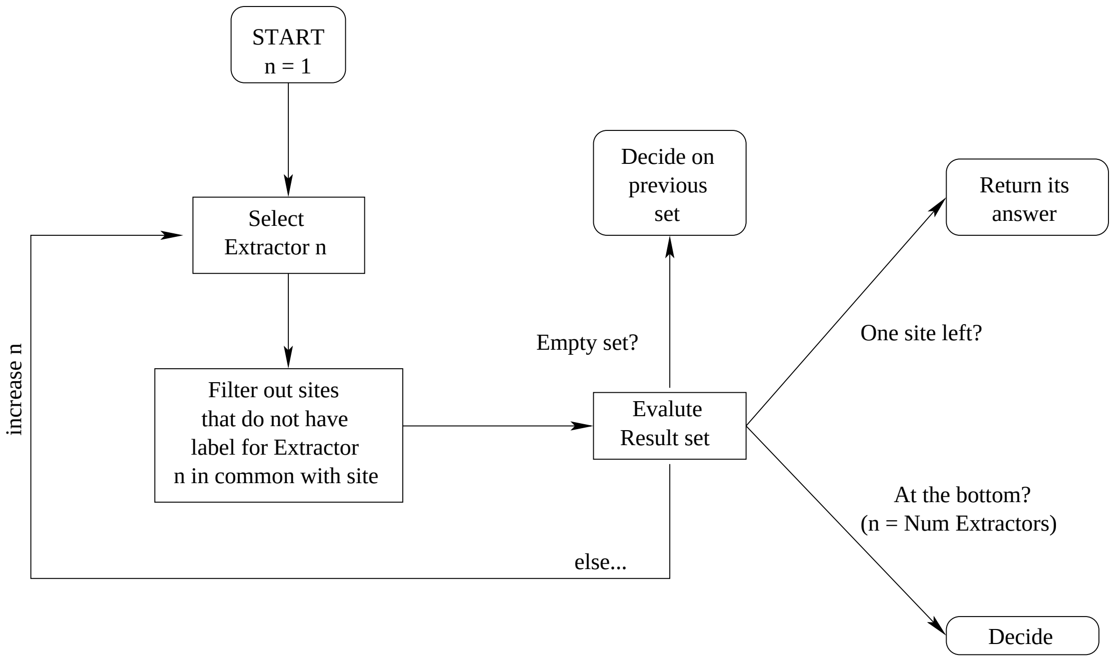

Figure 2.6: Flowchart describing the process of going down the Decision Tree

2.6 Moods & other inconsistencies

The methods discussed above are capable of learning human behaviour with respect to trust. They have difficulty with the fact that humans are inconsistent in this behaviour. These inconsistencies are mostly due to different moods or environmental issues. When, for example, a person has recently read about a major credit card scam he may be more suspicious than normal: the application does not know this of course and will remain scoring according to the old patterns. General changes of mood, time of the day and physical state (especially fatigue) will certainly influence trust behaviour as well. All these matters will not be taken into account in the application, nor in the model because they are, by definition, inconsistent. The application will give an answer based on the user's average behaviour and the user has to evaluate if he agrees with that or not at that particular moment in time.

Table 2.2: Comparison of methods

Criterium	*ANN*	*k-NN*	*TREE*
Performance with little data	Low	Medium	Low
Performance with lots of data	High	Medium	Medium
Sensitive to bad training examples	Little	Little	Yes
Compuational complexity (training)	High	Low	Low
Compuational complexity (runtime)	Low	Low	Low
Gives insight in decision	No	Little	Yes
Uses model knowledge	No	Little	Yes

Chapter 3

Theory in practice

In the previous chapter the theoretical basis for trust and the Web was laid out. In this chapter this theory is tested against reality and the specific case of WebTrust by means of two surveys. The two surveys will be described here together with their main results and conclusions drawn from them. A full conduct can be found in the appendices A & B and in the report that accompanies this document.

3.1 Initial survey

This survey was held in the beginning stages of the project. It was based on the initial ideas of trust and gave direction to defining the final ones, amongst which was the Feature Model presented in Chapter 4. In addition to this, it helped to scope the project towards commercial sites: it became clear that the 4 categories mentioned in Section 1.4 (Commerce, Information, Opinion and Personal) were too different to be put into one project.
The main purpose of the survey was to answer the following questions:

1. Is trust subjective?
2. What scale can be used to measure trust?
3. What do people look at when assessing trust of a site?

10 candidates from both Europe and the United States replied to the survey. Because of this low number no statistical importance is given to its outcome, it is purely indicative.

3.1.1 Design

Respondents were asked to score pages, give a ranking on some reasons *why* they gave these scores and what they looked for in the page. Of course there was also room for general comments. The scoring was done in four different parts; angles to look at the trustworthiness of a page as illustrated in Figure 1.3 on page 4. They were asked if they would:

- buy anything on the site;
- use information provided by the site for their purposes (e.g. research or cooking);
- consider information on the site as true (e.g. news or encyclopedia);
- supply the site with personal information.

Every part had 5 different sites, in total there were 20. After this, the respondent had to rank three 'drivers' according to their importance: "First impression", "Experience with the product" and "Information from friends and trusted parties". These drivers capture the order in which the person evaluates the site.

The survey was designed in HTML and used CGI to process the results. The full survey and its results are given in Appendix A.

3.1.2 Summary of results

The numerical results of the survey are shown in Table 3.1. For every site (a row) all ratings per respondent are displayed together with the angle of trust that was involved. In the information category, for example, one can find a scientific journal and an encyclopedia. The question asked for these sites were different from commercial ones for example.
Although the original ratings were in words (Not at all, Neutral and A lot) they have been transformed into numbers here for clarity.

Table 3.1: Site ratings

Site	*Respondent*										
	1	2	3	4	5	6	7	8	9	10	Category
Door to Door	3	2	3	2	3	3	3	2	2	2	Commerce
Launchpad 2000	2	3	2	2	3	2	3	2	2	3	
Silverbank	2	1	1	3	3	1	3	1	1	1	
Amazon	3	3	3	3	3	3	3	3	3	3	
Spacepen	2	2	2	2	3	2	3	1	1	2	
Encyclozine	2	1	-	1	3	2	3	3	2	2	Information
City Journal	2	1	-	1	3	1	3	2	2	3	
Radiocarbon	3	1	-	3	3	2	3	3	1	2	
Encarta Encyclopedia	3	2	-	3	3	3	3	3	3	3	
Bushido online	3	2	-	2	3	2	2	1	2	2	
Herald Tribune	3	2	3	-	3	3	3	3	3	3	Opinion
PeaceWatch	2	2	2	-	2	1	2	2	1	3	
Dauten, columnist	1	2	2	-	3	1	2	1	2	2	
Database on World Happiness	1	2	2	-	3	2	2	1	1	3	
Earthwatch Weather	3	2	2	-	3	1	3	2	2	3	
eBay	2	3	2	1	3	2	3	3	2	1	Personal
Killer roses	2	2	2	1	3	1	3	1	1	2	
XOIP	2	3	2	1	3	1	2	1	1	2	
FlowerHut	2	2	2	1	3	1	1	2	1	1	
ePoll	2	2	2	1	3	2	1	2	1	3	
1 = Not at all, 2 = Neutral, 3 = A lot											

Site ratings differ per site

Looking at the rows of Table 3.1 the following things can be noticed:

1. There are only 3 rows where zero or one answer is different from the rest (would suggest agreement) and they are all well-known sites.

2. 15 rows have all possible answers in them (suggests disagreement).
3. If the agreement would be defined as the number of times the most frequent answer is given divided by the number of answers given, there are only 10 sites that score more than 50%.

As mentioned before, this survey does not hold statistically, but it gives a good indication that site ratings can be very different per person. In addition to this it can be said that apart from the site itself these differences have two causes: the person rating it (his Basic Trust) and inconsistencies because of mood or fatigue. In this configuration of the survey it is not possible to examine these parts independently.

Table 3.2: Driver rankings

Driver	*Respondent*									
	1	2	3	4	5	6	7	8	9	10
First impression	1	1	2	3	3	2	2	3	1	3
Experience with product	3	3	3	3	0	3	3	2	3	1
Information from friends	2	2	1	3	0	1	1	1	2	2
0 = Lowest rank, 3 = Highest rank										

Drivers differ per person

When asked what they *did* when making up their assessment of a site, they sorted the three 'Drivers' depicted in Table 3.2 in order of importance. This means that they would look first at the most important one and then go down in this list.
When you add up all the rankings per driver (row) you get the overall ranking of it: the number of points all the respondents together gave it. The higher the score, the higher the average ranking. It is clear that some of these drivers were more important than others

1. First impression (total ranking of 24)
2. Experience with the product (total ranking of 21)
3. Information from friends (total ranking of 15)

The concept of order is captured by a decision tree, one of the methods uses in Chapter 5.

People look at many properties of a site when assessing trust

The respondents gave many different reasons for their page scores. They looked at the site or related issues (other organisations etc.) to make up their minds. Some of them are mentioned below:

- Do I know the company already or do I have experience with their products in either real or Web life?
- Does the page look good and provide clear, quality information?
- Does the site mention a privacy policy?
- Does the site provide any security or credit card seals?

- Do they have a proper URL (is it a www.company.com name or some ISP with a directory?)
- What country are they from (if that is clear) or what type (.net, .org or .com)?
- Performance of the site; does it work at all?
- Business model: what do they actually make money on?
- Do they have a real-life company as well?

These '*features*' of sites will be used later on in the Feature Model.

A three point scale does not give people enough space to answer

Some of the respondents indicated that the gaps between the ratings are too big. They have the choice between "Not at all", "Neutral" and "A lot". In Marsh ([18]) a continuous scale is used, which is good for internal workings but not for feedback from people.

3.1.3 Conclusions

- The differences in ratings per site suggest that trust may indeed be subjective as indicated by the theory: different people trust the same site differently. The next survey will give more proof for this.
- People look for features on a site: the Feature Model will have to capture them.
- The scale will be fixed on a five-point level: "Not at All", "A little", "Neutral", "A lot" and "Absolute".

3.2 Ratings survey

After the first survey, it was clear that the focus needed to be at *what* people looked for on a site. Many of these features were already known because of the work presented in the Background chapter or brainstorming, but there was a need to get a *complete* list.
Together with this, data was required to test the methods used in the prototype (see Section 2.5 & Chapter 6).
The Ratings survey queried 22 people on their trust ratings for 50 sites. Apart from the rating, they were asked to supply the reason for giving it. In addition to this they supplied many useful comments and hints for the rest of the project.

3.2.1 Design

Every week respondents received a set of 10 sites through email. This split-up was done to make sure the people did not get bored with it and stayed sharp.

The sites were chosen from a large amount of commercial sites available through Yahoo!. When picking the sites, attention was paid to diversity and representativeness. In other words: there were very normal commercial sites and some more awkward ones; very familiar sites and unfamiliar ones, sites that sell cars and sites that sell jojos, sites that looked as if they had cost a million dollar and sites that were made as a hobby. The full list of sites is available in Appendix B.
The sites were attached to the email as an Microsoft Web Archive (mht) that incorporates the whole page in one file. Respondents did not do the survey completely synchronous so in this

way the sites looked the same for everyone regardless of time.
For every site the following questions were asked:

How much would you trust this site (1-5)? This question could be answered on the 5-point scale as was discussed in the first survey: Not at all, A little, Neutral, A lot, Absolute, represented by the numbers 1 to 5 respectively.
Respondents were explained that to trust means the general feeling one has with the site: "do you feel comfortable to order something or to leave your personal information on it, things like that".

Did you know this site already (y/n)? It was important to know if people based their judgment purely on the site or on previous experiences.

What was the main reason for giving this score? The respondents had an open choice to fill in whatever they wanted in this field. Any constraints here in the form of a look-up list would reduce the diversity in answers.
After 20 sites (2 packages) this answer became optional because it was too much effort for the people to fill in. Luckily some people kept on until the end.

3.2.2 Summary of results

Because of their size, the tables with results are placed in a separate report that accompanies this thesis. Although a note on its statistical relevance has to be made, a lot of interesting information can still be drawn from it.

Notes on statistical relevance

- Selected sites were chosen manually, while trying to make them as diverse as possible. No measurement was used to test this though.
- There were only 22 respondents (from a population of about 400 million[1]) out of which 18 finished the whole trajectory. The respondents came from Western Europe and were all involved in IT-related business (Accenture or Computer Science student). Other people may have looked a lot differently at these sites than the respondents of this survey.
- Respondents had to take a lot of time filling in the whole survey. By the end (after more than a month) some said to have change their behaviour in rating. And, as mentioned in Section 2.6, moods are not included in the model. Respondents display inconsistent patterns in their ratings.
  This will however, also be true for the application: it has to work with a user that has different moods and therefore different ideas on what trusted is.

Categorised Reasons

All the reasons respondents filled in for their rating had to be categorised in order for them to be comparable. 23 of these categories were defined (see Table 3.6 on page 29).
Some of these categories will not be regarded as trust-related. The 'concept' category for example is a feeling the respondent has about the idea of buying this product online. It is often related to the law (is it legal to buy it) or just to his interest.
Cleared of these categories the most frequently mentioned are (in order):

1. Design

[1]ACNielsen estimated it at 379 million in March 2001 on http://www.eratings.com/news/20010430.htm

2. Reputation
3. Company
4. Privacy
5. Seals

It is remarkable to see that only one text-content-related issue is mentioned in this top 5: Company; the address information. The rest of the table also does not mention any text related issues. In other words: people do not read the page when assessing its trust.
This is probably due to the fact that most commercial sites do not contain a lot of text in the sense of proper sentences. It has a lot of words, but no real text.

Results per site

In Table 3.3 the average ratings that respondents gave to every site are displayed. They are quite different but have their average at 3.1 (almost the Neutral score) with a standard deviation of 0.7. This indicates that people have scored very conservative: in 50% of the cases their average rating was around 3.0 ($2.5 <= \text{score} < 3.5$).
The overall results are a little more spread. When we take the separate ratings into account we see that the answer '3' was given in 29% of the cases. See table 3.4 for more detail. This result is either due to the respondents or the sites.

Table 3.3: Average ratings per site

Sites 1-10									
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
2.5	4.5	3.6	2.2	2.4	2.9	4.6	2.5	3.2	3.2
Sites 11-20									
11	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*
3.7	2.4	2.9	4.1	3.7	3.2	4.0	4.4	3.6	4.1
Sites 21-30									
21	*22*	*23*	*24*	*25*	*26*	*27*	*28*	*29*	*30*
3.3	4.8	2.4	3.4	2.7	2.9	2.9	2.3	2.8	2.6
Sites 31-40									
31	*32*	*33*	*34*	*35*	*36*	*37*	*38*	*39*	*40*
2.5	3.3	2.0	2.1	2.8	2.7	2.2	2.2	2.7	3.0
Sites 41-50									
41	*42*	*43*	*44*	*45*	*46*	*47*	*48*	*49*	*50*
1.9	1.9	3.9	3.3	2.2	3.1	3.0	2.9	3.2	3.1

Table 3.4: Spread of the ratings

Not at All	*A little*	*Neutral*	*A lot*	*Absolute*
13%	21%	29%	24%	14%

One of the aims of the survey was to see if the theory on subjectivity on trust was correct. When a group of people is presented with a site, do they disagree? The question would first be what is (dis)agreement and to what detail can we go?

In Tables 3.7, 3.8 and 3.9 the ratings respondents gave are mapped onto Negative (score 1 & 2), Neutral (score 3) and Positive (score 4 & 5). For a site the relative number of negative, neutral and positive scores can be found. The maximum of these is defined as the agreement. In 58% of the cases does the agreement reach 50%, but only in 12% of the cases is the agreement more than 75%. This means that only in 6 cases 75% of the respondents agreed.

Results per respondent

The same can be done for the 19 respondents[2]. Their average scores are displayed in Table 3.5. After Marsh's model we can say that the people with an average score under the overall average will be the pessimists and above the optimists: it is their Basic trust. The respondents were not far off from the average, but still when looking at scores outside of the $< 2.5, 3.5 >$ range there were 3 pessimists and 4 optimists. The other 11 respondents scored too close to the average to be distinguished.

Table 3.5: Average ratings per respondent

Respondents 1-11										
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*
3.2	3.1	—	2.6	3.1	—	2.7	3.5	2.2	3.2	4.2
Respondents 12-22										
12	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*
2.9	3.5	2.5	—	3.1	2.9	3.5	3.3	3.4	2.7	2.8

Combining respondent results with the categorised reasons mentioned earlier, a vocabulary can be defined: what categories of reasons do respondents take into account. Per site an average of 1.6 categories were identified. This is not a number that one can use, because it is limited by the effort a respondent wants to put into the survey and not to how many there were.
It *is* useful to see that an average of 10.8 categories were used in total for respondents. This means that their vocabulary of categories is 10 to 11. They look at these things for all sites.

3.2.3 Conclusions

Although some remarks have been made on the statistical relevance of the survey, a number of things can be said about its outcome:

1. Respondents scored sites very neutral (around 3). This can be due to the sites or to the respondents. To be sure about this, a more detailed study would have to be undertaken.

2. The concept of trust seems very subjective if we take the agreement measure into account. There are only 6 sites where more than 75% of the respondents agreed on their judgment.

3. It is hard to find the Basic trust of people. If they would have a tendency to score above or below the average, this should be more clear from the data. With this result only 7 people were found to disagree structurally with the average.

[2]Respondent 3,6 and 15 have been taken out because they did not return enough data

4. People take 10 to 11 of the categories (defined in this chapter) into account. Some of these categories are used very often by the whole group (Design of a site for example) and some only by a few (Public). The ones that are used by everyone deserve a lot of attention in the next chapters where this will be used.

5. Textual content is not an important issue when assessing Trust. People do not seem to 'read' the page. This has two important consequences:

   (a) Language is not very relevant, although it does seem to scare people if the site is in another language.

   (b) The learning methods discussed in the previous chapter should not focus on text but on appearance.

Table 3.6: Categorised Reasons

Category	*Description*	*Frequency*
Capabilities	Capabilities of the site, the options it has, a user-account etc.	0.9%
Company	Information: address, telephone etc., also means that it is a real-world company	7.2%
Concept	Product type offered in general or specifically on the Web was not to the liking of the respondent	6.1%
Defined	Well defined products (e.g. like phones)	0.2%
Design	Professionalism / clear / organisation, description of products / multi-lingual quality of grammar / spelling	33.9%
Experience	Respondent has experience in using this site	1.9%
Feedback	Reviews on the site, feedback from users	0.4%
Location	Offices in your country (you can visit them), not with regards to shipping	0.3%
Model	Business model: ways of payment (cheques or only through their shops), shipment rules, process of ordering (back-office), order-tracking	4.9%
Partners	Affiliates of the site or company	2.1%
Privacy	Presence of a Privacy Policy	5.9%
Products	Brand products taken from real-world (not the company, but a shoe portal selling Nike for example)	2.1%
Public	On a stock exchange (NASDAQ, NYSE etc.)	0.4%
Recent	Site looks up-to-date	1.5%
Reputation	Real-world & Net reputation	10.9%
Risk	What one has to loose	1.4%
Seals	Credit-card seals, VeriSign. Negative means either 'not there' or 'untrusted'	5.5%
Security	Technical perspective (not seals) is often just because of word 'security'. Also includes cookies	5.2%
Service	Service offered by the company	1.5%
Similar	The product or site looks similar to something that is already trusted	0.6%
Size	The size of the site (amount of things they offer etc) and with that the size of the company	2.5%
Technical	Does the site work properly (mostly noticed when it does not) also when it is too slow	3.2%
URL	URL / directory / name (from a technical point of view), country extension	1.5%

Table 3.7: Ratings of respondents mapped onto Negative (-), Neutral (=) and Positive (+), X = no rating. Sites 1-17

Part.	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*
1	=	+	+	-	=	-	+	-	+	=	+	-	=	=	+	=	+
2	-	+	+	-	-	+	+	=	-	+	+	-	+	=	+	+	=
4	-	+	=	-	=	-	+	-	-	=	-	-	=	+	=	=	=
5	-	+	+	-	-	-	+	=	+	=	+	+	-	X	+	=	+
7	-	+	+	-	+	-	+	X	-	+	=	-	+	+	+	-	+
8	+	+	+	=	-	=	+	+	+	+	+	+	-	X	+	=	+
9	=	+	+	=	-	=	+	-	-	=	=	-	-	+	=	-	=
10	-	+	+	+	+	+	-	-	=	-	+	-	-	+	+	+	=
11	+	+	+	=	+	+	+	+	+	+	+	+	+	+	+	+	+
12	-	+	=	-	-	=	+	-	-	=	=	-	-	+	=	=	+
13	-	+	=	=	=	+	+	+	=	=	=	+	+	=	+	+	+
14	=	+	=	-	X	-	+	-	=	=	+	-	=	+	=	-	+
16	-	+	=	-	=	=	+	+	=	+	+	-	=	=	=	=	+
17	-	=	=	-	-	+	+	-	+	-	+	+	=	+	+	+	+
18	+	+	+	-	=	-	+	-	+	+	+	-	+	+	+	+	+
19	-	+	+	-	-	=	+	-	=	=	=	-	=	+	-	-	+
20	+	+	-	+	-	=	+	+	-	=	=	=	+	+	=	+	+
21	=	+	=	=	-	-	+	-	-	-	=	=	=	+	=	-	-
22	=	+	-	-	-	+	+	=	+	=	+	-	=	+	+	+	+
Agreement defined as the maximum percentage of people that share the same rating																	
	55%	95%	50%	65%	53%	35%	95%	58%	40%	50%	58%	63%	42%	76%	58%	42%	74%

Table 3.8: Ratings of respondents mapped onto Negative (-), Neutral (=) and Positive (+), X = no rating. Sites 18-34

Part.	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*	*27*	*28*	*29*	*30*	*31*	*32*	*33*	*34*
1	+	+	=	=	+	-	+	=	=	=	=	+	-	=	+	-	-
2	+	+	+	+	+	+	+	-	+	+	-	-	+	-	+	=	-
4	+	=	+	=	+	-	=	-	-	-	-	-	-	-	-	-	-
5	+	+	+	=	+	-	+	-	-	=	=	+	=	=	=	-	-
7	=	+	=	+	+	-	-	-	-	-	-	=	+	-	-	-	-
8	+	+	+	=	+	-	=	-	=	-	-	-	-	X	X	X	X
9	+	=	+	=	+	-	=	+	-	-	-	-	-	-	-	-	-
10	+	+	+	+	+	=	+	+	=	+	=	-	+	=	-	-	=
11	+	+	+	+	+	=	+	+	+	=	=	+	+	=	X	X	X
12	+	+	+	-	+	-	=	-	=	-	=	=	-	X	X	X	X
13	=	=	=	+	+	+	+	=	=	+	+	=	+	-	+	+	+
14	+	=	+	-	+	=	=	-	=	-	-	=	-	-	=	-	-
16	=	+	+	+	+	-	=	+	+	=	=	-	-	-	+	=	-
17	+	=	+	+	+	-	+	-	=	+	=	-	+	-	=	=	=
18	+	+	+	=	+	-	=	-	X	+	-	X	=	+	+	-	-
19	+	=	+	-	+	-	=	+	=	+	-	=	-	+	+	-	+
20	+	+	=	=	+	=	+	=	=	-	=	=	=	=	+	=	-
21	+	=	=	=	+	+	+	=	+	=	-	+	-	=	=	-	-
22	+	+	=	+	+	-	=	=	-	=	-	+	-	-	=	-	-
Agreement defined as the maximum percentage of people that share the same rating																	
84%	58%	68%	42%	100%	63%	47%	47%	50%	37%	53%	39%	53%	53%	44%	69%	75%	

Table 3.9: Ratings of respondents mapped onto Negative (-), Neutral (=) and Positive (+), X = no rating. Sites 35-50

Part.	*35*	*36*	*37*	*38*	*39*	*40*	*41*	*42*	*43*	*44*	*45*	*46*	*47*	*48*	*49*	*50*
1	=	-	=	+	=	+	=	-	+	+	-	+	=	=	+	=
2	-	+	-	-	+	=	-	-	+	-	-	=	=	+	+	-
4	=	=	-	-	-	=	-	-	=	-	=	=	=	-	=	=
5	-	-	+	+	=	=	-	-	+	-	-	-	=	=	=	+
7	-	=	-	-	-	=	X	X	X	X	X	X	X	X	X	X
8	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=
10	+	=	-	-	-	-	X	X	X	X	X	X	X	X	X	X
11	X	X	X	X	X	X	=	=	+	+	+	+	+	+	+	+
12	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
13	+	=	+	+	-	=	-	-	=	+	+	=	=	-	+	+
14	-	-	-	-	=	=	-	-	=	X	-	=	-	-	-	-
16	+	=	=	-	-	=	-	-	+	=	-	=	+	=	+	-
17	-	-	-	-	X	-	X	-	+	+	-	-	-	-	-	-
18	-	=	X	-	+	+	+	=	+	+	-	=	=	=	=	+
19	+	+	+	-	+	+	-	-	+	+	-	+	+	+	+	+
20	+	-	=	+	-	+	-	=	+	=	=	+	+	+	+	+
21	=	=	-	-	=	=	-	=	+	=	-	=	-	-	=	=
22	-	-	-	=	-	=	-	-	=	+	-	=	=	=	-	=
Agreement defined as the maximum percentage of people that share the same rating																
	50%	44%	60%	69%	47%	56%	79%	73%	67%	50%	73%	53%	47%	40%	47%	40%

Chapter 4

Feature Model

This chapter introduces the main framework for WebTrust: the Feature Model. It is built on top of the research done by others and the survey performed during the project.
First of all the parallel with humans is made, their behaviour forms the basis of the ideas presented. After this, concept of features will be formely introduced. The model itself consists of a categorised list of these features together with their metric for measurement as depicted in a number of tables. Finally a section is dedicated to the way sites are represented using features.

4.1 The basis: humans

In the Background Chapter and the surveys the way humans trust on the Web was laid out and discussed. It is clear now that people use a 'checklist' of site properties when assessing trust. All items on this checklist have a ranking of importance. This ranking is personal and is therefore not necessarily the same for different people. In addition to this, people can be more gullible than others and not use all elements on this list. Below a list of all identified elements is presented:

Experience; the user will see if he knows the company or site already. If he does, he already has an idea about the trust he can attribute this site.

Layout; after this the design of the site is taken into account. Especially for unknown sites this can be very important. It is a sign of professionalism site when it looks well. This will be reiterated later in this section.

Interaction; part of the layout are links and buttons. These are susceptible to errors. If the designer is not careful they may not work: the link does no longer exist or the button generates some kind of error. All these things can be discovered by interacting with the site: clicking on it.

Other people; finally the opinion of other people is important as shown by the eOpinions website, discussed in Section 2.3. In general these people can be friends, consumer magazines, newspapers or any other trusted party with an opinion. Their trust with regards to this particular site is of importance to the user.

All these questions asked and tests performed to assess trust are the basis of the *Feature Model*, which will be introduced in the next section. The 'Other people' item is part of the negotiating agents principle, which is laid out in Chapter 5.

4.2 Features

Features are used to describe properties of sites that people use to determine its trustworthiness. They are very similar to things we look at when we judge a person in an interview. Did he shave? What does he wear? Does he speak with an accent? What do other people say about him? Does he have diplomas? Did he perform well on the IQ tests?
Although not all of these features may be relevant to *you* they may be to others: they have a different meaning for everyone, but can be measured objectively.

The Feature Model is based on this idea of examining properties of a site and related issues: features of a site that are involved in the trust-decision. A significant difference with the 'components' from Cheskin (Section 2.2) is that these features are measurable entities. Although some of them can easily be transformed, Cheskin's are not as strict.

A **Feature** is a measurable entity present on, or related to a site or the organisation behind it.

See Figure 4.1 for an example site with its features.

Figure 4.1: Features of a webpage

The Feature Model consists of a list of all identified features together with their metric, split up into 4 categories defined for it:

Reputation is an easy way for people to decide on a site. If they know it already because they read about it in a magazine, saw the company on television or heard about it from a

friend they have an initial idea about its trustworthiness. Sometimes this idea is so strong that they will decide without really taking the site into account or even contradicting their feelings about other features (see the Ratings survey in 3.2). Reputation can be personal as well: the fact that they know it because they have used the site before (experience) is also included.
An automated application would miss out on all of these inputs used by its users when it is not aware of other sources then only the tested site. Reputation of a company and its site are therefore defined by other sites indexing it. The features in this category are all external sources of information that give an impression on how well-known it is and what people think of it. See Table 4.1 on the next page for the features included in this category.

Professionalism we seek in both brick-and-mortar companies and online ones. It is typically a vague subject people refer to. Even the Cheskin report talks about "Clarity of purpose" and "Craftsmanship" (page 11). The features mentioned in Table 4.2 on page 37 try to define things we have taken into account when we say that the site looks 'professional'

Technical issues are noticed when they do not work, or when one wants the site to conform to a certain technical standard. This can be true for more categories, but here is it really obvious. The site is not working, not very fast etc. Or one really wants the site to be secure before ordering anything on it. See Table 4.3 on page 38 for the features.

Other Some features mentioned by people were so hard to classify (i.e. implement) that they ended up in a separate category. The modeling of them will take a lot of research.
See Table 4.4 on page 38 for the features.

Some people will claim that they just do not want to buy product X over the Web. This is not regarded as a feature. Although buying a 2^{nd}-hand car over the Web involves a lot of trust in the company or the person selling it, the features one takes into account are still in the feature list and not in the fact that it is a 2^{nd}-hand car. Although one could argue that the weights for these features change when talking about a more risky product, it does not change the things one looks for in a site.

4.3 Representing a site: Feature Vector

All these features have been defined to be able to represent a site. This representation requires the features to be measured: *rated*.

> A **Feature Rating** is the objective outcome of a measurement of a feature on a site

Ratings are expressed on a boolean or an integer scale. The Boolean features typically have a boolean nature by themselves, like the 'presence of a privacy policy'.

It is very important to state that these ratings do not have a judgment on trust in them. In other words: a lot of pictures do not have to mean a lot of trust. In fact this feature is probably correlated through some parabola.

> The **Feature Vector** is an n-dimensional vector containing the Ratings of all features representing a site; n being the number of features defined

This Vector allows sites to be compared to each other.

The major assumption here is the span of the feature set: does it adequately represent a site with respect to trust? Ultimately the prototype will have to prove this, but unfortunately it is not able to measure *all* features. This will be explained later, but it is important to note that the prototype will work with a reduced feature set.
Finally people can say that it is possible to trick the system by adding 'good' features to their site. This is true, but also true in the sense that it justifies a higher trust for the page. When a company would engage in a relation with a 3^{rd}-party to obtain a seal, they have rightfully obtained a different trust position.

Table 4.1: Reputation Features

Name	*Description*	*Measurement*
External reviews	Consumer magazines for Internet users could give ratings to the quality of sites and their companies. Once they describe a site it will have a reputation in the eyes of the reader: their magazine writes about it. This is of course also dependent on the reputation of the magazine. Most reviews use a star system to measure their opinion, the more stars the better	The number of stars this site receives in a review.
Number of references to the site (implemented)	Based on the same principle as Google's PageRank, the number of or references to a site indicates how well it is known on the Web. It would also be interesting to include the trust for these referring sites in the rating to make recurrent trust values. Unfortunately some sites have 30 million sites referring to them, making this strategy intractable.	Number of references to this site.
Online revenues (implemented)	When a company has high online revenues, it probably has many customers buying its products. These customers have an idea of the company and communicate this to their friends. When revenues become very high, the company is likely to receive attention in the media.	Presence of this company in a ranking of 'best-performing' online companies
Partners	Partners are a special kind of external reviews. They indicate companies that are associated or behind the site. The trust the user attributes the site's partners is likely to be included in his judgment on the site. The partners also form the basis of a recurrent trust value: the trust of the sites that this site links *to*.	Trust values attributed to the sites of its partners.
Public	The fact that a company is on the NYSE or NASDAQ is a sign that it is a serious company. It takes a lot of effort to get there and after this the market will keep track of the health of this company. Stock values can change dramatically over night, so it may also be good to track their value over time.	The fact that this company is on the NYSE or NASDAQ.

Table 4.2: Professionalism Features

Name	*Description*	*Measurement*
3^{rd}-party seals (implemented)	3^{rd} parties are organisations that are either non-profit or that benefit from the fact that the user is content with his transaction (because they insure your purchases for example). They associate their name with commercial sites by allowing them to have a seal (a picture) on their site. The picture alone is very easy to fake, but the proper ones should link to the site of the neutral party, which tells you that this company is indeed registered. Well known companies in this field are VeriSign, BBB Online and the major credit-card providers (MasterCard, AMEX & VISA).	The presence of these seals on the site.
Capabilities of site (implemented)	A site can be enhanced with all kinds of scripts, frames and forms. Although this is not immediately apparent for the user, it can be seen in the overall picture of the site.	The number of scripts, frames and forms on the site
Information	Information provided on the company behind the site like address and phone numbers give the impression that this company is actually based somewhere.	The presence of address information on the site
Multi-lingual	Capabilities like these show that a big and / or professional company is behind the site. Sites often have a flag as a link on their site to popular word languages like Spanish, Italian, French and German.	The presence of links pointing towards a file similar in HTML tags as the main one. The name of this link should capture a language (e.g. espagnol-index.html)
Number of pictures (implemented)	Pictures are part of the design of a page.	The number of pictures on the site
Privacy policy (implemented)	When looking for the presence of a Privacy Policy, people want to make sure that their information is not sold to other companies or that they will not receive any other 'information'.	The presence of a link with a name related to 'privacy'
Quality of grammar	The number of spelling mistakes or the level of the language indicates the time spent on building the site and possibly the origin of its administrators.	The number of spelling mistakes on the site
Quality of URL	The URL of a site is the identity on the Web. Companies always try to use www.<companyname>.com, but this is not always possible. Sometimes you can also see where a company is from if it does not use the .com extension. A deep directory name with some ISP is also an indication of the company behind it (`www.isp.com/~companies/this_company.html`).	The fact that the URL is www.company.com
Refresh rate	When a site is not up-to-date, it displays sales from Christmas in January, or the fact that the site has been last updated some time ago. Some sites on the other hand, display news that is updated hourly.	The most recent change in the file as indicated by the file-system

Table 4.3: Technical Features

Name	*Description*	*Measurement*
Dead links	Links that take you nowhere are signs of a badly maintained site. When one clicks on a dead link, one gets the infamous 404 error: page not found. Missing pictures are also considered dead links since the code of a page links to them.	The number of dead links found on this site
Quality of HTML	This is often not seen by users, but says something about the editor (vi or MS Frontpage). HTML allows many different ways of writing. When the administrator is not skilled enough in writing, the probability grows that he will cross some of these vague boundaries and come into one of the error categories mentioned elsewhere (script errors etc.).	The number of HTML errors as reported by one of the applications mentioned in Section 2.4.2
Script errors	Sites get more and more complex and when the administrator does not use appropriate tools to keep track of all the connections, variables and databases, errors occur. They can often be seen as messages in the middle of a page or buttons that do not do anything.	The number of buttons that generate error messages when pressed
Security	Security is often regarded as the only feature of trust. When actually identifying oneself on a site or typing any other important information, the connection should be encrypted. This encryption makes sure that others cannot eavesdrop on the data and that authentication is done properly. For users the small lock on the bottom of their browser indicates the fact that a secure connection has been established.	The fact if the site is 'secure' as indicated by the browser
Speed of the site	This actually consists of the product of two variables. The speed of the server and the amount of traffic on the network. We can eliminate the latter by making more measurements at different times, but most of the time this will be ignored. Slow sites often indicate small companies.	The 'average round-trip time' of packets set to the site as indicated by the Ping command

Table 4.4: Other Features (difficult to measure)

Name	*Description*
Business model	People wonder why a company would do business the way they do it right now. When a company gives things away for free, they could question why; money needs to be made in some way. Together with the way of payment (by check or credit-card) they are the things that could raise suspicion.
Clear information on cost	The sooner the company comes up with the final bill the more clear the process will be. Some companies make a total bill under their 'shopping kart' including shipping and taxes, whereas others say it just before you check out.
Other sites or brands that this company has	The company behind the site can have more sites for different products or brands. They will often be taken into account as well; assuming that the users know this of course. In real-life this means that you trust Dockers because it is owned by Levi's.

Chapter 5

Automating trust

The main purpose of this thesis is to find a method to automate the finding and identification of trusted sites. In this chapter the WebTrust application is introduced to do this.
A general overview of the architecture will be given, accompanied by a more detailed design. The implementation of the 3 different methods for training explained in Chapter 2 is also highlighted.

5.1 The WebTrust application

The vastness of Internet has given us both good and bad things. We can buy almost everything online by now, but it is hard to find and difficult to assess the quality of the companies and their products. In practice this means that we will stick to companies we know and trust or the ones referred to by friends and magazines. Only a limited amount of shops and products will be available to us this way.

WebTrust tries to solve this problem and broaden our horizon by automatically retrieving sites from all over the Web and presenting the user with an indication on how much he would trust them. The workings of the prototype are based on two notions of the human way of assessing trust in a site:

- **It has a perception of what should be present on a trustworthy page and what not:** When looking at a site we can quickly get a first impression on its trustworthiness by looking at its appearance or the mere fact that we know it already. After this we will do a more thorough investigation with respect to less obvious things like the privacy policy or the way of ordering products.
  The prototype uses the Feature Model to extract all these *features* from the site and rate them according to the profile of its user. This profile is determined by the training process in which he implicitly reveals his trust-pattern.

- **It uses 'friends' to gain more knowledge on a site:** Before buying a product online people may ask friends about related sites or they find these sites by referral from magazines. A consumer magazine on cars for example, could rate sites that sell cars or parts. With that method it assigns an important index to these sites in the eyes of the readers of the magazine. Friends could be asked about sites because they have used them before or are considered 'experts' in the field.
  Apart from its own agent, users can add agents from other people or institutes to help him in his search. The consumer magazine could publish an agent trained by one of their

experts to be downloaded by readers. This agent will negotiate with the user's agent on what the trustworthiness should be as seen from both sides. Figure 5.3 on page 42 shows how you select these agents.

After the prototype has been provided with information on what agents to use and what URL to look at, it will try to assess the page's trustworthiness by asking its agents to investigate it. All of them come up with a rating and a *confidence level*. This level represents how sure they are of their rating. It will be used in the *process of negotiation* where one answer has to be found for the site. See the screenshot in Figure 5.1 on the facing page for the result and Figure 5.2 for the overview of the WebTrust process. All details in this picture are treated in the next sections.

5.2 Negotiating agents

On the highest level there are agents that can communicate with each other on their trust opinions and come up with an answer that suits them all. This approach allows users to use different agents than their own for specific purposes. There are two reasons for doing this:

Users do not know the area very well and want to ask an expert *they trust* to help them out on the details. A good example would be someone who wants to buy cigars online. He does not know much about cigars in general and therefore asks a cigar smoking friend. This friend knows what things to look for in a good tobacco store or site. His trust agent will represent this knowledge and looks for sites that look very Cuban and have the words handmade and 'real man' in them.
The first person also has his own agent that checks for things that *he* finds important like a credit card logo and a privacy policy. Together the two can come up with some trustworthy cigar-sites.

Some people may specialise in a certain field and sell their agents to users that trust them. An authority on the field of customer reliability like a cigar consumer magazine may have its agent on their website for downloading. In this way subscribers can use the agent from a magazine they trust to help them in finding proper sites.

As shown in Figure 5.4 on page 42 the agents both rate the page, negotiate and then come up with one rating. There are five different answers an agent (or the whole system) can give: Not at All, A little, Neutral, A lot and Absolute.
The process of negotiation is implemented in a basic way: as a blackboard. Every agent posts his answer and confidence level on this blackboard. When they have all done this, the system reads the information on the board and uses it to generate the overall answer as shown in Figure 5.5. The weights depicted here are all equal, but a reputation per agent can easily be implemented to reflect their expertise. At this moment all agents are anonymous.
In this Figure, 'Unreliable answers' are answers that have a confidence level below 30%. The agent was very unsure of its answer and it will therefore not be used in the negotiation process. It is discarded.
There are also 'Very Reliable' answers. These answers have a confidence level of at least 95%: the agents were very confident about them. When there are agents that have posted their answers with so much certainty, they will be the only ones that will be taken into account: only the highly reliable answers remain.
From the remaining answers the weighted average is calculated using the cumulative confidence as a weight. The cumulative confidence for the answer '1' is, for example, the sum of all confidence levels that were given with the answer '1'. The weighted average is returned as the overall answer.

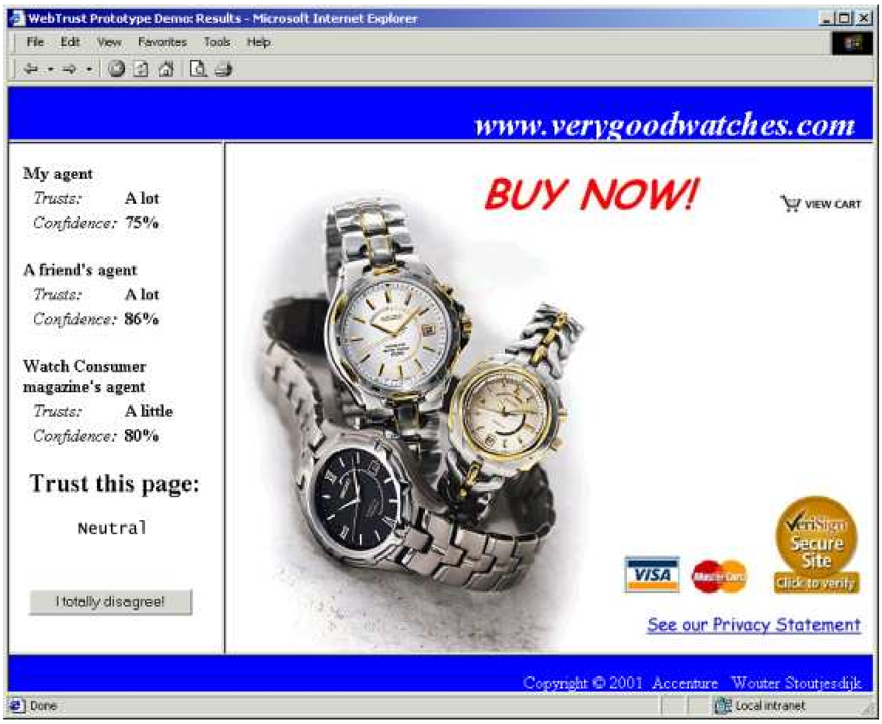

Figure 5.1: WebTrust Application: the Result

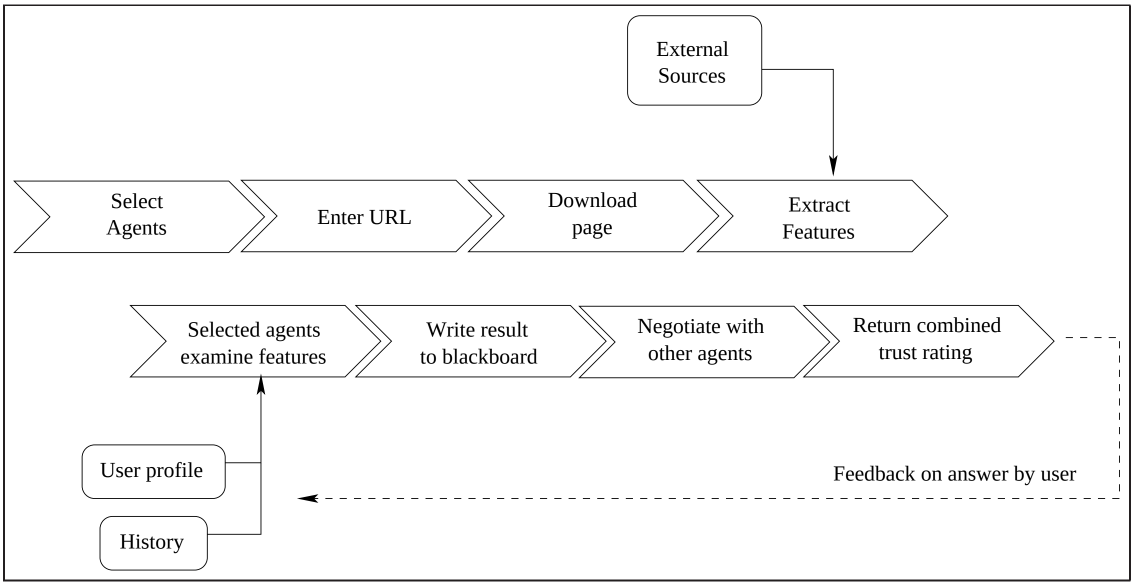

Figure 5.2: Process of WebTrust

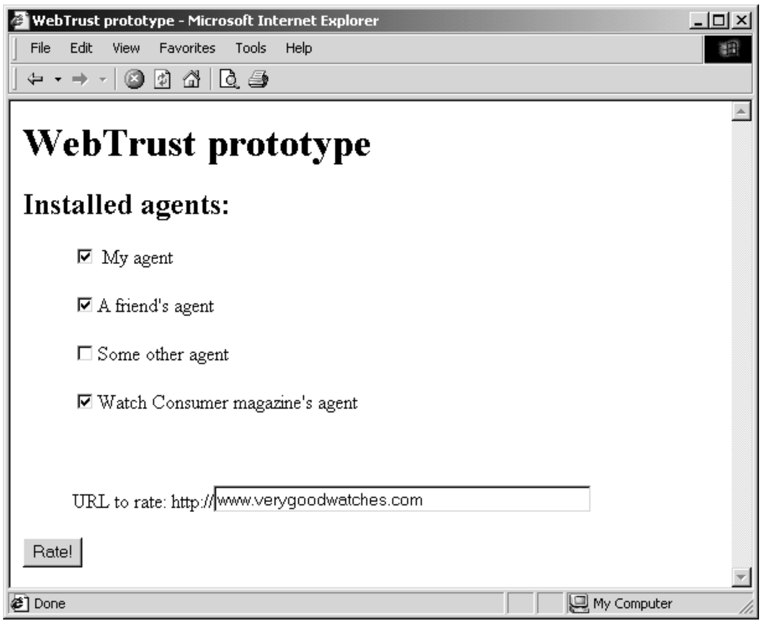

Figure 5.3: WebTrust Application: Selecting agents and a site

Figure 5.4: Agents looking at a website negotiating a common trust rating through the blackboard

Table 5.1: Agent ratings for a site

Agent	*Rating*	*Confidence level*
My agent	A lot	75%
A friend's agent	A lot	86%
Watch consumer magazine's agent	A little	80%

If we follow the ratings from the watches in figure depicted in Table 5.1 we would find:

- There are no agents UNRELIABLE, else it would have ignored them.
- There are no agents above the VERY RELIABLE level, else it would have taken only them in the rest of the process.
- The answers are transformed back into numbers to get 4 for a 'A lot' and 2 for 'A little'. The weighted average answer is then: $\frac{4*0.75+4*0.86+2*0.80}{0.75+0.86+0.80} = 3.3$.
- The system will return the rounded answer '3': Neutral.

This approach can be extended into a whole new project on how humans negotiate an answer. Enhancement is therefore left for further research.

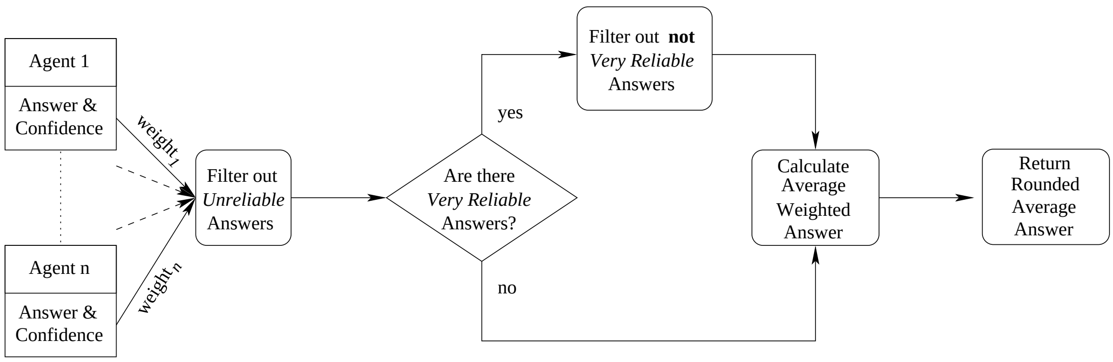

Figure 5.5: The process of negotiating agents

5.3 A trust agent

In this section *one* agent itself will be discussed. This trust agent can give a trust rating to a site based on its content and the agent's profile. The architecture for such an agent is displayed amid the total architecture in Figure 5.6 on the next page.
Once a site is found, its features are extracted by the extractors. They parse the site into a vector that represents the site. This vector is fed to the agent together with the user profile and the History. By means of the adaptive trust mechanism, which will be discussed later on in this chapter, the agent infers an answer and writes this to the blackboard. If the user does not agree with the answer, he has the opportunity to give his own rating. This will be put into the History and the profile can be retrained to reflect the update. All the details of this process are treated in the next sections.

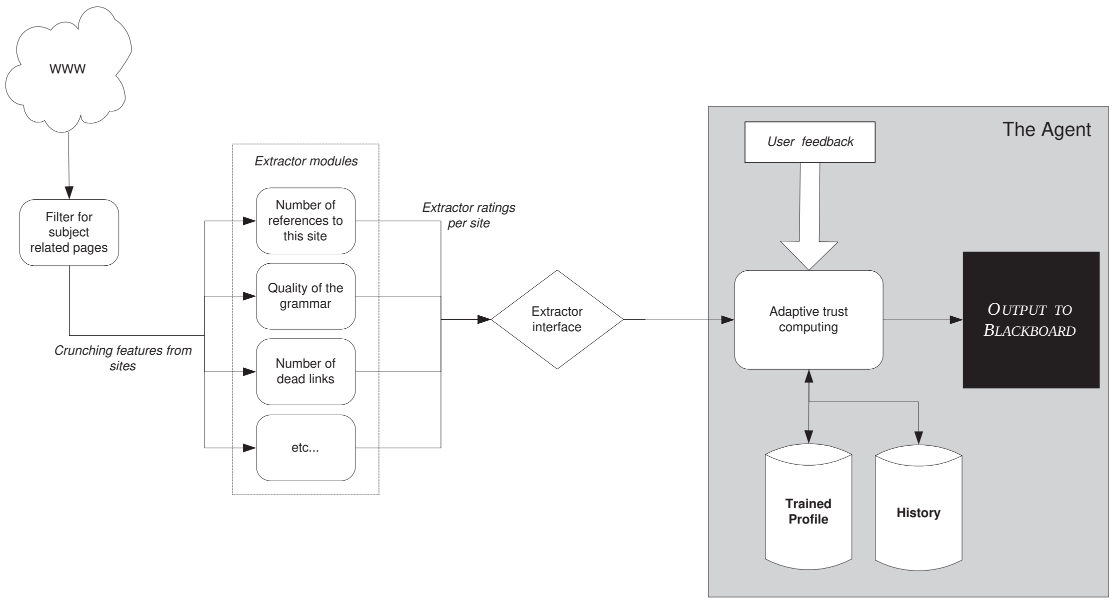

Figure 5.6: Architecture of a trust agent placed in the overall architecture of WebTrust

5.3.1 Finding sites on the Web

In the demo of the prototype, sites are selected by typing directly into a form (see Figure 5.3 on page 42). The idea behind WebTrust however, is to be able to automatically select these sites from the Web. A search engine looking for commercial sites with the topic would solve this problem. Reality has it that these results from search engines need another filter before being fed to the trust prototype. The prototype cannot - by itself - decide if the site is really selling something or not. This may lead to unexpected results in the ratings: a bookstore being as trustworthy as a library.

It is also important to note that the survey showed that most people will only take the main page of a site into account when assessing. The prototype therefore only looks at the first page of a site. This means that in practice a whole online company is represented by its main page!

5.3.2 Extractor modules

The purpose of the extractor modules mentioned in the architecture is to turn a URL into a feature vector representing the site. Each extractor focuses on a specific feature.

An **Extractor** is a module that rates a Feature on Webpages.

Some examples of extractor ratings are presented in Table 5.2 on the next page.
The features can be found on the main page itself or come from additional resources as depicted in Figure 5.7 on the facing page.
As mentioned in the Feature Model, the features fall into 4 categories: Reputation, Professionalism, Technical & Others. For the first two categories, 6 extractors have been implemented, which will be discussed below.

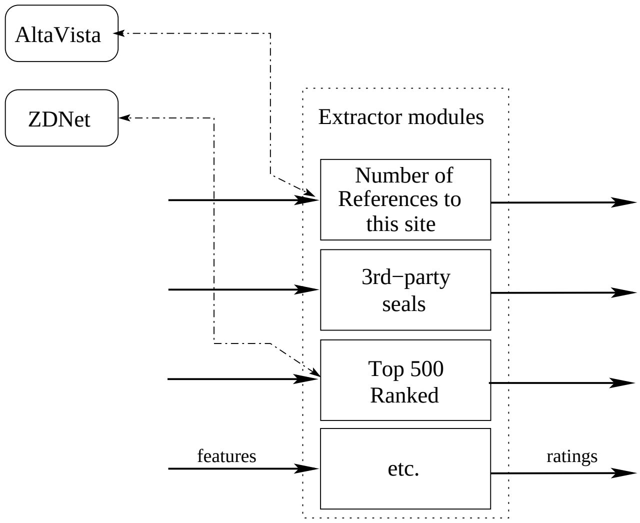

Figure 5.7: An agent extracts features directly from a site and from additional resources

Table 5.2: Presence of features is measured

Feature	*Value*
Pictures	0
Quality of Grammar	85
Number of references to this page	500421
Third-party trust signs	1

All the values that come out of these extractors are put into a six-dimensional-vector that will represent the site during the rest of the process. It is clear from both intuition and the feedback given in the survey that these 6 dimensions do not represent the site accurately enough. The aim of this thesis is not to implement the full Feature Model (which would be much more representative) but to identify its features and make a feasibility study. This is done in these 6 features. The modular approach of the extractors is chosen to facilitate the process of adding more extractors. They are all scripts called by one script. As long as they have the same interface, they can be added to measure anything one would like.

Number of referring sites (Reputation)

Counts the number of sites that have links pointing towards the site that is being examined. This count is seen as an indication of how important or well-known this site is: Does it stand all by itself, or do many people refer to it?
This measurement is also implemented in the search engine Google[1] as the PageRank. This search engine uses it to sort its results on relevance. The PageRank was not easy to extract. So another approach was chosen.
The prototype uses the AltaVista search engine to find this out. AltaVista can perform a reverse search for (for example) Amazon.com. When one enters `link:amazon.com` all links pointing

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

towards amazon.com are returned together with their count.
A few things happen here:

1. The URL of the site is stripped: in general a commercial URL looks like this `<server>.<company>.com`. On the opening page of the demo, the user would enter the WWW-address being `www.<company>.com`. Many companies however, use several servers (shop .yahoo.com is an example) for different services they offer.
   To make a clean search for references these servers are removed so that all references to this company are taken (i.e. `*.<company>.com`).

2. When the company is not very well known, one can see that most of the links that are found pointing towards it, are actually self references. Self references are references found on a site pointing towards another destination on the same site. Self references are subtracted from the total number by adding `-host:amazon.com` to the query: it skips hostnames of this form.
   The method does not work perfectly since some companies have completely different names. Barnes & Noble for example have both bn.com and barnesandnoble.com and mix them up constantly in their code. Also sites that use another scheme then the `<server>.<company>.com` are not captured.
   Since the typical number of references range from 0 to about 30 million the method needs not to be so accurate and still provides useful results.

3. The resulting page from AltaVista has to be searched for the resulting count of pages. The extractor looks for the following sentence: `We found about <number> results`. If AltaVista changes the layout of their page (which they have during the project) the extractor will not find anything. This is a weakness in the extractor, but it is also easily adapted.

★ *The outcome of this extractor is an integer signifying the (net) number of references to this site*[2].

Top 500 Ranked (Reputation)

In the ZDNet Top 500 of Internet companies[3] a listing can be found of companies ranked by online revenues in combination with their URL. These companies can be virtual or both real and virtual. It is seen as an indication of reputation.
This list has parsed before use in the prototype and put onto the server It is not done real-time because the list does not change very often. It could easily be changed to do so though (the script for automatically creating the list is already there). When a URL is entered the prototype searches this list for its presence.

★ *The outcome of this extractor is a boolean value signifying its presence in the Top500 of Internet companies.*

Complexity of the site (Professionalism)

The complexity of the site is not often mentioned explicitly by people, but is definitively part of the professionalism. When one encounters a site without these complexities, there is not much to see.
The complexities are defined in 4 HTML-tags. They are a symbol of the abilities of the site:

[3] http://www.zdnet.com/intweek/stories/0,4164,2652186,00.html

`FORM, FRAME, OBJECT & SCRIPT` are seen as complex tags. If goes too far to explain here what these tags do in HTML, but in general it can be said that they are the 'parameters' for the layout of the page. The HTML code of the main page is searched for these tags.

★ *The outcome of this extractor is an integer signifying the number of 'complex' tags in the main page of the site*

Number of pictures (Professionalism)

Another measurement of the professionalism of the site is the use of pictures. It is not said that more pictures always symbolize a more professional page (probably there is some optimum), but it is part of it. As with the complex tags, the extractor searches the HTML code of the main page for occurrences of the `IMG` tag.

★ *The outcome of this extractor is an integer signifying the number of pictures found in the main page of the site*

Presence of 3^{rd}-party security seals (Professionalism)

As shown by the Cheskin [5], [6] research and by the survey in Chapter 3 people look for 3^{rd}-party Seals on a site. They signify both professionalism in the fact that the site has taken the effort of getting these seals as well as the technical implications of them.
The extractor knows the top six of these organisations as identified by Cheskin: Visa, MasterCard, American Express, TRUSTe, VeriSign and BBB Online. It searches the HTML code for pictures that have a name with any of these words in it. It is important to look for pictures since seals have to be visible as pictures.
A problem here is that some sites have made the name unrecognisable for the prototype to read (e.g. `image1.gif`), these will not be found. A more sophisticated method of graphically comparing seals with originals could be implemented if desired.

★ *The outcome of this extractor is a boolean value signifying the presence of Seals on the main page*

Presence of a privacy policy (Professionalism)

Many sites make note of a privacy policy since this is an important topic on the Web (see also the surveys). They have a reference on their main page to some kind of privacy statement or policy. The extractor will look for occurrences of the word 'privacy' in the HTML text (this means that it has to be visible on the page as either plain text or a link). Looking for better words will result in less accuracy since sites tend to have different ways of describing their policy ('statement', 'policy', 'guidelines' etc.); the word privacy is always present though. Although the extractor makes no effort to actually understand this policy it is still a good measure, since most people do not do this either; they just see the reference on the page.

★ *The outcome of this extractor is a boolean value signifying the presence of a privacy policy on the main page*

5.3.3 The user profile

Once the feature vector has been compiled, the site has its formal representation in the system. It will be fed to the user profile that contains the trust behaviour of the user. This behaviour is represented by the user's history and a technique to infer a new answer from this history. The two will be treated below in detail:

History

In the training process, explained later in the Training section (5.3.5), the user builds up a history of ratings. He sees sites and gives his *explicit* opinion. Implicit feedback can be added in the future like 'Syskill & Webert' decscribed in Chapter 2. From the way the user interacts with the site, his opinion can be deducted. All the answers given by the user are stored in his History file together with the sitename and the ratings-vector:

```
www.amazon.com         [3, 49, 1, 1, 1, 32068133] Absolute
www.booksamillion.com  [1, 140, 0, 0, 1, 30242]   Neutral
...
```

This history is both used to infer trust as well as a direct reference: before any kind of inference is done, the prototype will examine the history to see if this site has already been rated. If it has, this is of course a very strong argument for its answer. The agent will need this argument to convince the other agents that it is right. It is the confidence it has in its answer. When a site is found in the history that was explicitly rated the agent has a 95% confidence-level.
People who publish their agent also provide valuable history with it. When a magazine trains an agent on a certain topic they may visit many sites and with that cover most of the sites that the average user would visit. In this way, when the agent is used by itself it behaves more like a seal, since it will return the answers of the magazine almost constantly: it fully reflects the opinion of the magazine.

Inference methods

When a site is not found in history, the agent will need to come up with an answer based on the feature vector of the site and the agent's knowledge of the user. This knowledge is based on patterns of answers found in its history. In the field of AI there are many methods of doing this. Part of the objectives of this project is to find a good method, therefore three different methods have been implemented and tested: Artificial Neural Network, k-Nearest Neighbour and a Decision Tree, but also an expert system, Bayesian Believe Networks or Case-based Reasoning could have been used. The basic theory of these methods was already explained in the Background chapter.
Below their implementation will be discussed. In the Results chapter a comparison will be made on their performance with real trust data from the survey. They will also be compared to the combined negotiating agents as discussed above.

ANN Since many Neural Network implementations are publicly available and easily adapted, a third-party program was used. The "Back Propagation Neural Net Engine v1.33u" by Patrick Ko Shu-pui.
This package consists of 2 executables that 'train' and 'recognise'. In the training phase data is offered to the system that trains until its error is sufficiently low. All neural nets were trained using the same parameters: single layer and an error level of 0.02.
It produces a file that represents the trained network or the *profile*. In usage, this file will be

supplied as input to the recognise program together with the feature vector of the site to rate. The system returns a floating point number that represents the answer. This is then rounded and used as the answer from the ANN.

k-NN The k-Nearest Neighbour algorithm was implemented in C++ in the course of the project. Making the *profile* for this method is trivial. It needs only feature vectors and their answers from the past. Using Euclidean distance measures it will decide what answer to choose in runtime according to the algorithms described in the theory.

Decision Tree Like the k-NN, the Decision Tree algorithm was implemented in C++ during the project. Again the training is relatively easy: the program stores the history file with one addition: the order of the leafs of the tree. This has to be calculated from the History as described in the theory. The leafs are the extractors.
A rating from an extractor is on either a boolean or a continuous scale. The latter poses a problem for the tree since a choice of branches has to be made (no or yes in the boolean case). Ratings that are not of a boolean nature will therefore be mapped onto labels. 4 Labels have been defined: "None", "Some", "Many" and "Max".
The mapping function is based on the quartiles of 77 pre-rated sites. Every quartile of this dataset is a label. This dataset has been chosen to represent as many different sites as possible and will cover most of the ratings that occur at run-time.
So every leaf in the tree has either 2 or 4 branches to go to the next leaf as depicted in Figure 5.8 on the next page.
With only 6 extractors of which 3 are boolean and 3 four-labeled $4*4*2*4*2*2 = 512$ different sites can be represented. This means that - as mentioned in the theory - the tree will be very sparse and in the beginning only a few labels will be used for the actual rating.
The answer will be calculated according to the algorithm described in the theory.

5.3.4 Confidence levels

To make the negotiation more realistic agents do no only submit their answer, but also a confidence level. This level is degree of certainty and is calculated according to how sure an agent is of its assessment using its method.
When, for example, we want to know the colours of the French flag and ask only one person, we are less certain then when we ask 10 and they all agreed. If we ask 10 people and they did not agree, our level of uncertainty could be even lower: confusion.
For every of the methods implemented, confidence level measurements have been defined:

Confidence for ANN

An easy way of defining the confidence level for a Neural Network would be to return the error-level to which it is trained. The prototype however, uses networks that have all been trained with the same parameters, the same error level. So this will not be used.

Another issue is rounding. The neural network is the only method that returns an answer that has to be rounded before it can be used. This error caused by the rounding will be the (inverse) confidence level.
The confidence will grow when less rounding has to be performed. This means that the answer the network gave has to be closer to the real answer, which is the case when the network has had more training (more data). When the unrounded answer is α and the rounded one ρ, the

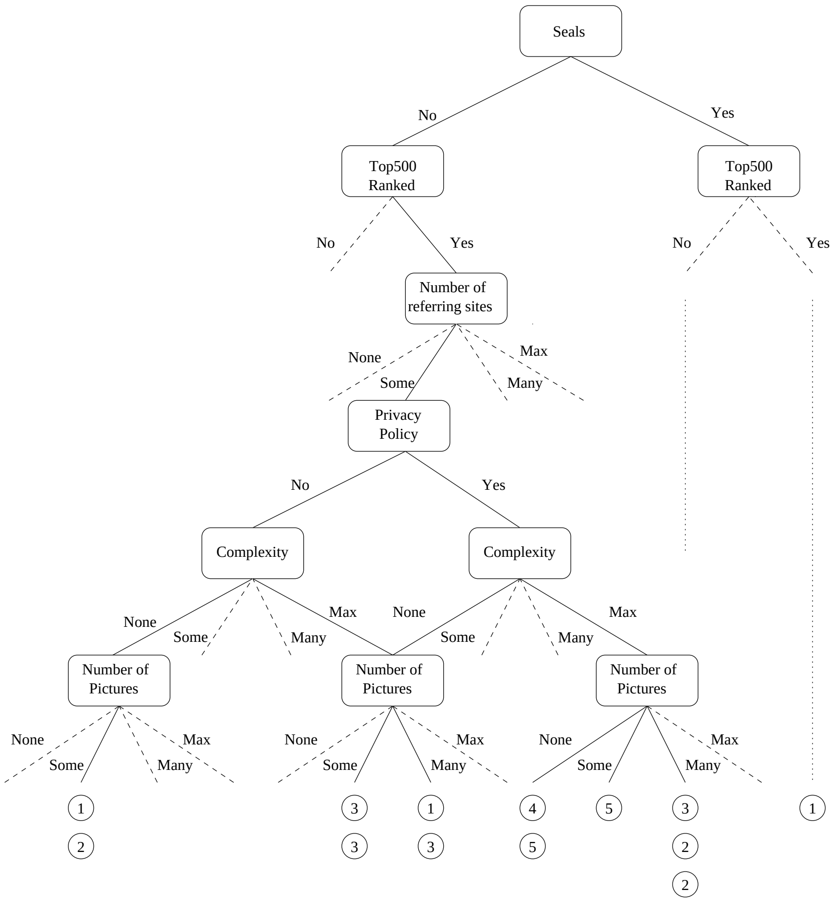

Figure 5.8: An example Decision Tree with 13 sites (their answers are displayed at the bottom).

confidence level is:

$$\text{Confidence} = 1 - |\alpha - \rho| \tag{5.1}$$

In other words the confidence level represents the quality of the network.

Confidence for k-NN

Different from the ANN, the k-NN gives insight into the decision and can therefore use a more sophisticated confidence level calculation. On a high level it is based on two properties of the solution rendered: the 'surprise' and the diversity of the neighbourhood.

The surprise is the difference between the answer given and the answer that was closest to the site: when the site that is tested is closest to answer 3, but the answer given was 5, the difference (δ) is $|5 - 3| = 2$. In other words, it is the answer that would be given when $k = 1$ (look at the nearest answer) and the one given.

The maximum distance between two answers is 4 (since there are 5 possible answers). So the surprise will be calculated as:

$$\text{Surprise} = \frac{\delta}{4} \tag{5.2}$$

The neighbourhood used for calculating the answer often contains more than one answer and can therefore contain *different* answers. When the neighbourhood is very diverse, say 1,1,5,4 and the answer is 1, there were 2 different answers from the one given. This is the diversity of the set.
The number of answers different from the one given in a set is defined as γ. This number can vary from 0 to $k-1$ (i.e. the size of the neighbourhood minus 1). The diversity is then calculated as follows:

$$\text{Diversity} = \frac{\gamma}{k-1} \tag{5.3}$$

The overall confidence level for the k-NN is now:

$$\text{Confidence} = 1 - \frac{\text{Surprise} + \text{Diversity}}{2} \tag{5.4}$$

This confidence level is based on the quality of the dataset on which the decision was taken.

Confidence for Decision Tree

At the point where the Decision Tree has to pick an answer it can be left with a same kind of neighbourhood as the k-NN method: there are more answers for sites that are similar. This is why the tree will also use diversity as part of its confidence. It is calculated the same way (see equation 5.3). Where k is the number of answers that are in this solution set.

When the tree is very sparse, it will not be traveled very deep when looking for an answer. This is due to the fact that the algorithm goes down as long as there are still sites in the solution set. In a sparse tree chances are that it will run out of sites before it has reached the bottom of the tree.
This implies that the decision is taken on limited information: only some of the extractor ratings have been taken into account. They are of course the most distinguishing ones as calculated when making the tree, but still limited information is used. This is why the depth traveled in the tree is also part of the confidence calculation. When the traveled depth is x and the maximum depth (number of leafs) y, the tree utilisation is:

$$\text{Utilisation} = \frac{x-1}{y-1} \tag{5.5}$$

$$\text{Confidence} = \frac{1 - \text{Diversity} + \text{Utilisation}}{2} \tag{5.6}$$

The confidence of the Decision Tree is based on both the quality of the data set and the sparseness of the Tree.

5.3.5 Training an agent

All the methods discussed above assume that data is available in the history of the user. To generate this data the user is presented with a number of sites and asked to what extend he trusts the page on the 5-point scale.
Sites that enter this process must be well selected in advance for they have to be representative of all possible sites to come. This means that it has to comprise a significant part of the feature space. If this is not done, some features may have been left out of the training of the profile will therefore not be properly recognised later on.

After this, the agent can be trained on this data (if necessary) and run from there on. The agent should continue to learn forever. In practice the latter does not happen. For 2 reasons:

1. It was decided that the program would not use explicit feedback from its user. This means that it will not ask the user constantly if it agrees because this would take away too much of the idea that the program actually advises its user. In other words it has no idea if, what it advised, was correct.
   There is a button on the result screen that allows the user to give direct feedback, but it does not need to be used always.

2. The button mentioned above gives the user the opportunity to say that he did not agree with the judgment of the agents. When he presses it to give his own input, this will not be trained upon, but merely added to the history. This means that the next time the user visits this page, he will see his own rating again with a very high confidence level; which makes sense.
   The agents do not update their profiles (by training) because it would take too much computing power when users would do this constantly on a server. Especially when the data sets grow, the training of the Neural Network will take too much time.

5.4 Implementation

Now that all parts of the prototype have been discussed, a more detailed view of the implementation can be given. It will be split up in the following parts:

1. User interface & interaction
2. Calculating the feature vector
3. The agent
4. The negotiation process

5.4.1 User interface & interaction

The whole user interface of the prototype is based on the Web. It is an online demo that can be run from a browser. In the current implementation all data is stored on a server. This means that the agents (user-profiles and history) are stored in a place that the users often cannot reach. For a true online demo, this is the only thing that should be changed.

All of the pages that make up the interface are dynamic. They are constructed by a series of CGI scripts. These scripts are all written in `tcsh` and C. In order of execution:

1. `index.cgi` reads the installed agents from a configuration file and displays checkboxes together with the URL field. This screen is shown in Figure 5.3 on page 42.
   The result from this page is put into another file and supplied as a parameter to the script that will rate the site (see next section).

2. After all the agents have done their work and their scores together with the negotiated one are known, `make_pages.tcsh` is run. This script actually makes the frame for the whole result page as shown in Figure 5.1 on page 41. The head contains the sitename (`make_header.tcsh`) and the left side the results of the agents and the overall score read from a file (`make_rating.tcsh`). This frame also includes the button for feedback (discussed below). The chosen site is included on the right side.

3. If the user disagrees with the agents' score, he can press the button on the results page to bring him to the training page. This page is constructed by `disagree.cgi` and enables the user to give a different score to the site, which is then added to the History of his agent[4]. The profile of the user's agent can now be recalculated to reflect the change. See Figure 5.9 for a screenshot.

Figure 5.9: Adapting the user's agent by feedback

5.4.2 Calculating the feature vector

When the URL is entered by the user, the site is rated by the extractors. These are the same for all agents and therefore done only once for every run. A number of things happen:

[4] The other agent's profiles are not updated, they are not his to train.

1. The page indicated by the user is downloaded using w3c. This is a command line browser developed by the WWW Consortium (http://www.w3.org). It saves the page in three ways: original HTML code, HTML tags and visible text. These three files represent different ways of looking at the site (see Figure 5.10). The HTML code contains everything, the HTML tags file is an interpreted version of the HTML code. This means that comments and errors have been taken out. The text version finally, only contains visible, plain text.

2. The page is ready to be rated. For this process another script (rate_now.tcsh) is started. This script, the extractor interface, will sequentially run all underlying extractors and write their results to a file: this is the feature vector of the site.

3. All these extractors are scripts as well. They can be implemented in a different way as long as they follow the rules set by this script: <script_name> URL HTML_FILE TAG_FILE <optional>. The latter is used for the text file mentioned before or any another parameter.
   Some of the extractors also use external sources of information. The 'number of references' extractor, for example, has to interact with the AltaVista site and parse the resulting page to come up with its result. Details of the extractors can be found in the implementation.

HTML

```
<A HReF="somewhere/index.html"><img src="pics/picture.
jpg"></a>
Look at the picture!
or just click <a href="somewhere_else/index.html">here</A>
```

HTML Tags

```
A HREF=somewhere/index.html
IMG SRC=pics/picture.jpg
A HREF=somewhere_else/index.html
```

Plain text

```
Look at the picture!
or just click
here
```

Figure 5.10: Three different formats for a site: HTML, TAGS and Plain text

5.4.3 The agent

Once the site is transformed into the feature vector the agent can start to examine the page. As mentioned before, the agent first looks in its History file for the URL. If this URL is present, it has a rating given by its user. This means that it is probably correct. It will return this answer together with a confidence level of 95%. However, if the URL is not found it will have to infer from the profile.
There are 3 different agents that use their own methods to infer trust from this knowledge. Together with their trust rating they return their confidence level. All agents are called with the same parameters, so a new type is easily added.

The ANN agent

The profile of this agent is a map of a trained neural network. Because this program was built in from another party some adjustments to the data were needed around the executable:

1. The data is scaled to the $[0, 1]$ interval. The network requires this. It is scaled with respect to a vector constructed from 77 example sites. This vector represents the maximum of all elements of these sites. In this way, all sites are scaled equally.

2. The network is run with this scaled feature vector as input. The output is a floating point number, again in the $[0, 1]$ interval, which is put into a file.

3. This floating point number is scaled and rounded onto the right interval. The confidence level is calculated as the difference between the scaled and the rounded number.

The k-NN agent

This agent is written in C++. The main two classes are the Case and the Casebase (not to be confused with Case-based Reasoning). They are depicted in Figure 5.11.

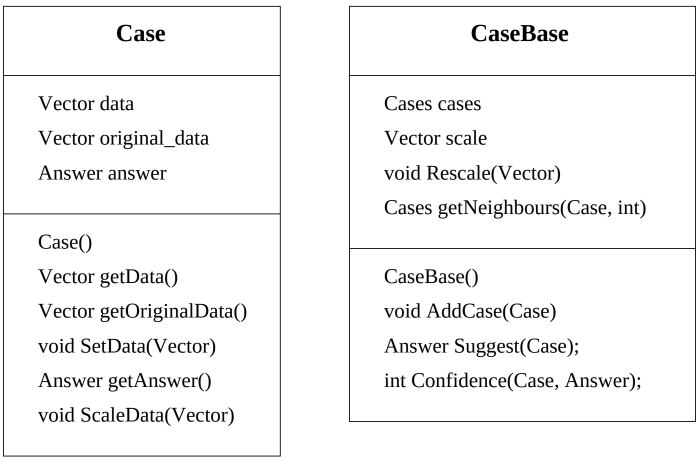

Figure 5.11: The two main classes from the k-NN: Case & CaseBase

A Case is a site. When taken from the History, it consists of a ratings vector (data) and an Answer. It has some methods to get and set data.
For reasons of comparison, this method also only uses scaled data. The ScaleData function will scale this Case to the level of the rest of the CaseBase, in which all the sites are found.
When the agent is asked for an answer it will call Suggest with the site ratings. Using the getNeighbours it will get the neighbourhood. Based on this neighbourhood it will come up with an answer. After this, the Confidence method is used to calculate the confidence level for this answer.

The Decision Tree agent

Again C++ is used for the implementation, but no classes are used for this agent. In Figure 5.12 on the next page the main part of the program is displayed to show its workings.

```
int main(int argc, char **argv) {
  History history; // Will hold all sites and answers
  Ratings ratings; // Will hold the ratings of this site
  if(ReadProfile(argv[1],history) != 0) return 1;
  if (ReadRatings(argv[2],ratings) != 0) return 2;
  int depth = 0, answer = -1;
  float confidence = 0.0;
  bool found = false;
  History filtered = history, old_filtered;
// Go through every leaf (extractor) of the tree until an answer is found
// or the end is reached.  The confidence level is made up of 2 factors:
// - Diversity:  in the answers left when this one was chosen
// - Depth:  the deeper we travel into the tree, the more confident we are
// Both of these factors make up 50% each
  while ((depth < NUM_FEATURES) && (!found)) {
// Go down the tree:
    old_filtered = filtered;
// After every leaf, the history is filtered for elements that have the
// same ratings.  So with every iteration the same or less remain
    filtered = Filter(filtered,ratings,depth);
    if(filtered.size() == 0) {
// If there are no elements left, we have to backtrack one step to the
// old_filtered history and come up with an answer from that information
      answer = Decide(old_filtered);
      confidence = Diversity(old_filtered,answer);
      found = true;
      depth--;} // Decrement the depth because we backed up one level
    if (filtered.size() == 1) {
// If only one answer is found, this is the one
      answer = filtered.front()->answer;
// The diversity is maximal since there is only one answer, so 50%
      confidence = 0.5;
      found = true;}
    depth++;}
  if (answer == -1) {
// If the end of the tree was reached without finding an answer,
// decide on what's left
    answer = Decide(filtered);
    confidence = Diversity(filtered,answer);}
// Add the depth to this confidence level
  confidence += (float)(depth - 1) / (float)(NUM_FEATURES-1) / 20;
  cout << answer << "#" << Round(100*confidence) << endl;
  return 0;}
```

Figure 5.12: The main algorithm of the Decision Tree agent

After some initialisations, the system goes down the tree, comparing the site with the sites in the History. All comparisons are done by labels. The Filter function mimics the descending: it takes out all sites from the list that do not have this label in common.
When the system encounters an empty set (no more leafs in the tree) it backtracks one leaf and decides on this. This is done by the Decide function. This function returns an answer based on the algorithm described in Section 2.5.3.
All the answers and confidence levels of the agents are put into one file: the blackboard. This will be explained in the next section.

5.4.4 The negotiation process

From all the answers & confidence levels of the agents, the negotiation process makes 1 answer. This process is implemented in C++ after Figure 5.5. Its primitive functions are depicted in Figure 5.13. After it has read the agent results from the blackboard, it will execute the code depicted in Figure 5.14 on the next page.

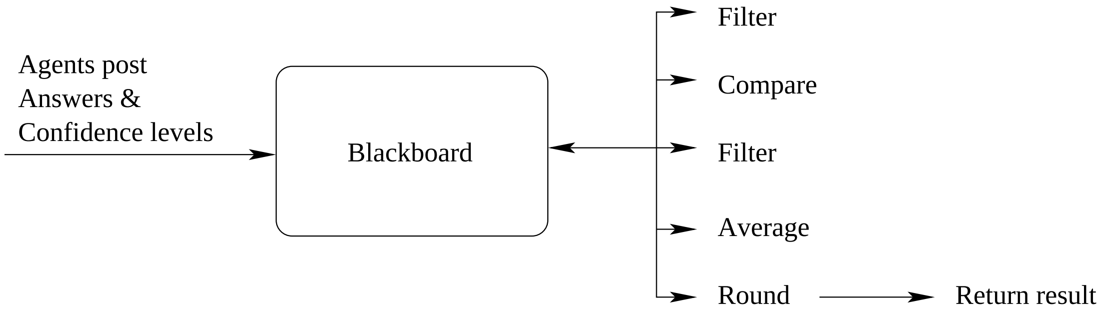

Figure 5.13: Functions act on the blackboard

```
NUMBER_OF_ANSWERS = 5 // (Not at all, A little, Neutral, A lot, Absolute)

// Define the threshold for an answer.  If the agent is less certain
// than this percentage, its answer is discarded.
UNRELIABLE = 30

// When the confidence is above this threshold, it is very sure.
VERY_RELIABLE = 94

// If high reliable answers are found, this boolean will tell.
bool HighReliable = false;
// Find out if there are Higly Reliable answers
While (Agents && !HighReliable)
  HighReliable = Agent.Confidence > VERY_RELIABLE

// The minimum threshold for agents to participate is greater than UNRELIABLE
int MinimumConfidence = UNRELIABLE;
// If there are High reliable agents, the threshold will be higher,
// only they can participate.
if (HighReliable)
  MinimumConfidence = VERY_RELIABLE;

// This array holds the total confidence level for each possible answer
int ConfidencePerAnswer[NUMBER_OF_ANSWERS];
int TotalConfidence; // and this one the the overall total.

// Add the Confidence Levels for all agents allowed to participate.
For all Agents
  if(Agent.Confidence > MinimumConfidence)
    ConfidencePerAnswer[Agent.Answer] += Agent.Confidence;
    TotalConfidence += Agent.Confidence;

float AverageAnswer = 0.0; // Will hold the nominator of the average

// Calculate the weighted Answer and round it properly.
For all Answers
  AverageAnswer += Answer * ConfidencePerAnswer[Answer];
return Round(AverageAnswer / TotalConfidence);
```

Figure 5.14: The negotiation process in pseudo-code.

Chapter 6

Test results

In this chapter all topics introduced in this thesis will be combined. Based on the data from the second survey, the 3 methods will be tested on the quality of their inference, learning rate and robustness. A benchmark will be introduced to be able to measure if the whole system actually does better than some 'guessing techniques'. This benchmark is also important for improving the system.

6.1 Comparison of the methods

In this document, 4 methods have been introduced for inferring Trust from the user profile: ANN, k-NN, Decision Tree (DT) and combinations of them by ways of negotiation. The latter actually consists of 4 methods, because there are four ways of combining these methods: ANN & k-NN, ANN & DT, k-NN & DT and all three. Yielding a total of 7 methods.

Using the survey results as input data, these 7 methods have been evaluated. An experiment was setup using a sliding window: From the 50 ratings each respondent gave, 40 were randomly selected to train on. The other 10 were used to test. In this way, the applications sees 80% of the behaviour and has to predict the other 20%.
As a measurement for the error of the system, the absolute difference to the correct answer was used. So if the system returned '3' and the answer given by the respondent was '4', the error would be 1.
A summary of these errors is given in the tables in Appendix C. In this chapter, multiple trainsets have been used (as will be explained in the next section). Trainset D is the one used for this measurement: 40/10. Each table shows the average error, the maximum error and the distribution of the errors made in the 10 predicted answers.

All these tables are averaged in Table 6.1 on the following page to get the overall performance. It presents the average over the 10 respondents that returned *all* ratings.
The average error mentioned in the first row is taken from all the errors. It can also be calculated (by definition) by taking the *expected* error based on the distribution shown in the lower 5 rows. The 'correct' row is equal to 'error = 0'. For the Neural Network column:
$1.030 = 0 * 0.32 + 1 * 0.43 + 2 * 0.18 + 3 * 0.04 + 4 * 0.03$

All (primitive) methods also have a confidence level with their answer. This signifies how sure the system is of its answer given its data. In Table 6.2 on the next page the averaged results for

Table 6.1: Averaged results for trainset D

	ANN	*k-NN*	*Tree*	*A & k*	*A & T*	*k & T*	*All*
Average error	1.030	0.840	1.100	0.860	1.120	1.030	0.960
Max error	2.4	2.2	2.3	2.3	2.5	2.3	2.4
# **Correct**	32%	39%	33%	39%	32%	34%	41%
# error = 1	43%	45%	41%	44%	41%	43%	35%
# error = 2	18%	11%	13%	11%	14%	12%	14%
# error = 3	4%	3%	9%	4%	9%	8%	7%
# error = 4	3%	2%	4%	2%	4%	3%	3%

this trainset are given. Again with the average, a distribution of the levels is given in intervals.

Table 6.2: Average confidence levels for trainset D

Average	50%	69%	77%
# > **75%**	29%	47%	68%
# [50,75] %	12%	46%	27%
# [25,50] %	39%	7%	5%
# < 25%	20%	-	-

It is important to take these confidence levels into account when comparing methods because the *combined* methods are based on both the ratings and the confidence levels. When an agent is very confident it is more likely to be taken into account with the overall answer and also has a higher weight in the calculation as explained in the implementation.
Given these two tables a comparison can be made. The only remaining problem now is the definition of quality of a method: how can they be compared?
Two metrics for the quality will be used:

1. The **Average error** gives an overall measurement of the answers given. When some answers are far out (outliers), it will be heavily influenced. It does not say a lot about the number of correct answers.

2. **Precision** is the 'correctness' of the method. It signifies the number of answers that were actually correct. As opposed to the average error, this metric does not say anything about the other answers.

Combined, these two metrics give a good insight in the quality of a method.

The k-Nearest Neighbour scores best with respect to average error (0.840). An answer taken from both an ANN and a k-NN agent give approximately the same result (0.860). This is due to the fact that the Neural Network is not very confident. In at least 20% of the cases, its answer is below the threshold of 30% and will not be taken into account at all. In the other cases, the k-NN will be more confident and therefore make up most of the answer.
With this insight, the bad results of combinations with a Decision Tree can also be explained (column 5 and 6): although this method scores very bad on average error, it is very confident. With its bad answer it still convinces the system.

Fortunately the precision score gives the same best method. The k-Nearest Neighbour is right in 39% of the cases. All agents together even score 41% precision. Their average error is a bit higher (0.960) than only k-NN.
This introduces a need for an overall ranking. This metric will be calculated as follows:
Combined quality metric = Average error * (1 - Precision)
The lower this number, the better the method. Values yielded from this metric are displayed in Table 6.3. The Nearest Neighbour has the lowest value and will therefore be considered the best method.

Table 6.3: Combined quality metric

ANN	*k-NN*	*Tree*	*A & k*	*A & T*	*k & T*	*All*
0.70	0.51	0.74	0.52	0.76	0.68	0.57

6.1.1 Random agent: benchmark

After having seen that the k-Nearest Neighbour has performed best on all the data supplied, it is interesting to see how well it did compared to some benchmark. A benchmark is a measurement that can be seen as the bottom line. Any method thought of, ought to be better than this benchmark. In the case of the architecture implemented a random agent would be a good one. The application has to (at least) outperform an agent that gives random answers.

The experiment was done on the basis of the data supplied by the respondents and two distributions. The first random agent gave uniformly distributed answers and the second one approximately normally distributed answers as depicted in Table 6.4

Table 6.4: Probability distributions for answers given by the random agents

	Answers				
Agent	*1*	*2*	*3*	*4*	*5*
Uniform (1)	0.2	0.2	0.2	0.2	0.2
Normal (2)	0.1	0.2	0.4	0.2	0.1

The distribution of the answers of a random agent is P_{agent}. The distribution of the answers from the respondents can be calculated and will be called P_{data}. With these two distributions a predicted agent can be setup by calculating its expected errors: (A = Answer)

$$
\begin{aligned}
P\{\text{error} = 0\} = P\{|A_{agent} - A_{data}| = 0\} &= \textstyle\sum_{i=1}^{5} P_{agent}\{A = i\} * P_{data}\{A = i\} \\
P\{\text{error} = 1\} = P\{|A_{agent} - A_{data}| = 1\} &= \textstyle\sum_{i=1}^{4} P_{agent}\{A = i\} * P_{data}\{A = i + 1\} + \\
&\quad \textstyle\sum_{i=2}^{5} P_{agent}\{A = i\} * P_{data}\{A = i - 1\} \\
P\{\text{error} = 2\} = P\{|A_{agent} - A_{data}| = 2\} &= \textstyle\sum_{i=1}^{3} P_{agent}\{A = i\} * P_{data}\{A = i + 2\} + \\
&\quad \textstyle\sum_{i=3}^{5} P_{agent}\{A = i\} * P_{data}\{A = i - 2\} \\
P\{\text{error} = 3\} = P\{|A_{agent} - A_{data}| = 3\} &= \textstyle\sum_{i=1}^{2} P_{agent}\{A = i\} * P_{data}\{A = i + 3\} + \\
&\quad \textstyle\sum_{i=4}^{5} P_{agent}\{A = i\} * P_{data}\{A = i - 3\} \\
P\{\text{error} = 4\} = P\{|A_{agent} - A_{data}| = 4\} &= P_{agent}\{A = 1\} * P_{data}\{A = 5\} + \\
&\quad P_{agent}\{A = 5\} * P_{data}\{A = 1\}
\end{aligned}
$$

Table 6.5: Distribution of answers in the dataset from respondents

1	*2*	*3*	*4*	*5*
0.116	0.212	0.324	0.256	0.092

P_{data} is based on all answers given by the respondents and is given in Table 6.5.
Based on the distributions and the equations, the 3 metrics for the quality of the random agents can be calculated:

- Average error = E(error) = $\sum_{i=1}^{5} i * P\{\text{error} = i\}$
- Precision = $P\{\text{error} = 0\}$
- Combined quality metric as before: Average error * (1 - Precision)

Table 6.6 compares the methods.

Table 6.6: Comparison of k-NN with random agents

	k-NN	*Uniform*	*Normal*
Average error	0.840	1.460	1.237
# Correct	39%	20%	24%
# error = 1	45%	36%	41%
# error = 2	11%	26%	24%
# error = 3	3%	14%	9%
# error = 4	2%	4%	2%
Combined metric	0.51	1.17	0.94

The comparison of the combined metrics of the best implemented method with the two benchmarking agents shows that the k-Nearest Neighbour outperforms them by 56% and 46% respectively.

6.1.2 Conclusions

This section has shown that the k-Nearest Neighbour performs best on the data provided by the respondents of the survey and its sites with the current implementation. The main notes that have already been made here are that the quality of the data from the respondents can be questioned. Moreover, not all extractors have been implemented: the application does not have full 'visibility'.

The other conclusion that can be drawn is that even the worst method (combination of ANN & Decision Tree) performs better than the benchmarks set by the random agents. The best one (k-NN) outperforms the normally distributed random agent by 46% measured in the combined metric defined earlier.

A few other points that can be noticed from this section are the fact that the confidence level heavily influences the outcome of the combined methods:

- The Neural Network often has a low confidence level (in 59% of the cases below 50%) and is therefore not weighted very well in negotiations. Even if its score is good, the level is not very high.

- The Decision Tree does not perform very well, but *is* very sure of itself. This means that its answers are weighted heavily in the negotiations which therefore results in bad answers.

6.2 Quality of learning

In addition to find the best method, it is important to see if it is actually capable of getting better. If it learns from data presented.
As can be seen in Appendix C, 4 trainsets have been used. Every set had a different trainset to runset ratio. Trainset A had 10 sites, 40 to predict, set B 20/30, C 30/20 and D 40/10 as mentioned in the previous section. These sets were used to test if there was an upward trend in the quality of the predictions. The summary of these tests is shown in Tables C.9, C.10, C.11 and C.12. Here, in Table 6.7 these results are expressed in the combined metric as defined in the previous section.

Table 6.7: Comparison of trainsets A, B, C and D by combined quality metric

	ANN	*k-NN*	*Tree*	*A & k*	*A & T*	*k & T*	*All*	*Total*
Trainset A	0.78	0.74	0.69	0.71	0.71	0.75	0.72	5.10
Trainset B	0.82	0.61	0.69	0.66	0.70	0.64	0.61	4.73
Trainset C	0.81	0.74	0.77	0.72	0.78	0.73	0.75	5.31
Trainset D	0.70	0.51	0.74	0.52	0.76	0.68	0.57	4.48

In every row, the quality measurements are displayed combined with their total score in the last column. This score is the sum of all elements in the row. From this table it is clear that for the ANN and k-NN, their lowest score (=best) is indeed with the biggest trainset D. The Decision Tree does not have this. It blurs all the combined scores in which it participated as well.

Except for trainset C, which is probably not chosen very well (its score is higher than its A and B), there is an increasing trend in quality of the answers: the more traindata, the better the predictions. For the k-Nearest Neighbour the increase in quality is 18% from set A to B and 16% from B to D. Both times the trainset size increased with 50%.

6.2.1 Conclusions

- Again the Decision Tree method decreases results. It does not increase the quality of its answers when the trainset is increased. As a matter of fact, it does the opposite. With this, it also decreases quality of the methods in which is participates.
- The other two methods (ANN & k-NN) *do* increase in quality. The method pointed out as the best in the last section, k-NN, increased the quality of its answers measured in the combined metric by 16% and 18% when doubling the trainsets.
- Trainset C was not very well chosen. It is not used in the statistics.

6.3 Robustness of the methods

In this section the results of the robustness test is discussed: what does the application do when the data is a little different? In this section the quality of the prediction is tested when the site is slightly changed.

6.3.1 Altering the site

Many of the respondents mentioned the fact that a site could be trusted because it looked like another one. If this similarity is purely mathematical, then this test will show if the application can handle this. The feature vector of one of the sites was changed by 10% to see if the system would still come up with the same answer as with the original.

The comparison showed that only the Neural Network gave different answers and even these were small. In 4 cases (out of 10) it gave a difference of 1 in the case where the ratings were increased by 10%. The other case where the ratings were decreased by 10%, there were only 2 of these cases where the difference was 1.
It looks as if the methods handle change in the same way, but this is not true:

- The differences noted for the *ANN* with its originals are again due to the fact that it has to take all its sites into account: it is a weighted average of all of them. Even a slight change has an effect on this average.
- For the *k-NN* the fact that nothing changed is because it uses an area to decide its answer. If the site stays within this area, the answer will remain the same. Within the area the answer is not, in the first place, based on distance from the site offered. It is based on the majority of answers and that does not change by altering the ratings.
- The *Decision Tree* is clearly aided by the fact that it uses labels. The label intervals are big enough to cover the change in ratings: they remain the same and will therefore yield the same answer.

6.3.2 Conclusions

Sites that are similar to each other will be recognised very well. The k-NN and the Decision Tree even made no mistakes at all.

6.4 Conclusions

In this chapter the learning methods as proposed by Chapter 2 have been tested and 3 important properties of the system have been examined.

The tests that have been performed suggest that the k-NN technique works best on this system. Both the ANN and the Decision Tree performed less in the comparisons. The combinations of methods (results from agents negotiating) are not better than the pure ones. This fact is partly due to the confidence levels of the agents, they do not accurately match the actual quality of their answer: a bad method like the Decision Tree is overconfident for example.
Fortunately it can also be said that even the worst method (combination of ANN & Decision Tree) performs better than the benchmarks. These benchmarks were set by the random agents that were introduced.

Apart from the Decision Tree, the other two perform better when the trainset grows: they learn. Finally the tests have shown that a slightly changed site is still recognised. When a site is similar for a human, it may also be for the system.

Chapter 7

Conclusions & Recommendations

The research presented in the Background Chapter together with the surveys make up the foundations for the Feature Model. This model explains how people assess their trust in a site and what can be done to measure this. It looks at aspects of sites that people indicated were important to trust.
This model is used later on as an input to a user-profiling system: the agent. This agent weighs the aspects of the page according to the user's behaviour. It knows which pieces of a website are important to its user. When the agent has generated an answer based on the site and the profile, it can negotiate with other agents to get a trust assessment from different angles.
For the implementation of this model a number of things had to be investigated, designed and implemented. This thesis presents an overview of this work. For the field of commercial websites we

- validated related research and laid the basis for the Feature Model by surveying people;
- defined this Feature Model that contains a full list of aspects of sites that people take into account when assessing their trust in a site, together with a way of measuring them;
- designed and implemented an agent, based on the Feature Model that can predict a person's trust in a site;
- selected and tested 3 methods that were used for learning the trust behaviour of a user;
- designed and implemented a platform on which these agents negotiate on their trust measurements and come up with an overall trust indication;
- combined this into an online demo for trusting websites.

Conclusions

From this work a number of conclusions can be drawn. The main conclusions are presented below:

Conclusion 1: We found strong support for our Feature Model describing properties of sites that influence trust behaviour. It presents a full listing of features that appear on, or are related to websites.

Conclusion 2: Using this Feature Model, it proved possible to automate the labeling of commercial websites with respect to trust. The agents that were implemented, automated the way humans assess trustworthiness of a website by learning from their behaviour.

Conclusion 3: k-Nearest Neighbour is the best method of the three tested to learn from human trust behaviour. Even combinations of these methods (by negotiation) did not perform better.

Recommendations

This thesis presents a blueprint for a bigger WebTrust project. It has defined the foundations and identified some of the difficulties. For further research a number of things can done:

1. The agent needs more extractors to measure all the features in the model. Once this is done, better results will be achieved.

2. Other AI methods can be used to train on the trust data presented to the agent. Case-based reasoning would be an interesting choice since it is, in fact, an extension to k-NN.

3. A more elaborate negotiation scheme needs to be designed (including the confidence levels of agents) to make negotiation resemble the human process better.

4. It is important to prove that the features are really the cause of the ratings. To do this, a survey can be done with manipulated sites (i.e. manipulated feature vectors). In this controlled environment, respondents have to rank sites to trust level. A better insight in the cause and effect of features will be given by this.

5. The WebTrust application can be used as the online service proposed in the beginning of this thesis. When users upload their agent profiles they can start using the application to rate websites for personal use.

6. The surveys have shown that trust is independent of textual content. A more international test of the system could prove that it is indeed globally applicable.

7. Rule extraction or data-mining can be used on the user profiles to find heuristics on trust. The common factors in these profiles hold information on what people *in general* find trustworthy on a website. This information can be used to improve commercial websites.

8. When extended with features that go beyond the ones people take into account, WebTrust can be taken to a higher level advising on trust. It will be more like a decision support system using information from all over the Internet to prevent the user from trusting only on what he sees.

References

[1] Berkeley. Evaluating web pages: Experience WHY it's important. Technical report, Berkeley University Library, Sept. 2000.
http://www.lib.berkeley.edu/TeachingLib/Guides/Internet/EvalForm.pdf (Nov. 2000).

[2] D. Billsus and M.J. Pazzani. Revising user profiles: The search for interesting web sites. In *International Multi-Strategy Learning conference*, 1996.
http://www.ics.uci.edu/~pazzani/Publications/RevUserProfiles.pdf (Dec. 2000).

[3] K. Brancolini and E. Okada. Critical thinking and the World Wide Web. Technical report, Indiana University Library, July 1997.
http://www.indiana.edu/~librcsd/eval/checklist.html (Nov. 2000).

[4] Webpage Builder.
http://www.geocities.com/EnchantedForest/3278/webpagebuilder.html.

[5] Cheskin. Ecommerce trust study. Technical report, Cheskin Research, Jan. 1999.
http://www.studioarchetype.com/cheskin/assets/images/etrust.pdf (Dec. 2000).

[6] Cheskin. Trust in the wired americas. Technical report, Cheskin Research, July 2000.
http://www.cheskin.com/think/studies/trustIIrpt.pdf (Oct. 2000).

[7] M. Deutsch. Cooperation and trust: Some theoretical notes. In *Nebraska Symposium on Motivation*, 1962.

[8] F. Egger. Human factors in electronic commerce: Making systems appealing, usable & trustworthy. In *Graduate Students Consortium & Educational Symposium, 12th Bled International E-Commerce Conference*, June 1999.
http://www.ipo.tue.nl/homepages/fegger/bled99.htm (Oct. 2000).

[9] F. Egger. Towards a model of trust for E-Commerce system design. In *CHI2000 Workshop: Designing Interactive Systems for 1-to-1 E-commerce*, Apr. 2000.
http://www.zurich.ibm.com/~mrs/chi00/contributions/egger.html (Oct. 2000).

[10] Website Garage.
http://websitegarage.netscape.com.

[11] E. Gerck. Toward real-world models of trust: Reliance on received information. Technical report, Meta-Certificate Group (MCG), 1998.
http://www.mcg.org.br/trustdef.htm (Dec. 2000).

[12] E. Grassian. Thinking critically about World Wide Web resources. Technical report, UCLA College Library, Aug. 1998.
http://www.library.ucla.edu/libraries/college/help/critical/index.htm (Nov. 2000).

[13] A.H. Harcourt. Cooperation and trust in animals. pages 15–26 of: Hinde, Robert A.& Groebel, Jo (eds), cooperation and prosocial behaviour., 1991.

[14] Webdevelopers Journal.
http://webdevelopersjournal.com/software/linkbot.html.

[15] B. Krulwich. Lifestyle finder. intelligent user profiling using large-scale demographic data. *AI Magazine*, 18, Summer 1997.

[16] N. Luhmann. Trust and power, new york: Wiley., 1979.

[17] N. Luhmann. Familiarity, confidence, trust: Problems and alternatives. chap. 6, pages 94–107 of: Gambetta, diego (ed), trust. blackwell., 1990.

[18] S.P. Marsh. *Formalising Trust as a Computational Concept.* PhD thesis, University of Sterling, UK, Apr. 1994.
http://www.iit.nrc.ca/˜steve/pubs/Trust/PhD/Trust.pdf (Dec. 2000).

[19] T.M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[20] Netmechanic.
http://www.netmechanic.com.

[21] M.J. Pazzani, J. Muramatsu, and D. Billsus. Syskill & Webert: Identifying interesting web sites. In *Proceedings of the National Conference on Artificial Intelligence*, 1996.
http://www.ics.uci.edu/˜pazzani/Publications/aaai-webert.pdf (Nov. 2000).

[22] S. Russel and P. Norvig. *Artificial Intelligence, a modern approach.* Prentice Hall, 1995.

[23] K. Schrock. The ABCs of website evaluation, Jan 1999.
http://www.capecod.net/schrockguidei/abc.pdf (Oct. 2000).

[24] TRUSTe.
http://www.truste.com.

[25] VeriSign.
http://www.verisign.com.

Appendix A

Survey 1

A.1 Layout

Introduction

I'm working on a filter for search engines that will give its user only those pages that contain trustworthy information. Trust can be approached from 4 different angles (everything is commentable on this page, so please do so at the bottom of this page if you feel that this is not correct):

- *(e)Commerce*: Would I buy anything on this website?
- *Information*: Would I use information provided on this website?
- *Opinion*: Do I consider what this website tells me as true?
- *Personal*: Would I give this website any personal information?

I will ask you about the things you trust from these four angles (see Sites). Afterwards I will ask some questions on how you got these values of sites that you trusted. This will only take about 15 minutes. Thank you very much.

Sites

Below, I will present you with 5 sites per category (trust angle). Please take a good look at all of them and ask yourself the question mentioned in the topic. Don't look too long at a page, if you would not do that normally. You can answer "A lot", "Neutral" and "Not at all". Of course some of the information presented on the page may not be interesting to you, but try to place yourself in the seat of somebody who *is* interested.

Commerce

"Would you buy anything on this website?"

- Door to Door http://www.doortodoor.com/mainframe.htm[1]
- Launchpad 2000 http://www.launchpad2000.com

[1]Pages may have changed by now

- Silverbank http://www.silverbank.com
- Amazon http://www.amazon.com
- Spacepen http://www.astropen.com

Information

"Would you use information provided on this page?"

- Encyclozine http://encyclozine.com
- City Journal http://www.city-journal.org
- Radiocarbon http://www.radiocarbon.org
- Encarta Encyclopedia http://encarta.msn.com/reference/
- Bushido online http://www.bushido.ch

Opinion

"Do you consider what this website says as true?"

- Herald Tribune http://www.iht.com
- PeaceWatch http://www.ariga.com/peacewatch/index.htm
- Dauten, columnist http://www.dauten.com
- Database on World Happiness http://www.eur.nl/fsw/research/happiness/
- Earthwatch Weather http://www.earthwatch.com

Personal

"Would you give this site any personal information?"

- eBay http://www.ebay.com
- Killer roses http://www.killerroses.com
- XOIP http://www.xoip.co.uk
- FlowerHut http://www.flower-hut.com
- ePoll http://www.epol.com

How did you do that?

It already tells me a lot *that* you like a page, but I'm also interested in how you came to this answer. In the first place I would like to know how the process of evaluation went (do you go for first impression etc) and then what you physically looked at when reviewing a page. Please rank the following statements in the order you use them to evaluate pages:

Driver	*Rank*
First impression	-
Experience with the product / company	-
Information from friends / trusted parties (e.g. consumer guides)	-

Do you have any other suggestions?
What did you look at when evaluating the site?
Do you have any other comments / tips?
Finally I would like to have your Notes id to make sure that I get every answer only once.
Notes id:

Press the button if you think that everything is done. Thanks a million, Wouter

A.2 Results

On the following pages the results of the survey are displayed in 4 tables:

1. **Site ratings**: are the scores people gave to all the sites. They numbers 1, 2 and 3 are numerical representations of "Not at all", "Neutral" and "A lot".
   The sites listed in the first column fall into the four different categories mentioned earlier: Commerce, Information, Opinion and Personal. With every category the question to the respondent was different: he had to take another angle of trust.
2. **Driver rankings**: displays the order in which people evaluated the site. The numbers range from 0 (last ranking or not taken into account at all) to 3 (highest ranking: evaluated first).
3. **What did they look at?**: here respondents were free to fill in which properties of the sites they looked at to make up there minds. The table displays the results
4. **Comments**: general comments made during the survey are displayed here.

Table A.1: Site ratings

Site	*Respondent*										
	1	2	3	4	5	6	7	8	9	10	Category
Door to Door	3	2	3	2	3	3	3	2	2	2	Commerce
Launchpad 2000	2	3	2	2	3	2	3	2	2	3	
Silverbank	2	1	1	3	3	1	3	1	1	1	
Amazon	3	3	3	3	3	3	3	3	3	3	
Spacepen	2	2	2	2	3	2	3	1	1	2	
Encyclozine	2	1	-	1	3	2	3	3	2	2	Information
City Journal	2	1	-	1	3	1	3	2	2	3	
Radiocarbon	3	1	-	3	3	2	3	3	1	2	
Encarta Encyclopedia	3	2	-	3	3	3	3	3	3	3	
Bushido online	3	2	-	2	3	2	2	1	2	2	
Herald Tribune	3	2	3	-	3	3	3	3	3	3	Opinion
PeaceWatch	2	2	2	-	2	1	2	2	1	3	
Dauten, columnist	1	2	2	-	3	1	2	1	2	2	
Database on World Happiness	1	2	2	-	3	2	2	1	1	3	
Earthwatch Weather	3	2	2	-	3	1	3	2	2	3	
eBay	2	3	2	1	3	2	3	3	2	1	Personal
Killer roses	2	2	2	1	3	1	3	1	1	2	
XOIP	2	3	2	1	3	1	2	1	1	2	
FlowerHut	2	2	2	1	3	1	1	2	1	1	
ePoll	2	2	2	1	3	2	1	2	1	3	
1 = Not at all, 2 = Neutral, 3 = A lot											

Table A.2: Driver rankings

Driver	*Respondent*									
	1	2	3	4	5	6	7	8	9	10
First impression	1	1	2	3	3	2	2	3	1	3
Experience with product	3	3	3	3	0	3	3	2	3	1
Information from friends	2	2	1	3	0	1	1	1	2	2
0 = Lowest rank, 3 = Highest rank										

Table A.3: What did they look at?

Respondent	
1	- The organisation behind a site. For example a university seems trustable. - The level of specialisation, for example a site on Bushido made by a guy that has it as a hobby seems more trustable than a very general website on science that pretends to know everything in the field of science. - The way people make the money. I often ask myself how a certain website make its money when it gives away free information. In other words: "free seems crap", except when it's supported by another type of media (Herald Tribune = paper media for example). - Identically a site that *sells* something is somehow more trustable, because if they serve you bad, they loose clients. - The name, if I know it I am more inclined to believe or trust them (which is stupid, I admit). - The banners, like "Secure Payments" "secure site" because it means that they care about it somehow.
2	-
3	- Known real world brand. Graphical design (i.e. has money been spent on this site). - What questions are being asked (if they ask my sexual orientation or whether I own expensive equipment at home, I will get suspicious)
4	Performance
5	Did it look like a 'real' site? All the URLs were reasonable rather than www.blob.ac.za/^rsole/ripoff.htm
6	Professional Design vs Bad taste, fancy graphics. Privacy policies etc. Page not working
7	-
8	- Background images graphics and their linked partners affect my decisions heavily. - If it looks cheap or poorly designed or has partners that I can base a decision on, I'll think less of them immediately.
9	Design, look-and-feel. A poorly designed page doesn't make you think they'll be much better with security or with researching information.
10	The main page only

Table A.4: Comments

Respondent	
1	- 30 sec per site is a little bit too short to decide if I trust a website or not. Most of the trust comes from either your experience or the one of friends. - Except Amazon, I had never heard about these sites.
2	-
3	- You need more detail on the scales you use above (i.e.between A Lot and Neutral, there is an awful lot of ground in terms of trust). - Also, from a trust perspective it's more what they want from me rather than who they are, or it's what they tell me rather than who they are...
4	I would trust IHT, the rest would be neutral most likely.
5	- The one I gave neutral, was merely due to the fact that it didn't load in a reasonable time. - You need some context. How much do I need to trust them, for information not much, but if I am going to give them $200 a lot more
6	I would like the options: Trust/Distrust page a little.
7	-
8	-
9	-
10	When you say if I would buy something from a web site and that website either does not come up because their server would be down - although this is not to be taken for granted or crashes in passing from a section to the other I reply (not at all) because I imagine it could go that way when I submit personal information. Where has it gone if the process crashed in the between two steps. Also I think it is reassuring to find some sentence section about security and privacy. Some sites do provide this some other do not bother. When you ask if I could use the information contained on a web site it really depends on what kind of information especially if you would like to take into account the look of such a site. For example I could trust a lot RadioCarbon if I know there is an article I want to read i.e. a priori I am looking for some content it is there then I do not really care about the look and I take that content and read it.

Appendix B

Survey 2

B.1 Layout

```
Dear respondents,

Thank you for your positive responses I appreciate it a lot.

For the process:
I will ask you for 3 things with every site.  When I speak about trust, I mean the
general feeling with the site.  Do you feel comfortable to order something or to
leave your personal information on it, things like that.
```

1. How much do you trust this site on a 1-5 scale:
   1 = not at all, 2 = a little, 3 = neutral, 4 = a lot, 5 = absolutely
2. Did you know this site or company already? With this question I'd like to see how much reputation of a company counts. If you see a site that you know already and either trust or not because of you're experiences, please make not of this; it's very useful.
3. The number one reason for giving this score. Anything that you may find was the reason for rating this site the way you did can be written down here.

```
I have not been able to make a database and form for this email so I'm afraid that
it will be the old-fashioned way:  you can put your answers next to the questions
in the original email.  I will try to make something for this later on.

The sites are attached in a format readable only by MS Internet explorer.  It contains
the full first page of the site.  If you click on any links / pictures, you will
be transferred to the real site (so expect delays), but this is not always necessary;
it depends all on the way you would like to do the assessment.

Site number 1:

How much would you trust this site (1-5)?
Did you know this company or site already (y/n)?
```

```
What was the main reason for the score you gave?

Site number 2:
⋮
```

B.2 Sites

For the whole survey, 50 sites were used. Every respondent received 5 sets of 10 sites during the course of the survey. In the table below, all the sites can be found.
Some of them may not exist anymore or have changed content dramatically. In the survey this problem was overcome by sending the respondents web archives that contained the whole first page of the site. In this way they looked the same for everyone.

Table B.1: Sites that were used in the survey

1	anyknockoffs.com	26	www.mgcpuzzles.com
2	store.palm.com	27	www.theuniquebride.com
3	www.1-800-flowers.com	28	www.venicewatch.com
4	www.1stinballoons.com	29	www.webswap.com
5	www.4-giftbaskets.com	30	www.yoyoguy.com
6	www.amazingsocks.com	31	fujipub.com/macabi
7	www.amazon.com	32	members.iinet.net.au/~htlk/index
8	www.an-electronic-store.com	33	www.angelfire.com/biz/movieposters/index
9	www.ashford.com	34	www.asis.com/radiantheart
10	www.autoacessory.com	35	www.bcn.net/~demaio
11	www.autobytel.com	36	www.cupojoes.ws
12	www.autorentnet.com	37	www.designerfakes.com
13	www.baronbelts.com	38	www.exclamation-2-rolex.com/index
14	www.bestbuy.com	39	www.factory370cigaro.com
15	www.bestnest.com	40	www.flagnet.com
16	www.bestpriceaudiovideo.com	41	www.hooliganweb.com
17	www.blockbuster.com	42	www.jbrau.com/bnb-01
18	www.bn.com	43	www.jugglingstore.com
19	www.booksamillion.com	44	www.sandollars.com
20	www0.jcpenny.com	45	www.seiko-citizen-bulova-wenger.com
21	www.cufflinks.com	46	www.semcycle.com
22	www.easyjet.com	47	www.skyroll.com
23	www.econvergence.net	48	www.thehennapeople.com
24	www.ehats.com	49	www.toddsmith.com
25	www.justzero.com	50	www.topbrands.net

Appendix C

Comparing results

On the following pages all the tables used for comparing the training methods can be found. There are four different tables:

- The first 8 tables are the measurements on the quality of the answers predicted by the prototype. These measurements have been done on 4 different datasets and are split into two parts (two groups of respondents).
  Each table deals with 5 respondents from bottom to top. The columns are the methods used to predict these answers. They are ANN, k-NN, Decision Tree, ANN & k-NN, ANN & Decision Tree, k-NN & Decision Tree and a combination of all of them.
  For every respondent a summary of the answers predicted is given in the form of some statistics:
  - *Average error* is the average error made in all predictions the prototype made. The error is calculated in absolute distance to the answer from the respondent.
  - *Max. error* is the maximum error made in predicting.
  - *# Correct* signifies the number of correct answers given. Error 0.
  - *# error* $= i$ signifies the part of the predictions that had a distance to the respondent answer (error) of i.
- In the following 4 tables these results are summarised per dataset (A to D). They have the same layout as the previous ones.
- Another 2 tables depict the average confidence levels give by the different methods for the datasets and respondents.
  The combined methods do not use confidence levels for this test so here only the three primitives are given. The first row is the average confidence level of this method, for this user, given this dataset and the other four are the intervals over which the levels are spread out.
- The last table is the average confidence level for all of the trainsets and respondents. It gives an overview of how confident the methods were. This will be compared with their predictions: ideally the match in the sense that good answers are given with high confidence levels. See Chapter 6 for the results.

Table C.1: Performance of prototype. Trainset A, participant 1-13

Performance prototype on participant 1, trainset A							
	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	0.875	0.800	0.775	0.775	0.775	0.800	0.800
Max error	2	2	2	2	2	2	2
# **Correct**	35%	33%	33%	35%	30%	35%	35%
# error = 1	43%	55%	58%	53%	63%	50%	50%
# error = 2	23%	13%	10%	13%	8%	15%	15%
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 2, trainset A							
Average error	1.525	1.700	1.450	1.575	1.400	1.625	1.600
Max error	4	4	3	4	4	4	4
# **Correct**	18%	23%	23%	25%	25%	23%	23%
# error = 1	35%	20%	33%	23%	33%	25%	25%
# error = 2	30%	25%	23%	25%	23%	23%	25%
# error = 3	13%	30%	23%	25%	18%	28%	25%
# error = 4	5%	3%	-	3%	3%	3%	3%
Performance prototype on participant 4, trainset A							
Average error	1.075	0.675	0.675	0.650	0.775	0.675	0.625
Max error	3	2	2	2	2	2	2
# **Correct**	23%	38%	43%	40%	38%	38%	43%
# error = 1	50%	58%	48%	55%	48%	58%	53%
# error = 2	25%	5%	10%	5%	15%	5%	5%
# error = 3	3%	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 9, trainset A							
Average error	1.325	1.500	1.050	1.375	1.025	1.350	1.250
Max error	4	4	3	4	3	4	4
# **Correct**	23%	18%	33%	18%	28%	20%	25%
# error = 1	38%	38%	38%	48%	48%	40%	40%
# error = 2	28%	25%	23%	18%	20%	28%	23%
# error = 3	10%	18%	8%	15%	5%	10%	10%
# error = 4	3%	3%	-	3%	-	3%	3%
Performance prototype on participant 13, trainset A							
Average error	1.075	1.000	1.025	1.075	0.975	0.975	1.100
Max error	3	2	2	2	2	2	2
# **Correct**	25%	25%	25%	23%	28%	25%	23%
# error = 1	45%	50%	48%	48%	48%	53%	45%
# error = 2	28%	25%	28%	30%	25%	23%	33%
# error = 3	3%	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-

Table C.2: Performance of prototype. Trainset A, participant 16-22

Performance prototype on participant 16, trainset A							
	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	1.075	0.575	0.875	0.700	1.075	0.650	0.825
Max error	3	2	3	3	3	3	3
# **Correct**	30%	53%	30%	45%	33%	48%	38%
# error = 1	38%	38%	58%	43%	40%	43%	48%
# error = 2	28%	10%	8%	10%	15%	8%	10%
# error = 3	5%	-	5%	3%	13%	3%	5%
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 19, trainset A							
Average error	1.950	1.525	1.600	1.525	1.575	1.700	1.525
Max error	4	4	4	4	4	4	4
# **Correct**	15%	15%	20%	13%	25%	13%	15%
# error = 1	28%	43%	35%	45%	38%	38%	43%
# error = 2	25%	20%	13%	23%	10%	20%	23%
# error = 3	13%	20%	30%	18%	10%	28%	15%
# error = 4	20%	3%	3%	3%	18%	3%	5%
Performance prototype on participant 20, trainset A							
Average error	0.700	0.675	0.725	0.625	0.975	0.750	0.700
Max error	2	2	2	2	4	2	2
# **Correct**	45%	40%	35%	48%	38%	35%	40%
# error = 1	40%	53%	58%	43%	43%	55%	50%
# error = 2	15%	8%	8%	10%	8%	10%	10%
# error = 3	-	-	-	-	10%	-	-
# error = 4	-	-	-	-	3%	-	-
Performance prototype on participant 21, trainset A							
Average error	0.750	0.800	0.900	0.775	0.850	0.850	0.775
Max error	2	3	3	3	3	3	2
# **Correct**	43%	40%	40%	40%	40%	40%	38%
# error = 1	40%	43%	35%	45%	38%	40%	48%
# error = 2	18%	15%	20%	13%	20%	15%	15%
# error = 3	-	3%	5%	3%	3%	5%	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 22, trainset A							
Average error	0.800	1.375	1.000	1.250	1.000	1.350	1.175
Max error	3	4	3	4	4	4	4
# **Correct**	43%	20%	35%	25%	40%	25%	33%
# error = 1	38%	40%	33%	40%	33%	35%	33%
# error = 2	18%	25%	30%	23%	18%	23%	23%
# error = 3	3%	13%	3%	10%	8%	15%	10%
# error = 4	-	3%	-	3%	3%	3%	3%

Table C.3: Performance of prototype. Trainset B, participant 1-13

Performance prototype on participant 1, trainset B							
	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	0.867	0.700	0.833	0.700	0.800	0.800	0.833
Max error	2	2	2	2	2	2	2
# **Correct**	33%	37%	30%	37%	33%	33%	30%
# error = 1	47%	57%	57%	57%	53%	53%	57%
# error = 2	20%	7%	13%	7%	13%	13%	13%
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 2, trainset B							
Average error	1.767	1.500	1.300	1.467	1.533	1.600	1.567
Max error	4	4	3	4	4	4	4
# **Correct**	17%	17%	33%	23%	27%	23%	23%
# error = 1	27%	47%	27%	37%	27%	30%	30%
# error = 2	30%	13%	17%	17%	20%	13%	20%
# error = 3	17%	17%	23%	17%	20%	30%	20%
# error = 4	10%	7%	-	7%	7%	3%	7%
Performance prototype on participant 4, trainset B							
Average error	0.867	0.767	0.733	0.767	0.733	0.733	0.733
Max error	2	2	2	2	2	2	2
# **Correct**	37%	37%	33%	37%	37%	40%	40%
# error = 1	40%	50%	60%	50%	53%	47%	47%
# error = 2	23%	13%	7%	13%	10%	13%	13%
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 9, trainset B							
Average error	1.633	1.000	0.833	1.467	1.267	0.833	1.033
Max error	4	3	2	4	4	3	3
# **Correct**	17%	30%	37%	20%	30%	37%	33%
# error = 1	30%	43%	43%	37%	33%	47%	37%
# error = 2	30%	23%	20%	23%	20%	13%	23%
# error = 3	20%	3%	-	17%	13%	3%	7%
# error = 4	3%	-	-	3%	3%	-	-
Performance prototype on participant 13, trainset B							
Average error	1.100	0.567	0.933	0.567	0.767	0.667	0.667
Max error	3	2	2	2	2	2	2
# **Correct**	30%	50%	30%	50%	43%	53%	53%
# error = 1	37%	43%	47%	43%	37%	27%	27%
# error = 2	27%	7%	23%	7%	20%	20%	20%
# error = 3	7%	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-

Table C.4: Performance of prototype. Trainset B, participant 16-22

Performance prototype on participant 16, trainset B							
	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	1.067	0.833	0.833	0.933	1.033	0.733	0.800
Max error	4	2	2	4	4	2	4
# **Correct**	27%	40%	33%	37%	33%	43%	47%
# error = 1	50%	37%	50%	40%	40%	40%	33%
# error = 2	17%	23%	17%	20%	20%	17%	17%
# error = 3	3%	-	-	-	3%	-	-
# error = 4	3%	-	-	3%	3%	-	3%
Performance prototype on participant 19, trainset B							
Average error	1.433	1.667	1.933	1.600	1.567	1.967	1.433
Max error	4	4	4	4	4	4	4
# **Correct**	30%	27%	17%	30%	30%	17%	33%
# error = 1	30%	23%	27%	20%	23%	27%	23%
# error = 2	13%	20%	23%	23%	20%	23%	20%
# error = 3	20%	17%	13%	13%	13%	10%	13%
# error = 4	7%	13%	20%	13%	13%	23%	10%
Performance prototype on participant 20, trainset B							
Average error	0.733	0.600	0.700	0.600	0.633	0.600	0.667
Max error	2	2	2	2	2	2	2
# **Correct**	40%	50%	43%	47%	47%	53%	47%
# error = 1	47%	40%	43%	47%	43%	33%	40%
# error = 2	13%	10%	13%	7%	10%	13%	13%
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 21, trainset B							
Average error	0.967	0.867	0.967	0.833	1.067	1.000	0.933
Max error	4	3	3	3	3	3	3
# **Correct**	37%	43%	30%	43%	23%	30%	40%
# error = 1	40%	37%	50%	37%	53%	47%	33%
# error = 2	17%	10%	13%	13%	17%	17%	20%
# error = 3	3%	10%	7%	7%	7%	7%	7%
# error = 4	3%	-	-	-	-	-	-
Performance prototype on participant 22, trainset B							
Average error	1.233	1.000	1.033	1.167	1.067	1.067	1.067
Max error	4	3	3	4	4	3	4
# **Correct**	30%	30%	30%	27%	27%	27%	30%
# error = 1	33%	43%	47%	40%	50%	47%	43%
# error = 2	27%	23%	13%	27%	17%	20%	20%
# error = 3	3%	3%	10%	3%	3%	7%	3%
# error = 4	7%	-	-	3%	3%	-	3%

Table C.5: Performance of prototype. Trainset C, participant 1-13

Performance prototype on participant 1, trainset C							
	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	0.750	0.850	1.000	0.750	0.800	0.950	0.900
Max error	3	2	2	3	3	2	2
# Correct	40%	35%	25%	40%	45%	30%	35%
# error = 1	50%	45%	50%	50%	35%	45%	40%
# error = 2	5%	20%	25%	5%	15%	25%	25%
# error = 3	5%	-	-	5%	5%	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 2, trainset C							
Average error	1.600	1.400	1.850	1.550	1.950	1.900	1.950
Max error	4	4	4	4	4	4	4
# Correct	30%	25%	10%	30%	15%	10%	20%
# error = 1	20%	35%	35%	15%	25%	30%	15%
# error = 2	20%	20%	20%	35%	20%	25%	25%
# error = 3	20%	15%	30%	10%	30%	30%	30%
# error = 4	10%	5%	5%	10%	10%	5%	10%
Performance prototype on participant 4, trainset C							
Average error	1.000	0.850	0.500	0.800	0.500	0.750	0.700
Max error	3	2	2	2	1	2	2
# Correct	25%	30%	55%	35%	50%	35%	40%
# error = 1	60%	55%	40%	50%	50%	55%	50%
# error = 2	5%	15%	5%	15%	-	10%	10%
# error = 3	10%	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 9, trainset C							
Average error	1.250	0.750	1.000	0.900	1.150	0.650	0.800
Max error	4	2	3	2	4	2	2
# Correct	30%	40%	30%	30%	25%	45%	35%
# error = 1	35%	45%	45%	50%	45%	45%	50%
# error = 2	20%	15%	20%	20%	25%	10%	15%
# error = 3	10%	-	5%	-	-	-	-
# error = 4	5%	-	-	-	5%	-	-
Performance prototype on participant 13, trainset C							
Average error	1.300	0.800	0.800	0.950	1.050	0.800	0.900
Max error	4	2	2	2	2	2	2
# Correct	15%	40%	35%	30%	20%	40%	35%
# error = 1	50%	40%	50%	45%	55%	40%	40%
# error = 2	30%	20%	15%	25%	25%	20%	25%
# error = 3	-	-	-	-	-	-	-
# error = 4	5%	-	-	-	-	-	-

Table C.6: Performance of prototype. Trainset C, participant 16-22

Performance prototype on participant 16, trainset C							
	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	1.000	0.850	0.850	0.850	0.950	0.750	0.700
Max error	3	2	3	3	3	2	3
# **Correct**	30%	35%	45%	40%	45%	45%	50%
# error = 1	45%	45%	30%	40%	25%	35%	35%
# error = 2	20%	20%	20%	15%	20%	20%	10%
# error = 3	5%	-	5%	5%	10%	-	5%
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 19, trainset C							
Average error	1.700	2.200	1.750	2.000	1.800	1.900	1.950
Max error	4	4	4	4	4	4	4
# **Correct**	35%	10%	25%	20%	30%	20%	20%
# error = 1	10%	10%	30%	10%	15%	15%	15%
# error = 2	20%	45%	10%	35%	20%	35%	35%
# error = 3	20%	20%	15%	20%	15%	15%	10%
# error = 4	15%	15%	20%	15%	20%	15%	20%
Performance prototype on participant 20, trainset C							
Average error	0.800	0.650	0.850	0.700	0.750	0.700	0.700
Max error	3	2	2	2	2	2	2
# **Correct**	45%	40%	25%	40%	30%	35%	35%
# error = 1	35%	55%	65%	50%	65%	60%	60%
# error = 2	15%	5%	10%	10%	5%	5%	5%
# error = 3	5%	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 21, trainset C							
Average error	1.350	1.150	1.250	1.050	1.150	1.100	1.200
Max error	4	3	3	3	3	3	3
# **Correct**	25%	20%	20%	25%	25%	25%	25%
# error = 1	35%	50%	40%	50%	40%	45%	35%
# error = 2	25%	25%	35%	20%	30%	25%	35%
# error = 3	10%	5%	5%	5%	5%	5%	5%
# error = 4	5%	-	-	-	-	-	-
Performance prototype on participant 22, trainset C							
Average error	0.950	1.150	1.150	0.950	1.250	1.200	1.250
Max error	3	4	4	2	4	4	4
# **Correct**	30%	30%	30%	25%	25%	30%	25%
# error = 1	50%	40%	45%	55%	45%	40%	45%
# error = 2	15%	20%	10%	20%	15%	15%	15%
# error = 3	5%	5%	10%	-	10%	10%	10%
# error = 4	-	5%	5%	-	5%	5%	5%

Table C.7: Performance of prototype. Trainset D, participant 1-13

Performance prototype on participant 1, trainset D							
	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	0.400	0.500	1.100	0.600	1.100	1.000	1.000
Max error	1	1	2	1	2	2	2
# **Correct**	60%	50%	20%	40%	20%	30%	30%
# error = 1	40%	50%	50%	60%	50%	40%	40%
# error = 2	-	-	30%	-	30%	30%	30%
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 2, trainset D							
Average error	1.400	1.400	1.900	1.300	1.800	1.900	1.800
Max error	4	3	4	4	4	3	4
# **Correct**	30%	20%	30%	40%	40%	20%	40%
# error = 1	40%	40%	10%	20%	-	20%	-
# error = 2	10%	20%	10%	20%	10%	10%	10%
# error = 3	-	20%	40%	10%	40%	50%	40%
# error = 4	20%	-	10%	10%	10%	-	10%
Performance prototype on participant 4, trainset D							
Average error	0.800	0.700	0.500	0.700	0.700	0.700	0.600
Max error	2	2	1	2	2	2	2
# **Correct**	40%	40%	50%	40%	40%	40%	50%
# error = 1	40%	50%	50%	50%	50%	50%	40%
# error = 2	20%	10%	-	10%	10%	10%	10%
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 9, trainset D							
Average error	1.500	0.900	0.500	1.000	0.700	0.700	0.700
Max error	2	2	2	2	2	2	2
# **Correct**	-	20%	60%	10%	40%	40%	40%
# error = 1	50%	70%	30%	80%	50%	50%	50%
# error = 2	50%	10%	10%	10%	10%	10%	10%
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 13, trainset D							
Average error	1.200	0.700	1.100	0.700	1.100	0.700	0.800
Max error	2	2	2	2	2	2	2
# **Correct**	10%	40%	30%	40%	30%	40%	40%
# error = 1	60%	50%	30%	50%	30%	50%	40%
# error = 2	30%	10%	40%	10%	40%	10%	20%
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-

Table C.8: Performance of prototype. Trainset D, participant 16-22

Performance prototype on participant 16, trainset D							
	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	0.900	0.800	0.700	0.800	0.900	0.800	0.800
Max error	3	2	2	3	3	2	2
# **Correct**	30%	40%	40%	50%	30%	40%	40%
# error = 1	60%	40%	50%	30%	60%	40%	40%
# error = 2	-	20%	10%	10%	-	20%	20%
# error = 3	10%	-	-	10%	10%	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 19, trainset D							
Average error	1.700	1.400	2.400	1.600	2.300	2.000	1.700
Max error	4	4	4	4	4	4	4
# **Correct**	30%	40%	30%	30%	30%	30%	40%
# error = 1	20%	20%	-	20%	-	20%	10%
# error = 2	10%	20%	-	20%	10%	-	10%
# error = 3	30%	-	40%	20%	30%	20%	20%
# error = 4	10%	20%	30%	10%	30%	30%	20%
Performance prototype on participant 20, trainset D							
Average error	0.800	0.500	0.800	0.600	0.700	0.600	0.600
Max error	2	1	1	1	1	1	1
# **Correct**	40%	50%	20%	40%	30%	40%	40%
# error = 1	40%	50%	80%	60%	70%	60%	60%
# error = 2	20%	-	-	-	-	-	-
# error = 3	-	-	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 21, trainset D							
Average error	0.500	0.800	0.800	0.400	0.800	0.700	0.500
Max error	2	3	2	2	2	2	2
# **Correct**	70%	50%	30%	80%	40%	40%	70%
# error = 1	10%	30%	60%	-	40%	50%	10%
# error = 2	20%	10%	10%	20%	20%	10%	20%
# error = 3	-	10%	-	-	-	-	-
# error = 4	-	-	-	-	-	-	-
Performance prototype on participant 22, trainset D							
Average error	1.100	0.700	1.200	0.900	1.100	1.200	1.100
Max error	2	2	3	2	3	3	3
# **Correct**	10%	40%	20%	20%	20%	20%	20%
# error = 1	70%	50%	50%	70%	60%	50%	60%
# error = 2	20%	10%	20%	10%	10%	20%	10%
# error = 3	-	-	10%	-	10%	10%	10%
# error = 4	-	-	-	-	-	-	-

Table C.9: Average performance for trainset A

	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	1.115	1.063	1.008	1.033	1.043	1.073	1.038
Max error	3.000	2.9	2.7	3	3.1	3	2.9
# Correct	30%	30%	32%	31%	32%	30%	31%
# error = 1	39%	44%	44%	44%	43%	44%	43%
# error = 2	24%	17%	17%	17%	16%	17%	18%
# error = 3	5%	8%	7%	7%	7%	9%	7%
# error = 4	3%	1%	-	1%	3%	1%	1%

Table C.10: Average performance for trainset B

	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	1.167	0.950	1.010	1.010	1.047	1.000	0.973
Max error	3.300	2.7	2.5	3.1	3.1	2.7	3
# Correct	30%	36%	32%	35%	33%	36%	38%
# error = 1	38%	42%	45%	41%	41%	40%	37%
# error = 2	22%	15%	16%	16%	17%	16%	18%
# error = 3	7%	5%	5%	6%	6%	6%	5%
# error = 4	3%	2%	2%	3%	3%	3%	2%

Table C.11: Average performance for trainset C

	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	1.170	1.065	1.100	1.050	1.135	1.070	1.105
Max error	3.500	2.7	2.9	2.7	3	2.7	2.8
# Correct	31%	31%	30%	32%	31%	32%	32%
# error = 1	39%	42%	43%	42%	40%	41%	39%
# error = 2	18%	21%	17%	20%	18%	19%	20%
# error = 3	9%	5%	7%	5%	8%	6%	6%
# error = 4	4%	3%	3%	3%	4%	3%	4%

Table C.12: Average performance for trainset D

	ANN	**k-NN**	**Tree**	**A & k**	**A & T**	**k & T**	**All**
Average error	1.030	0.840	1.100	0.860	1.120	1.030	0.960
Max error	2.400	2.2	2.3	2.3	2.5	2.3	2.4
# Correct	32%	39%	33%	39%	32%	34%	41%
# error = 1	43%	45%	41%	44%	41%	43%	35%
# error = 2	18%	11%	13%	11%	14%	12%	14%
# error = 3	4%	3%	9%	4%	9%	8%	7%
# error = 4	3%	2%	4%	2%	4%	3%	3%

Table C.13: Confidende levels prototype. All trainsets, participant 1-13

Confidence level prototype for participant 1, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	34%	79%	68%	33%	73%	72%	52%	69%	71%	45%	69%	82%
# > **75%**	13%	55%	33%	17%	60%	53%	35%	40%	55%	20%	50%	90%
# [50,75] %	20%	45%	68%	3%	40%	43%	20%	45%	25%	30%	50%	10%
# [25,50] %	15%	-	-	23%	-	3%	20%	15%	20%	10%	-	-
# < 25%	53%	-	-	57%	-	-	25%	-	-	40%	-	-
Confidence level prototype for participant 2, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	51%	69%	51%	59%	74%	70%	67%	62%	66%	46%	61%	74%
# > **75%**	23%	13%	40%	47%	63%	50%	50%	20%	45%	20%	10%	40%
# [50,75] %	35%	88%	33%	13%	37%	37%	15%	50%	45%	10%	70%	60%
# [25,50] %	28%	-	-	17%	-	13%	20%	30%	-	50%	20%	-
# < 25%	15%	-	28%	23%	-	-	15%	-	10%	20%	-	-
Confidence level prototype for participant 4, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	60%	92%	67%	33%	79%	68%	56%	73%	67%	41%	80%	77%
# > **75%**	53%	85%	40%	10%	83%	47%	25%	40%	50%	20%	80%	80%
# [50,75] %	8%	15%	43%	10%	17%	43%	35%	60%	30%	10%	20%	10%
# [25,50] %	20%	-	18%	43%	-	10%	25%	-	15%	30%	-	10%
# < 25%	20%	-	-	37%	-	-	15%	-	5%	40%	-	-
Confidence level prototype for participant 9, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	41%	71%	50%	63%	73%	69%	59%	62%	60%	49%	71%	72%
# > **75%**	23%	25%	40%	57%	70%	40%	45%	10%	25%	40%	40%	60%
# [50,75] %	20%	75%	28%	7%	30%	57%	15%	70%	45%	-	60%	30%
# [25,50] %	13%	-	5%	20%	-	3%	15%	20%	30%	30%	-	10%
# < 25%	45%	-	28%	17%	-	-	25%	-	-	30%	-	-
Confidence level prototype for participant 13, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	44%	72%	63%	40%	71%	64%	47%	72%	66%	33%	69%	79%
# > **75%**	18%	33%	33%	20%	50%	30%	30%	40%	35%	-	50%	80%
# [50,75] %	20%	68%	60%	20%	50%	53%	10%	55%	40%	10%	40%	20%
# [25,50] %	38%	-	8%	10%	-	17%	35%	5%	25%	80%	10%	-
# < 25%	25%	-	-	50%	-	-	25%	-	-	10%	-	-

Table C.14: Confidende levels prototype. All trainsets, participant 16-22

Confidence level prototype for participant 16, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	57%	75%	45%	49%	70%	64%	36%	76%	70%	60%	73%	82%
# > **75%**	30%	43%	25%	13%	57%	27%	10%	55%	55%	30%	70%	60%
# [50,75] %	30%	58%	30%	40%	43%	57%	10%	40%	35%	20%	20%	40%
# [25,50] %	23%	-	13%	17%	-	17%	40%	5%	10%	50%	10%	-
# < 25%	18%	-	33%	30%	-	-	40%	-	-	-	-	-
Confidence level prototype for participant 19, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	53%	77%	47%	69%	63%	67%	64%	65%	57%	63%	57%	68%
# > **75%**	40%	53%	28%	63%	23%	43%	60%	30%	40%	50%	10%	40%
# [50,75] %	13%	48%	35%	3%	70%	40%	-	60%	30%	-	60%	50%
# [25,50] %	15%	-	10%	17%	7%	17%	15%	10%	20%	40%	30%	10%
# < 25%	33%	-	28%	17%	-	-	25%	-	10%	10%	-	-
Confidence level prototype for participant 20, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	40%	75%	55%	43%	73%	63%	33%	75%	68%	40%	76%	75%
# > **75%**	10%	45%	40%	10%	60%	33%	15%	50%	55%	10%	60%	70%
# [50,75] %	25%	55%	33%	30%	40%	50%	5%	50%	15%	10%	40%	30%
# [25,50] %	25%	-	-	30%	-	17%	40%	-	30%	70%	-	-
# < 25%	40%	-	28%	30%	-	-	40%	-	-	10%	-	-
Confidence level prototype for participant 21, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	46%	74%	59%	56%	74%	68%	46%	72%	62%	75%	73%	74%
# > **75%**	8%	43%	43%	43%	73%	37%	15%	50%	40%	70%	60%	70%
# [50,75] %	28%	58%	30%	13%	27%	50%	30%	45%	25%	10%	40%	10%
# [25,50] %	65%	-	-	20%	-	13%	30%	5%	30%	20%	-	20%
# < 25%		-	28%	23%	-	-	25%	-	5%	-	-	-
Confidence level prototype for participant 22, trainset A-D. Every trainset uses ANN, k-NN & Tree												
	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
Average	49%	72%	53%	47%	70%	71%	54%	71%	71%	48%	65%	84%
# > **75%**	18%	30%	25%	20%	53%	43%	25%	45%	35%	30%	40%	90%
# [50,75] %	28%	70%	30%	20%	43%	50%	35%	45%	60%	20%	60%	10%
# [25,50] %	28%	-	13%	40%	3%	7%	20%	5%	5%	10%	-	-
# < 25%	28%	-	33%	20%	-	-	20%	5%	-	40%	-	-

Table C.15: Average confidence levels for prototype. All trainsets

	Trainset A			*Trainset B*			*Trainset C*			*Trainset D*		
	ANN	**k-NN**	**Tree**	**ANN**	**k-NN**	**Tree**	**ANN**	**k-NN**	**Tree**	**ANN**	**k-NN**	**Tree**
Average	47%	75%	56%	49%	72%	68%	51%	70%	66%	50%	69%	77%
# > **75%**	23%	42%	35%	30%	59%	40%	31%	38%	44%	29%	47%	68%
# [50,75] %	23%	58%	39%	16%	40%	48%	18%	52%	35%	12%	46%	27%
# [25,50] %	27%	-	7%	24%	1%	12%	26%	10%	19%	39%	7%	5%
# < 25%	28%	-	20%	30%	-	-	26%	1%	3%	20%	-	-