2014 14th International Conference on Quality Software

A Method to Test the Information Quality of Technical Documentation on Websites

Oleksandr Shpak, Welf Löwe, Anna Wingkvist, and Morgan Ericsson
Dept. of Computer Science
Linnaeus University
Växjö, Sweden
Email: {oleksandr.shpak | welf.Lowe | anna.wingkvist | morgan.ericsson}@lnu.se

***Abstract*—In software engineering, testing is one of the cornerstones of quality assurance. The idea of software testing can be applied to information quality as well. Technical documentation has a set of intended uses that correspond to use cases in a software system. Documentation is, in many cases, presented via software systems, e.g., web servers and browsers, and contains software, e.g., Javascript for user interaction, animation, and customization, etc. This makes it difficult to find a clear-cut distinction between a software system and technical documentation. However, we can assume that each use case of a technical documentation involves retrieval of some sort of information that helps a user answer a specific questions.**

To assess information testing as a method, we implemented QAnalytics, a tool to assess the information quality of documentation that is provided by a website. The tool is web-based and allows test managers and site owners to define test cases and success criteria, disseminate the test cases to testers, and to analyze the test results. This way, information testing is easily manageable even for non-technical stakeholders.

We applied our testing method and tool in a case study. According to common perception, the website of Linnaeus University needs to be re-engineered. Our method and tool helped the stakeholders identify what information is presented well and which parts of the website that need to change. The test results allowed the design and development effort to prioritize actual quality issues and potentially save time and resources.

I. Introduction

Website quality assessment is commonly based on analytics of usage traces, i.e., click sequences. This works well when a website has a single well-defined purpose, e.g., an online retailer. For such sites, the number of users that reach an order confirmation page is a measure of successful use, and thus of the quality. This approach is not easy to adapt when a website provides information to potentially different user groups and quality translates into how easy it is to find and to understand this information.

We have developed a method to test information to assess its information quality where a test manager defines a set of test cases, each corresponding to a specific scenario. Each test case can be assigned to and run by a set of test users of a specific target user group. What makes a test run successful can be adjusted to each test case and user group. The success of each test run can then be checked automatically. The information testing method allows test managers to analyze whether or not specific user groups can easily find and understand the information provided on their website.

Our method is based on software testing and the assumption that a documentation/website has a set of intended uses. These uses correspond to software use cases or user stories, and hence, one or more test cases should correspond to an intended use of the website. Since our method is based on software testing, we use terminology analogously, such as test suite, coverage, etc., unless otherwise specified. To improve readability, whenever we use the terms website and (web)page, we refer to a technical documentation presented as a website and a single document within that documentation, respectively.

Information testing was introduced as a concept by Wingkvist et al. [1]. In this paper we complement the concept with a tool to support the testing and perform a case study to determine whether information testing is an effective and efficient means to assess information quality. The rest of the paper is organized as follows. Section II introduces such a tool and Sections III and IV describe the design and the results of a case study applying information testing, the method supported by the tool, in a practical context. Section V presents an overview of website evaluation systems, and Section VI concludes the paper.

II. An Information Testing Tool

An information test case can be defined analogously to a software test case. More specifically a test case of a use case of technical documentation is defined as:

1) A set of test users of the technical documentation that represent a general class of users for a use case.
2) A question that represent the set of questions that this class of users ought to get answered with the help of the information technical documentation.
3) An expected answer.
4) Optionally, the expected completion time, i.e., the time needed to find the correct answer to the question using the technical documentation.
5) Optionally, the expected usage path(s) from the entry point of the technical documentation to the point in the documentation providing the necessary information to answer the question.

Information testing succeeds if all test users of all test cases answer the questions with the expected answers, optionally, within the expected time and using the expected paths through the documentation.

1550-6002/14 $31.00 © 2014 IEEE
DOI 10.1109/QSIC.2014.48

To make large-scale testing feasible, there is a need for tool support. We investigated existing website analytics tools, such as *Pikwik* and *Google Analytics* (cf. Table I and Section V). These could possibly be used to support information testing, either directly or as a platform to support the implementation of a tool. Table I presents our requirements on an information testing tool and an evaluation of existing analytics tools with respect to these.

The General analyses provide an understanding of the overall use of a website, e.g., visit counters can determine popular pages and content. Together with visit counters, visit distributions allow us to analyze continuous activity changes and determine structural problems in a website as well as issues within individual pages/. If there is a problem within an individual page, heatmaps can help identify the source of the problem. The Specific analyses offer additional insights into the quality of the overall structure and individual pages.

One particularly important requirement is goal conversions, i.e., the ability to determine whether a target user visited a page or a section of a document that contains the correct answer. We separate the goals into page visits, clicks, and scrolls to allow for fine-grained measures; we want a target user to not only load the page but also to scroll to the right part and spend a reasonable amount of time there for a test to succeed.

As depicted by Table I, neither of the existing tools fulfil all the requirements, so we designed and developed our own tool, QAnalytics. The tool is agnostic with respect to both documentation type and structure, but assumes that it is presented as a website.

A. Design and Use Cases

We distinguish supervised and unsupervised testing. In unsupervised testing, test users are unaware of event tracing, i.e., the tester will not experience any difference from the original website. Sessions are defined automatically; heuristics for that are based on leaving a website and timeouts. Every session is added to a default test case and the network address (IP address) of the user is used to name the session. When used for unsupervised testing, QAnalytics behaves like a regular website analytics tool.

In supervised testing, website owners set up test cases to collect test data. Each test case requires a name (e.g., the question connected to the test case) and a URL to the website to test as input. This URL is the entry point of each test. When the tests are set up, the website owner can invite testers by sending them a link that connects to the entry point via a proxy. When a tester follows the link, he or she is prompted for a session name (e.g., his or her real name) and can then explicitly start and stop the session. Between start and stop (events) all interaction events are recorded but the test user does not experience any difference compared to interacting with the original website (except for the stop button that is always displayed).

Each test case eventually contains a set of sessions, sequences of events generated by the website when a test user interacts with it to answer the related question. A documentation/website owner can define arbitrarily many test cases for different testing purposes. When sessions (event sequences) are collected, a documentation/website owner can run analyses and visualizations over the set of sessions of a single test case or several test cases. Our analysis tools provides a set of standard visualizations:

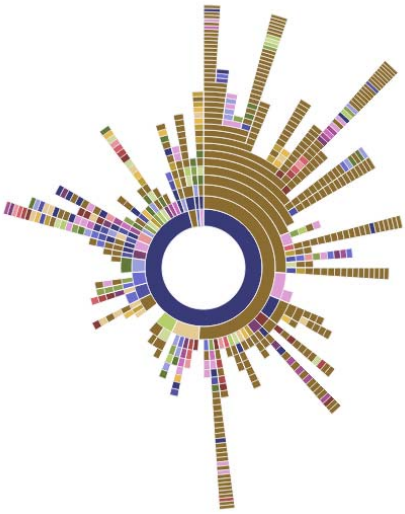

Fig. 1: Sunburst chart of all test cases of the case study. Click events correspond to colored blocks. Sessions start in the center of the chart.

- Sunburst chart — shows the path separation for a set of sessions, cf. Figure 1. It highlights common event sequence prefixes of sessions. It offers an initial insight into commonalities and differences in test users' behavior. It works fine with large sets of sessions.
- Event alignment chart — shows aligned event sequences using Levenshtein's edit distance to compute minimal alignment. This alignment offers detailed insights into commonalities and differences in test users behavior. It is complementary to the sunburst chart and works best when comparing a small set of sessions.
- Times chart — shows distribution of time spent on pages and fractions thereof for a set of sessions. A page is divided into 10 segments, and the time that a segment is visible in any session of the set is accumulated and displayed. This chart offers insight into which parts of a website that are read in detail and which are just scrolled over.
- Clicks chart — shows distribution of clicks to a page in a set of sessions. It is complementary to the Times chart view and provides information about the pages visited.
- Sequence chart — each sequence of events is displayed for debugging purposes and detailed analysis. It shows events types, additional information per event, and corresponding time steps.

TABLE I: Web Analytics Tools and their Assessment for Support of Information Quality Testing.

	Piwik[1]	Google Analytics[2]	OWA[3]	Woopa[4]	Clicky[5]	HitLinks[6]	KissMetrics[7]	Clicktale[8]
Goal Conversions								
User visits a page	Yes	Yes	Yes	Yes	Yes (not default)	No	Yes	Yes
User clicks to element	No (link only)	Yes	Yes	Yes	Yes (not default)	No	?	Yes
User scrolls to	No	Yes (not default)	?	Yes (not default)	Yes (not default)	No	?	Yes
General analyses								
Visits counter by time	Yes	Yes	Yes	Yes	Yes	Yes	Yes	?
Visits distribution by time	Yes	No	No	No	No	No	?	?
Heatmaps	No	No	Yes	No	Yes	No	No	Yes
Specific analyses								
Spent time on pages	No	Yes	No	Yes	Yes	Yes	No	No
Clicks on pages	No	Yes	Yes	Yes	Yes (not default)	No	No	Yes
Page traces	No	Yes	No	No	Yes	Yes	?	Yes
Sessions comparison	No	No	No?	No	No	No	No	No
Recognize user groups	Yes	Yes	Yes	Partly	?	Partly	Yes	No
Entry/Exit points	Yes	Yes	Yes	No	Yes	Yes	?	No

For the Sunburst, Times and Clicks charts, we adapted and integrated example charts from http://d3js.org. In addition to the visualizations, QAnalytics provides a set of statistical analyses, such as the precision and recall (cf. Section III for a complete list together with their interpretation for information quality).

B. Architecture and Implementation

QAnalytics provides the tester with an HTML proxy to the website, cf. Figure 2. The proxy and analysis tools are standalone parts.

Tracing is implemented using an event generator and listener architecture. When a session starts the browser captures all interaction events and eventually sends these to the analysis tool. Corresponding listeners receive the events, process them, and then store them in a database for further analysis. The database has a simple schema that allows for future extensions; it consists of four collections: users, test cases, sessions, and events.

- Each documentation owner (user) has a set of test cases;
- Each test case has a name, entry point, and a set of test sessions;
- Each session has name and list of events. A session can be marked as gold standard (discussed later).

Events are generated as a test user displays and interacts with webpages, e.g., when topics are displayed, cross-references are followed, or text is inserted into forms. To support a wide range of web servers and browsers, we restrict event types to those supported in the HTML standard (cf., Table II). The event type is captured together with the event time and (possibly) additional data such as topic id, cross-reference target, and text inserted.

[1]http://piwik.org
[2]http://www.google.com/analytics
[3]http://www.openwebanalytics.com
[4]https://www.woopra.com
[5]http://clicky.com
[6]http://www.hitslink.com
[7]http://www.kissmetrics.com
[8]https://www.clicktale.com

TABLE II: Event types and sources generated by the proxy.

Source	Type	Data
body	Click	href, origin attributes, time, tag
body	Select	time, tag
form	Submit	time, tag
a, button, input, textarea	Focus	time, tag
a, button, input, textarea	Blur	time, tag
input, textarea	Keyup	field value
document	Scroll	position, page and screen size
document	Ready	page url, screen size

Eventually, event sequences need to be transferred to the analysis tool and stored in the database. This works differently for supervised and unsupervised testing. In the former case, a stop event forces the proxy to send the event sequence together with the specific test case id. In the latter case, a Javascript script integrated in the original webpage sends the event sequence at regular time intervals (alternatively continuously) together with the default test case id.

III. Case Study Design

In order to assess our information testing method and tool, we designed and conducted a case study. We follow the description framework outlined by Runeson and Höst [2].

A. Research Questions

There are two main research questions that we seek answers to in this study:

(i) Is information testing an effective method to assess information quality?
(ii) Is QAnalytics an efficient information testing tool?

Question (i) can be answered positively if stakeholders get new insights into the quality of their technical documentation. Question (ii) can be answered positively if QAnalytics makes it more efficient to get these insights.

B. Case and Subject Selection

The case we selected is the website of Linnaeus University (http://lnu.se). The website was developed in 2010 when Linnaeus University was founded as the merger of the universities

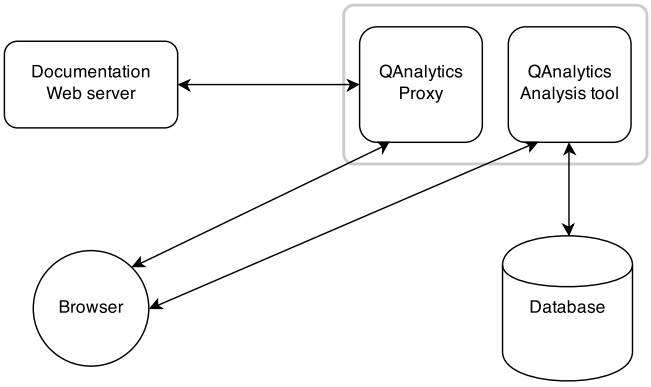

Fig. 2: The architecture of QAnalytics.

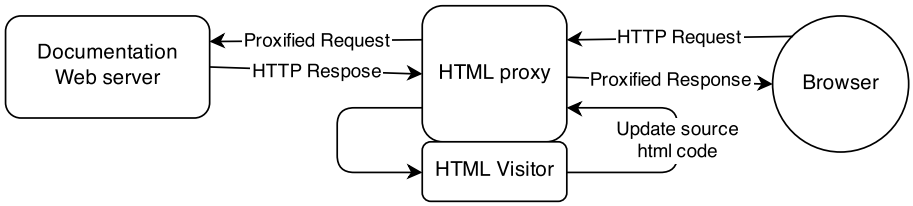

Fig. 3: The architecture and data flow of the HTML Proxy.

in Växjö and Kalmar. According to the stakeholders, the website was design happened with haste, which, in turn, resulted suboptimal (information) quality. The site owners have decided to re-engineer the website and want to get an overview of the specific quality issues. We selected this case for several reasons: 1. it is a large documentation that is used by thousands of users, 2. it was straight forward to apply our method and tool to the case, and 3. there was stakeholder interest in the outcome of the testing (and not the tool specifically).

We use both supervised and unsupervised testing. Since unsupervised testing is in the scope of other website analytics tools our main focus is to evaluate supervised testing. We opted to use students as testers for the supervised testing since they represent the largest target group. We used convenience sampling and selected the 17 students enrolled in a course in Applied Program Analysis (Spring term 2014) as test users. The first assignment of the course was to complete six test cases and submit the answers. As test questions, we selected the following six:

1) Which study programs are available?
2) Where is room Myrdal?
3) Where does professor Welf Löwe have his office?
4) Who works in Media Technology?
5) What support can I get when organizing a conference?
6) Where is the guide for Eduroam?

We selected the questions because the answers are all present on the website and it is easy to manually determine if an answer is correct or not. The questions represent information targeting both students (questions 1–3) and researchers (questions 4–6). Since we only use students as test users, we will gain some insight into whether it matters or not if test users are selected from an appropriate target group. The expected answers are not important for the case study, but are all short and well defined.

The expected completion time and the expected usage path are defined for each test case. In order to find these, we let an expert user — who could be expected to know where to find the answers to the above questions but not necessarily the answers — find the information on the website. In fact, we collected data from this expert user just like for the test users using the same data collection procedure, cf. Section III-C, and promoted the expert session as the gold standard for both the expected times and paths for each question.

C. Data Collection Procedure

For unsupervised testing, we measured how frequently pages were loaded by clicking on a link and the overall time a user spent on a page. We traced and collected corresponding events on the entire http://lnu.se website.

For supervised testing, the data collection procedure was planned as follows. All test cases should be predefined in QAnalytics and corresponding links should be sent to the test users. For each test case, a test user (including the expert user defining the gold standard) has to create a test session first with a session name equal to the test user's real name for easy distinction. Each test users should then start the session, use the website to answer the test case question (starting at the main entry page http://lnu.se for each question), and stop the session explicitly when they either found the answer or decided to give up. This process should be repeated for each

test case. The test users should then write down and submit their answers.

Test users should be instructed how to perform data collection and that test is supposed to assess the quality of the website and not their own ability to search the website. We explicitly encourage them to stop searching when they feel it is difficult to find the answer and to mention this in their report.

D. Analysis Procedure

For unsupervised testing, we wanted to highlight the difference between pages displayed and the time users spent on the pages. Therefore, we defined two simple metrics. Let T_{10} (T_{100}, resp.) be the top 10 (top 100) pages that the users spent the most time on. Let C_{10} (C_{100}) be the top 10 (top 100) pages that most users visited (clicked on a link to). We can now determine the ratio of pages that are among the top 10 (top 100) in both sets, $common_{10}$ ($common_{100}$).

$$common_{10} = \frac{|T_{10} \cap C_{10}|}{10} \tag{1}$$

and

$$common_{100} = \frac{|T_{100} \cap C_{100}|}{100} \tag{2}$$

For supervised testing, we want to highlight the differences between intended and actual user behaviour. Let S be the set of all captured sessions for a test case. For each test case, we define the *Right Answer Sessions* (RAS) to be the set of sessions that report correct answers. We use Eq. 3 to determine the ratio of successful sessions, $pass$.

$$pass = \frac{|RAS|}{|S|} \tag{3}$$

We want to distinguish sessions that report correct answers from sessions where the right answer is displayed to the test user. The latter can be computed automatically using the gold standard session; if the (segment of the) last page of the gold standard session is contained in a regular session (and it is displayed for sufficiently long time to read it) we conclude that the correct answer was displayed to the user. For each test case, we define the *Success Criteria Sessions* (SCS) to be the set of sessions that displayed the correct answer and $find$ their ratio (cf. Eq. 4). Note that we determine the right answer from the gold standard, so if the correct answer is found on a different page (as well), that session is not in SCS (but it will be in RAS).

$$find = \frac{|SCS|}{|S|} \tag{4}$$

$$P = \frac{|RAS \cap SCS|}{|SCS|} \tag{5}$$

$$R = \frac{|RAS \cap SCS|}{|RAS|} \tag{6}$$

$$A = \frac{2PR}{P + R} \tag{7}$$

We define precision (P) as the ratio of sessions with a correct answer *and* the right information displayed over the set of all sessions with right information displayed (cf. Eq. 5), i.e., the percentage of sessions where the right information was displayed and the test user found it. A low precision might indicate that information is easy to find but hard to understand. We define recall (R) as the ratio of sessions with a correct answer *and* the right information displayed over the set of all sessions with correct answers (cf. Eq. 6), i.e., the percentage of sessions where the right information was found where the expert (gold standard) expected it. A low recall suggests that the requested information can be found on multiple pages. Accuracy (A) is defined as the harmonic mean of P and R according to Eq. 7. Note that $P = R = A = 1$ iff $RAS = SCS$.

We determine the average time (μ_{time}) and number of clicks (μ_{clicks}) for a test case and their standard deviation (σ_{time} and σ_{clicks}). Longer times, more clicks, and greater standard deviations indicate that the information is hard to find.

Finally, we analyze deviations from the gold standards with respect to click events for each test case. We compute the Levenshtein distance between the sequence of click events of each session and the sequence of click events of the corresponding gold standard session. The distance is defined as the minimum number of edits needed to transform one event sequence into the other; edit operations are insertion, deletion, or substitution of a single click event. High distance from the expectation indicates that the website is not used as expected or intended. We determine the average distance (μ_{dist}) and the standard deviation (σ_{dist}) for each test case.

IV. Results

A. Execution

We collected data for the unsupervised part of the case study between May 22nd and Aug 9th, 2014. The collected data contains 2,556,284 relevant events (clicks, scrolling, page loads, etc.). Users loaded 864,896 pages from the http://lnu.se domain. These collected events and page loads were grouped into 319,798 sessions based on the network addresses and heuristics to determine session start/end. We identified 114,966 unique network addresses. Note that the network address is not a good way to identify unique users since it can be dynamically assigned and thus reused for different users. This is, for example, the case on the internal wifi networks at Linnaeus University.

We executed supervised testing part of the case study as planned in March 2014. We set up the test cases and defined the gold standard session before we handed out the test case URLs to the test users along with oral and written instructions. The preparations took less than 30 minutes, excluding oral instructions.

Out of the 17 test users there was one who created several sessions per test case (up to five per test case instead of one as expected). Only some of the sessions are in SCS for their respective test cases and those sessions are not always the last created. We decided to exclude the test user and his sessions

TABLE III: Results of Analysis of the Case Study.

Test Case	$pass$	$find$	P	R	A	μ_{time}	σ_{time}	μ_{clicks}	σ_{clicks}	μ_{dist}	σ_{dist}
1	0.875	0.31	1.0	0.357	0.526	111	81.872	5	3.230	8	3.156
2	1.0	0.19	1.0	0.187	0.315	171	269.46	7	3.889	10	3.887
3	1.0	0.44	1.0	0.437	0.608	117	123.69	8	8.986	14	8.972
4	0.875	0.06	1.0	0.071	0.132	143	118.65	9	6.359	14	6.334
5	0.188	0.19	1.0	1.0	1.0	429	558.57	18	12.656	23	12.65
6	1.0	0.06	1.0	0.062	0.116	142	147.26	9	9.024	18	9.013

from further analysis. The (remaining) test users conducted the tests in less than 20 minutes (on average) including setting up sessions (excluding the time it took to write the report). Altogether we collected data for six test cases, each with 16 sessions of events and one gold standard session.

B. Analysis and Interpretation

1) Unsupervised Testing: Figures 4 and 5 show the results of the unsupervised testing. The x axes of the two diagrams show the sets T_{100} and C_{100}, respectively, and the y axes show the time spent on each page and the clicks on a link to each page, respectively.

There is a large difference between pages selected and the time spent on these pages. If we consider the top 10, only 1 page is common to the two sets ($common_{10} = 0.1$), and if we consider the top 100, $common_{100} = 0.31$. This clearly suggests that the most time is not spent on the most frequently loaded pages (and the opposite).

Interpretation of Results for the Website Stakeholders: The pages in T_{10} are either most relevant for users or difficult to understand. A re-engineering of the website should focus on these pages and make them easy to find, understand, and use. Although the time spent on each of the 10% fractions of the pages is rather evenly distributed, this information might provide additional information for longer pages on exactly where the issues can be found.

Interpretation of results for our research questions:

The low values of $common_{10} = 0.1$ and $common_{100} = 0.31$ support our idea that it is indeed important to consider the times spent on the pages in addition to page clicks.

2) Supervised Testing: Table III presents the results of the supervised testing. The first column contains the number of the test case, and the following contain the results of the analyses defined in Section III. In the discussion below, we distinguish a possible interpretation of the results for stakeholders from an interpretation for our two research questions.

Interpretation of Results for the Website Stakeholders: The fraction of successful answers $pass$ is quite high (except for Test case 5). This sample of test cases suggests that only part of the website needs to be re-engineered. The ratio of sessions that displayed the right answers, $find$, is quite low (except for test case 5). This suggests that the information could be found elsewhere or that the test users already knew the answers.

All test cases have ideal precisions. This means that when a user finds a page with the right information he or she can easily understand it. This suggests that the page design itself is acceptable.

In most test cases, the recall is rather low. Again, this means that requested information can be found in different ways on different pages. This suggests that information is quite redundant and the number of webpages might be reduced, which, in turn, makes the website more maintainable. In Test case 5 the right answer can only be found on one specific page, so the recall is $R = 1$.

The average search time and number of clicks vary quite a bit between the sessions. For instance, the answer to Question 5 was much more difficult to find than, for example, the answer to Question 1. This is perfectly fine since students ought to find study programs quickly, while finding conference support is a lesser issue for this target group. However, the high time and clicks deviations show that the test users interacted with the website in very different ways. There is no natural way to try to get answers to the questions. There were a few extreme outlier sessions that we analyzed in detail. These show that the corresponding test users went back and forth on the website, scrolled up and down on pages and spent more time on an individual page before they continued. The Levenshtein distance metrics also indicate a quite large deviation of the actual from the expected behavior.

In conclusion, the results of this limited study suggest that the website of Linnaeus University can be improved with respect to information organization and by removing duplicated information. Test users search for information in different ways; a re-engineering should simplify and streamline search paths. The information on each individual page seems to be appropriate.

Interpretation of Results for our Research Questions: In general, we found information about the website that the stakeholders considered relevant. Information testing offers insights into how easy it is to find and understand information. It allows to differentiate this information for the website structure and individual pages. All these are indications that (i) information testing is an effective method to assess information quality.

The time to set up test cases (approx. 5 minutes per test case), test (on average less than 4 minutes per test case) and analyze the outcome (less than a seconds per test case) are low due to the automation provided by QAnalytics. All the testing was done in parallel and in parallel with regular use of the website. The website did not need any modifications. These are all indications that (ii) QAnalytics is an efficient information testing tool.

However, there are also some insights that will improve our method. The accuracy is low, i.e., RAS and SCS are quite

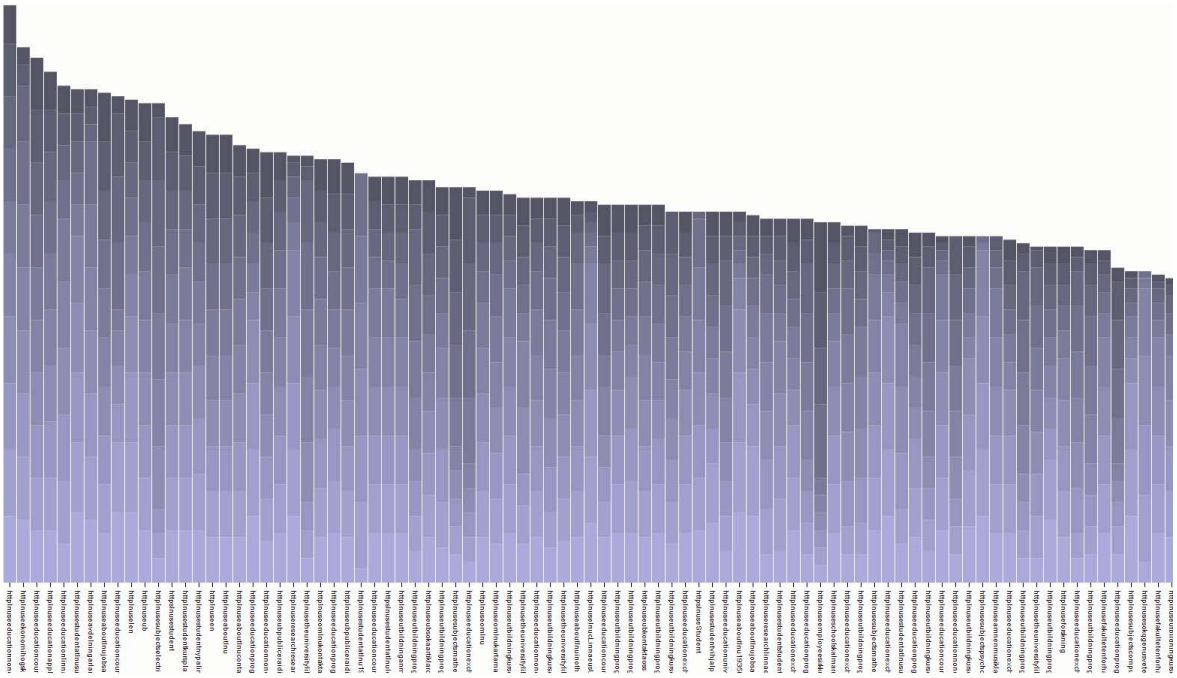

Fig. 4: The top 100 pages according to time (in seconds) spent on these pages. For each page, the total time is divided into 10 segments (each using a different grey scale) that each correspond to the time spend on that specific 10% of the page. All times are given on a log scale.

disjoint. So we cannot use SCS instead of RAS, i.e., we cannot approximate $pass$ with $find$ and, hence, we cannot automate the decision of whether a session was successful or not. Moreover, the successful sessions used different paths to the information, e.g., followed the site structure or used direct search. Deviations in time and clicks from the gold standard can either be interpreted as problems to find the information or as different (valid) paths to the information.

Together, the two observations suggest that we should add a pre-testing phase where a small group of test users finds answers to a test question and reports success manually. Manual inspection of successful test sessions by experts will identify different valid paths. Successful sessions with different paths are promoted to gold standard sessions until $SCS \approx RAS$. Now the actual test phase with a larger group of test users does not require manual reports. There is already support for this modification in QAnalytics; more than one session can be marked as gold standard and the distance is calculated as the *smallest* distance from a session to any of the gold standards.

Finally, we observed considerably worse success indicators for test cases 4–6 (where the target users are researchers) compared to test cases 1–3 (where the target users are students). This suggests that selecting test users from the target user group indeed has an impact on the test result.

C. Threats to Validity

The test cases only cover a fraction of the website, so we cannot make any general claims about its information quality. Hence, we should at the current stage refrain from any general statement on the information quality of the whole website. Moreover, an assessment of test coverage analysis (as a method or as part of the QAnalytics tool) was not applicable in this case study.

We tested the functionality of QAnalytics on many different websites prior to this case study and we are confident that it works as specified. However, the case study only concerned a single website. This means that we must not overstate the findings about information testing and QAnalytics as a support tool.

The group of test users was quite small and many of them are neither Swedish nor English natives (the two languages used on the website). They all have a computer science background and most are male. Hence, the selection of test users was neither a large enough nor a representative sample of the target users of the website. The developers of information testing and QAnalytics set up the test cases and instructed the test users. The test users were all enrolled in an advanced (M.Sc.-level) course in computer science. Hence, the assessment of the efficiency of the tool is based on expert users.

V. Related work

There are several tools that rely on a proxy and/or client-side scripting to evaluate website use to overcome the limitations posed by traditional log file analysis, such as the inability to capture user interaction on a single page, e.g., scrolling. Early efforts, such as the Web Variable Instrumenter Program (We-

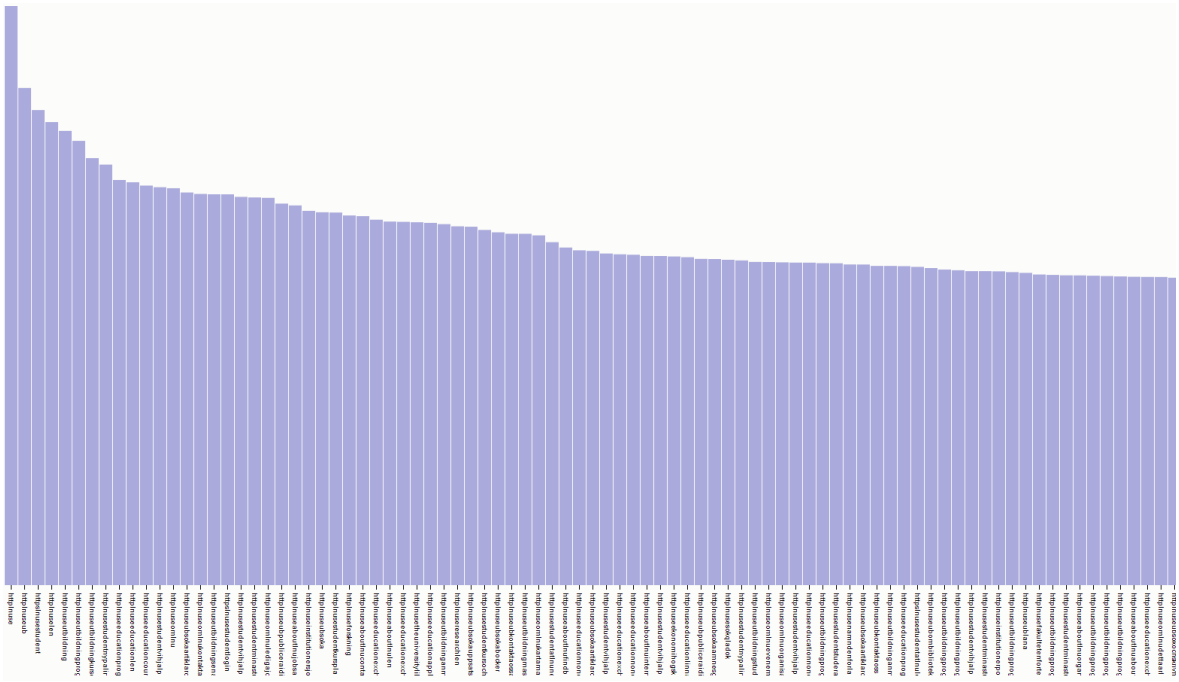

Fig. 5: Top 100 pages according to clicks on a log scale.

bVIP) [9], intruments various HTML elements with Javascript scripts that log interactions. These logs can be processed by VISVIP [3] to visualize the navigation paths taken by users. The automated testing tool WET [4] relies on a similar approach but only requires a single Javascript to be added to each webpage, which often can be done automatically by the webserver.

Tools such as WebQuilt [5] instead relies on an (http) proxy to log page visits. The proxy makes it possible to test several sites, without any modification to the original pages. MouseTrack [6], UsaProxy [7], WebinSitu [8], and our QAnalyticsuse a proxy to instrument webpages dynamically. This allows for a fine-grained logging of user events without any modification to the original website. Many tools, e.g., WELFIT [9] transfer the logs to a (server side) database to enable mining. The tools presented in Table I rely on modifications to the webpages, since the information is collected continuously and of interest to the website owners.

Ivory and Hearst [10] distinguish between formal and informal use of a website during testing; this is similar to our distinction between supervised and unsupervised testing. Most the of the tools discussed above focus on informal use or unsupervised testing. In WebRemUSINE [11] the user interaction is logged and the resulting logs can be compared to a previously configured "optimum" task model. This allows for formal use or supervised testing.

[9]http://zing.ncsl.nist.gov/WebTools/WebVIP/

VI. Conclusions

The case study presented in this paper indicates that information testing is an effective method to assess the information quality of a website, and that our tool, QAnalytics, provides and efficient way to perform the assessment. We suggest information quality metrics and their interpretation by analyzing the information test data. For instance, precision and recall can be used to determine how understandable individual pages are and how many of the information and paths to it are duplicated, respectively. We also found that deviations in the number of clicks and time spent can be used support these findings. Finally, the case study pointed out ways to further improve our method, e.g., to handle several valid paths to information and to help further automate success detection.

There are, however, a number of validity concerns. We have only applied the method and tool to a single website and a homogeneous group of testers. We currently applying the method and tool to other cases, for example a large documentation for a commercial telecom product (i.e., not a traditional website). This will expose the tool to larger group of test users and managers. We also need to consider coverage and other quality indicators for the testing. While we currently define coverage analogously to software testing, it is not obvious that this is a sound approach if we for example consider page and path coverage.

References

[1] A. Wingkvist, M. Ericsson, W. Löwe, and R. Lincke, "Information quality testing," in *Perspectives in Business Informatics Research*, ser. Lecture Notes in Business Information Processing, 2010, vol. 64, pp. 14–26.

[2] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, Apr. 2009.

[3] J. Cugini and J. Scholtz, "VISVIP: 3D visualization of paths through web sites," in *Proceedings of the 10th International Workshop on Database and Expert Systems Applications*, 1999, pp. 259–263.

[4] M. Etgen and J. Cantor, "What does getting WET (web event-logging tool) mean for web usability," in *Proceedings of Fifth Human Factors and the Web Conference*, 1999.

[5] J. I. Hong, J. Heer, S. Waterson, and J. A. Landay, "Webquilt: A proxy-based approach to remote web usability testing," *ACM Trans. Inf. Syst.*, vol. 19, no. 3, pp. 263–285, 2001.

[6] F. Mueller and A. Lockerd, "Cheese: tracking mouse movement activity on websites, a tool for user modeling," in *CHI'01 extended abstracts on Human factors in computing systems*. ACM, 2001, pp. 279–280.

[7] R. Atterer and A. Schmidt, "Tracking the interaction of users with ajax applications for usability testing," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2007, pp. 1347–1350.

[8] J. P. Bigham, A. C. Cavender, J. T. Brudvik, J. O. Wobbrock, and R. E. Lander, "Webinsitu: a comparative analysis of blind and sighted browsing behavior," in *Proceedings of the 9th international ACM SIGACCESS Conference on Computers and Accessibility*, 2007, pp. 51–58.

[9] V. F. De Santana and M. C. C. Baranauskas, "Summarizing observational client-side data to reveal web usage patterns," in *Proceedings of the 2010 ACM Symposium on Applied Computing*. ACM, 2010, pp. 1219–1223.

[10] M. Y. Ivory and M. A. Hearst, "The state of the art in automating usability evaluation of user interfaces," *ACM Computing Surveys (CSUR)*, vol. 33, no. 4, pp. 470–516, 2001.

[11] L. Paganelli and F. Paternò, "Intelligent analysis of user interactions with web applications," in *Proceedings of the 7th international conference on*