EIAO
European Internet Accessibility Observatory
Project no.: IST-2003-004526- STREP

UWEM Indicator Refinement

Version:	0.9
Date:	2007-05-25
Author:	Annika Nietzio, Nils Ulltveit-Moe, Terje Gjøsæter, Morten Goodwin Olsen
Dissemination Level:	internal
Status:	RC

This document consists of 55 pages including this cover

Version Control

Version	*Status*	*Date*	*Change*	*Author*
0.1	DRAFT	2007-05-02	Initial version based on minutes from complementary discussion in Brussels (2007-04-25) First version of Requirement section (with input from Nils Ulltveit-Moe, Morten Goodwin Olsen, Terje Gjøsæter, and Mkael Snaprud)	Annika Nietzio
0.2	DRAFT	2007-05-02	Intermediary version of Requirement section Integrated comments from Nils Ulltveit-Moe and Morten Goodwin Olsen	Annika Nietzio
0.3	DRAFT	2007-05-03	Stabilised version of Requirements section	Annika Nietzio
0.4	DRAFT	2007-05-03	Added compatibility and dependency information to Requirement section	Annika Nietzio
0.5	DRAFT	2007-05-07	Updated according to external review.	Nils Ulltveit-Moe Annika Nietzio
0.51	DRAFT	2007-05-08	Update of Web Acc requirements according to external review	Annika Nietzio
0.52	DRAFT	2007-05-08	Update of Maths requirements according to external review	Annika Nietzio
0.53	DRAFT	2007-05-08	Update of Statistical requirements according to external review	Annika Nietzio
0.54	DRAFT	2007-05-08	Added analysis of sampling and stop criteria requirements.	Nils Ulltveit-Moe
0.55	DRAFT	2007-05-08	Separated sampling and stop criteria, renumbered them and moved analysis of requirements to discussion after each chapter.	Nils Ulltveit-Moe
0.56	DRAFT	2007-05-09	Added analysis of web accessibility and mathematical requirements	Annika Nietzio
			Added discussion of implementation requirements	Terje Gjøsæter
0.6	DRAFT	2007-05-09	Added References	Annika Nietzio
0.61	DRAFT	2007-05-09	Revision of Introduction, added appendix with Mind Map provided by Terje Gjøsæter	Annika Nietzio
0.62	DRAFT	2007-05-10	Updated after inspection by Terje Gjøsæter and Morten Goodwin Olsen, eloborated discussion of statistical requirements	Annika Nietzio
0.63	DRAFT	2007-05-10	Updated after inspection by Nils Ulltveit-Moe	Annika Nietzio
0.64	DRAFT	2007-05-10	Updated according to comments by Gottfried Zimmermann, and added sampling algorithm feasibility study.	Nils Ulltveit-Moe
0.65	DRAFT	2007-05-11	Updated section on Stop Criteria	Nils Ulltveit-Moe
			Updated section on Implementation requirements	Terje Gjøsæter
			Added section on score function design	Annika Nietzio
0.66	DRAFT	2007-05-11	Updated after inspection	Annika Nietzio, Nils Ulltveit-Moe, Terje Gjøsæter, Morten Goodwin Olsen
0.67	DRAFT	2007-05-11	Updated section on score function design	Annika Nietzio
0.7	RC	2007-05-11	Updated after internal inspection.	Nils Ulltveit-Moe
0.71	DRAFT	2007-05-14	Updated sections 2.2, 2.3, 2.4, and 4 according to comments from external	Annika Nietzio,

Version	*Status*	*Date*	*Change*	*Author*
			experts. Added Appendix B	
0.72	RC	2007-05-16	Updated after internal inspection (sections 3 and 4) Added acknowledgement.	Annika Nietzio
0.73	DRAFT	2007-05-18	Updated sections 2.1 and 3	Nils Ulltveit-Moe
0.74	DRAFT	2007-05-18	Updated section 3 with a new algorithm for detecting change	Morten Goodwin Olsen Anis Yazidi
0.75	RC	2007-05-19	Added experimental results Added comments from Nils Ulltveit-Moe	Anis Yazidi Morten Goodwin Olsen
0.76	RC	2007-05-22	Updated after comments from GZ	Terje Gjøsæter
0.77	RC	2007-05-23	Updated after new input from reviewers	Terje Gjøsæter
0.78	RC	2007-05-23	More updated after more new input from reviewers	Terje Gjøsæter, Nils Ulltveit-Moe
0.8	RC	2007-05-23	Boiled down section 3 and improved section 4.	Nils Ulltveit-Moe
0.81	RC	2007-05-25	Some fixes, corrections and improvement suggestions from Morten and Nils, and AW's comments added to text marked with yellow. Added a bit to section 5.	Terje Gjøsæter
0.9	RC	2007-05-25	Additional fixes and comments after external review.	Nils Ulltveit-Moe

Table of Contents

1 Introduction

1.1 Scope of this document

EIAO has been asked to produce an indicators paper, based on the previous specifications for the current implementation and on the ideas discussed in the review meeting and complementary discussion which took place in Brussels on 2007-04-25.

The problems identified by the experts concern three different parts of the Observatory / UWEM:

1. sampling algorithm
2. aggregation model, metrics
3. visualisation of statistics (GUI, scorecards)

The EC has a strong interest that the results presented by EIAO are statistically sound. Therefore the underlying indicator calculations have to be well motivated and without statistical flaws. EIAO has been asked to develop a concept for the revision of the UWEM aggregation model.

As one of the reviewers pointed out :"The crucial thing about indicators is, that you have to know what you want to do." To find out what we want (or need) to do the following **refinement process** was suggested:

- Identify requirements and desirable properties of the model.
- Some requirements may be incompatible. Make a choice based on your expertise in the area of web accessibility. Motivate your choices and give reasons in your report. It might be helpful to look at pathological cases to identify (un-)desirable properties.
- Build a mathematical model and select algorithms that fulfil the requirements.

Note that the basic need for sampling, aggregation and presentation as score card is seen as a prerequisite that will not be further discussed in this paper. In detail the prerequisites are the following:

1. The score from a sample of the pages within a site can be used as a predictor of the score for the whole site.
2. The Observatory calculates the score for a web site (where higher score means worse accessibility).
3. The calculation is based on output of UWEM tests (= EIAO B-WAMs).
4. Scores are mapped to score cards for presentation.

Furthermore there are some prerequisites which are of special importance for the Observatory:

5. The web site score lends itself naturally to further aggregation over groups of web sites, like regions or sectors.
6. There can be multiple functions giving scores at the site level.

7. Algorithms must be compatible with the copyright licence of the Observatory (GPL), which means that patented technologies should be avoided.
8. Functions and algorithms can be implemented and computed efficiently, to allow for large-scale automatic evaluation.
9. The score can accommodate results from heuristic tests. (The results of heuristic tests are probability values instead of binary "pass" or "fail" results. The probability is indicating the confidence of the result.)

Furthermore there are some assumptions about the properties of web accessibility barriers.

- The occurrences of barriers of different type are independent.

Most of the methods and findings in this paper belong to the underlying methodology and should be integrated in the upcoming version of UWEM (version 1.2). But some methods and requirements are EIAO specific (e.g. statistical analysis beyond web site level, presentation of the results in the GUI) and should be described in the relevant EIAO project deliverables.

1.2 Related work and readers instructions

This document is related to the following documents:

- **UWEM 1.0** (D-WAB4 Unified Web Evaluation Methodology)
- **D5.1.1.1-2 Functional Specification** (of EIAO Release 2)

The rest of this document is organised as follows: Section 2 Contains analysis of requirements for sampling, and requirements related to web accessibility, mathematics and statistics of indicators. In section 3, possible sampling algorithms are presented and discussed. Section 4 discusses aggregation models, and section 5 concerns presentation and visualisation of results.

1.3 Terms and notation

In this paper we will use the following terms and notation.

A *web page* is a resource that is referenced by a URI and is not embedded in another resource, plus any other resources that are used in the rendering or intended to be rendered together with it.[1]

The *score* is a single number that indicates the accessibility of a web site. The score is calculated by the *score function*. Note that we use the term score to refer to the output of the score function.

A *UWEM test result* is the outcome of a single UWEM test that can be attributed to the application of a specific UWEM test to a specific (unique) location within a specific resource. The outcome is PASS or FAIL. If heuristic tests are included it can also be a probability value.

[1] Definition from WCAG 2.0 http://www.w3.org/WAI/GL/WCAG20/WD-WCAG20-20070220/#webpagedef

The table below summarises the mathematical notation that is used in this paper.

Notation	***Definition***
p	web page
s	web site
$n(s)$	sample size (number of pages sampled from site s)
$m(s)$	margin of error for site s
t	UWEM test
T	total number of UWEM tests
$f(p)$	score of page *p*
$F(s)$	score of site *s*
A_{pt}	Binary report (1 if any application of t within p failed, 0 if all tests passed).
B_{pt}	Number of fail results of *t* within *p*
N_{pt}	Number of applications of test t within *p*
R_{pt}	Ratio (calculated as B_{pt}/N_{pt})
A_p	sum of all A_{pt}
B_p	Number of failed tests in *p* (sum of all B_{pt})
N_p	Number of all tests carried out in *p* (sum of all N_{pt})
R_p	Ratio (calculated as B_p/N_p)
D_p	Barrier diversity of web page *p* $D_p = 1 - \sum_{t=1}^{T} (\frac{B_{pt}}{B_p})^2$
B_{st}	Number of fail results of *t* within sample from *s*
N_{st}	Number of applications of test *t* within sample from *s*
R_{st}	Ratio (calculated as B_{st}/N_{st})
B_s	Number of failed tests in *s* (sum of all B_{st})
N_s	Number of all tests carried out in *s* (sum of all N_{st})
R_s	Ratio (calculated as B_s/N_s)
D_s	Barrier diversity of web site *s* $D_s = 1 - \sum_{t=1}^{T} (\frac{B_{st}}{B_s})^2$

Table 1: Overview of mathematical notation

1.4 Acknowledgement

The EIAO team would like to thank the external experts Gottfried Zimmermann and Andrew Westlake for their useful proposals and comments on several versions of this paper. Their expertise in the areas of web accessibility and statistics has been very helpful.

2 Analysis of requirements and desirable properties

This section is based on the outcome of the ***EIAO Indicators workshop*** held by EIAO on 2007-04-30. The participants were Mikael Snaprud, Nils Ulltveit-Moe, Morten Goodwin Olsen, Terje Gjøsæter, and Annika Nietzio. Subsequently, it has been updated according to ***comments from the external reviewers*** Gottfried Zimmermann and Andrew Westlake in phone conferences and through email.

For easier overview the identified requirements and desirable properties are grouped by category. The Appendix A contains mind map figures depicting the dependencies and incompatibilities of the requirements.

2.1 Sampling requirements and desirable properties

This section contains requirements on how to acquire a sample from the web site and how to define the stop criteria for sequential sampling of web sites.

2.1.1 Sampling algorithm

SAMPLING 1

The sampling algorithm (especially the stopping criterion) does not have effectively deterministic effects.

SAMPLING 2

The sampling is based on web pages.

SAMPLING 3

The sampling algorithm selects a near uniform random sample from the web site.

SAMPLING 4

The sampling algorithm selects a representative sample from the web site.

SAMPLING 5

The sampling algorithm should avoid being biased.

(a) The sampling should not favour pages with a large number of in-links.

(b) The sampling algorithm should work in a dynamic environment, where there will be a flow of updates of the web pages.

(dependent on **SAMPLING 3** and **4**)

Sampling 6

The order of the links in a page should not affect how the links are chosen. (This is in contrast to the "reading" approach of the random walk used in EIAO release 2 beta).

Sampling 7

The sampling algorithm does not depend on the size of a web site.

Sampling 8

The sampling precision can be estimated.

There are parameters to control and tune the sampling precision. (E.g. decrease the margin of error or increase the confidence level).

Sampling 9

Sampling is independent of the number of tests.

Sampling 10

The sampling algorithm can be carried out in an automatic fashion (with no humans involved).

Sampling 11

The sample should be based on, or contain the UWEM core resource set.

(Discarded in discussion)

Sampling 12

Sampling is agnostic to the underlying document format.

2.1.2 Discussion of sampling algorithm requirements and desirable properties

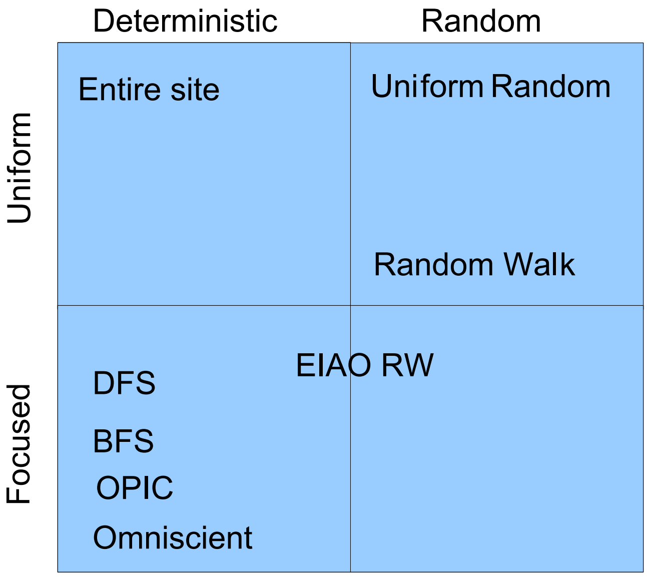


Illustration 1: Overview over different sampling strategies.

Sampling strategies can be subdivided into focused and uniform selection strategies. A focused strategy typically uses a page importance measure, e.g. PageRank, to crawl the most important web pages first. The PageRank measure is based on the assumption that web pages that have many URLs pointing to them, are important.

Search strategies can also be grouped according to whether they use deterministic or random strategies when selecting web pages. Many of the focused crawling strategies are based on deterministic algorithms, whereas the only deterministic algorithm that would provide a uniform selection of a web site would be to crawl the entire web site.

Examples of sampling strategies [PR1][COMP06], are:

- Uniform deterministic sampling
    - Entire site: Sites are chosen uniformly at random with a certain probability, and all pages within a site are included in the sample, e.g. by an exhaustive breadth-first (BFS) or depth-first (DFS) search.
- Uniform random sampling
    - Uniform random sampling: Pages are chosen uniformly at random with a certain probability[2].
    - Random walk: A near-uniform random sample can be achieved by performing a random walk through interconnected web sites. The random walk algorithms have built-in mechanisms to avoid bias for pages that many links point to. Several random walk based algorithms exist. The algorithm proposed by Henzinger et al

[2] Note that implementing this would require complete knowledge of the web graph in advance.

[HENZINGER00] is based on a random sampling stage, to select a random URL from the URL repository and a walking stage, where it performs a random walk where the probability to visit a page is weighted inversely to the pagerank, to counteract the bias from web pages with many in-links; i.e. many links pointing to it. This algorithm also takes a random decision to start a new walk, which avoids too deep walks. The algorithm proposed by Bar-Yossef et al [BYossef] was developed in parallel with the Henzinger algorithm and is based on modifying the web graph to be undirected and regular. Regularity is achieved by adding sufficiently many self-loops, so that every page ends up with the same number of links as the node with maximum number of links pointing to it. This approach requires an initial mixing time before the distribution can be approximated as uniform. The Bar-Yossef approach performs relatively deep crawls, which makes it vulnerable to getting stuck in areas of the graph with few links out and have been shown to be biased towards pages with many in-links. The Paat R. algorithm[PAATR] is a more recent algorithm that builds upon and improves the Henzinger and Bar-Yossef approaches.

- Focused deterministic sampling
    - Omniscient order (or quality-first order): the crawler uses a queue prioritised by PageRank values; in other words, it chooses to visit the page with highest quality among the ones in the frontier.
    - Depth-first search (DFS): the crawler chooses the next page as the last that was added to the frontier; in other words, the visit proceeds in a LIFO fashion.
    - OPIC (online page importance computation): an algorithm for ranking pages while discovering them. It can be seen as a biased breadth-first search in which the pages that are highly interlinked are more likely to be chosen.
    - BFS (Breadth-first search): the crawler chooses the next page as the first that was added to the frontier; in other words, the visit proceeds in a FIFO fashion.

The current algorithm used by EIAO was intended to be a near-uniform sampling algorithm, but has been shown to be focused towards highly interlinked web pages. Furthermore, it is biased towards the first URLs on a web page and there is a dependency between the evaluated web pages due to consecutive interlinked web pages being evaluated. The EIAO algorithm is therefore placed almost in the middle of the figure – it is neither properly focused, nor uniform and is random but with effectively deterministic characteristics. We therefore need to consider alternative sampling algorithms.

The first decision is whether we should base the sampling on a focused or a uniform selection strategy. Our position, is that a uniform selection strategy is preferable. Among the sampling requirements, **Sampling 11** – Sampling based on/containing the UWEM core resource set contradicts this position.

Sampling 11 assumes that some web resources (e.g. the home resource, Contact information, Help, Site Map) are more important, and that the sampling algorithm should sample a representative set of pages with different technologies and services. It would imply some kind of focused crawling strategy, like e.g. a BFS search, that would start on the home resource and, continue its way deeper through the web site, and stop after a certain number of web pages have been acquired. With such an algorithm it is assumed that the most important web pages are close to the home resource, and that the sample would be large enough to include the web resources in the core resource set. Other possible focused approaches would be to use the PageRank measure as a measure of importance, indicating the the web pages with most links to them are most important or define importance as how close (in number of links) a web page is to the home page, assuming that pages are more important the closer they are to the home resource.

Basically, following **Sampling 11** could be compatible with the following requirements:

- **Sampling 2** Sample based on web pages.
- **Sampling 4** Representative sample (assuming that **Sampling 11** defines representativeness).
- **Sampling 9** Sampling is independent of number of tests.
- **Sampling 12** Sampling is agnostic to the underlying document format.

The problem with a focused approach, is that it is inherently biased towards a certain set of web pages, which means it is not statistically sound on web site level. It would violate **Sampling 1, 3, 5, 8.** In addition, the **Sampling 11** requirement is only suitable for expert evaluation, since detecting parts of the core resource set is not automatable **Sampling 11** is therefore in conflict with **Sampling 10**.

The conclusion is therefore that we should focus on getting a uniform sample from the web automatically, and not a focused sample. Because of this we will drop requirement **Sampling 11**.

Illustration 1 indicates 3 ways to achieve a uniform sample from the web site:

- Sample the entire web site
- Uniform sample
- Random walk strategies

We consider each of these possibilities. 1. Sampling and evaluating the entire web site would give the most exact measure, however it can be disputed whether this is the most cost-effective way of sampling, since a random uniform sub-sample would provide sufficient precision without evaluating all web pages.

2. Achieving a uniform sample may be possible if the web site is pre-scanned to identify and store all URLs. A uniform sample could then be drawn from this complete URL repository. If the URL repository stores web page timestamps, and uses the if-modified-since HTTP header to conditionally load only changed web pages, then a considerable amount of effort could be saved in subsequent URL pre-scannings, since most web sites can be expected to update only a fraction of the site between monthly measurements. It should also in principle be possible to reuse evaluation results for web pages that have not changed since the last evaluation. A problem with this approach, is that some websites may be very large and some may not even be possible to crawl exhaustively, e.g. due to dynamically generated content, so 2. may violate prerequisite 8 in some cases; i.e: that functions and algorithms can be implemented and computed efficiently for large-scale automatic evaluation. This will be discussed further in the algorithm design in section 3.

3. A near-uniform random walk algorithm is a good way to get a moderately representative sample from a web site, but it can never be a fully random sample from the full set of pages within a site, because at each stage, the set of accessible pages is conditional on the path already traversed. However, there will be an initial bias with a near-uniform random walk

algorithm, and such an algorithms should avoid selecting dependent web pages, by separating the discovering of structure and random sampling. Because the selection and discovery of structure needs to be separated, then there will be more efficient approaches for filling the URL repository than using a random walk strategy, since a random walk strategy will converge slowly towards knowledge of all URLs. We will therefore not consider any new random walk strategies in the algorithm feasibility study in section 3.

Another challenge with one the Henzinger near uniform random walk algorithm[HENZINGER00], is that it is patented, so it violates the prerequisite 7.

Patent holder	***Algorithm***	***Patent reference***
Hewlett-Packard	System and method for near-uniform sampling of web page addresses	US pat. 6594694

This means that a near uniform random walk algorithm based on inverse PageRank weighting is not compatible with the copyright license of the observatory (GPL).

2.1.3 Sampling with or without replacement

The most commonly used formulas for calculating sample variance is based on sampling with replacement. An advantage with sampling with replacement, is that independence can be assumed between the different samples, which simplifies the calculations. However, this requires that a large population and a small sample fraction of the total population is assumed.

In our case, the sample fraction will often be a substantial part of the population, and in some cases the entire population, which means that sampling without replacement will give the most correct picture of the variance and error margin. The margin of error will be smaller in this case, than if the formula for sampling with replacement is used. A problem with using the formulas for sampling without replacement, is that it requires that the total number of web pages *N* is known, which violates requirement **Sampling 7**. To avoid this problem, we can require that the sampling algorithms *always* performs sampling without replacement from the set of known URLs. The error margin calculations will then take into account the total number of pages *N*, if all pages are known and they will assume that the population is large, and use the simpler formulas for sampling without replacement, if the total number of pages *N* is unknown.

$$m(s)=\pm\left\{\begin{matrix} z\frac{\sigma}{\sqrt{n}}, N \, unknown \\ z\frac{\sigma}{\sqrt{n}}\sqrt{\frac{N-n}{N-1}}, N \, known \end{matrix}\right\}$$

Where *m(s)* indicates the margin of error, *z* is the number of standard errors to achieve a given confidence interval. *z=2.58* gives a 99% confidence interval. σ is the standard deviation, *N* is the total number of web pages in the web site, and *n* is the sampled number of web pages.

This strategy would overestimate the error margin, if *N* is unknown, and would estimate the correct error margin otherwise. The strategy would also be in compliance with requirement **SAMPLING 7**.

2.2 Stop criteria

STOPCRIT 1

The stopping criterion of the sampling algorithm is statistically sound.

STOPCRIT 2

It should work for sequential sampling of web pages.

STOPCRIT 3

It should be based on well known statistical constructs, like confidence interval and error margin.

STOPCRIT 4

The actual coverage of the confidence interval should be close to the required level[3].

STOPCRIT 5

The stop criterion does not have effectively deterministic effects.

2.2.1 Discussion of stop criteria requirements and desirable properties

Neither of the stop criteria have been disputed. Solutions for stop criteria is discussed in section 3.6.

2.2.2 Final selection of sampling requirements and desirable properties

As outlined in the discussions above we have reached the following conclusion on the final selection of sampling and stop criteria requirements:

- **SAMPLING 1 – 10, 12**
- **STOPCRIT 1 – 5**

[3] The confidence interval calculation in requirement 4 does not have the statistical properties of a single calculation, which is the reason for the additional requirement for confidence interval coverage.

2.3 Web accessibility related indicator requirements and desirable properties

This section contains requirements that describe the relationship between the score and the features and properties of the web page (or web site resp.). The requirements have been subclassified into four categories:

1. requirements related to definition of web page (sample unit)
2. requirements related to web page score
3. requirements related to web site score
4. requirements related to further parameters

Within each category the dependencies and incompatibilities will be discussed separately.

2.3.1 Definition of web page

According to the WCAG definition, a web page does not consist solely of an HTML file but includes "any other resources that are used in the rendering or intended to be rendered together with it". Therefore the methodology (UWEM) and any additional specifications used by the Observatory need to address the issue of handling Cascading Style Sheets (CSS) and frames in web pages.

Web Acc – CSS 1

A CSS file should contribute to the score for every web page in which it is referenced.

Web Acc – CSS 2

The score for a web page that references a CSS file should be the same as if all styles were defined in-line (i.e. with style attributes).

Web Acc – CSS 3

The score for a CSS file should be evaluated separately from that of the pages to which it contributes. The contribution of the CSS score to that of the site may be weighted by the number of pages referencing the sheet.

(Discarded in discussion)

Web Acc – Frame 1

A web page that is presented in the context of a frame-set should contribute to the score for the whole frame-set (and not be evaluated separately).

Web Acc – Frame 2

The score for a frame-set should be the same as for a single page containing all the material in the set.

Web Acc – Frame 3

The score for each frame in a frame-set should be evaluated separately from that of the sets to which it contributes. The contribution of the frame to the score of the site may be weighted by the number of frame-sets referencing the frame.

(Discarded in discussion)

2.3.2 Discussion of CSS requirements and desirable properties

CSS is an important technology in web design. It is assumed that most users browse the web with CSS enabled in their user agent. Therefore it should be assessed during the web accessibility evaluation process. This is reflected in requirement **Web Acc – CSS 1**.

According to UWEM 1.0 CSS code is evaluated independently of the html code. Especially, it is evaluated independently of whether it is really applied in the rendering of the page or not. This strategy is easy to implement but has some shortcomings when it comes to interpreting its meaning.

(1) All styles are included in the evaluation, even those that are not applied because of the cascading effect, or because the styled element is not used in the current web page.

(2) The results do not reflect how often a CSS rule, which induces potential accessibility problems, has been applied. (This is in contrast to the strategy used for html tests.)

(3) The overall result does not reflect the dependencies caused by the fact that the same CSS file is referenced from multiple web pages.

Requirement **Web Acc – CSS 2** addresses the first two of the shortcomings. Its implementation will be more complicated than the current approach but nevertheless feasible. To keep the methodologies for automated and expert testing well aligned, the strategy described in UWEM would have to be changed accordingly. If the number of occurrences of a potential barrier caused by a CSS rule is counted, then there are no longer any implicit dependencies, and an additional weighting as suggested by **Web Acc – CSS 3** is not necessary.

Another strategy can be derived from **Web Acc – CSS 3**. The implementation effort would be less because a file based evaluation is already implemented in EIAO (release 2 beta) and the only missing part are the weighting parameters. Although this approach solves the third issue, it still has the shortcomings (1) and (2). Thus it is incompatible with **Web Acc – CSS 2**. It also introduces the additional complication that CSS rules included in the html file (via the <style> element) are treated differently from CSS rules in separate files – even if both contain the same code. Furthermore, this strategy is not compatible with the WCAG definition of web page.

2.3.3 Discussion of frame requirements and desirable properties

Note that the current version of UWEM (version 1.1) doesn't contain any instructions for the evaluation of frame-based web sites. This needs to be resolved.[4]

To be aligned with the choices in the CSS part **Web Acc – Frame 1** and **2** should be selected.

[4] Information to be included in UWEM:

- How to identify an frame-based web page? (reporting the URL is not enough)
- How to calculate a score for a frame-based web page? (outcome of this paper)

2.3.4 Web page score

A straightforward approach to score calculation is the aggregation of single test results on web page level. The following requirements have been identified in this category:

WEB ACC – PAGE 1

The page score gets worse if more (different types of) tests have failed. The score improves if fewer (different types of) tests have failed. In particular, a page where a number of failures occurs of several types has a worse score than one where the same number of failures is found but in fewer types of test.

(Discarded in discussion)

WEB ACC – PAGE 2

The score gets worse if a test fails more often (anywhere within the site). The score improves if a test fails less often.

(Also applies to site level)

WEB ACC – PAGE 3

The score reflects the number of failed UWEM tests (Bp), i.e. the score is an increasing function of the number of failures.

(Also applies for Bs to site level)

WEB ACC – PAGE 4

The score reflects the percentage of failed UWEM tests (Rp), i.e. the score is an increasing function of the percentage of failures.

(Also applies for Rs to site level)

WEB ACC – PAGE 5[5]

The complexity of a page is taken into account. The term complexity refers to the number of tests that are applied to a web page, i.e. the score is an increasing function of the number of tests applied to the page.

WEB ACC – PAGE 6

The score is insensitive to page size (i.e. size of the source file).

[5] Discarded in discussion.

Web Acc – Page 7

> The impact of a failed test (i.e. the amount it contributes to the page score) is independent of its position within the page.

2.3.5 Discussion of web page score requirements and desirable properties

The requirements listed in the previous section describe the behaviour of the web page score in the presence of a varying test results. We now discuss the test result properties for which an effect on the overall score is desirable (and what kind of effect this is), as well as those properties that should not have any impact on the score.

One property that could potentially be used in the score calculation is the size of the web page. This can be measured either as

(a) the size of the source file (in bytes) or

(b) by the number of tests that are applied to the page (Np).

Approach (a) is problematic because file size is not necessarily directly related to the accessibility of a page. For example, if a page consists mostly of text with only basic markup, even a long file will not have major accessibility problems, whereas complicated layouts, navigation constructs of forms might cause accessibility problems even in a small file. The desirable behaviour is that the score should be insensitive to page size (**Web Acc – Page 6**).

Option (b) which postulates a correlation between accessibility and number of tests (Np) does not have the shortcoming of option (a) but it is still problematic. A page with more test applications will usually be more complicated than a page with fewer test applications (i.e. smaller Np) and is therefore also likely to have more accessibility problems. However, the indicator that is relevant for the final score is the number or ratio of tests that really failed. These indicators (Bp and Rp) are available and should be preferred to the less accurate Np. Requirement **Web Acc – Page 5** is therefore discarded.

The position of a failed test within the page is not taken into account for the score calculation. On the one hand the position would be difficult to determine because the position in the source file doesn't necessarily correspond to the position in the rendered version of the web page. Depending on the user agent positions might even differ between different renderings. On the other hand, even if the position was known, there is no straightforward way to determine the weights that should be applied. Furthermore, weighting occurrences of barriers differently would be conflicting with Web Acc – Parameter 2 (all barriers contribute equally to the score). Thus the position information will not be taken into account. This is described in **Web Acc – Page 7**.

The requirements **Web Acc – Page 3** and **4** cannot both be fulfilled. The decision, which one to select is closely related to the construction of the score function and will be discussed in conjunction with it (see section 4).

It is a useful property that the score gets worse if a test fails more often. The requirement **Web Acc – Page 2** is undisputed. Note that **Web Acc – Page 2** is compatible with **Web Acc – Page 3** as well as with **Web Acc – Page 4**.

To be able to take into account the diversity of barriers (**Web Acc – Page 1**: The score gets worse if more (different types of) tests have failed.) we have investigated how to measure the diversity.

In demographical research the "Index of diversity" is used.[6] In our notation the formula is:

$$D = 1 - \sum_{t=1}^{T} \left(\frac{B_{pt}}{B_p}\right)^2$$

The range of values is [0;1] where 0 is homogeneous and values closer to 1 mean that the population is more heterogeneous. This number has the property $max\, D = 1 - \frac{1}{T}$ which occurs when all species are present in equal numbers.

Another field that has developed measure of diversity is ecology.[7] *Simpson's diversity index* is similar to the Index of diversity described above. It is 1 – D. *Shannon's diversity index*[8] is simply the ecologist's name for the communication entropy introduced by Claude Shannon.

$$H = -\sum_{t=1}^{T} \frac{B_{pt}}{B_p} \ln \frac{B_{pt}}{B_p}$$

This number has the property $max\, H = \ln T$ which occurs when all species are present in equal numbers.

In the context of web accessibility evaluation diversity measures are an interesting approach to score calculation that meets requirement **Web Acc – Page 1.**

We define the *barrier diversity* of a page as:

$$D_p = 1 - \sum_{t=1}^{T} \left(\frac{B_{pt}}{B_p}\right)^2$$

If only one type of test failed, we have B_{pt} = B_p and thus D_p = 0 (low barrier diversity). If more types of test fail the value increases, and converges to the limit $1 - \frac{1}{T}$ (high barrier diversity).

A better version of the index (from a distributional perspective) would be 1 - sqrt(1-D), since this changes the accumulated statistic from a variance to a standard deviation, and should also make the distribution more symmetric. I.e:

$$D_p' = 1 - \sqrt{\sum_{t=1}^{T} \left(\frac{B_{pt}}{B_p}\right)^2}$$

The diversity index will be smaller the more homogeneous the set of test results are, and larger the more heterogenous the set of test results are. This was intended to address the assumption that lead to the requirement **Web Acc – Page 1**; that it is worse with many different barrier indicators failing than many instances of one barrier indicator. The reasoning behind this assumption is that different types of tests may apply to people with different types of disability. So if all the problems are of one type, it will probably affect less users.. Where many types of problems are found, it is more likely that more users will be affected, so inclusion of a diversity component might contribute to increase the corellation between the score and the rate at which users perceive the page to have accessibility problems. See

[6] http://en.wikipedia.org/wiki/Index_of_diversity

[7] http://en.wikipedia.org/wiki/Diversity_indices

[8] http://en.wikipedia.org/wiki/Shannon%E2%80%93Wiener_index

section 2.3.9for further discussion on this issue. Note that the diversity index does not weight any particular disability differentially, it merely says that if the type of problem is restricted, then this is likely to affect fewer people, so the score should be lower. This is consistent with making statements about the average experience. Also, different types of tests are already treated differently, not by choice but because they are evaluated in different ways (see 4.2.1). This means (for example) that tests that can appear many times (N > 1), carry more weight, in determining the value of the barrier ratio, than those that can occur only once. If these different types of tests have more impact on different types of disability, then the barrier ratio is biased.

This means that the definition of the diversity index is neutral with respect to different disability groups. However, when we look at the effect on real sites, since the incidence rates of particular types of problem are not uniform and different types of problem do affect different groups differently, then the score values could be more or less appropriate for these different groups.

Web Acc – Page 1 may, however, under some circumstances be in conflict with requirements **Web Acc – Page** 3 and **Web Acc – Page** 4, because a smaller number of tests in a more homogenous set of test results may be smaller than larger number of tests in a more heterogenous test set. This may not be a serious objection, if the the diversity indicator can be expected to posess overall useful properties as an addition to, or in addition to the score value, as the discussion above indicates. Technically, the requirement**Maths – Output 3** cannot be fulfilled, because the diversity function is limited to $1-\frac{1}{T}$ in case of high diversity, which is less than 1. This also means that there is a small dependency to the number of tests, which means that **Statistics – Stability 4** cannot be completely fulfilled. However, in practice, the number of automatable UWEM tests is so large that these two problems probably can be ignored. The diversity index will therefore be discussed as a possible part of a score function in section 4.

2.3.6 Web site score

Web Acc – Site 1

The score is insensitive to the size of the site (i.e. the number of pages).

Web Acc – Site 2[9]

The contribution of a page to the overall site score can be weighted by parameters that depend on known (or knowable) characteristics of the page (such as its distance from the home page, frequency of access, main language, etc.).

(maybe incompatible with **Web Acc – Site 3** and **Maths – Input 4**)

Web Acc – Site 3

The score is robust to structural changes of the web site, i.e. moving the pages within the site doesn't change the result (for the site).

[9] Discarded in discussion.

(dependent on **Web Acc – Site 2**)

Web Acc – Site 4

The score is robust to structural changes of the web site, i.e. moving content among pages doesn't change the result (for the site), unless the changes introduce new accessibility problems. In particular, if only the locations of the single results change, but Bst and Nst are constant, the score of the site doesn't change.

(a) The score depends only on the number of failed test within the site (Bst), and not on how they are split among the pages (i.e. not on Bpt with p in s).

(b) The score depends only on the ratio of failed test within the site (R_{st}), and not on the ratio for the pages (i.e. not on R_{pt} with p in s).

2.3.7 Discussion of web site score requirements and desirable properties

Note that the score for a web site also depends on the properties of the sample. Therefore, some of the requirements in this category are analysed together with the sampling requirements.

If new pages are added to a site without improving the accessibility (i.e. the new pages have similar accessibility features as the old ones) the score of the site shouldn't change. Therefore, **Web Acc – Site 1** is needed.

If **Web Acc – Site 2** were selected, the crawler would need to provide the relevant information and **Maths – Input 6** has to be fulfilled.
The usefulness of **Web Acc – Site 2** can be disputed. While it is often the case that users visit the pages higher up in the site hierarchy first (and would therefore be confronted with accessibility problems on those pages more often), it is not clear that the pages higher up in the hierarchy really are more important to the user or that there is any correlation of the position in the hierarchy and the relevance at all.**Web Acc – Site 2** is dependant on **Web Acc – Site 3**. In general structural changes will not have any major effect on the percieved accessibility of a web site, provided that the changes to not cause new accessibility problems and the link structure is up-to-date. Therefore **Web Acc – Site 3** is selected, but **Web Acc – Site 2 (distance from home page)** is discarded. Also other versions of **Web Acc – Site 2** are discarded for the time being, since it unclear how the characteristics should be reflected in the weights. Note that score functions using a two stage approach will usually support **Web Acc – Site 2** (see section 4.2.2).

Moving content among the pages of a web site is a major change that can introduce new accessibility problems (e.g. the IDs, and titles are not longer unique because of duplicates in added content, or crossreferences are missing because part of a page has been removed.). In case no new accessibility problems are introduced, **Web Acc – Site 4** is a useful requirement that should be considered in the score function development.

2.3.8 Further parameters

Web Acc – Parameter 1[10]

Test results can be weighted according to severity of the associated barrier type (dependent or independent of disability group).

(incompatible with **Web Acc – Parameter 2, Maths – Input 5**)

(Discarded in discussion)

Web Acc – Parameter 2

Each barrier type contributes equally to the score.

(dependent on **Maths – Input 5**, incompatible with **Web Acc – Parameter 1**)

2.3.9 Discussion of parameters requirements and desirable properties

The objective of the indicators is to be as accurate as possible and to present useful and meaningful information. It would be highly desirable to have different indicators for different types of disabilities, as proposed in **Web Acc – Parameter 1**. However, if the parameters cannot be estimated with sufficient accuracy, problems will arise on several levels.

- High effort to estimate / define parameter values (see also **Maths – Input 5**).
- "Political issues". The credibility of the Observatory might be undermined because the diversity of disabilities can not be represented adequately by a small number of disability groups.[11]

Taking into account these risks **Web Acc – Parameter 2** should be selected. Note that formally **Web Acc – Parameter 2** can be viewed as special case of **Web Acc – Parameter 1** with all parameters equal.

2.3.10 Final selection of web accessibility related requirements and desirable properties

As outlined in the discussions above we have reached the following conclusion on the selection of web accessibility related requirements:

- **Web Acc – CSS 1** and **Web Acc – CSS 2**
- **Web Acc – Frame 1** and **Web Acc – Frame 2**
- **Web Acc – Page 2, 6, 7**
- **Wec ACC – Page 3 OR 4**
- **Web Acc – Site 1, 3,** and **4**
- **Web Acc – Parameter 2**

[10] Discarded in discussion.

[11] This was also the main reason for the dropping of user testing from UWEM 0.5.

2.4 Mathematical indicator requirements and desirable properties

The mathematical requirements are mainly related to the behaviour of the score function (i.e. the properties of the output values) and the way input values and parameters are handled.

2.4.1 Output of score function

Maths – Output 1

The score covers a continuous range of values.

Maths – Output 2

The score is bounded, i.e. there are minimum and maximum values.

Maths – Output 3

The score can get arbitrarily close to the extreme values (taking into account the behaviour of the function as well as the expected properties of the input data).

2.4.2 Discussion of Maths Output requirements and desirable properties

The use of continuous score values (**Maths – Output 1**) is a primary requirement of the Observatory. A coarse quantification e.g. on score card level, would be too course in the intermediary calculations of averages and other statistics. Furthermore, this property enables the creation of differentiated ranking lists. Apart from ranking the presented data, it should also support comparisons in general, for instance to compare different versions of the same site and thus monitor changes over time. A bounded score (**Maths – Output 2**) facilitates comparison over time because the extreme values of the maximum and minimum score establish a frame of reference for the results. Saturation of the range of values (**Maths – Output 3**) backs up the frame of reference. Additionally, this is a useful requirement when it comes to presentation of the data (see section 5).

2.4.3 Input of score function

Maths – Input 1

The score can accommodate results from tests that never have "pass" results due to their applicability criteria.

Maths – Input 2

The score calculation allows integration of UWEM tests that are applied on site level (e.g. tests for navigation consistency).

(dependent on **Sampling 2** and **StopCrit 2**)

Maths – Input 3

It is possible to calculate scores for individual UWEM tests. These scores are related to the overall score in a well-defined way.

Maths – Input 4

The score function can be applied to arbitrary sets of web pages (including sub domains of a web site, or language specific samples) as long as all pages belong to the same site.

Maths – Input 5

The score function should avoid unknown parameters, if it is not viable to estimate these parameters with the resources available (e.g. barrier severity parameters).

Maths – Input 6

The score function can take into account the site complexity information[12] provided by the sampling.

(Rejected in discussion)

Maths – Input 7

The score function is agnostic to the underlying document format.

2.4.4 Discussion of Maths Input requirements and desirable properties

The score calculation is based on the results of single UWEM tests (as described in section 1.3). Whereas most UWEM tests produce "pass" or "fail" results for unique locations within a web page, there are some tests that have a different behaviour. On the one hand there are some tests that produce only "fail" results (e.g. tests that are either not applicable or fail). This is reflected in requirement **Maths – Input 1**. There are several potential solutions to this problem:

- The score is calculated only from "fail" results.
- If a test is not applicable to a web page, it counts as a single "pass" result.
- Tests that can't produce "pass" results are added to the score separately (maybe with a weighting parameter).

The advantages and disadvantages of the solution will be discussed in the context of algorithm selection (see section 4).

[12] Site complexity information is information about the structure of the web site graph, e.g. link degree d-values (in random walk of EIAO release 2 beta), or length of path from home page.

On the other hand there are tests that are applied on site level[13] yielding only one result for the whole site, which cannot be attributed to a single page (e.g. navigation consistency). This is reflected in **MATHS – INPUT 2**. Note that **MATHS – INPUT 4** in the general reading of "arbitrary sets of web pages" is incompatible with **MATHS – INPUT 2** because site level tests usually cannot be applied to arbitrary sets of web pages. However the limited version (concerning only the calculation of a score for a sub domain and not for arbitrary sets of web pages) does not cause any conflict.

The presentation of scores for individual tests as required in **MATHS – INPUT 3**, is very important in the presentation of results to give a more detailed picture of the accessibility problems of a web site or page. Depending on how **MATHS – INPUT 1** and **MATHS – INPUT 2** are implemented, there might be some consequences for **MATHS – INPUT 3**. For tests that produce "pass" and "fail" results on page level this requirement unproblematic.

The requirement **MATHS – INPUT 5** has already been discussed in section 2.3.9 .

The requirement **MATHS – INPUT 6** is in conflict with requirement **WEB ACC – PARAMETER 2**, which means that it should be rejected. It has also been discussed in section 2.3.7.

The requirement **MATHS – INPUT 7** requires the score function to be agnostic to the underlying document format.

If a web site provides download files in other formats (e.g. pdf, Microsoft Word, or OpenOffice.org), it is possible to sample and evaluate these files also, given that relevant accessibility tests are available.

However, Until UWEM supports tests of these formats, these evaluation results should not be included in the UWEM accessibility score.[14]

We do think that use of other files on the web should influence the accessibility score of the web site. For instance, a web site could choose to refer to untagged .PDF or Word documents, instead of doing the effort of converting the document to HTML, however UWEM tests currently only detect barrier indicators in HTML/CSS.

In practice, assuming barrier density as measure, this means:

- Similar UWEM tests would have to be defined on e.g. the PDF tag structure as the tests UWEM perform for (X)HTML.
- Some checks may not be possible for other document formats, some checks (at least WCAG 1.0) are HTML specific. So the total number of tests would probably differ between (X)HTML and other formats. This means that if an average barrier density for a web site is to be used, then the barrier density measure would have to be calculated on a page level and averaged for a web site, alternatively some weighting scheme might to be used to counteract the difference in barrier density between e.g. HTML and PDF pages due to different number of tests. Take the extreme case, that only one test that checks for existence of PDF tags exists. In this case, the PDF page would be deemed as accessible if it was tagged, and it would be deemed as inaccessible if it was untagged. If aggregated on web page level, this strategy should

[13] In the current version 1.1 of UWEM none of the site level tests is marked as "fully automatable". There are activities within the WAB cluster (in the BenToWeb project) targeted at the implementation of testing modules for such tests (navigation consistency).

[14] In the future development of UWEM the possibility to include the PDF in the overall score of a web site will also be considered. However, as long as UWEM is based on WCAG 1.0, which does not include accessibility tests for PDF, separate scores are more suitable and supportive of result transparency.

work well. If not, each PDF test would have to be weighed according to the number of HTML tests.

- It is still an open question how the indicators for other document formats and HTML/CSS can be merged in a sensible way, if the test coverage is not similar or if tests behave differently in the different documents. So while it is technically possible to merge the results, it is important to also ensure that the end result will be meaningful.

We propose to stick to the strategy agreed in the review meeting, to include PDF as a part of the content statistics to collect for a web site. This way we can collect data about PDF documents and possibly other formats, and later, based on this information, elaborate a score to combine different document formats. **Maths – Input 7** requires that the score aggregation function will support such a combined score.

Note that external objects embedded into the page (like Flash or other technologies requiring plug-ins) are also interpreted as part of the web page from which they are referenced. UWEM 1.0 does not specify any fully automatable tests for external objects, but if such tests become available in the future, they will be handled as any other test results for the page.

2.4.5 Final selection of mathematical requirements and desirable properties

As outlined in the discussions above we have reached the following conclusion on the selection of mathematical requirements:

- **Maths – Output 1, 2,** and **3**
- **Maths – Input 1, 2, 3, 4, 5, 6** and **8**

2.5 Statistical requirements and desirable properties

This section investigates requirements related to the statistical properties of the web site score. Stability and robustness are very important for the presentation of the data in the Observatory.

Additionally, the section addresses the need to investigate the relationship between the indicators determined by automated evaluation and web accessibility as perceived by users with disabilities and expert testers. This will support a meaningful interpretation of the Indicators.

2.5.1 Stability and robustness

Statistics – Stability 1

Scores on site level are stable and repeatable. Evaluating the same site twice produces similar results within a defined tolerance.

(dependent on **Sampling 3**, and **Sampling 4**)

Statistics – Stability 2

Enlarging the sample from a site (such that the sampling requirements are still met) does not significantly change the score of the site.

Statistics – Stability 3

Scores are comparable over time, i.e. measuring improvement or changes over time is possible.

(dependent on **Sampling 3, 4, 7**, and **9**)

Statistics – Stability 4

Scores are stable (i.e. still comparable) if a new test is added to the test set.

(c) The model takes into account the number of tests.

(d) If the score function is updated[15] to accommodate new tests, it still fulfils the same requirements as before the update.

(e) Ranking remains if a test is added that has the same percentage of failure as the average of the previously existing tests for a specific Website. If two sites have the same relative failure rate with regard to the newly introduced tests, then their relative ranking under the new score function is the same as under the old one.

(dependent on **Sampling 9**)

Statistics – Stability 5

Scores are robust, i.e. the score of a site is not unduly affected by outliers.

2.5.2 Discussion of stability requirements and desirable properties

The requirements in this section describe stability needs in different dimensions. On the time dimension the repeatability and comparability of results over time is postulated (**Statistics – Stability 1** and **3**). These two requirements are related to the sampling strategy and should be verified (in theory or in an experiment) when the sampling algorithm is specified and implemented respectively.

The stability of the score values with regard to changes in the sample (**Statistics – Stability 1, 2,** and **5**) is a property of the score calculation and should be verified (in theory or in an experiment[16]) once the score function is defined and implemented respectively.

Finally, **Statistics – Stability 4** is a requirement concerns the stability regarding changes in the test set, and should be taken into account during the design of the score function.

[15] The update of the test set and score function affects only new evaluations. None of the old results is recalculated or changed in other ways.

[16] The difference between **Statistics – Stability 1** and **Statistics – Stability 5** is that **Statistics – Stability 5** can be verified based on artificial data (where outlier results have been added on purpose), whereas **Statistics – Stability 1** needs to be verified in the crawl of real web sites.

2.5.3 Interpretation of score

STATISTICS – INTERPRETATION 1

> The results are correlated[17] to the accessibility of the web site as perceived by a (disabled) user.

STATISTICS – INTERPRETATION 2

> The results are correlated to the accessibility of the web site as assessed by an expert.

STATISTICS – INTERPRETATION 3

> The results for automatic evaluation are correlated to the full UWEM score for the web site as computed by an expert.

2.5.4 Discussion of interpretation requirements and desirable properties

One of the most important prerequisites of UWEM-based automated web accessibility monitoring is that the score can be used as an indicator of the web site's accessibility. Requirements **STATISTICS – INTERPRETATION 1** and **STATISTICS – INTERPRETATION 2** describe two different ways to conceive the link between the score and the accessibility of a web site. The former tries to relate the accessibility score to user experience. The latter postulates a correlation of automated score and expert assessment.

In the decision between the two the following questions should be taken into account:

(1) How can the fulfilment of the requirement be measured?

(2) How does it relate to the underlying methodology (UWEM)?

(3) Is the requirement compatible with the other requirements discussed in the paper?

Both approaches need a large amount of data (i.e. results from several hundred web sites) for a statistically sound correlation analysis. Interpretation 1 may require a larger set of data than interpretation 2 since the perceived accessibility will probably have more variation among users than among experts. We expect it to be easier to acquire expert evaluations of the web sites because several partners in the WAB cluster perform expert evaluations on a regular basis. Existing user evaluations of web sites will be more difficult to come by. Furthermore they are often less structured and therefore not so suitable for comparisons. The only resort would be to conduct a survey with a sufficiently large number of participants. However, this is beyond the possibilities of EIAO.

In general adherence to guidelines (e.g. WCAG 1.0) is a target that can be measured with higher accuracy than personal accessibility experience. Regarding question (1) **STATISTICS – INTERPRETATION 2** is the better choice.

[17] We use "correlated" to indicate the existence of an underlying monotonic functional relationship, not necessarily linear. The presence of such correlation makes it possible to use the score results as predictors of the other values, together with an estimate of the precision of such a prediction.

With regard to question (2) **Statistics – Interpretation 3** has advantages over **Statistics – Interpretation 1** because the UWEM methodology contains instructions for expert testing, but user testing is not covered. Additionally, **Statistics – Interpretation 3** has advantages over **Statistics – Interpretation 2** because it requires the application of UWEM explicitly.

The stability requirements (see section 2.5.1) are very important for the Observatory. The investigation of correlation with user or expert evaluation makes sense only if those evaluations also meet some stability requirements. We expect the expert testing to fulfil these requirements better. Thus for question (3) **Statistics – Interpretation 2** or **3** are the better choice as well.

Statistics – Interpretation 3 is a special case of **Statistics – Interpretation 2**. It describes one way in which the correlation of expert evaluation and automated testing can be performed. The first part of **Statistics – Interpretation 3** states that the expert evaluation should be based on UWEM as well. This is seen as a very useful property. The second part requires the calculation of a "full UWEM score" from the expert results. If the selected score function fulfils the necessary requirements (mainly **Statistics – Stability 4**), a comparison and correlation analysis is feasible.

The number of automatable tests in UWEM is relatively small. The automatic results should always be the same as the expert result for a fully automatable test. There must then be a correlation between the automatic results and the full expert evaluation results since the automatic tests form a subset of the complete expert test set. An interesting question is whether there is any correlation between the automatable and ther non-automatable results. It will not be possible to preserve the order of the scores between expert and automatic measurements since that would require an exact monotonic relationship, which is clearly impossible. Note that the existence of a correlation does not require that the ranking is always exactly the same, merely that there is a general trend in the ranks. So **Statistics – Interpretation 2** and **3** might not be impossible to fulfil.

2.5.5 Final selection of statistical requirements and desirable properties

As outlined in the discussions above we have reached the following conclusion on the selection of statistical requirements:

- **Statistics – Stability 1, 2, 3,** and **5**
  There are no incompatibilities among the stability requirements. Furthermore, there are no conflicts with the selected sampling requirements (see section 2.1).
- The dependencies between **Statistics – Stability 4** and the **Web Acc – Page** and **Web Acc – Site** requirements have to be investigated.

3 Sampling algorithm

This chapter will first propose a set of possible new sampling algorithms, including the old EIAO Random Walk algorithm for comparison and will then perform a feasibility study, which concludes with the recommended sampling algorithm. The algorithms are divided into two groups. The first group (3.1) investigates algorithms based on random exploration of the web site, and the second group (3.2) investigates various possibilities based on exhaustive scanning of the web site. Finally, the conclusion section concludes which algorithm should be selected.

3.1 Preconditions and definitions

a) It is presumed that the URL repository is grouped into a two-level structure, consisting of web sites and URLs within each web site.

b) A *new* URL is a URL that points to a web page that has never been downloaded before.

c) Random URL selection is limited to the currently selected web site being crawled by a crawler instance.

d) For *all* the algorithms, significant efficiency gain can be achieved by storing the timestamps for when the web page was downloaded, and only querying the web server for web pages that have a newer modification time than the stored timestamp (e.g. by using http-modified-since). Existing evaluation results can be reused if the web page is not changed.

3.2 Algorithms based on random exploration of the web site

This section describes various algorithms that are designed around the idea of random exploration of the web site, and uniform selection from the set of URLs explored. It starts with the approach suggested by Andrew Westlake (3.1.1), which is a one-phase approach for random exploration and uniform evaluation of web sites. The second approach (3.1.2), is a two phase solution, that adds a URL extension phase, to provide faster exploration of the set of URLs in the web site than the one-phase solution. The third and fourth approaches (3.1.3 and 3.1.4) elaborate on the second approach, in an attempt to improve the algorithms ability to detect new web pages. The third approach uses a simple flag to indicate change, and the fourth is a more elaborated solution based on Learning Automata. Finally, a discussion section concludes with the most viable alternative.

3.2.1 The Andrew Westlake algorithm

The following is taken from a draft of “UWEM/EIAO Review Appendix - Statistical issues” by Andrew Westlake:

“The following algorithm is proposed. (though it may need to be preceded by a preliminary scan to detect changes.)

1. Initialise the calculation of metrics for the web site
2. *Repeat:* Randomly select a URL from the repository for the site, omitting any already visited at this scan (sampling without replacement)
3. Identify **all** linked URLs from the selected page, (if it has not already been scanned at this visit), and add any new ones to the repository.
4. Evaluate appropriate tests and other measures for the page. Note that pages that have not changed, do not need to be re-evaluated.
5. Update the site metrics with information from the current page
6. Until: Check for Stopping rule is met (principally whether adequate precision has been achieved for the site metrics)
7. Complete computation of site metrics and store

This algorithm does not explore chains, and so does not suffer from dependency between the evaluated pages, except at the early stages of the first visit to a site. At early visits (while the repository is incomplete) it will probably visit more pages close to the home page than the current algorithm. Similarly, in the absence of cross-links, broad branches of a sub-tree are likely to be explored before deep ones. It is guaranteed to eventually reach every accessible page within the site, though it will probably take longer to reach deeper parts of the site. Once all pages within the site have been identified (so all are in the repository) it produces a strictly independent random sample of the site pages. The algorithm does not know any characteristics of a page before it is visited, so must use equal probability in the selection of pages. If it is desired to associate some form of importance with the results from a page, this can be done through the use of weights in the statistical calculations of metrics.

The algorithm is an invention, so its properties will need to be explored. An issue to be considered (amongst others) is whether the algorithm will perform as wished at a repeat visit to the site, when the site structure has changed.“

Advantages:

- Simple, and works.
- Nice on the crawlers and web servers, does not use an excessive amount of bandwidth.

Disadvantages:

- The selection is biased in the beginning when only web pages close to the home page are known. This can be mitigated by requiring a minimal number of URLs before any selection occurs.
- It is slow to explore the site. One can not guarantee that a site is ever fully explored unless it is finite and unchanging. Pre-scanning of the site could be used to expand the set of known URLs.

- This algorithm will give a bias for new web pages (lag behind), if a substantial flow of new web pages are added to the web site. If all known URLs have been checked, then these new web pages will be randomly detected during the first phase, which may take too long time.

3.2.2 Random selection/URL extension phase solution

1. While sampling stop criteria are not met:
2. *Random selection phase:*
   2.1 Choose a random URL uniformly from the set of currently known URLs. (sampling without replacement.) Download the web page, extract all URLs, and set the *new* flag to false for the downloaded URL when it is stored in the set of known URLs.

   2.2 If the chosen URL cannot be downloaded (e.g. HTTP 404), remove URL from the set of known URLs, and go to 1.

   2.3 The *new set of URLs* for the selected web page, is the set of URLs within the same site and are unregistered in the set of known URLs. Store all URLs in the *new set of URLs* for the selected web page in the set of known URLs for the currently selected web site, with *new* flag set to true.

   2.4 URLs that point to other web sites may cause new web site structures to be created, if necessary, and previously unregistered URLs should be stored as seed URLs for the respective new or existing web sites with the *new* flag set to true.

   2.5 If the number of URLs in URL repository for the current web site is larger than a minimum number N, or all URLs in the set of known URLs have *new* flag = false (already downloaded[18]) then:

   Send web page to accessibility evaluation.
3. *URL extension phase:*

   3.1 If the *new set of URLs* is empty, go to 1

   3.2
   - Select a random URL uniformly from the *new set of URLs* within the same web site available from the selected web page.
   - Download the web page available from the selected new URL. Extract all URLs available from this web page.

   3.3 If the chosen URL cannot be downloaded (e.g. HTTP 404), remove URL from the set of known URLs, and go to step 1.

   3.4 Store all URLs in the *new set of URLs* for the selected web page in the URL repository of the currently selected web site, with *new* flag set to true.

   3.5 URLs that point to other web sites may cause new web site structures to be

[18] The latter criteria is to allow evaluation of web sites with less than N pages.

created, if necessary, and previously unregistered URLs should be stored as seed URLs for the respective new or existing web sites with the *new* flag set to true.

Go to step 1.

The purpose of this two-phase approach, is to force the URL repository to grow quickly, to eventually map all visible URLs within the web site. This should make it possible to make a stronger claim for a representative random selection of URLs, assuming that no URL weighting is involved. Note that the URL repository will continue to grow although a bit slower when the minimum number of URLs N is reached, since the pages randomly selected for evaluation will be scanned for links.

Note also that the selection of web pages for evaluation is only performed in the random selection phase and not in the URL extension phase. This avoids bias in the calculated score due to dependencies between the two linked web pages, in contrast to subsequently crawling pages that are linked together. Newly identified URLs will in the next iteration of the algorithm compete on equal terms with all other known URLs in the URL repository for being selected. If newly identified URLs are selected, then the cached version of the web page will be sent to evaluation, so the web page does not need to be reloaded.

This algorithm will perform a uniform random selection of known web pages from the URL repository of the currently selected web site. The algorithm will further extend knowledge about new URLs by choosing new web pages that the system has no prior information about by selecting a random web page uniformly among all URLs available from the chosen web page. In addition, deep links from other web sites linking to our web site will be added, which will provide several seeds for random exploration of the web site in addition to the initial seed resources (usually the home URL.) This would mean that the algorithm might possess near-uniform random selection properties until all URLs have been found, and it will have uniform property if all URLs within the web site have been found, however it is in reality not possible to verify that all URLs have been downloaded, due to the possibility of disjoint parts of the web that are not interlinked which the observatory currently does not support.

The indicated solution is simplified compared to the current random walk algorithm, since it does not attempt to balance the page selection properties. or attempt to weigh inversely proportional to the pagerank, as the patented Near Uniform Random Walk algorithm by Heinzinger et al does[HENZINGER00].

The URL extension phase of this algorithm could be considered a random walk of length 2, that only explores unknown parts of the web during the walk phase.

Advantages:

- Bias is avoided by having an initial URL extension phase. Evaluations is only started after the URL repository has been filled with N URLs.
- If the web site has less than N URLs, then the web site is crawled exhaustively, and the algorithm can perform an exact calculation of the accessibility indicator.
- It explores the site relatively quickly, and will eventually guarantee that a static site will been crawled exhaustively.
- Nice on the crawlers and web servers, does not use an excessive amount of bandwidth.

Disadvantages:

- This algorithm will give a bias for new web pages (lag behind), if a substantial flow of new web pages are added to the web site. If all known URLs have been checked, then these new web pages will be randomly detected during the first phase, which may take too long time.
- There may be biases as long as the web site has not been fully explored. However, if N is set to infinity (or in practice a very large number), then this should be possible to avoid, since this effectively would be an exhaustive crawl.

This algorithm may be improved by introducing weighted random URL selection in the URL extension phase, weighted according to update probability. Since the request distribution to a web site normally is Zipf-like (power law) [ZIPF] [19] , it is a reasonable assumption that new URLs will mostly be available from a certain set of web pages. (E.g. the index page.) [knap]. URL extension can then be focused on those web pages that have a high probability of changing and thus have a high probability of new URLs being available at each visit. The next algorithm is a variant of 3.2.2, that explores a different way to identify the web pages that change, and perform more frequent checking of these web pages for new URLs, to improve the freshness of the set of identified URLs.

3.2.3 Random selection/URL extension phase solution with learning automata based approach for detecting web page updates.

This algorithm is based on 3.2.2 and it introduces a strategy for tracking web page changes via learning automata (LA), so that the parts of the web that are updated are checked more frequently than parts of the web that do not change.

In this approach the algorithm and preconditions are the same as in 3.2.2 with exception of the URL extension phase and the additional shown below. Because of this we will only discuss the part of the algorithm related to the URL extension phase.

Additional preconditions

a) Each site is crawled at regular intervals. I.e. the site to crawl is not chosen randomly from the complete set of sites, but traversed as a list.

b) Each known URL must also include an integer number for the state of the Learning Automata (LA).

Environment feedback

In this alternative we propose to connect a Learning Automata (LA) to each known URL with the intention to learn the update probability of the web page the URL points to[20].

What this achieves is the elimination of the request to get the last-modified date of the page. The last-modified date is not always reliable, using if-modified-since may cause bias of up to 17% according to the findings in [effe].

[19] A zipf distribution follows a linear curve if plotted with logarithmic scale for X and Y axis.

[20] For details, including proof of convergence of this LA we refer the reader to [search]. For a similar approach using LAs to detect web page changes, including empirical results, we refer the reader to [knap].

The probability of an update is often in literature considered as an unknown stochastic function of which it is only possible to observe the effects. I.e. we will never know a priori according to which scheme a page within a site is updated, we can only identify whether a downloaded page has been updated or not. Based on this feedback, the algorithm intends to learn the probability of a page changing.

If a page i is updated or not is modelled according to $f_i(x_i)=\{1,0\}$ where 1 is when a page is updated, while 0 is whenever a page is visited but not updated since last crawl. Thus, in order to detect as many updates as possible, the intention is to maximise $\sum_1^n f_i(x_i)$, where n is the number of known web pages.

It is assumed that the evaluations will be performed at regular, and not too short intervals, e.g. monthly. The proposed scheme therefore needs to work with very little feedback from the environment. This means that the each LA needs to converge and find a viable visit probability, based on very little feedback from the environment. Rapid convergence is achieved by having few states in the LA, which means that the automata also will be more prone to “forget” web pages, if they change too seldom. It is suggested to have a LA-scheme with only four states. Appendix C shows results from updates with LAs of different number of states.

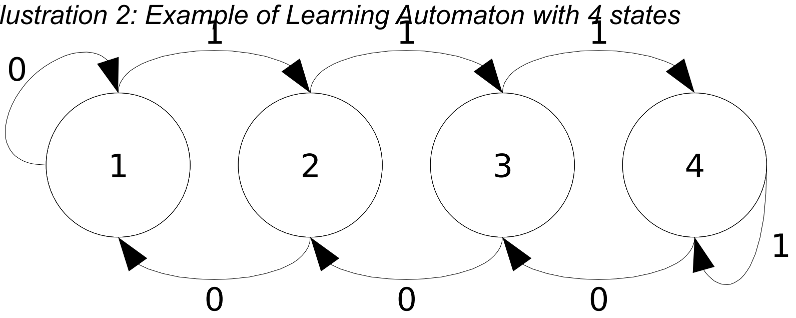
Learning Automata Update Scheme

Whenever a page is downloaded , the automaton state is updated based on the feedback of $f_i(x_i)$. $f_i(x_i)=1$ makes the automaton move one state closer to the top state, while $f_i(x_i)=0$ makes the automaton move one state closer to the lowest state.

Example of such an automaton with 4 states can be seen in Illustration 2. Literature has shown such an automaton converges towards the optimal manner state whenever the feedback from the environment is correct more than 50% of the time [search].

We further propose, to assign visit probabilities based on which state the LA is in. State 4 has the highest visit probability, while state 1 has the lowest probability. This is also done according to the findings in [knap]. For each state there exists a visiting number $\frac{s^2}{N^2}$ where s is the current state and N is the number of states available. E.g. a LA in state 4 would have a visiting number $\frac{4^2}{4^2}=1$ while a LA in state 1 would have visiting number of $\frac{1^2}{4^2}=0.0625$. All new URLs will start in state 4, thus having a probability of 1 whenever the maximum number of visits has not yet been exceeded.

Illustration 2: Example of Learning Automaton with 4 states



1. When maximum number of visits M is not exceeded;
2. Sequentially accumulate the LAs connected to pages within the current site that has not yet been visited in the current crawl according to $\frac{S_i^2}{N^2}$. See for an example of such an accumulated distribution.
3. Choose a random number between 0 and $\sum_1^n \frac{s_i^2}{N^2}$ where n is the number of pages that has not yet been visited.[21]
4. Visit and download the page of the LA which the accumulated visiting number corresponds to the chosen random number.
5. If the chosen page cannot be downloaded (e.g. HTTP 404), remove URL from the URL repository, and go to step 1.
6. Update the selected LA state based on environment feedback ($f_i(x_i)$).
7. If the page is changed since last evaluation, extract all new links and add these to the list of known URLs / URL repository. The automaton connected to the new URL will automatically start in the highest possible state. This ensures that the newly discovered URLs are prioritised. Note that a page that has been visited will not be visited again in the current crawl.
8. Go to step 1

The visit probability of selecting a LA o , and thus visiting the corresponding page, would then $\frac{\frac{s_o^2}{N^2}}{\sum_1^n \frac{s_i^2}{N^2}}$ for each iteration.

[21] Note that this number will be continuously updated after iteration as the the LAs / pages that have already been visited is not considered to be part of this number while newly discovered pages will also be part of this.

Advantages in addition to the advantages with 3.2.2:

- The algorithms should track web page updates reasonably well, which means that the bias for new pages (lag behind) should be less.

Disadvantages:

- It may take long time for the algorithm to converge, since only the sampled sub-set of automata will be updated.
- If update is resumed on a web page that is not among the “popular” ones that are being prioritised by the automata, then this change may go undetected for a very long time. The algorithms can therefore not give any guarantees for the freshness of the set of URLs, even though it is better than 3.2.2 and 3.2.1 when it comes to freshness.

3.3 Exhaustive search alternatives

Exhaustive link-search crawl in combination with random uniform selection of pages for evaluation. Two different approaches are discussed. First an exhaustive search with evaluation of the entire web site, and then a solution with exhaustive URL search and random uniform selection for evaluation of a subset of the web pages identified in the URL search.

3.3.1 Exhaustive search with evaluation of the entire site:

1. Exhaustive scan of the full web site.
2. Add all detected pages from the site to the URL repository.
3. Select all web pages from the site from the URL repository.
4. Send selected web pages to accessibility evaluation.

Store the result.

Advantages:

- Simple, and works.
- Gives an exact, complete and up-to-date picture of conformance to the automatable parts of UWEM.
- Not biased
- It is possible to conserve a significant amount of I/O after the initial mapping of all URLs by only downloading web pages that are marked as changes using the if-modified-since HTTP header element.

Disadvantages:

- Very processor intensive
- Bandwidth and I/O intensive during initial URL mapping.

3.3.2 Exhaustive URL search with evaluation of a sampled set of web pages:

Version with evaluation of a number of samples:

1. Exhaustive scan of the full web site.
2. Add all detected pages from the site to the URL repository.

3. Randomly select a web page from the URL repository for the site. (sampling without replacement)
4. Send selected web pages to accessibility evaluation.
5. Check stop criterion, if not reached, go to step 3.

Advantages:

- Simple, and works.
- Gives a complete and up-to-date picture of web accessibility within a given tolerance.
- Not biased
- It is possible to conserve a significant amount of I/O after the initial mapping of all URLs by only downloading web pages that are marked as changes using the if-modified-since HTTP header element.

Disadvantages:

- Bandwidth and I/O intensive during initial URL mapping.

3.4 Conclusion

The random exploration algorithms have the advantage that they initially are bandwidth-friendly, since they do not perform exhaustive crawls of the web. However, this comes at the cost of having a less fresh picture of updates to the web, which is in conflict with requirement **Sampling 5**. The other problem with these algorithms, is that they potentially will have a bias towards the seed URLs (e.g. start page) initially. It is important that the UWEM score presents an updated picture of the web. In addition, the exhaustive evaluation algorithms is initially very bandwidth- and CPU-intensive, but subsequent evaluations can benefit from using timestamps and query the web server only for those web pages that have changed, which means that after the initial full mapping, using one of the exhaustive scan algorithms is the best and least resource intensive way to maintain an updated picture of the web. It is therefore recommended to not use any of the random exploration algorithms.

The exhaustive search algorithm in section 3.3.1 is the most resource intensive algorithm. Evaluation of automatable UWEM tests is the most CPU intensive part of this algorithm, and will be the limiting factor for large scale evaluation of web sites, since some web sites can be very large. For instance, the web site regjeringen.no has 440.000 pages according to Google[22], which means that exhaustive evaluation of web sites in this size category will be very resource intensive. The alternative algorithm 3.3.2 would provide results that essentially are the same with evaluation of a much smaller random uniform sample than the complete web site. Experiments have also shown that initial exhaustive pre-scanning for URL extraction seems viable. One possible objection against 3.3.2 may be that site-level UWEM indicators, e.g. navigation consistency, would need to consider all web pages, to get a reliable indicator. However it may equally well be that it is possible to estimate a site based indicator from a random, uniform sample from the web site, so no conclusion can be drawn on this. It is therefore recommended to use 3.3.2 “Exhaustive URL search with evaluation of a sampled set of web pages” as the UWEM sampling algorithm.

[22] http://www.google.com/search?q=site%3A.regjeringen.no&start=0&ie=utf-8&oe=utf-8&client=firefox&rls=org.mozilla:en-US:unofficial

3.5 Stop criteria

Several different variants of stop criteria for the sampling algorithms are conceivable:

- Sample until all pages are visited.
- Fixed number of samples.
- Sample to a certain length of path from home resource.
- Sample a certain percentage of all web pages in the site.
- Sequential sampling until the web site score is within a given error margin for a given confidence interval, and with a given confidence interval coverage.
- 99% confidence that each site is placed in the correct score-card group.
- Maintain ordering of sites; I.e. require that the sign of the difference between two sites which have a real difference is unlikely to change between the previous and the current measurements purely by chance when the sites have not been changed.

(1) Sample until all pages are visited, implies an exhaustive search and evaluation of all pages, which would be too expensive in terms of evaluation effort, so it violates prerequisite 8.

(2) Acquiring a fixed number of samples is useful as a minimum requirement, in addition to other criteria like (5) or (6).

Sampling to a certain length of path from the home resource (3) would work fine with a BFS sampling strategy, but is not compatible with the chosen near random uniform sampling strategy, since the requirements **Sampling 1, 3, 4, 5, 6, 8, 11, 12** are violated, so (3) is not considered a viable alternative.

(4) Sampling a certain percentage of all web pages in the site, is possible with a near uniform random sampling algorithm, however it is considered a better alternative to use solution (5) or (6) for sampling of single web sites, since these solutions fulfil all the stop criteria requirements, whereas sampling a certain percentage would not support the requirements **StopCrit 3, 4, 7.**

(5) Sequential sampling until the average result is within a given error margin for a given confidence interval, and with a given confidence interval coverage fulfills the requirements well for single web sites, however we will need to elaborate on how to handle **StopCrit 4** in the algorithm discussion. Approach (5) is therefore viable.

(6) '99% confidence that each site is placed in the correct scorecard group could be implemented as an error margin less than the length to the closest adjacent scorecard boundary within a 99% CI. This would mean that the lowest number of samples would be needed in the middle of a score, and the number of samples would increase as the sample average approached the scorecard boundary, and would eventually end up with an exhaustive search when the score was near a scorecard boundary. A nice property of this, is that one could assure that a score would be placed in the correct scorecard group. However significantly more samples would be needed close to a scorecard boundary, since the error margin is proportional to $1/\sqrt{n(s)}$ where $n(s)$ is the sample size. It would also be possible to combine this approach with approach (5), to achieve a smaller maximum limit than the error margin at the centre of the score. A concern with this approach, is that the precision comes at a cost – it is computationally expensive to evaluate all web pages of web

sites that are close to scorecard boundaries, which, depending on the size of the web site, may be in conflict with prerequisite 8.

(7) 'Maintain ordering of sites': Any measure evaluated over a sample has a confidence interval or error margin associated with it. This is true for differences (between sites, or between visits to a site over time). So we can have a criterion that says that we have 99% (say) confidence that a given (absolute or relative) positive difference between two scores represents a real difference that is greater than zero. This confidence then applies to the ranking as evaluated.

The most precise stop criterion, both for presenting results as scorecards, and for presenting exact *F(s)* score values, would be a combination of 2,5, 6 and 7; i.e. a minimum requirement on the number of samples N, then having a maximum requirement on the error margin, 99% confidence that the error margin was within a specific scorecard group. and 99% confidence that a difference between two score values represents a real difference. However, 6 may be in conflict with prerequisite 8, that the functions and algorithms can be implemented efficiently for large-scale evaluation, since web sites in principle can contain infinitely many web pages, so this approach will not always converge. 7 would need further investigations, to identify an implementable strategy, since it presumes that any information between two sites always is available. This would be problematic for historic values, if more samples were needed on historic values to have 99% confidence of a real change. It may also, depending on implementation, be problematic for comparing differences between web sites, because it means that any web site may be required to acquire more samples to have a positive confirmation about a potential difference. It is therefore suggested to not include 7 in the stop criterion.

It is therefore recommended to go for a combination of (1) and (5), i.e. a minimum requirement on number of samples, e.g. N=50 and sampling until the result is within a given error margin, e.g. 0.02 for 99% confidence interval.

4 Score function design

The table below presents the score functions that are discussed in this section. They can be grouped in the two categories "one step calculation" and "two step calculation". Furthermore, a rough classification of how simple or advanced they are, is given.

	One step ***(test results – site)***	***Two step*** ***(test results – page – site)***
simple	barrier ratio per site	barrier ratio per page UWEM 1.0 WABscore (Zeng)
advanced (complexity)		EIAO internal
advanced (diversity)	barrier ratio and diversity per site	barrier ratio and diversity per page

Table 2: Overview of potential score functions discussed in this section

Below we define the potential score function in detailed formulae. The mathematical notation is explained in Table 1 in the introduction section.

Barrier ratio per site

$$F_1(s)=\frac{B_s}{N_s}$$

Barrier ratio per page (described in Sullivan & Matson 2000)[23]

$$f_2(p)=\frac{B_p}{N_p} \qquad F_2(s)=\text{average of page scores}$$

UWEM 1.0 (described in UWEM 1.0)

$$f_3(p)=1-(1-s_t)^{A_p} \qquad F_3(s)=\text{average of page scores}$$

WABscore (Zeng) (described in ZENG 2004)[24]

$$f_4(p)=\sum_{t=1}^{T} s_t\frac{B_{pt}}{N_{pt}} \qquad F_4(s)=\text{average of page scores}$$

EIAO internal (described in Bühler et al. 2006)[25]

[23] **Terry Sullivan and Rebecca Matson.** *Barriers to use: Usability and content accessibility on the web's most popular sites.* In Proceedings of ACM Conference on Universal Usability – CUU, 2000.

[24] **Zeng, X.**: *Evaluation and Enhancement of Web Content Accessibility for Persons with Disabilities*. PhD thesis, University of Pittsburgh (2004)

[25] **Christian Bühler, Helmut Heck, Olaf Perlick, Annika Nietzio, and Nils Ulltveit-Moe.** *Interpreting results from large scale automatic evaluation of web accessibility*. In Proceedings of ICCHP 2006, 2006.

$$f_5(p)=1-\prod_{t=1}^{T}(1-s_t)^{C_p} \text{ where } C_p=\frac{B_{pt}}{N_{pt}}+\frac{B_{pt}}{B_p} \qquad F_5(s)=\text{average of page scores}$$

Barrier ratio and diversity per site (introduced in this paper)

$$F_6(s)=w_R \cdot R_s+w_D \cdot D_s, \text{ with } w_R+w_D=1$$

Barrier ratio and diversity per page (introduced in this paper)

$$f_7(p)=w_R \cdot R_p+w_D \cdot D_p, \text{ with } w_R+w_D=1 \qquad F_7(s)=\text{average of page scores}$$

It remains to be investigated how the two parts Rp and Dp should be combined. The simplest way to combine the two measures is to calculate the mean, using wR=wD=0.5. However other weightings are also possible, e.g. giving more weight to the barrier ratio.

This question is a parameter calibration problem that needs to be solved using real data. A first experiment giving an overview of the behaviour of the functions and the resulting distributions can be found in .

Another way to combine barrier ratio and barrier diversity is to calculate the product:

$$f_8(p)=R_p \cdot D_p, \qquad F_8(s)=\text{average of page scores}$$

Note that the Ratio-Diversity score functions can be defined for the one step approach as well was for the two step approach.

4.1 Easy to verify requirements and desirable properties

4.1.1 Mathematical requirements and desirable properties

All of the functions discussed in this section meet requirement **Maths – Output 1**.

The WABscore function proposed by Zeng does not meet requirement **Maths – Output 2** because it doesn't have an upper boundary. It just adds up the ratios for all test without any normalisation. This will also lead to problems with requirement **Statistics – Stability 4** since adding new tests will always increase the score. Furthermore, the WABscore function uses severity weights for the test. As discussed in section 2.3.9, this feature is considered problematic. For these reasons the WABscore function is rejected and will not be considered in the discussions in the remainder of this section.

Apart from the WABscore all other suggested functions meet requirement **Maths – Output 2.**

The two functions UWEM 1.0 and EIAO internal do not meet **Maths – Output 3**. The maximum value is[26]

$$1-(1-s_t)^T<1$$

[26] This problem could be mitigated by normalising the output of the function of [0;1].

4.1.2 Web Accessibility related requirements and desirable properties

Requirement **Web Acc – Page 2** ("Score gets worse if a test fails more often") is not fulfilled by the UWEM 1.0 score function because this function uses only the value Ap (number of failed barrier types) and doesn't include any information on the number of failed instance. This is a major shortcoming of the UWEM 1.0 score function. For this reason the UWEM 1.0 function is rejected and will not be considered in the discussions in the remainder of this section.

Requirement **Web Acc – Page 1** and the combined Diversity calculation has been rejected in section 2.3.5. A separate Diversity calculation may optionally be presented.

4.2 Design decisions

This section describes the decisions made during the development of the score functions. We start from the general questions affecting all functions and subsequently cover the more differentiated parts.

4.2.1 General questions

Question 1: How are the input values Bpt and Npt composed? How are the results from the different resources that constitute a web page p put together?

- for tests that have pass and fail results for several instances within page p (including tests for CSS rules that are applied in several different locations)
  Bpt = number of fail results for t within p
  Npt = sum of fail and pass results for t within p
- for tests that have exactly one pass or fail result per resource file[27] (including CSS tests that are applied on page level)
  Bpt = 0 or 1
  Npt = 1
- for tests that have only fail results because of their applicability criteria
  Bpt = Npt = number of fail results for t within p
- for heuristic tests yielding a probability value per location
  Rpt = average of probability values per location
  Npt = number of locations in which t was applied
  Bpt = Rpt * Npt
- for heuristic tests yielding a probability value per page
  Bpt = Rpt = probability value
  Npt = 1
- for tests that have only one result per site / sample
  Bst = 0 or 1 (or probability value)

Question 2: Which properties of the web page, web site, sample are not taken into account for the calculation?

- The following properties are not taken into account in the calculations:
  - size of the page (i.e. file size)
  - number of pages in site
  - position where test was applied

[27] This includes for example checking the validity of an html or css file.

All the score functions mentioned in this section handle their input as outlined in the answers to the questions. This leads to the conclusion that the following requirements are always met:

- **Web Acc – CSS 1** and **2**
- **Web Acc – Frame 1** and **2**
- **Web Acc – Page 6** and **7**

4.2.2 One or two step approach

Question 4: Should the web site score be calculated in one step (directly from the test results) or in two steps (with intermediary calculation of page score)?

The table below compares the one step and the two step approach. All functions in the two step approach are based on the principle that the site score is calculated as (weighted) average of the page scores.

(related requirement)	***One step (test results – site)***	***Two step (test results – page – site)***
	The main focus of the Observatory is the evaluation of sites (and groups of sites).	People think on page level. It is natural to average over pages.
StopCrit	The proposed site score function is compatible with adaptive sampling.[28]	Average page result yields useful stop criterion.
Web Acc – Site 3	Always fulfilled.	Depends of function. (Pages could be weighted according to their position in the calculation of the site score.)
Web Acc – Site 4'	Always fulfilled.	In general not fulfilled.
Statistics – Stability 1	Should be OK	Pages with small N have more variability, but are treated equally, so tend to inflate variability of the site score.
Statistics – Stability 2	Needs to be checked.	Always fulfilled.
Maths – Input 2	Needs to be explored. (It is of advantage that only site level results are calculated.)	Needs to be explored. (Potentially problematic since site score cannot longer be calculated as simple average of page scores.)

[28] The following requirement can be derived from this observation: "Site score can be calculated from any number of pages and yields a value that converges / stabilises when the sample is large enough."

(related requirement)	***One step (test results – site)***	***Two step (test results – page – site)***
MATHS – INPUT 6	If you can apply weights to the page score then you can also apply weights to the page components as they contribute to the site score.	Always fulfilled.

Table 3: Comparison of one step and two step approach

The following requirements need to be checked for each potential score function separately:

- **WEB ACC – PAGE 2, 4,** and **5**
- **WEB ACC – SITE 1**
- **WEB ACC – PARAMETER 2**
- **MATHS – OUTPUT 3**
- **MATHS – INPUT 3**

4.2.3 Simple or advanced approach

The choice between the simple (barrier ratio) and advanced approach (barrier ratio and diversity) was done based on the rejection of the diversity parameter. The diversity parameter may optionally be presented as a separate score value. **MATHS – OUTPUT 3** leads to the rejection of the EIAO internal function. Then we are left with the barrier ratio function. The question is then if this function should be calculated on a page or a site basis. Advantages by calculating the barrier ratio on page basis, and averaging it to a site basis, is that operating on a page basis is a well known concept for accessibility experts, and this approach should work with test sets from different document types, provided that the test sets have equal, or close to equal test coverage. Statistically, the barrier ratio per site $F_1(s)$ is a more efficient estimator of a single site characteristic than the average barrier ratio per page $\overline{f_2(p)}$ for all pages *p* in *s*. It is therefore suggested to use one step approach with calculation of barrier ratio per site directly from the underlying barrier data, instead of the two-step approach that takes the average of page level barrier ratios. Note that it still is possible to calculate the barrier ratio per page independently of this decision.The barrier ratio with diversity function will be in conflict with requirement **MATHS – INPUT 5**, because unknown weighting parameters are specified. It is therefore, for the time being, recommended to keep the barrier ratio and barrier diversity as two separate measures to avoid the unknown weighting factors. The recommendation is therefore to use the barrier ratio per site function as the UWEM aggregation function, optionally with the additional diversity parameter presented separately.

5 Presentation and visualisation of results

For visualising the results in a graphical user interface, scorecards may be used, for example with a colour code and an associated short description of the score. A score (page score, site score or aggregated score for a group of sites) will then fall into one of several scorecard categories. The limits between these scorecard categories still need to be investigated. One option, is to use limits based on percentiles of the site score distribution.

We may divide the scores into a number of percentiles based on a Gaussian distribution. If we assume that we will have 5 score card categories; *Excellent*, *Good*, *Medium*, *Poor,* and *Very Poor*, we can then choose the following percentiles as limits between the 5 score cards: P20, P40, P60, P80.

There are two options for calculating these percentile based cut off values:

1. We may calculate new cutoff values for each monthly crawl, based on the distribution of the results for individual web sites. This means that all web site scores will compete about their rank within the distribution. For example, if all web sites improve their score, then the web sites that improve fastest will get a higher score, and those with lower than average improvement speed will risk to lose their score. An advantage with this approach, is that it scales itself dynamically. A possible disadvantage, is that the scale might become too sensitive if web most designers fixed all the problems, so that very little separated a good and a bad web site. However, this may be counteracted by maintaining, improving and extending the UWEM test set. Note that old scores should retain their cut-off values, and the procedure for calculating them should be clearly disclosed, to avoid doubt about the consistency from users.
2. Fixed cutoff values. It would then be necessary to perform initial crawling experiments on a suitable representative set of web sites in order to get a good estimate for mean and standard deviation of the distribution of the results. From this information, the percentiles can then be calculated. A disadvantage is that if the accessibility situation changes, too many results may end up in one or two categories and the full scale will not be used as intended.

Version 1. is preferred, because it provides dynamic re-scaling of the cut-off values under changes in the distribution function e.g. caused by like addition or removal of UWEM tests, web sites or an overall change in score distribution over time due to changes in technology.

6 Appendix A: Mind maps of the requirements

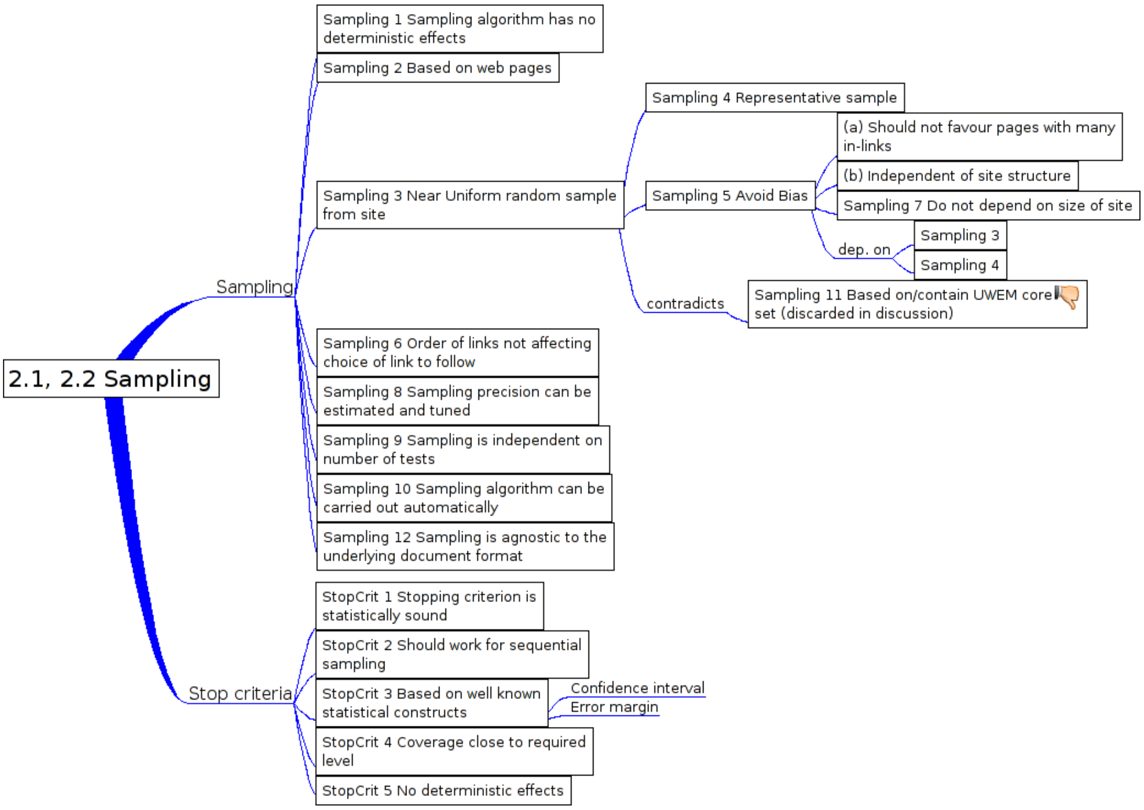

2.1, 2.2 Sampling
Sampling
Sampling 1 Sampling algorithm has no deterministic effects
Sampling 2 Based on web pages
Sampling 3 Near Uniform random sample from site
Sampling 4 Representative sample
Sampling 5 Avoid Bias
(a) Should not favour pages with many in-links
(b) Independent of site structure
Sampling 7 Do not depend on size of site
dep. on
Sampling 3
Sampling 4
contradicts
Sampling 11 Based on/contain UWEM core set (discarded in discussion)
Sampling 6 Order of links not affecting choice of link to follow
Sampling 8 Sampling precision can be estimated and tuned
Sampling 9 Sampling is independent on number of tests
Sampling 10 Sampling algorithm can be carried out automatically
Sampling 12 Sampling is agnostic to the underlying document format
Stop criteria
StopCrit 1 Stopping criterion is statistically sound
StopCrit 2 Should work for sequential sampling
StopCrit 3 Based on well known statistical constructs
Confidence interval
Error margin
StopCrit 4 Coverage close to required level
StopCrit 5 No deterministic effects

Figure 3: Mind map of sampling requirements

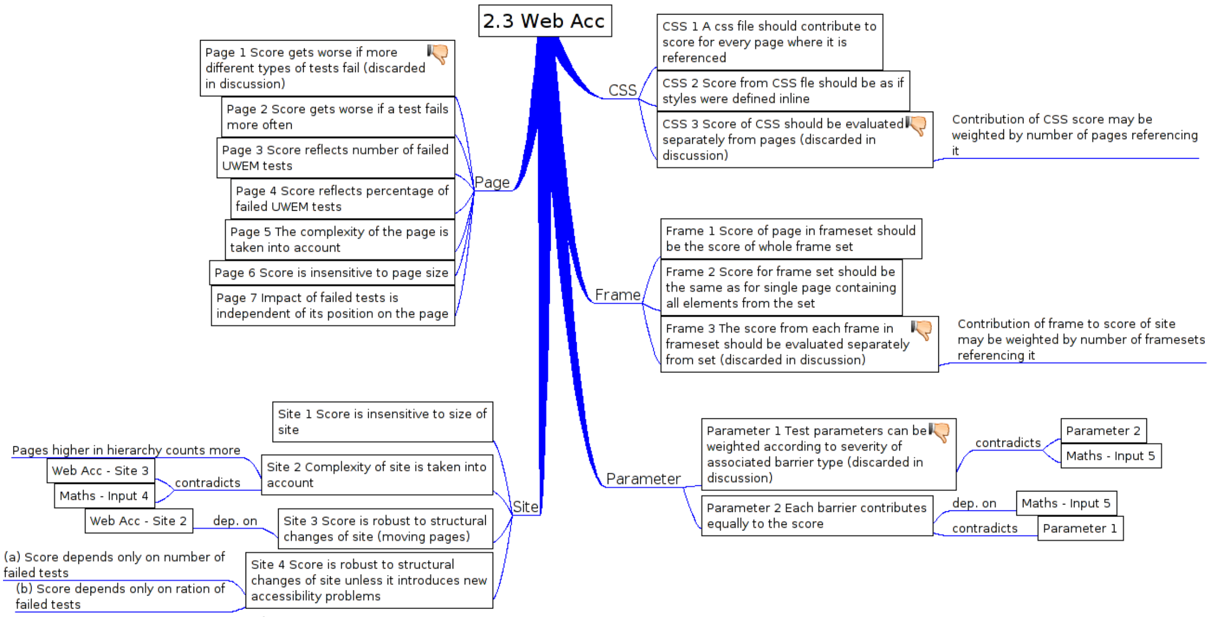

2.3 Web Acc
CSS
CSS 1 A css file should contribute to score for every page where it is referenced
CSS 2 Score from CSS fle should be as if styles were defined inline
CSS 3 Score of CSS should be evaluated separately from pages (discarded in discussion)
Contribution of CSS score may be weighted by number of pages referencing it
Page
Page 1 Score gets worse if more different types of tests fail (discarded in discussion)
Page 2 Score gets worse if a test fails more often
Page 3 Score reflects number of failed UWEM tests
Page 4 Score reflects percentage of failed UWEM tests
Page 5 The complexity of the page is taken into account
Page 6 Score is insensitive to page size
Page 7 Impact of failed tests is independent of its position on the page
Frame
Frame 1 Score of page in frameset should be the score of whole frame set
Frame 2 Score for frame set should be the same as for single page containing all elements from the set
Frame 3 The score from each frame in frameset should be evaluated separately from set (discarded in discussion)
Contribution of frame to score of site may be weighted by number of framesets referencing it
Parameter
Parameter 1 Test parameters can be weighted according to severity of associated barrier type (discarded in discussion)
contradicts
Parameter 2
Maths - Input 5
Parameter 2 Each barrier contributes equally to the score
dep. on
Maths - Input 5
contradicts
Parameter 1
Site
Site 1 Score is insensitive to size of site
Site 2 Complexity of site is taken into account
Pages higher in hierarchy counts more
Web Acc - Site 3
Maths - Input 4
contradicts
Site 3 Score is robust to structural changes of site (moving pages)
Web Acc - Site 2
dep. on
Site 4 Score is robust to structural changes of site unless it introduces new accessibility problems
(a) Score depends only on number of failed tests
(b) Score depends only on ration of failed tests

Figure 4: Mind map of Web Acc requirements

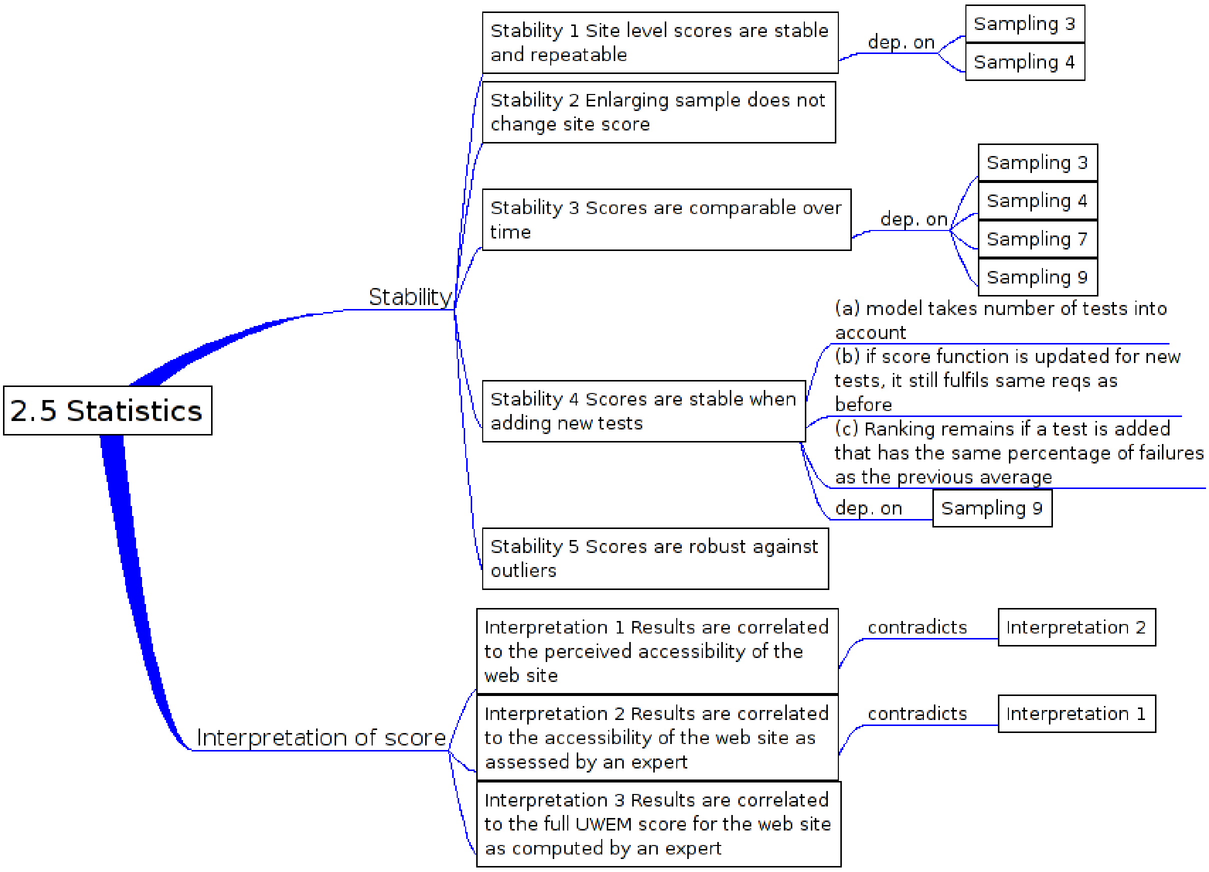

2.5 Statistics
Stability
Stability 1 Site level scores are stable and repeatable
dep. on
Sampling 3
Sampling 4
Stability 2 Enlarging sample does not change site score
Stability 3 Scores are comparable over time
dep. on
Sampling 3
Sampling 4
Sampling 7
Sampling 9
Stability 4 Scores are stable when adding new tests
(a) model takes number of tests into account
(b) if score function is updated for new tests, it still fulfils same reqs as before
(c) Ranking remains if a test is added that has the same percentage of failures as the previous average
dep. on
Sampling 9
Stability 5 Scores are robust against outliers
Interpretation of score
Interpretation 1 Results are correlated to the perceived accessibility of the web site
contradicts
Interpretation 2
Interpretation 2 Results are correlated to the accessibility of the web site as assessed by an expert
contradicts
Interpretation 1
Interpretation 3 Results are correlated to the full UWEM score for the web site as computed by an expert

Figure 5: Mind map of statistical requirements

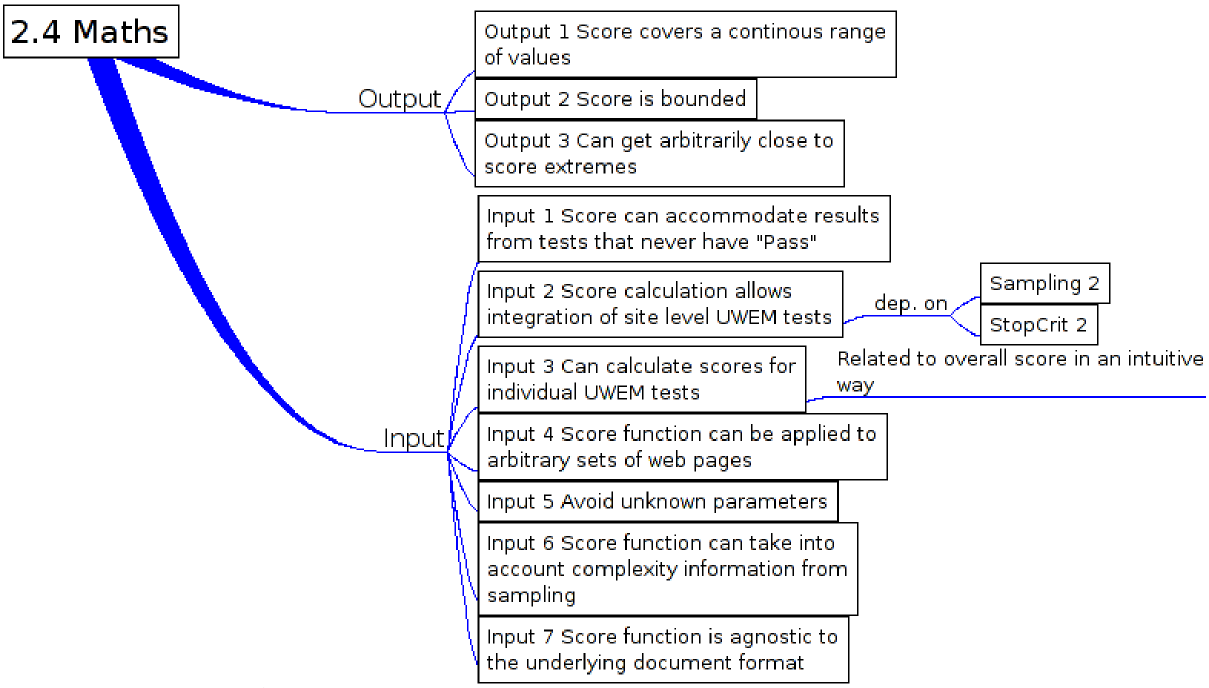

2.4 Maths
Output
Output 1 Score covers a continous range of values
Output 2 Score is bounded
Output 3 Can get arbitrarily close to score extremes
Input
Input 1 Score can accommodate results from tests that never have "Pass"
Input 2 Score calculation allows integration of site level UWEM tests
dep. on
Sampling 2
StopCrit 2
Input 3 Can calculate scores for individual UWEM tests
Related to overall score in an intuitive way
Input 4 Score function can be applied to arbitrary sets of web pages
Input 5 Avoid unknown parameters
Input 6 Score function can take into account complexity information from sampling
Input 7 Score function is agnostic to the underlying document format

Figure 6: Mind map of mathematical requirements

]

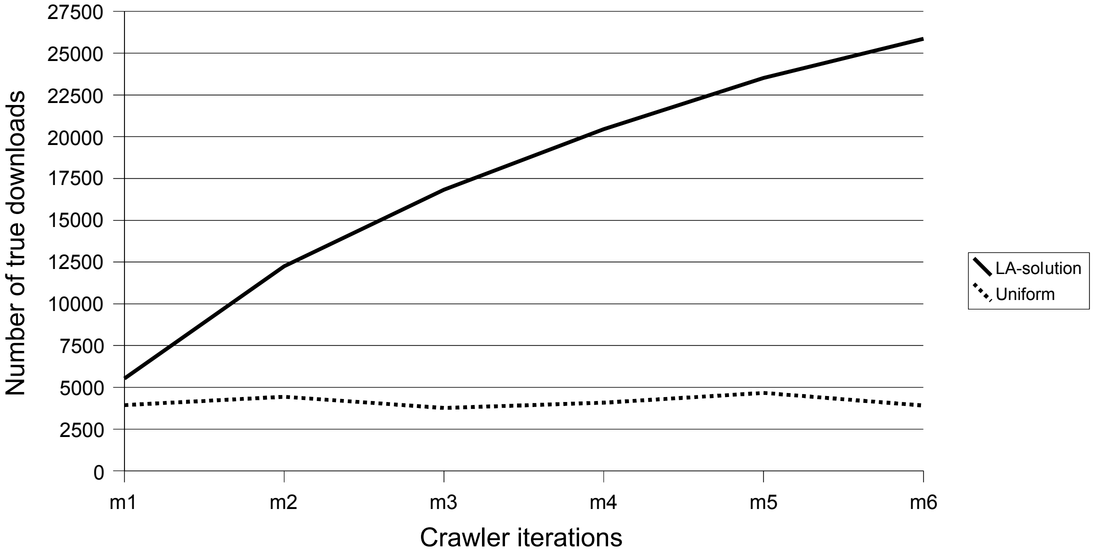
7 Appendix B: Plot of update detection

The following experiment show how an learning automata based solution, proposed in 3.1.2 operates compared to uniformly selecting pages in a simulated environment. This experiment only includes a simulation of the URL extraction phase, not the evaluation phase. A true download is defined as when a page is visited and is has changed since last crawl. In this experiment M was set to 35 pages. The true number of unique pages within each site was set to 135. The pages within each site had a distribution of update probability according to zipf where *s* set to 1.3, *k* set to 1, and N is 50.

$$f(k;s,N) = \frac{1/k^s}{\sum_{n=1}^{N} 1/n^s}$$

This is according to the experiments done in [knap].

Note that in this experiment, the cost of downloading a page that has not been updated and a page that has been update is considered equal.



8 Appendix C Requirement overview table

Requirement ID	***Requirement text***	***Compliant?***
Web Acc – CSS 1	A CSS file should contribute to the score for every web page in which it is referenced.	Yes, with new sample definition.
Web Acc – CSS 2	The score for a web page that references a CSS file should be the same as if all styles were defined in-line (i.e. with style attributes).	
Web Acc – Frame 1	A web page that is presented in the context of a frame-set should contribute to the score for the whole frame-set (and not be evaluated separately).	
Web Acc – Frame 2	The score for a frame-set should be the same as for a single page containing all the material in the set.	
Web Acc – Page 2	The score gets worse if a test fails more often (anywhere within the site). The score improves if a test fails less often.	Barrier ratio, yes, Diversity index, no
Web Acc – Page 4	The score reflects the percentage of failed UWEM tests (Rp).	Yes
Web Acc – Page 6	The score is insensitive to page size (i.e. size of the source file).	Yes
Web Acc – Page 7	The impact of a failed test (i.e. the amount it contributes to the page score) is independent of its position within the page.	Yes
Web Acc – Site 1	The score is insensitive to the size of the site (i.e. the number of pages).	Yes
Web Acc – Site 3	The score is robust to structural changes of the web site, i.e. moving the pages within the site doesn't change the result (for the site).	Yes
Web Acc – Site 4	The score is robust to structural changes of the web site, i.e. moving content among pages doesn't change the result (for the site), unless the changes introduce new accessibility problems.	Yes
Web Acc – Parameter 2	Each barrier type contributes equally to the score.	Yes
Maths – Output 1	The score covers a continuous range of values.	Yes
Maths – Output 2	The score is bounded, i.e. there are minimum and maximum values.	Yes
Maths – Output 3	The score can get arbitrarily close to the extreme values (taking into account the behaviour of the function as well as the expected properties of the input data).	Barrier ratio only: yes Barrier diversity+barrier ratio: No, but close.
Maths – Input 1	The score can accommodate results from tests that never have "pass" results due to their applicability criteria.	No, however this can be mitigated. (see also Question 1)
Maths – Input 2	The score calculation allows integration of UWEM tests that are applied on site level (e.g. tests for navigation consistency).	Yes
Maths – Input 3	It is possible to calculate scores for individual UWEM tests. These scores are related to the overall score in an intuitive way.	Propose simple percentage calculation for single test score. (Can be derived from **Web Acc – Page 4**.

Requirement ID	***Requirement text***	***Compliant?***
Maths – Input 4	The score function can be applied to arbitrary sets of web pages (including sub domains of a web site, or language specific samples) as long as all pages are from the same site.	Yes
Maths – Input 5	The score function should avoid unknown parameters, if it is not viable to estimate these parameters with the resources available (e.g. barrier severity parameters).	No unknown parameter if the barrier ratio and optionally the barrier diversity is presented as two separate measures.
Statistics – Stability 1	Scores on site level are stable and repeatable. Evaluating the same site twice produces similar results within a defined tolerance.	Yes
Statistics – Stability 2	Enlarging the sample from a site (such that the sampling requirements are still met) does not change the score of the site.	Yes
Statistics – Stability 3	Scores are comparable over time, i.e. measuring improvement or changes over time is possible.	Yes
Statistics – Stability 4	Scores are stable (i.e. still comparable) if a new test is added to the test set. (a) The model takes into account the number of tests. (b) If the score function is updated to accommodate new tests, it still fulfils the same requirements as before the update.	Barrier ratio, yes Diversity function has a small dependency.
Statistics – Stability 5	Scores are robust, i.e. the score of a site is not unduly affected by outliers.	The score should be relatively robust.

References

Monika R. Henzinger, Allan Heydon, Mikael Minzenmacher, Marc NajorcOn Near-Uniform URL Sampling2000
Ziv Bar-Yossef, Alexander Berg, Steve Chien, Jittat FakcharoenpholApproximating Aggregate Queries about Web Pages via Random Walks2000http://www.ee.technion.ac.il/people/zivby/papers/webwalker/webwalker.ps
Paat Rusmevichientong, David M. Pennock, Steve Lawrence, C. Lee GilesMethods for Sampling Pages Uniformly from the World Wide Web2001
Lee Breslau, Pei Cao, Li Fan, Graham Phillips, Scott SchenkerWeb Caching and Zipf-like Distributions: Evidence and Implications1999http://linkage.rockefeller.edu/wli/zipf/breslau99.pdf
John Oommen, Ole-Christoffer Granmo, Svein Arild Myrer and Morten Goodwin OlsenLearning Automata-based Solutions to the Nonlinear Fractional Knapsack Problem with Applications to Optimal Resource Allocation2007
C. CastilloEffective Web Crawling2004
J. B. OommenStochastic searching on the line and its applications toparameter learning in nonlinear optimization1997