MEASURING THE TIMELINESS OF WEBSITES

Yanlong Zhang, Qingning Huo, Lu Zhang, Hong Zhu and Sue Greenwood

School of Computing and Mathematical Sciences,
Oxford Brookes University, Gipsy Lane, Headington, Oxford, OX3 0BP, UK
Email: (yzhang | qhuo | lzhang | hzhu | sgreenwood) @brookes.ac.uk

ABSTRACT – Timeliness is an important quality factor for websites. In this paper, we present an automated approach to measuring the timeliness. Four metrics for measuring website timeliness are proposed in this paper, which include the change frequency, the structural change of a website, the editing distance and the vector distance between web pages. A prototype system has been designed and implemented to realise the metrics. Some preliminary results of applying the prototype system are also reported.

Keywords: website quality, measurement, metrics, timeliness.

1. INTRODUCTION

One of the principal objectives of software engineering is to improve the quality of software products [3]. The abstract notion of software quality can only be measured when it is broken down into a number of specific software attributes. A number of software quality models have been proposed to establish frameworks for classifying software quality attributes and to understand the relationships between them. The most well-known software and information system quality models include McCall's model [12], Boehm's model [1], COQUAMO [10] (which is based on the ideas proposed in [9] and [4]), the ISO 9126 standard quality model [7], and SOLE model [2], etc. The interrelationships between quality attributes and metrics for quantitative measurement of them have also been addressed in the literature, e.g. [3, 15, 5].

In recent years, the Internet and World Wide Web have become a major medium of software applications. There has been some research on the quality issues of web-based software systems. The differences between the web-based software systems and the conventional software systems are discussed in [11] from the perspective of software quality. Efforts have been made to set up new quality models for web-based software systems. For example, in [14], a quality model, called Website QEM, is proposed to break down the quality of websites into more than a hundred attributes. Among the many quality attributes of web-based applications, *timeliness* has been identified as of particular importance to web-based software systems, especially for those websites serve as a medium such as online newspapers and online magazines [14, 17]. However, automated measurement of timeliness is difficult because of the high complexity, large volume of information, and dynamic nature of websites. In [17], a framework of using software agents for quality management of web-based software systems is proposed. In this paper, we present an application of the approach to measuring the timeliness of web-based information systems. In section 2, we propose some metrics for measuring timeliness of websites. Section 3 presents a prototype system that implements the metrics. Section 4 reports the preliminary results of experiments with the metrics. Section 5 is the conclusion of the paper. Future work is discussed.

2. METRICS FOR MEASURING TIMELINESS

The timeliness of a website depends on two factors: (1) the time delay in updating the information in order to keep the system consistent with the real world, and (2) the percentage of the information that is up to date. For example, the timeliness of the website of a daily newspaper is the time delay in deploying the information published or to be published in a paper to the web and the percentage of the information updated daily. In this section, we propose some metrics for both aspects of timeliness and discuss how these metrics can be combined to provide meaningful measurements of timeliness.

2.1. Time delay and update frequency

Intuitively, time delay is the length of the time period between the real time that an event happens and the publication of the information about the same event on a website. Unfortunately, without an

automatic mechanism that enables us to obtain the real time of an event happened in the real world, it is obvious that time delay cannot be directly measured automatically. Therefore, instead of measuring time delay directly, we measure the frequency that information is updated on a website. By update frequency, we mean the number of updates in a given period of time, i.e.

$$Frq_{Update}(Website) = \frac{Number\ of\ Updates}{Length\ of\ Time\ Period} \quad (1)$$

Although update frequency is not a direct measure of time delay, it is closely related to time delay. We can identify two types of websites according to their strategies of updating information. The first type of websites updates their information periodically, such as websites of daily newspapers. Another type of websites updates their information as soon as the information is available and ready to be published on the web. As shown in the Figure 1 below, for websites that update their information periodically, one would expect that the time delay should be less than that of those that have a higher frequency of update.

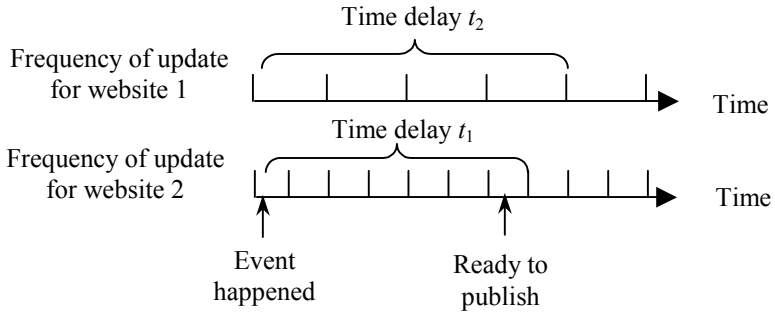


Figure 1. The effect of update frequency on time delay

Assume that the same time length was taken to prepare the publication of information on a website, the more frequent one website updates its information, the less one can expect the time delay for that information is to be published.

For websites that do not update their information periodically, but update the information whenever it is available and ready, the average update frequency also indicates the time delay, as shown in Figure 2, where the real world events are indicated by capital letters and the publication of the event is indicated by lower case letters. The time delay for an event x is denoted by the symbol δ_x. The figure shows that website 1 has a higher average update frequency than website 2. Its total time delay, i.e. the summation of δ_x for $x = a, b, c$ and d, is larger than the total delay on website 2.

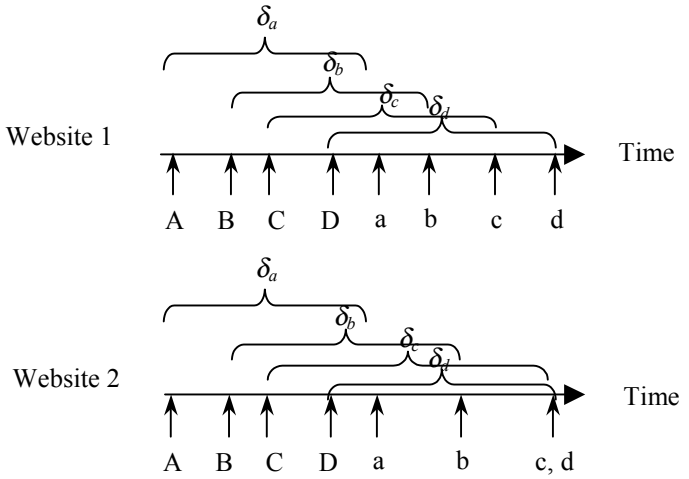


Figure 2. The update frequency indicates the time delay

However, the update frequency of a website cannot be measured accurately because web publication is passive in the sense that the information on a website can only be obtained when the user requests the information. The only practical approach to measuring the update frequency is to make a number of requests of the information in certain period of time and to find out how many times the information has changed. Therefore, a metric for update frequency is the probability of change defined as follows.

$$\Pr_{Change}(Website) = \frac{Number\ of\ Different\ Versions}{Total\ Number\ of\ requests} \quad (2)$$

Assume that the requests of information from a website is made regularly with a fix gap of time between two consecutive requests, the update frequency can be calculated approximately from the probability of change. It is easy to see that when the sampling frequency is high enough, detected update frequency should be close to the real average update frequency. That is, if $Frq_{Update}(Website) \prec Frq_{Sampling}$, we have that

$$Frq_{Update}(Website) \approx \Pr_{Cahnge}(Website) \times Frq_{Sampling} \quad (3)$$

where $Frq_{Sampling}$ is the sampling frequency, which is defined as the number of samples of the website obtained in a given period of time.

The requirement of request information from a website at a high sampling frequency implies that the detecting the information obtained is difference from the previous one must be computed very efficiently. To meet this requirement, we use the MD5 algorithm [16] to generate a fingerprint for each web page. Every time an html file is obtained from a website, its fingerprint is generated and compared with the last fingerprint. The MD5 algorithm can create a unique 'fingerprint' for a file. If the content of such file has been changed, the fingerprint changes accordingly. The MD5 algorithm therefore can be used to identify whether the page of remained URL has changed or not. Because the fingerprint of a file is significantly shorter than the original file, and the MD5 algorithm is very efficient, comparing fingerprints is much more efficient than comparing the original files.

Although the use of MD5 algorithm and fingerprints of files significantly improved the efficiency of detecting changes in a website, detecting changes by comparing all the pages of a website is still not practical. Therefore, we only detecting the changes in the home page because main changes in the contents of a website are almost always reflected in the homepage. For example, headline news are always listed in the home pages of daily newspaper web sites.

In the experiment, we found that the homepage of some website changed every time the website was visited. Such phenomenon is caused by the random numbers generated by the website when the homepage were requested. Such random info formation should be discovered and filtered before applying MD5.

2.2. Structural Changes

As discussed above, timeliness is not only related to the time delayed in updating the information, but also depends on to what extend the information is up to date. Similar to the measure of time delay, whether the information published on a website is accurate with respect to the real world cannot be directly measured automatically. Therefore, we measure how much information is updated each time the information is changed. For example, if a website publishes the prices of all shares on London Stock Exchange market, we would expect all the prices of the shares are updated each time the website updates its information. If only a proportion of the shares' price are updated, we would not consider the website is good at timeliness. Similarly, for a daily newspaper website, we would expect the news items are all updated daily, rather than that old news stay on the web front page for a few days. In this and the next subsections, we will discuss three different metrics of changes.

Firstly, let's see how to measure structural changes of a website. A website can be viewed as a hypermedia consists of a set of web pages linked one to another by clickable hyperlinks. As all hypertext systems, the structure of a website can be represented as a directed graph as follows [17].

- A node in the graph represents a page that is physically stored as an HTML or XML file and denoted by a URL in the website.
- An arc from a node *a* to another node *b* represents a hyperlink between page *a* and page *b*.

Therefore, a website can be modelled as a directed graph $G=(V, E)$ where V is the set of URLs representing web pages, E is the set of edges representing hyperlinks between web pages. An example of the graph model is depicted in Figure 3.

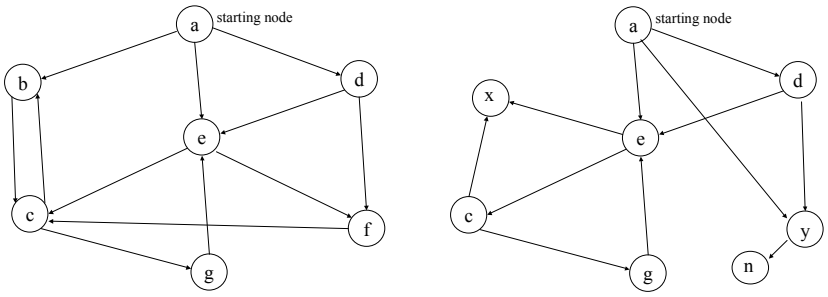
For a website under constant changes, the structures of the website at two time points can be represented as two graphs. Therefore, the structure change of a website can be calculated by comparing two graphs $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ that G_2 is obtained from G_1 by modifications such as deleting pages and links and adding new pages and new links. The changes to the contents of a page does not affect the structure of the website.

For example, supposing G_1 and G_2 are the graphs given in Figure 3 (a) and (b), respectively. Then we have that $V_1=\{a, b, c, d, e, f, g\}$ and $V_2=\{a, c, d, e, g, x, y, n\}$, where each element in V_1 and V_2 is a URL. The set of unchanged URLs is $\{a, c, d, e, g\}$. The set of deleted URLs is $\{b, f\}$. And, the set of newly added URLs is $\{x, y, n\}$. The structure change can be measured via calculating the number of

changed pages. By calculating the change of pages including the newly added pages and the deleted pages, we can therefore observe the structure change of the website according to the following formula.

$$Chg_{Structure}(Website) = \frac{Number\ of\ Changed\ Pages\ and\ Links}{Total\ Number\ of\ Pages\ and\ Links} \times 100\% \quad (4)$$

For example, Figure 3 (a) has 7 nodes (or pages) and 11 links. If it is changed to the graph given in Figure 3 (b), it has 3 new pages and 5 new links, and 2 pages and 6 links are removed. Then, the structure change is (3+5+2+6)/(7+11+3+5) = 61.5%.



(a) (b)

Figure 3. Example of website structure change

2.3. Changes within a web page

Measuring the changes in web pages is of prime importance to the timeliness measurement. Here, we propose two different metrics to measure the change of contents in a web page.

The first metric uses the 'keyword frequency vector' of a web page as an representation of its content. A keyword frequency vector consists of a set of keywords and their frequency of occurrences in a text. It is widely used in information retrieval from texts. Given a keyword frequency vector $f : W \to N$, where W is a set of keywords and N is the set of natural numbers, the frequency of a keyword w occurred in a text is represented by f_w. We use equation (5) to normalise the vector. This ensures that the measuring results are within the unit interval [0, 1] of real numbers.

$$C_w = \frac{f_w}{\sum_{w \in W} f_w^{\,2}} \quad (5)$$

Assume that a web page has normalised keyword frequency vector C before a change and C' after a change, the difference between the two versions can be measured by the distances $\|C, C'\|_V$ between the two vectors as follows.

$$\|C, C'\|_V = \tfrac{1}{2} \sqrt{\sum_{w \in W_1 \cup W_2} (C_w - C'_w)^2} \quad (6)$$

The second metric considers a web page as a string of characters. The difference between two pages is therefore the number of editing actions that one needs to obtain one web page from another. To do so, we calculate the longest common sub-string of two strings. A common sub-string of two strings S_1 and S_2 is a string L such that by subtracting some characters from either S_1 or S_2 can get S. The longest common sub-string is the longest one among all the sub-strings. For example, supposing S_1=*abcdefabgxyz* and S_2=*abcdefdefgxabz*, the longest common sub-string is *abcdefgxz*. Algorithms for computing longest common sub-string can be found in [13].

Supposing the longest common sub-string of two strings S_1 and S_2 is L, the editing distance $\|S_1, S_2\|_E$ between the two strings can be calculated according to the following formula,

$$\|S_1, S_2\|_E = \frac{\|S_1\| + \|S_2\| - 2 \times \|L\|}{\|S_1\| + \|S_2\|} \quad (7)$$

where $\|X\|$ is the length of string X, i.e. the number of characters in X.

For the above example, the editing distance between S_1 and S_2 is (12+14-2*9)/(12+14)=30.7%.

2.4. Combining content and structural changes

To measure the changes to a website by combining the changes in structure and contents, consider the situation that a website is changed from $G_1=(V_1, E_1)$ to $G_2=(V_2, E_2)$. For each page $v \in V_1 \cup V_2$, a change rate r_v can be calculated with either metrics defined by equation (6) or (7) in the section 2.3. If $v \notin V_1 \cap V_2$, let $r_v = 1$. Then, we can define the combined content and structural change rate as follows:

$$Chg_{Combined}(Website) = \frac{\sum_{v \in V_1 \cup V_2} r}{\|V_1 \cup V_2\|} \tag{8}$$

It can be simplified as follows:

$$\frac{\sum_{v \in V_1 \cup V_2} r_v}{\|V_1 \cup V_2\|} = \frac{\sum_{v \in V_1 \cap V_2} r_v + \sum_{v \in (V_1 - V_2)} r_v + \sum_{v \in (V_2 - V_1)} r_v}{\|V_1 \cup V_2\|} = \frac{\sum_{v \in V_1 \cap V_2} r_v + \|V_1 - V_2\| + \|V_2 - V_1\|}{\|V_1 \cup V_2\|} \tag{9}$$

3. THE WQMTT SYSTEM

We have designed and implemented a prototype system called Website Quality Measurement Tool for Timeliness (WQMTT), as a part of the agent-based quality management system proposed in [17]. This section presents the structure and function of the prototype.

Figure 2 depicts the structure of the WQMTT system. In WQMTT, there are three types of agents and a knowledge base. Their functions are presented below.

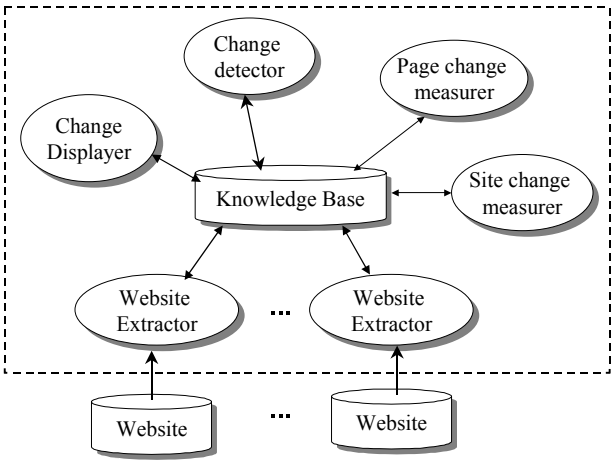


Figure 4. The structure of WQMTT system

The change detectors implements the calculation method discussed above, and stored the results in the knowledge base. The change displayer displays the results of measurement to the users of the system.

In this section, we briefly present the design and implementation issues of the internal knowledge base, the website extractor, the website comparer, and the change displayer.

- Website extractor

For each website, we create one agent to periodically retrieve the information of the structures and contents from the website, and to store the information in the knowledge base. An agent retrieves the information every minute. We exploit the technique of retrieving structure information from a website investigated in [8] and [6] to extract the structure of a website. Based on the structure, we further extract all the pages on the website. The information is used by the change measurers to calculate the timeliness.

- Knowledge base

There are two kinds of information stored in the knowledge base. The first is the information of the structures and contents of each website under measurement. The second is the information of the results of measurement. As the main goal of the system is to measure the timeliness, only the information of the structures and pages of the latest two versions and the results of measurements and comparisons of previous versions of each website are stored in the knowledge base.

- Change measurers

The change measurers implement the metrics of timeliness discussed in section 2. As the latest two versions of each website are stored in the knowledge base, the change measurer compares the two versions to calculate the change of the website. The results of the measurements are also stored in the knowledge base.

- Change displayer

We also provide a web-based interface for users of the system. The change displayer can generate web pages based on the results of the comparison to display the results to users. We display the result of comparison according to the three metrics proposed above.

4. EMPIRICAL RESULTS

To evaluate the feasibility of our method and to validate the metrics as well, we have conducted some preliminary experiments with the WQMTT system to measure the timeliness of five UK newspaper websites and one Internet stock index website. These websites and their web addresses are:

- The Times (www.times.co.uk)
- The Guardian (www.guardian.co.uk)
- The Daily Telegraphy (www.dailytelegraph.com)
- The Independent (www.independent.co.uk)
- The Sun (www.thesun.co.uk)
- The Internet.com ISDEX (www.wsrn.com/apps/ISDEX)

4.1. Homepage change frequency

We monitored the changes of the homepages of above six web sites for 24 hours from 00:00 to 23:59 on March 12th 2001 GMT. The sampling frequency is one request per minute. The results are listed in Table 1. Failed times record the number of times of failure to access a homepage. The total measured results are less than 1440 as we sometimes could not connect to the homepage within one minute.

Table 1. Homepage change frequency of selected websites

Website	Number of Requests	Changed	Unchanged	Failed	Homepage change frequency (%)
The Times	1411	39	1364	8	2.8
The Guardian	1436	36	1400	0	2.5
Daily Telegraph	1438	8	1426	4	0.6
The Independent	1415	79	1311	25	5.6
The Sun	1420	64	1351	5	4.5
ISDEX	1428	427	994	7	30.0

When we studied the results in more details, we found some interesting results.

Comparing the five newspaper websites in Figure 5, we found that they have the similar patterns of change frequencies. Sharp peaks appear around 12:00 and 17:00. At other times, most homepages kept changing at a lower frequency. The explanation of this phenomena could be that a website would keep changing when new information was provided. When the information is received and edited, it would be put online as soon as possible. Noon (12:00) is perhaps the time when the preparation work in the morning is finished and ready to publish online. Five o'clock in the afternoon is perhaps the time when the preparation work for the afternoon would finish and ready to publish on the web.

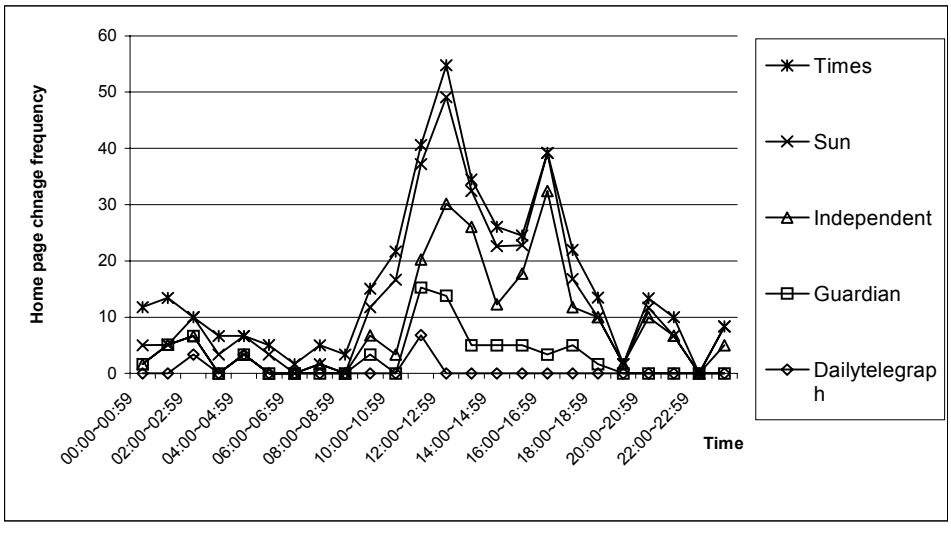


Figure 5. Pattern of change frequencies of newspaper websites

When we studied the data of the Internet Stock Index, shown in Figure 6, we found that its shape was near to a rectangle. Between 14:00 and 21:00 GMT, the homepage changed every minute. As this website is in America, that time should be the working time there. The other time, when the stock market is closed, the homepage seldom changed.

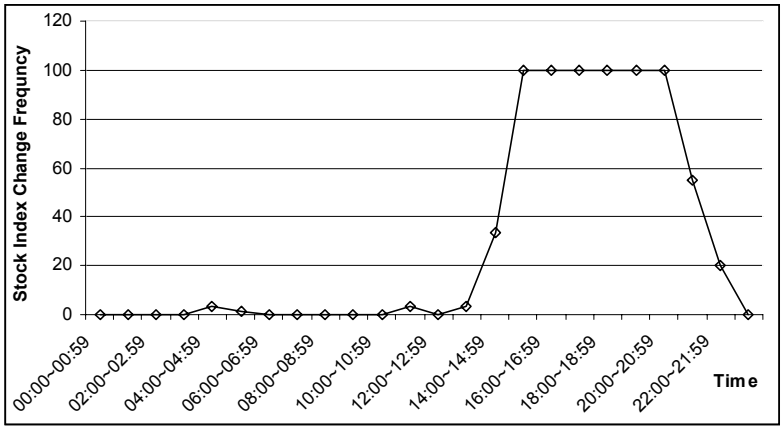


Figure 6. Pattern of change frequency of online website of share prices

4.2 Website structure change

The Guardian website consists of over 20,000 pages on March 10, 2001. We only take one directory (www.guardian.co.uk/footandmouth/story/) about the foot-and-mouth events as an example to test our metric and method. The result is shown in Table 2[**].

Table 2. Results of measuring structure change

Time	Total pages	Added pages	Deleted pages	Unchanged pages	Structure Change (%)
March 10	165	N/A	N/A	N/A	N/A
March 11	172	7	0	165	4.1
March 12	186	14	0	172	7.5

Each day, latest news were added and the old ones remained in the website. Readers can access such pages to read the latest news and old information.

4.3 Website content change

We implemented the algorithm presented in section 3.4 and measured the same directory of the Guardian. On March 10, 2001, the directory had 165 pages. When we measured the following day, only three pages showed the changes of content. Other 162 pages remained the same. The results of measurement are listed in Table 3.

Table 3. Results of measuring the vector distances between consecutive days of a website

Time	Total pages	Combined content and structure change (%)
March 10	165	N/A
March 11	172	4.1
March 12	186	7.7

We also tested our method by calculating the distances between the home pages of two different websites: The Sun and The Guardian. The vector distance between them was 60.3%, which is much bigger than the distance between the same web page of different versions, which are normally about 2~3%. This is what we expected, as the bigger difference of two websites, the bigger the measurement result is.

[**] The implementation does not take the changes of links into account, because the majority of links are only related to advertisements.

5. CONCLUSION

With the rapid growth of the World Wide Web, many issues of measuring websites have been raised. In this paper, we focus on the timeliness, which is essential to many websites. We proposed four metrics of timeliness: the change frequency, the website structure change, the web page editing distance, and the web page vector distance. For a website, each of the above metrics can be automatically calculated. Based on the methods, we implemented a prototype system to automatically measure the timeliness of websites and conducted some experiments with the system.

Measuring the timeliness of websites is complicated and difficult. There is few existing work in the literature. Our research is still very preliminary. There are a number of future works worth pursuing.

An immediate future work is to perform a thorough empirical study: to measure more websites and compare the results of our approach and the results of user feedbacks. We will also test the performance of the WQMTT system when measuring large websites.

We can also identify the following issues of measuring the timeliness that need to be further investigated.

First, when measuring a large website, we think that multi-agents that can cooperate with each other may be needed to achieve the goal. In such a paradigm, each agent may be in charge of one part of the website. We think this paradigm may enhance the performance of our system.

Secondly, in some cases the timeliness of a medium is viewed as the response time to an important incident, but our current method cannot deal with this kind of timeliness. Although in our fourth metric, we take the 'keywords' of a website as an indicator to express the semantic of web content, more work will be done to investigate semantic comprehension techniques for web contents and integrate them within our method.

Thirdly, we have developed four metrics of timeliness of websites. Do these four metrics have some relations? If so, what are the relations? This is a very interest direction for further research.

Fourthly, a website is usually more archival than its real world counterpart. For example, a daily newspaper usually only provide up-to-date information, but an online daily newspaper may provide much archival information. We will further investigate how much this may affect the precision of our approach and how to improve our approach accordingly.

REFERENCES

1. Boehm, B.W., Brown, J., Kaspar, H., Lipow, M., MacLeod, G. and Merrit, M. (1978) Characteristics of Software Quality, *TRW Serious of Software Technology*, Vol. 1, North-Holland, New York.
2. Eriksson, I. and Torn, A. (1991) A Model for IS Quality, *Software Engineering Journal*, July, pp. 152-158.
3. Fenton, N.E. and Pfleeger, S.L. (1996) *Software Metrics: A Rigorous and Practical Approach*, Second Edition, International Thomson Computer Press.
4. Gilb, T. (1988) *Principles of Software Engineering Management*, Addison-Wesley, Reading MA.
5. Gillies, A. (1997) *Software Quality: Theory and Management*, Second Edition, International Thomson Computer Press.
6. Huo, Q. (1999) *Testing of Hypertext Application*, Master's degree Thesis, Nanjing University, June. (in Chinese)
7. ISO (1991) *ISO 9126: Information Technology – Software Product Evaluation – Quality Characteristics and Guide Lines for Their Use*, ISO/IEC IS 9126, Geneva, Switzerland.
8. Jin, L., Zhu, H. and Hall, P. (1997) Adequate Testing of Hypertext Applications, *Journal of Information and Software Technology*, Vol. 39, No. 4, pp225-234.
9. Kitchenham, B.A. and Pickard, L.M. (1987) Towards a Constructive Quality Model. Part II: Statistical Techniques for Modelling Software Quality in the ESPRIT REQUEST Project, *Software Engineering Journal*, 2(4), pp. 114-126.
10. Kitchenham, B.A. and Walker, J.G. (1989) A Quantitative Approach to Monitoring Software Development, *Software Engineering Journal*, 4(1), pp. 2-13.
11. Lindroos, K. (1997) Use Quality and the World Wide Web, *Information and Software Technology*, Vol. 39, pp. 827-836.
12. McCall, J., Richards, P. and Walters, G. (1977) *Factors in Software Quality*, *Technical report CDRL A003*, US Rome Air Development Centre, Vol. I.
13. Myers, E.M. (1986) *An O(ND) Difference Algorithm and its Variations. Algorithmica* 1, 2, 251-266.
14. Olsina, L., Godoy, D., Lafuente, G. and Rossi, G. (1999) Specifying Quality Characteristics and Attributes for Websites, *First ICSE Workshop on web Engineering*, 16 -17 May Los Angeles, USA.
15. Perry, W. (1987) *Effective Methods for EDP Quality Assurance*, 2nd Edition, Prentice-Hall.
16. Rivest, R.L. (1992) *RFC 1321: The MD5 Message-Digest Algorithm*, Internet Activities Board.

17. Zhu, H., Greenwood, G., Huo, Q. and Zhang, Y. (2000) Towards Agent-Oriented Quality Management of Information Systems, *Workshop Notes of Second International Bi-Conference Workshop on Agent-Oriented Information Systems* (AOIS-2000) at AAAI'2000, Austin, USA, July 30, pp.57~64.