Extracting Unstructured Information from the WWW to Support Merchant Existence in eCommerce

Farid Meziane and Mohd Khairudin Kasiran
School of Sciences, Computer Science
University of Salford, Salford M5 4WT
f.mczianc@salford.ac.uk, mkasiran@aol.com

Abstract: In [KM02]a model has been developed to support trust in eCommerce. The model is composed of four main modules where each module is a set of factors the consumer is looking for to trust a virtual merchant. These four modules represent the merchant existence, affiliation, policy and performance. In this paper, we present a model to implement the existence module by developing an information extraction system which aims at localising the required information on the merchant's website. The system is based on rules that reflect the different ways the information is represented on the websites, their structures and the layout of the websites. The extracted information is then stored in a database to be used for a future evaluation of the trust associated with the merchant website.

1 Introduction and Motivation

It is estimated that there are more than 200 millions of web pages available on the WWW [CDF98, Fre98] and this number keeps increasing on a daily basis [1]. It is also stated that the computer cannot understand any of these pages [CDF98] and humans can hardly make use of this wealth of information without the use of tools that guide them towards the desired information. Initially, search engines were used and lately research has shifted to information extraction system. These systems attempt to localise information within a document rather then just finding the document. There is a lot of interest in extracting information from the WWW [VTT01, Sod97, Fre98, CDF98, Ham97]. In this paper we present an information extraction system in the context of eCommerce.

Electronic commerce, also known as eCommerce, is a business arrangement where the merchants place their business in the cyber world through web technology. There was a lot of optimism and excitement when first introduced. However, eCommerce did not leave to its promises and has yet to be widely accepted. In fact, we assisted in the last two years to the crash of many dot companies. Like other technological breakthrough, eCommerce may require some time to be accepted by the society. However, there are many other reasons why eCommerce is not becoming the success story that many predicted when first

[1] Google search engine (`http://www.google.com`) claims to search more than 3,083,324,652 web pages on the 18th of March 2003

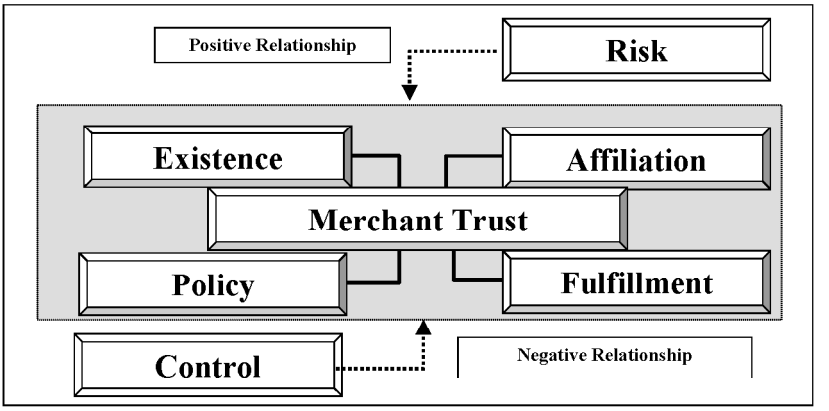

Figure 1: The Trust Model

launched. Lack of trust has been identified as one of the major obstacles to the success of eCommerce [TT01].

Although there are many successful eCommerce organisations, a day can hardly pass without us hearing about a negative story related to transactions made through the WWW. This range from delays in deliveries, quality of the goods and fraud. Indeed, It has been reported that in 1999, consumers lost to Internet fraud was over US$3.2 millions and this is increasing every year [Ba01]. This has affected consumers' trust towards online business. The question that many customers are asking is "who to trust in the cyber space?" Many models of eCommerce trust have been developed in the last few years [CA99, LB96, MT01, MDS95, SSC92, TT01]. In [KM02] we presented a trust model based on the information present on the merchant website [2]. As shown in Figure 1, the model identified four major factors that need to be identified before dealing with unknown cyber merchants. Establishing the existence of merchants is an important factor in trusting them. Indeed the customer feels more comfortable when he knows that there is more than just an email behind a merchant's website. The information the customer needs to collect to satisfy the existence factor have been identified as: the merchant's Telephone Number, Fax Number and Postal Address. However, if done manually this process would be time consuming and may discourage many customers checking the information or performing a transaction with the merchant.

The remaining of this paper is organised as follows: section 2 provides a global overview of the system's architecture. Section 3 describes the rules used for extracting the various trust factors. Section 4 presents the navigation process used for searching the existence factors. The evaluation of the system is summarised in section 5 and in section 6 we present the main conclusions of this work.

[2] For more information on how this model was derived, the reader is invited to consult [KM02]

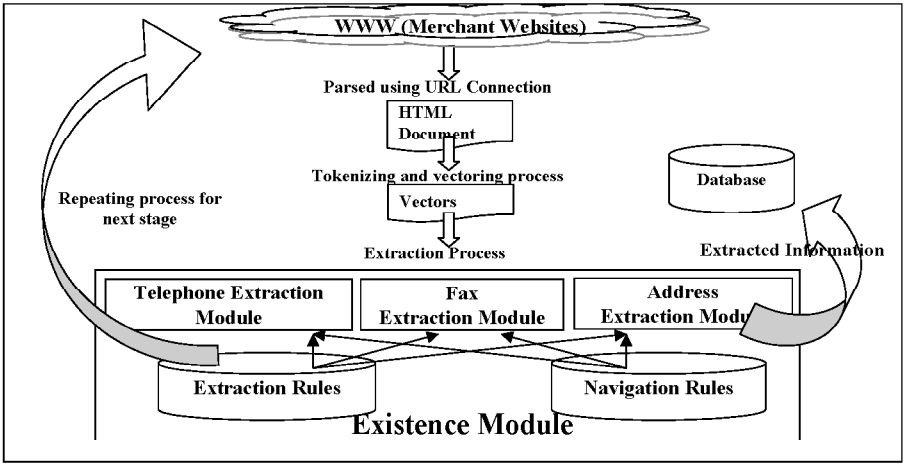

Figure 2: System Architecture

2 Overall Approach and System Architecture

The work presented in this paper is the implementation of the existence factor described in the previous section. At its current implementation, the entry to the system is the eCommerce website's URL. The Existence subsystem is composed of 3 modules each module's aim is to extract one existence factor. Once the top level of the website is loaded, extraction rules will be applied for each module. If any module fails to extract the required information then the links of the page are collected and navigation rules are applied for a better selection of the links to be used first. The extracted information is then stored in a database. Although, not described in this paper, the final aim is to collect all the information extracted from the remaining factors and then evaluate the merchant website. The final output of the system will be an evaluation score associated with the website. The overall architecture of the system is shown in Figure 2.

We adopted a learning approach for the system. We have first identified 50 websites to define rules for extracting the three existence factors. The stores are selected randomly using several ShoppingBot. Requests are then made to purchase few items such as book, a digital camera and chocolate. These 50 sites have been studies and the rules extracted manually.

3 Extraction Rules

When trying to extract an existence factor item, we take into account the information that precede the item, the structure of the item and the information that may follow the item. Each extraction rule will be composed by a "precede expression", a valid "structure" of

the item and a "follow_expression". A "precede_expression" is an expression we expect to find before the item and a "follow_expression" is an expression the system expects to find after the item. An extraction rule will have the format:

$$extraction\_rule = precede\_expression, item\_structure, follow\_expression$$

In the following subsections we define for each item, a set of *precede_expression*, a set of *item_structures* and a set of *follow_expressions*.

3.1 The Telephone Number Rules

From the studied sample , the common expressions found to precedea telephone number are:

```
TPList = [By Phone, by phone,
          Call, call,
          Call 24 hours a day,
          Call us, call us,
          Call us at, Call us at:,
          Call us on, call us on:,
          Call Toll Free, call toll free,
          Call sales at, Call sales at:,
          Phone, Phone:, phone, phone:,
          Telephone, Telephone:,
          Telephone Number, Telephone Number:,
          Toll Free, Toll Free:, toll free, toll free:
          Toll Free Order
         ]
```

A telephone number may have many formats on a company's web site. Its simplest form is a string of digits. The length of the string varies from one country to another. The country code is also used on some website which makes it difficult to rely on the length of the phone number. The country code is sometimes written inside brackets. On some websites periods (.) and dashes (-) are used to separate groups of digits example $800.123.3456$ or $800-123-5678$. To make the phone number easy to remember, some organisations writes it using characters. For example, $varsitybooks.com$ which sells books has its phone number written as $1.877.VAR.BOOK$. Modern telephone hand sets have letters associated with each key digit. The previous phone number should be read as $1.877.827.2665$. Based on these assumptions, the rules used for defining the structure of a telephone number are:

```
telephone_number = numeric_value
telephone_number = telephone_number,{.telephone_number}
```

```
telephone_number = telephone_number,{-telephone_number}
telephone_number = (telephone_number),telephone_number
telephone_number = (telephone_number), char_value
```

We do not associate any length to the numeric value. However, when the information is extracted, the telephone number is stored as a string of digits only.

In some websites, the telephone number was followed by the expression given in list `TFList`.

```
TFList = [ Mail Order,
           Outside US
         ]
```

3.2 The Fax Number Rules

The Fax number is extracted using rules similar to those defined for a telephone number. The common expressions that precede a fax number are:

```
FPList = [By fax, By Fax, by fax:, By Fax:,
          Fax, Fax:, fax, fax:,
          Fax 24 hours a day:,
          Fax on demand,
          our fax number is:
         ]
```

There was less problems with the fax number as all of them used a string of digits. However, to be on the safe side, we used the same rules as the phone number. We also found that some fax numbers are followed by some expressions. These are given in the `FFList` list.

```
FFList = [Fax US,
          Fax outside US
         ]
```

3.3 The Address Rules

The address part of the existence factor represents the physical location of the Company. Some companies are only web based other do have physical stores and an internet presence. In this last category of companies, most websites will point the customer to their physical stores through a list of stores or a search engine where the customer can localise the nearest store by providing a town name or a post code (Zip). Although, for the

eCommerce model this information is important, for the current work which presents an information extraction system, this information is omitted and not used in the extraction process. All the stores returned are either based in the US or the UK. Hence, only addresses from these two countries are used. The following are the common expressions found to precede a postal address:

```
@PList = [About Us, About Us:, about us, about us:,
          Address
          By Mail, by mail, By Mail:, by mail:,
          By regular mail,
          Contact Us, contact us,
          Headquarters,
          Mailing,
          Mailing Address:, mailing address:,
          Mailing address for return,
          Our address is:
          Return Instructions:,
          Send a request to:
          Send your returns to:
          Store Location:,
          Warehouse,
          Write Us, write us, Write Us:, write us:,
          You can write us at:, you can write us at
         ]
```

The structures of US addresses have two formats depending on weather they are physical addresses or Postal Box (Po Box) addresses. A US Address will have the following format:

```
Organisation name, (numeric value|PO Box, numeric value),
String, numeric value(5)
```

A bracketed number that follows a numeric value indicates its length. We expect the Value of String to contain words such as Street, Avenue, Square, etc. and the last numeric value indicates the post code, or ZIP as called in the US.

A UK Address will have similar structure but the post code is not all numeric. It is a combination of characters and digits. The general format of a UK post code is: `CDD DCC` where `D` is a digit and `C` a character.

There are no common expressions following a postal address.

3.4 The Rules Formulation

The information we want to extract will not be represented by web pages or the links between the web pages. They represent small parts of text embedded in pages. We formulate the rules used for the information extraction process using Horn clauses similar to

[CDF98]. We illustrate only the rules for the telephone number extraction as the rules for extracting other existence items are similar.

```
telephone(String):- before(String, String1),
                    member(String1, TPList),
                    tstructure(String1),
                    after(String, String2),
                    member(String2, TFList).

telephone(String):- before(String, String1),
                    member(String1, TPList),
                    tstructure(String1).

telephone(String):- tstructure(String),
                    after(String, String2),
                    member(String2, TFList).

telephone(String):- tstructure(String).
```

where

- `before(String1, String2)` : means `String2` appears just before `String1` in the text.
- `after(String1,String2)` : means `String2` appears just after `String1` in the text.
- `tstucture(String)` : means that `String` has the structure of a telephone string.
- `member(X,List)` : is the usual membership relation where `X` is member of the list `List`.

4 The Navigation Process

Websites contain collections of hypertext documents. A hypertext document is typically composed of nodes and links. The nodes are the documents part and the links are the relationships between documents. A node contains the information and a link allows the navigation of other documents of the hypertext collection. A link $\lambda(n_1, n_2)$ therefor represents a connection between the source node n_1 and the destination node n_2 [FS92]. Thus, a hypertext document is better represented as a directed graph. Furthermore, we distinguish two types of links [FS91], referential links and semantic links. Referential links are used for a better organisation and easy reading of a document. However, the purpose of semantic links is to provide more details, additional or similar information about a specific topic.

Any content-specific retrieval system should consider both nodes and semantic links. In our current approach, we give a more restricted definition of semantic links. A semantic link is a link that can be index by one or more words from a predefined set of keys. This restricted view of semantic links is used to target primarily those links that have a high probability of containing the information the system is after. Hence, improving the overall search time of the extraction process as a single node may contain hundreds of links.

In addition to the source node and destination node already associated with a link, we now associate a list of indices for each link. Furthermore, semantic links are divided into three subtypes depending on how they appear on the node and the nature of the destination node. The link name can appear as simple text or an image. The target node can be a static HTML node or a dynamic one (the result of querying a database for example). Our system does not deal with extracting information from images and needs therefore to understand the URL (Universal Resource Locator) of the target node. The following are some types of semantic links:

```
<A HREF="http://www.1bst.com/res_comp.asp?scat=CP&mid=6">
   Computers and Internet</A>

<A HREF="http://www.1bst.com/xt\_add.asp?"><IMG
   SRC="/cart.gif" ALT="--" BORDER="0" WIDTH="34"
   HEIGHT="40"></A>

<A HREF="http://www.1bst.com/book1.html">A Dog Lover's
   Collection</A>
```

The indexation process starts with the tokenisation of the link name and target URL. Each token is then compared to a predefined list of indices. If the link name is textual we index both link name and target URL however, if the link is an image we just index the target URL.

Again using the list of the first 50 websites selected, we extracted the following indices given in lists NameIndexList and UrlIndexList below. We did not differentiate between the three existing items as most indices are common to all of them.

A link whose NameIndexList or UrlIndex Name is not empty will be considered as a semantic link. Otherwise, it will be considered as a referential links. During navigation process, only semantic nodes are used.

```
NameIndexList = [About XXX, About Us, Company Info,
                 Company Information, Contact Us,
                 Customer Service, Customer Support,
                 FAQ, Help, Location, Private Policy,
                 Policies, Privacy Statement
                ]

UrlIndexList = [about, aboutus, CompanyInfo, contact,
```

Table 1: System Precision

	Actual	Extracted	Precision
Telephone Number	45	33	73%
Fax Number	25	16	64%
Address	29	18	62%

```
    contactus, contact_us,
    customerservicemain, customer_service,
    CustomerService, customer_support,
    cservice, faq-page, ftr_cotact,
    help, info, service, location,
    policies, privacy
  ]
```

The extraction process can then be summarised as follows:

```
extract(page, existence-item)
 {
  apply extraction rules to page
  if existence item found
     return item
  else
    {
     extract all links in the current page
     index all links in current page
     let S be the set semantic links
     for each link ks in S
         extract(ks, existence-item)
    }
    return "not found"
 }
```

5 Evaluation

Once the extraction and navigation rules have been defined and implemented, we used the same process as before to select another set of 50 commercial websites for the evaluation of the system. Table 5 summarises the performance of the system in terms of the precision with regards to the extracted information.

We have also conducted an experiment, to localise the level where the information is extracted. We assume that the top level page (the one we access when we use the website's URL) is level 1. We note here that sometimes the information is available earlier than the

Table 2: Extraction Level

	Level 1		Level 2		level 3		level ¿=4	
	Total	%	Total	%	Total	%	Total	%
Tel.Number	13	39%	18	55%	2	6%	0	0%
Fax Number	0	0%	12	75%	4	25%	0	0%
Address	1	6%	10	56%	6	33%	1	6%

level where it is extracted. This is particularly true for those websites that display the information as an image at level one and then the information is repeated textually at another level. Table 2 summarises our findings.

This experiment is important as we felt the need to know the exact place where the information is stored to improve the efficiency, in terms of time, of the system. We have noticed that once the level where the information is stored is greater than 3 the system becomes very slow and it takes up to fifteen minutes to extract the information. This experiment has shown that in general if the information is not found after searching level three then it is more likely that the information is not present at all. There was only one case where the address was extracted at level 4. Such findings will surely influence the implementation of our system to make it more usable. We believe that we need a larger sample to have a better understanding about the localisation of the information before making any changes to the current implementation of the system.

6 Conclusion

In this paper, we reported the results of the first prototype of an information extraction system that has been developed to support a trust model in eCommerce. Although the precision is not high at this stage we believe it can be improved with a larger sample as more rules will be included. The efficiency of the system will be surely improved if more techniques are used. We have learned that when it comes to the WWW the information is not always textual. Many websites use images to convey information such as the one considered in this paper. It is worth including in futures implementation a module to extract text from images. This is something we felt will improve the overall search and extraction process. The second aspect that needs to be improved is the indexation process of the links. For the moment we are giving the same weight for all the link's indices. A better approach will be to weight these indices and start with the links that have the most significant index. This significance will depend on the type of information we are trying to extract. Different extraction modules need different weights for the indices. For example we should use the weight differently if we are looking for a telephone number or an address.

References

[Ba01] S. Ba. Establishing online trust through a community responsibility system. *Decision Support Systems*, (13):323–336, 2001.

[CA99] Cheskin Research and Studio Archtype/sapient. eCommerce Study Trust. WWW, *http://www.studioarchetype.com/cheskin/assets/images/etrust.pdf*, 1999.

[CDF98] Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew K. McCallum, Tom M. Mitchell, Kamal Nigam, and Seán Slattery. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence*, pages 509–516, Madison, US, 1998. AAAI Press, Menlo Park, US.

[Fre98] Dayne Freitag. Information Extraction from HTML: Application of a General Machine Learning Approach. In *Proceedings of the Fifteenth Conference on Artificial Intelligence AAAI-98*, pages 517–523, Madison, Wisconsin, USA, 1998.

[FS91] H. P. Frei and P. Schuble. Designing a Hyper-media information system. In *DEXA'91*, pages 449–454, Wien, 1991. Springer-Verlag.

[FS92] H. P. Frei and D. Stieger. Making Use of Hypertext Links when Retrieving Information. In *Proceedings of the ACM Conference on Hypertext and Hypermedia*, pages 102–111, Milan, Italy, 1992.

[Ham97] Joachim Hammer, Hector Garcia-Molina, Junghoo Cho, Arturo Crespo, and Rohan Aranha. Extracting Semistructured Information from the Web. In *Proceedings of the Workshop on Management fo Semistructured Data*, pages 18–25, Tucson, Arizona, USA, 1997.

[KM02] Mohd Khairudin Kasiran and Farid Meziane. An Information Framework for a Merchant Trust Agent in Electronic Commerce. In Hujan Yin, Nigel Allinson, Richard Freeman, John Keane, and Simon Hubbard, editors, *Third International Conference on Intelligent Data Engineering and Automated Learning (IDEAL02)*, Lecture Notes in Computer Science, LNCS2412, pages 243–248, UMIST, Manchester, UK, August 2002.

[LB96] R. J. Lewicki and B. B. Bunker. *Developing and maintaining trust in work relationship*, pages 114–139. 1996.

[MDS95] R.C. Mayer, J.H. Davis, and F.D. Schoorman. An integrative model of organizational trust. *Academy of Management Review*, 20(3):709–734, 1995.

[MT01] K. O. Matthew and E. Turban. A trust model for consumer Internet shopping. *International Journal of Electronic Commerce*, 6(1):75–91, 2001.

[Sod97] Stephen Soderland. Learning to Extract Text-Based Information from the World Wide Web. In *Proceedingd ofthe third international conference on Knowledge Discovery and Data Mining*, pages 251–254, California, USA, 1997.

[SSC92] D. Shapiro, B. H. Sheppard, and L. Cheraskin. Business on a handshake. *The Negotiation Journal*, pages 365–378, October 1992.

[TT01] T. Tao-Huan and W. Theon. Towards a Generic Model of Trust in Electronic Commerce. *International Journal of Electronic Commerce*, 5(2):61–74, 2001.

[VTT01] Lakshmi Vijjappu, Ah-Hwee Tan, and Chew-Lim Tan. Web StructureAnalysis for Information Mining. In *Proceeding of the First International Workshop on Web Document Analysis*, pages 15–18, Seattle, Washington, USA, 2001.