eGovMon
eGovernment Monitoring
Project no.: Verdikt 183392/S10

eGovMon System Design Specification

D4.2.1

Version:	1.0
Date:	2009-11-25
Author:	Morten Goodwin Olsen
Dissemination Level:	Public
Status:	Final

Abstract:

In this document we present the design specification of the first iteration of the eGovMon implementation. The eGovMon tool is built on the results from the FP6 project European Internet Accessibility Observatory (EIAO), and is intended to include approaches to measure accessibility, transparency, efficiency and impact.

The main focus of this paper is to present the architecture updates from EIAO to eGovMon motivated by changes needed to: (1) evaluate transparency and (2) simplify the data storage approach and improve the capacity compared to the results achieved in the EIAO project.

This document consists of 29 pages including this cover

Version control:

Version	***Status***	***Date***	***Change***	***Author***
0.1	DRAFT	2009-05-21	Initial Draft.	Morten Goodwin Olsen
0.2	DRAFT	2009-08-21	Updated Architecture description and figure. Added implementation anatomy.	Morten Goodwin Olsen
0.3	RC	2009-08-31	Updated based on comments from Annika Nietzio and Mikael Snaprud	Morten Goodwin Olsen
0.4	RC	2009-08-31	Minor comments from Annika Nietzio and Mikael Snaprud	Morten Goodwin Olsen
1.0	Final	2009-11-25	Finalized	Morten Goodwin Olsen

Table of Contents

1 Introduction

In this deliverable we present the system architecture for the eGovernment Monitoring tool. The architecture is largely based on the existing European Internet Accessibility Observatory architecture (EIAO [D5.2.2.1-2]). The final eGovMon architecture will need to cover all four eGovATEI indicators namely accessibility, transparency, efficiency and impact.

The main focus in this report is on how to extend the EIAO tool, created for measuring accessibility, for transparency measurements. Note that efficiency and impact will be addressed in later eGovMon reports.

In addition to adding transparency measurements, we suggest to simplify the architecture from EIAO in order to:

- Remove the duplicate data in the system,
- Make it possible to run all components in parallel – allowing for a drastic performance increase,
- Simplify the entire tool making it easier to maintain and extend.

This deliverable is organised as follows. First we present the architecture both for accessibility evaluation and transparency as well as work flows. Subsequently, we present the individual components including assigned developers and maintainers responsible, estimated effort and priority. Finally, we present an anatomy chart of the implementation tasks. The deliverable is meant both as a specification document for the eGovMon tool and an ongoing paper for eGovMon deliverable.

Note that we do not present a detailed database schema in this deliverable. To design an efficient database we need to know exactly what the data will look like (what to store for each indicator), granularity of the data and which data we will present in a user interface. The challenge is that we need some initial data to build a better understanding of the nature of the data and its use. Therefore this will be our initial design for the first implementation to collect initial data.

Because of this, we implement a simple database. When we have more knowledge, we can make decisions on which data will be useful for reports and which to present in a user interface etc. In short, this allows us to get to know the data before starting to implement a more full-fledged version.

Specification for user interfaces and how to present data are covered in a separate deliverable D6.4.1. For details on the EIAO architecture please refer to [D5.2.2.1-2].

1.1 Related Deliverables

This deliverable is related to the following eGovMon Deliverable:

- **D3.2.1,** Annika Nietzio, eGovMon Transparency Indicators (part 1)
- **D4.1.1,** Morten Goodwin Olsen, Web-mining methods for measuring transparency
- **D6.4.1**, Mandana Eibegger, Large Scale Online Reporting Tool specification

2 Architecture

2.1 Accessibility Measurements

The eGovMon architecture is largely based on the already existing EIAO architecture [D5.2.2.1-2]. However, there are several simplifications. Most significantly, the URL repository, RDF databases and Data Warehouse (DW) are merged into one new database called eGovMonDB. In EIAO, the URL repository and Data Warehouse consisted of redundant URL data which sometimes led to inconsistencies. Furthermore, the RDF databases and Data Warehouse both contained the same results.

EIAO was designed with an ETL (Extract Transform Load) work flow designed to read information from the URL repository and the RDF databases and write the information to the Data Warehouse. This was time consuming as only one ETL could run simultaneously. A merge of the databases, can both remove duplicate data and simplify the entire architecture.

These changes are also presented in Figure 2.1. An overview of the architecture including both accessibility and transparency is available in Figure 2.3.

Consequences of merging URL repository and DW:

- revise URL repository to also include results,
- remove DW,
- remove ETL,
- remove RDF-database,
- let the sampler write results directly to the eGovMonDB instead of to many temporary RDF databases.

This will have a performance increase as we no longer need RDF, ETL etc. We remove some of the most time consuming tasks and get the possibility to run the entire tool in parallel.

Note that sanity checking, such as only writing site results to the Data Warehouse if there has been enough samples drawn from the web site, are done in the Sampler and not the ETL. This functionality will remain unchanged.

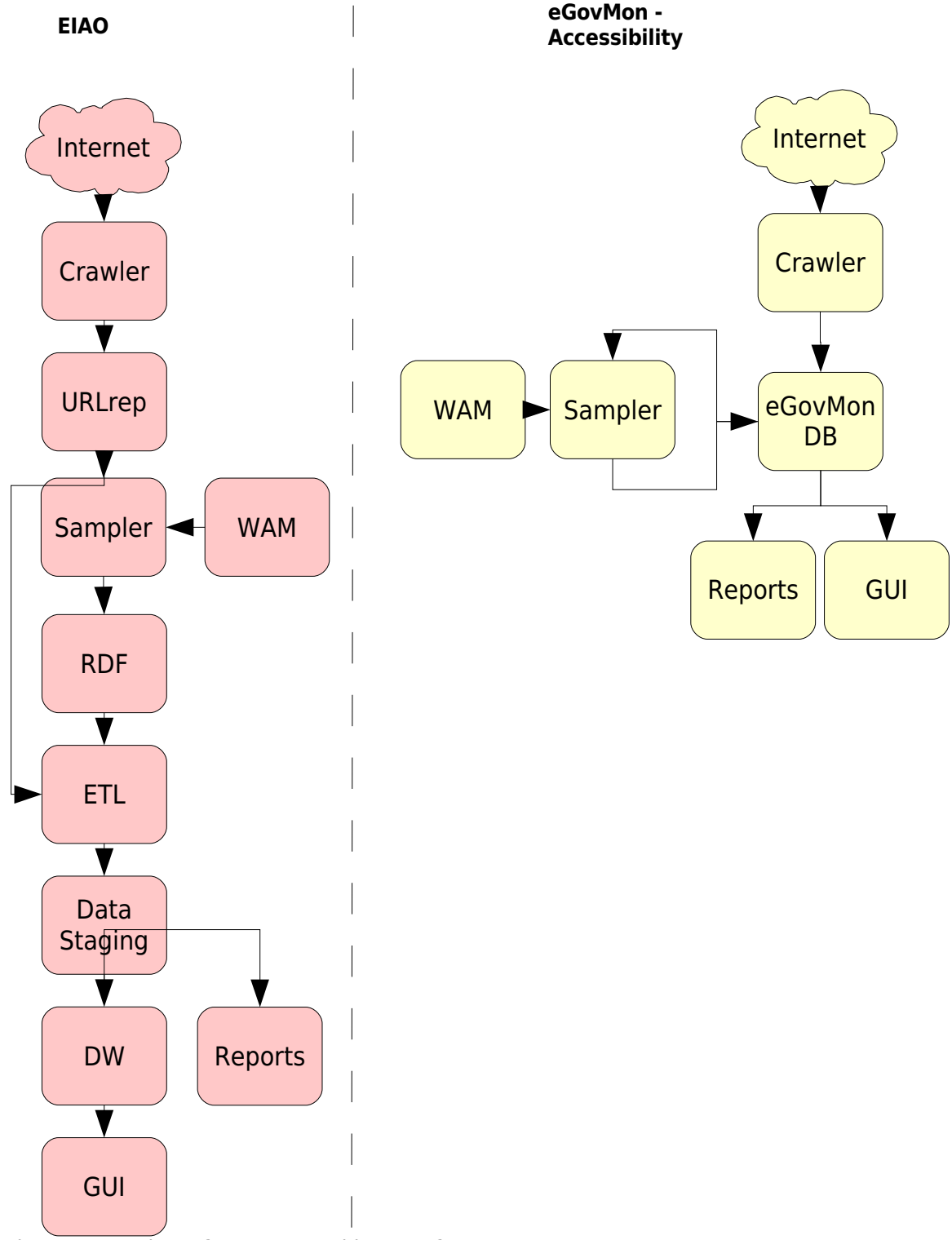

Figure 2.1: Update of eGovMon Architecture from EIAO

2.2 Transparency Measurements

Figure 2.1 contains the architecture for accessibility evaluation. For measuring transparency, the architecture would be slightly different. For transparency, the tests are not targeted directly towards individual elements of web pages, but rather properties of web sites and services.

This means, in contrast to accessibility measurements, there are no direct differences between Sampling and Evaluation. Therefore the metrics for measuring transparency have to be designed differently.

An overview of the architecture seen from the point of view of transparency can be found in Figure 2.2 and Figure 2.3.

The architecture is divided into three main parts:

1. **Crawler(s)** responsible for exploring URLs for each web site.
2. **Transparency Metrics (TraM)** responsible for both sampling and applying the transparency tests. Note that this corresponds to the combination of the WAM and the Sampler in the accessibility measurements.
3. **eGovMonDB** storing URLs from the Crawlers and results from the transparency metrics.

Note that the system is designed keeping all ATEI indicators in mind. The Crawler and eGovMonDB[1] are intended to remain the same for all metrics while the metrics (TraM), is intended to be exchangeable. By doing this, we avoid having to maintain duplicate branches for each metric. Further note that the transparency metric consists of several components (see Figure 2.4).

1 The eGovMonDB needs to be continuously maintained to so that results from all metrics can be stored.

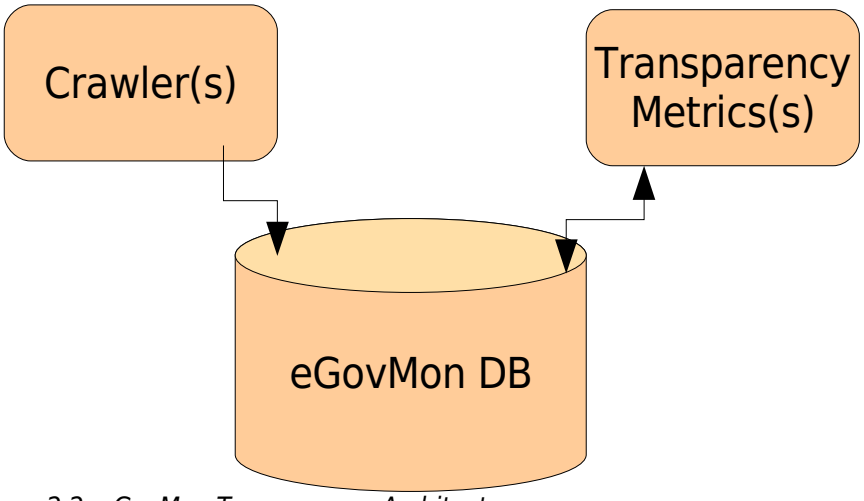

Figure 2.2: eGovMon Transparency Architecture

2.2.1 Transparency Metric (TraM),

The main component of the tool for evaluating transparency is the transparency metric (TraM) responsible for doing the actual evaluations. An overview of the architecture design for the TraM can be seen in figure Figure 2.4. For an overview of the tests, we refer the reader to D3.2.1 and D4.1.1.

Manually Collected Data

Prior to any evaluation, data is collected manually as input to the learning algorithms (training data). Such training data could include examples of mail record pages which the algorithms can use to identify the features of mail record pages. The results from the manually collected data are stored in the eGovMonDB while the pages and URLs are stored in the eGovMonDB and files on disk.

Controller

During the evaluation, a controller keeps track of pages to evaluate (sampling), what features are extracted, as well as the interaction between the components such as sending the results from the TraM to the eGovMonDB. The controller is also responsible for the order of applying the tests. For example, testing if a mail record has a search functionality should only be done after a mail record has been detected.

TraM

The TraM itself consists of several individual components.

For each web page / web site to evaluate, features are extracted. These features are intended as input to the algorithms both the deterministic tests and learning algorithms. Examples of features from a web page could be the individual words in each web pages.

Subsequently, both the deterministic and learning algorithm tests are applied based on the features extracted.

The deterministic tests take the features as input and have a deterministic boolean result as output. Example of such deterministic test would be finding a phone number (eight subsequent integer numbers) from input features. It has true/false as output.

The learning algorithm tests take both the previously stored manually collected data and features as input. Example of a learning algorithm would be to detect if the evaluated page is a mail record. This would be done by comparing the page to be evaluated with manually collected data of mail records. Details regarding the classifications can be seen in D4.1.1.

When the tests have been applied, an aggregation function (described in D3.2.1) is applied. The intention with this is to aggregate the intermediate results for a single indicator. Example of such an aggregation would be finding which web page, if any, is most likely to be a mail record.

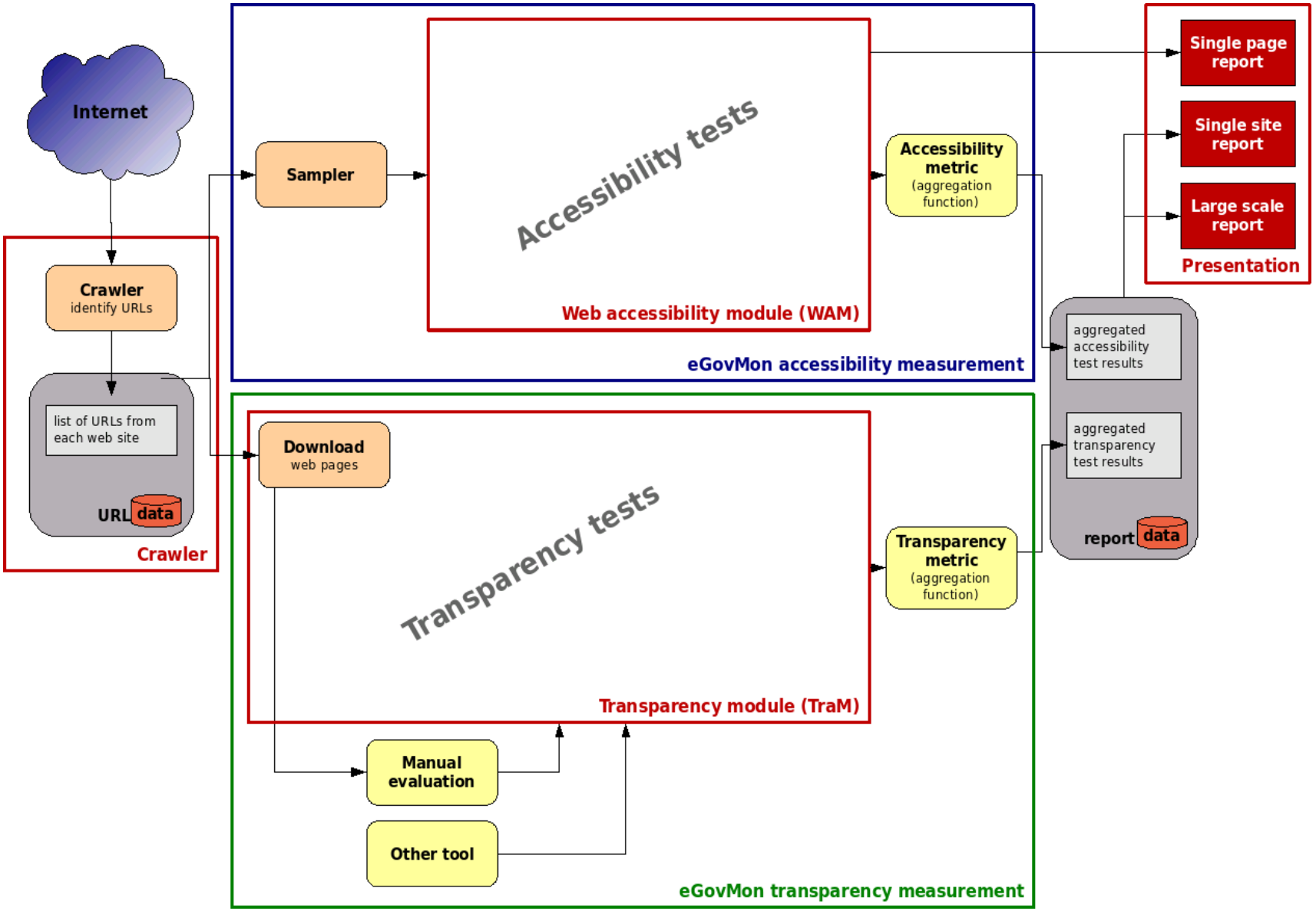

Figure 2.3: eGovMon Tool

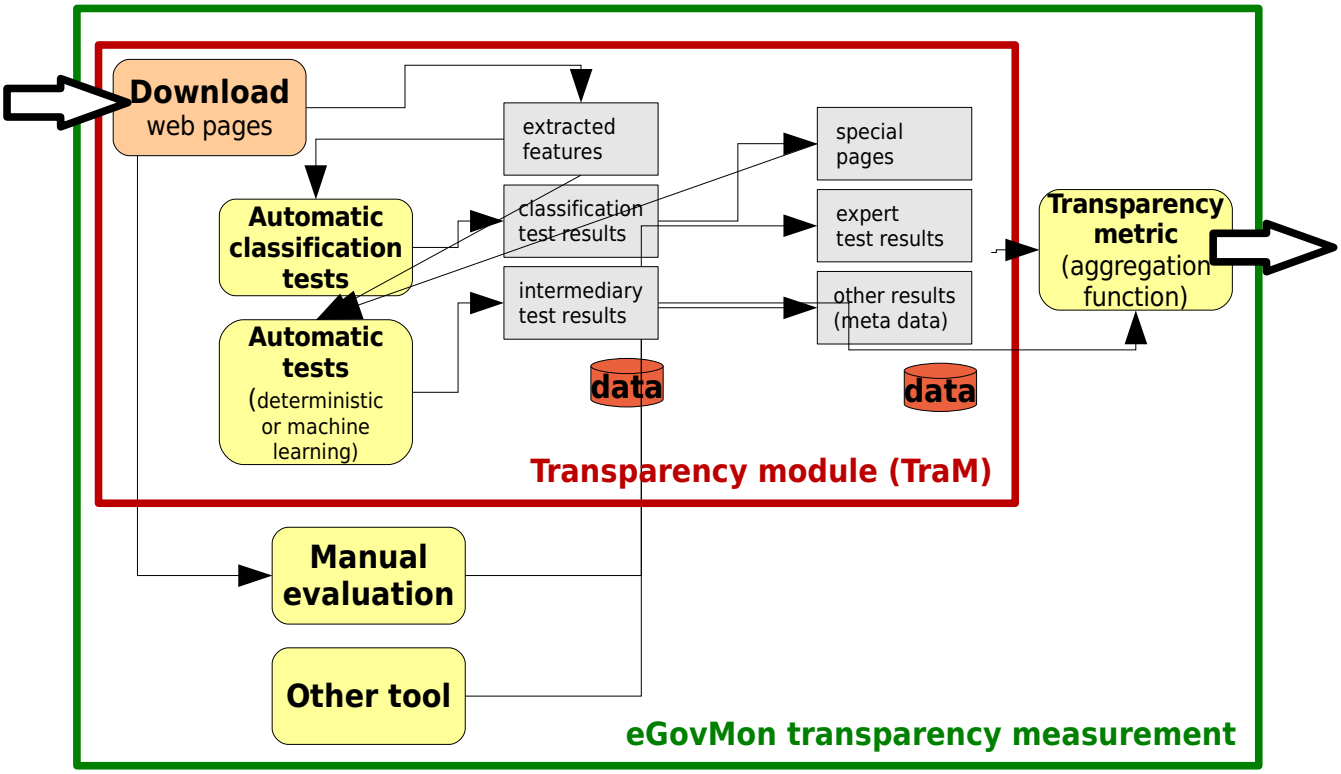

Figure 2.4: eGovMon Transparency Measurements

3 Work Flow

In this section we present two work flows; the first showing the entire tool while the second shows how tests are applied. Note that only transparency measurements are presented here.

In figure 3.1 we present the main work flow for transparency measurements. Note that for reasons of clarity, only the main components are presented here.

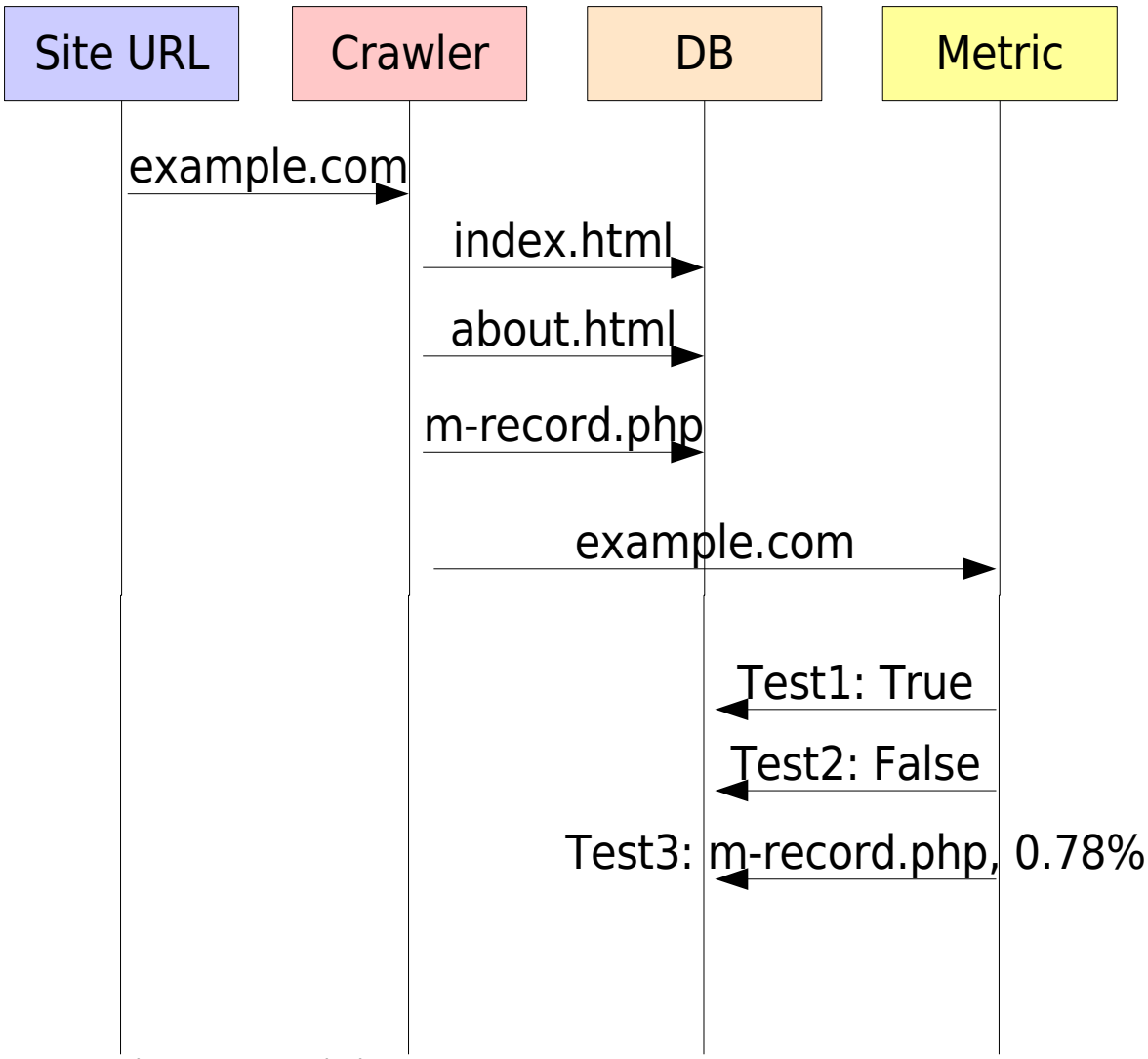

Figure 3.1: Main eGovMon Work Flow

In Figure 3.2 we present the work flow for applying tests. In this situation we assume the web site consists of two simple web pages, namely index.html and m-rec.php. There exist two deterministic tests; finding an RSS link and finding a phone number. Additionally there are two heuristic learning algorithm tests; finding mail record and finding accessibility statement.

Index.html

First index.html is downloaded (before this work flow starts). Then the features are extracted. In this case the features are only *91 23 45 67, velkommen, les* and *mer.* These features are sent to both the deterministic tests and heuristic tests. The deterministic tests find a phone number but no RSS feed and sends this to the controller. The heuristic tests claims this is a mail record with probability 14% (very low) and accessibility statement with 2%.

m-rec.php

Secondly m-rec.php is downloaded and the features extracted. In this case the features are postliste and 2009-05-15. These are again sent to both the deterministic tests and heuristic tests. The deterministic finds no phone number and no RSS feed, and sends this to the controller. The heuristic tests claims this is a mail record with probability 93% (very high) and accessibility statement with 3%.

Finally, the controller knows that all pages have been evaluated, commits the following:

- a mail record is in m-rec.php with probability 93%,
- no accessibility statements where found (because the probabilities are so low), [2]
- phone found in index.html and
- no RSS link found.

The controller keeps track of

- The state (e.g. all pages finished) and commits to the database when a site or test is finished.
- Test committed to the DB which may consist of many individual tests.
- Test results used as input to other tests.
- Which tests should be run and in sequence. As an example, if contact information has already been found in a web site, it is no longer needed to evaluate the remaining pages in order to find if contact information is available.

2 The threshold values are derived from the training data.

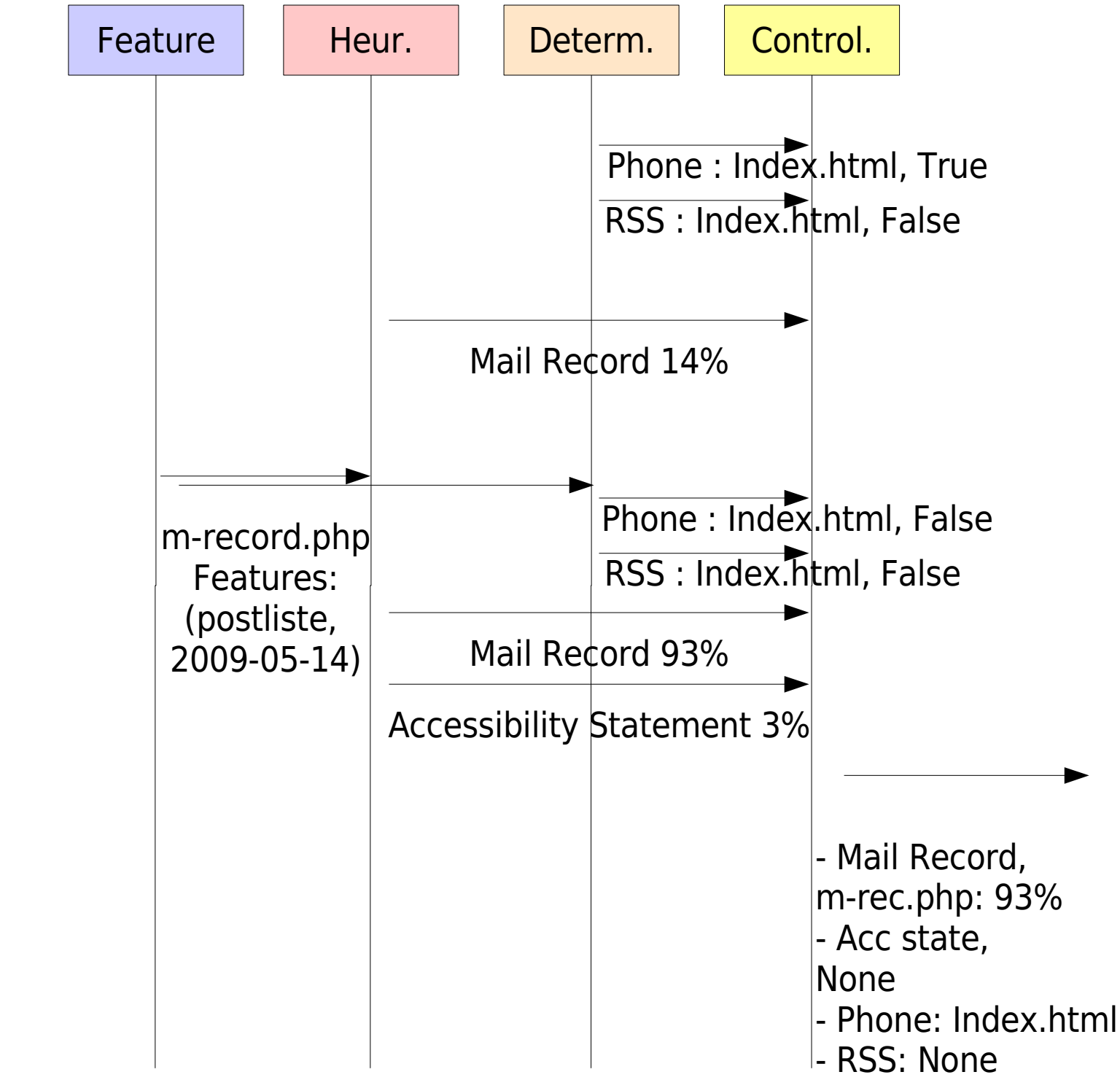

Figure 3.2: Work Flow for applying tests

4 Description of individual components

In this section we list the individual components part of accessibility and transparency evaluation as well as how these are changed from EIAO. The components which require little or no change are only briefly mentioned here.

For each of the listed components, the following information is available:

- **Applicable:** Whether it is applicable for accessibility, transparency or both. Note that components regarding efficiency and impact will be dealt with at a later stage.
- **Short Description:** Short description of the component.
- **Changed from EIAO:** Changes from EIAO if any.
- **Status:** Status of the development.
- **Priority:** Priority of the development.
- **Estimated Effort:** Estimated effort in number of days.
- **Work Package:** Responsible work package
- **Location:** URL to subversion where the component is located.
- **Responsible:** Responsible person. For components already updated, the person who did the changes is marked as responsible.

Note that all components should be unit tested. Every night unit tests of all components are run automatically. If code breaking the tool is committed, and this breaking code makes some of the unit tests fail, this will automatically be known quickly. For each component a test.py should be added testing the unit. Additionally, for more elaborate tests, such as bugs discovered, separate test files could be created and added to test suite: http://svn.egovmon.no/svn/eGovMon/tests/

Any bug discovered and fixed should be replicated with a unit test and attached to the test suite. Issues should be not closed unless a unit test exits. This allows us to make sure the same bug isn't repeated at a later stage.

It is worth noticing that the most significant changes are the eGovMonDB (see section 4.22 eGovMonDB) and Transparency Metrics (see section 4.23 Transparency Metrics).

4.1 System Configuration

Applicable: Accessibility and Transparency

Short Description: Contains configuration of machine dependant static information used by eGovMon such as database passwords, location of log files. All such information should be part of the System Configuration.

Changed from EIAO: Drastically simplified. Instead of using an RDF database, we have the RDF only in a rdf-text file and keep this in memory in the tool. This makes it easier to update/install, more maintainable, faster etc.

Status: Done.

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/SystemConfiguration/

Responsible: Morten Goodwin Olsen

4.2 Adaptive Sampling

Applicable: Accessibility

Short Description: The name was misleading. The Adaptive Sampler calculates site score etc..

Changed from EIAO: Merged with Sampler and deleted.

Status: Done.

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/ETLServer

Responsible: Morten Goodwin Olsen

4.3 Add URLs

Applicable: Accessibility and Transparency

Short Description: Adding URLs to the database. Add scoping of URLs. Add seed URLs.

Changed from EIAO: Needs to be compatible with the eGovMonDB (see section 4.22 eGovMonDB). Minor.

Status: Not started.

Priority: High

Estimated Effort: Minor update.

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/AddURLs

Responsible: To be determined

4.4 Crawler – HarvestMan

Applicable: Accessibility and Transparency

Short Description: Crawling and Downloading the web sites.

Changed from EIAO:
No Changes. The crawling process will remain the same. See D3.2.1 for further details.

Status: -

Priority: -

Estimated Effort: None

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/Crawler/ and http://svn.egovmon.no/svn/eGovMon/trunk/HarvestMan/

Responsible: Anand B. Pillai

4.5 Crawlers

Applicable: Accessibility and Transparency

Short Description: Daemon keeping track of the individual crawlers. (Note the difference with Crawler/HarvestMan). Name should be updated to CrawlerServer to be in line with SiteURLServer, Sampling Server etc.

Changed from EIAO: Simple name change, otherwise unchanged.

Status: Done

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/Crawlers/

Responsible: Christian Amble

4.6 CronWAM

Applicable: Accessibility

Short Description: Checking that the WAM is running. Restart if not.

Changed from EIAO: None

Status: -

Priority: -

Estimated Effort: -

Location: http://svn.egovmon.no/svn/eGovMon/trunk/CronWAM/

Work Package: WP4

Responsible: -

4.7 DBCleaner

Applicable: Accessibility and Transparency

Short Description: Cleaning up logs, databases which are too old. Avoids having too many links.

Changed from EIAO: Almost unchanged. Cleaning up RDF (mysql) databases should be removed (minor) since the RDF databased will be removed.

Status: Not started.

Priority: Low

Estimated Effort: Very low.

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/DBClearer

Responsible: Morten Goodwin Olsen

4.8 Data Warehouse

Applicable: Accessibility and Transparency

Short Description: Storing the data.

Changed from EIAO: Removed and merged with eGovMonDB. See eGovMonDB for more details. (see section 4.22 eGovMonDB)

Status: Not Started.

Priority: See section 4.22 eGovMonDB

Estimated Effort: See section 4.22 eGovMonDB

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/DataWarehouse.

Responsible: -

4.9 ETL Server

Applicable: Accessibility and Transparency

Short Description: Keeping track of ETLs.

Changed from EIAO: Removed since ETL is removed. (see section 4.22 eGovMonDB)

Status: Not started.

Priority: See section 4.22 eGovMonDB

Estimated Effort: See section 4.22 eGovMonDB

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/ETLServer

Responsible: -

4.10 Install

Applicable: Accessibility and Transparency

Short Description: The tool should be installable automatically. This allows us to:

(1) Quickly install the tool on new machines,

(2) Check if errors exists by nightly build

(3) Make it possible for other people to install it.

Changed from EIAO: The installation for EIAO did not work. It was always a challenge and a lot of effort to install the tool – especially for other people outside of EIAO/eGovMon. Note that we should additionally separate out a stable release separating unit tests for development software and install script.

Status: Done.

Priority: High

Estimated Effort: 5 days

Location: http://svn.egovmon.no/svn/eGovMon/trunk/INSTALL/

Work Package: WP4

Responsible: Christian Amble

4.11 PyTripleStore

Applicable: Accessibility

Short Description: Storing RDF information from the Sampler.

Changed from EIAO: Remove. (see section 4.22 eGovMonDB and 4.12 Sampling Algorithm)

Status: Not started.

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/DataWarehouse.

Responsible: -

4.12 Sampling Algorithm

Applicable: Accessibility

Short Description: Sampling the URLs from the URL repository, sending the URL to the WAM and writing it to the RDF repository. Automatic load balancing of the WAMs.

Changed from EIAO:
(1) Should not write to RDF repository but to eGovMonDB (see section 4.22 eGovMonDB)

Priority: High

Estimated Effort: 5 days

Status: Not started.

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/SamplingAlgorithm

Responsible: To be determined

4.13 SamplingServer

Applicable: Accessibility and Transparency

Short Description: Sending out site URLs to the Sampler / Transparency Metrics.

Changed from EIAO: Mostly unchanged. Minor Update that it need to work with transparency metrics in addition. (see section 4.23 Transparency Metrics). Because of this the name should change to avoid confusion.

Status: Not started.

Priority: Low

Estimated Effort: Minor

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/SamplingServer

Responsible: To be determined

4.14 Site URL Server

Applicable: Accessibility and Transparency

Short Description: Sending out site URLs to the Crawlers.

Changed from EIAO: Unchanged. Works well.

Status: -

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/SiteURLServer

Responsible: -

4.15 URL Repository

Applicable: Accessibility

Short Description: Strong URLs and some information.

Changed from EIAO: Removed and merged with eGovMonDB (see section 4.22 eGovMonDB).

Status: Done.

Priority: See section 4.22 eGovMonDB

Estimated Effort: See section 4.22 eGovMonDB

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/URLRepository.

Responsible: Morten Goodwin Olsen

4.16 ETL

Applicable: Accessibility

Short Description: Reading from RDF and writing to datastaging of DB. (see section 4.22 eGovMonDB)

Changed from EIAO: Removed.

Status: Not started.

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/DataWarehouse.

Responsible: -

4.17 HTML WAM

Applicable: Accessibility

Short Description: Evaluating accessibility of HTML with corresponding CSS.

Changed from EIAO:

(1) Allowed code extraction (DONE).

(2) Better reference to line and column (or similar reference). The current implementation does not work very well which results in wrongly presented line/column (and corresponding code extraction) in the eAccessibility Checker. (Not done)

(3) Possibility to return B-WAM results instead of EARL. This allows for quicker writing for the sampler. (Not done). RDF is removed in the rest of the tool and should only be optional here. [3]

(4) Move CSS evaluation from Batik-CSS to using the W3C CSS validator. This makes the CSS validation consistent between the W3C validator and HTML WAM.

(5) Update to latest version of W3C HTML validator.

3 Note that the eAccessibility Checker relies on EARL output. Thus, the eAccessibility Checker will need to be updated read B-WAM results. The eAccessibility Checker is not addressed in this deliverable, but is part of D6.4.1.

Status: Not started.

Priority: High

Estimated Effort: 10 days

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/WAMs/relaxed_wam. and http://svn.egovmon.no/svn/eGovMon/trunk/RelaxedWAM/ and http://svn.egovmon.no/svn/eGovMon/trunk/WAMs/wamlib/

Responsible: Anand B. Pillai, Christian Amble and Morten Goodwin Olsen

4.18 PDF WAM

Applicable: Accessibility

Short Description: Evaluating accessibility of PDF.

Changed from EIAO: Add test for checking if the PDF is scanned (mostly from producer tag - Minor).

Status: Ongoing.

Priority: Low

Estimated Effort: To be determined

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/WAMs/pdf-wam

Responsible: Anand B. Pillai

4.19 egovmontime

Applicable: Accessibility and Transparency

Short Description: Avoiding bug with This is to avoid the unknown error 514 http://www.tummy.com/journals/entries/jafo_20070110_154659

Changed from EIAO: Changed name from EIAOTime to eGovMonTime. Other than that, unchanged.

Status: Done.

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/egovmontime

Responsible: Morten Goodwin Olsen

4.20 Logit

Applicable: Accessibility and Transparency

Short Description: Common log functionality. Note that errors should never be concealed with this functionality. Complete trace-backs should always be presented when applicable.

Changed from EIAO: New.

Status: Done

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/trunk/logit

Responsible: Morten Goodwin Olsen

4.21 Unit Testing Framework

Applicable: Accessibility and Transparency

Short Description: Running all unit tests automatically each night and sends out e-mails. This makes it easy to detect any commits which break the code.

Changed from EIAO: New.

Status: Done

Priority: -

Estimated Effort: -

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/tests

Responsible: Morten Goodwin Olsen

4.22 eGovMonDB

Applicable: Accessibility and Transparency

Description:

A common database for both URLs (old URL repository), accessibility results (old DW) and transparency results. This is intended to be an in-production database and should not be used for efficient presentation of data in a user interface or similar. I.e. we do not at this point prioritise which data should be extracted quickly. This will come later when we have more knowledge of the Data.

The intention is to allow easy production of data with very little change to the existing software. Furthermore, the schema can be updated when we have more knowledge of the actual data and when we want something which can present quickly in e.g. a GUI.

The implementation should be as following in prioritised order:

1. Merging of existing databases.
Copy the existing tables from the URL repository (http://svn.egovmon.no/svn/eGovMon/trunk/URL_repository/tables.sql), existing tables in datastaging the datastaging-part (http://svn.egovmon.no/svn/eGovMon/trunk/Datawarehouse/DataStagingSchemaR20.sql) of the DW and EIAODW (http://svn.egovmon.no/svn/eGovMon/trunk/Datawarehouse/EIAODWR20Schema.sql)

This makes sure we do not loose any information we have available today.

2. Clean up of data

Remove all data which is overlapping and do a general clean up. Removed references which are not presented in any user interface (http://svn.egovmon.no/svn/eGovMon/UserInterface/) or report (http://svn.egovmon.no/svn/results/). Example of such is: barrier probability, key-use-scenario etc. Change references from EIAO to eGovMon.

3. Add transparency data

For transparency measurements, all data will be on web site level D3.2.1. This means few tables are needed for this. This should include:

- Time/date
- Testrun ID
- Web Site
- TestID (D3.2.1)
- Deterministic/Learning Algorithm/Expert Testing/User Survey/Automatic test confirmed by expert testing.
- Type of results. Results may consist of several parts including: boolean (pass/fail), count and percentage.
- Result.
- URL (if available)[4].
- Confidence (if available) E.g. probability of how sure we are the results is correct.
- Learning Algorithm used (if available) Naïve Bayes, Maximum Likelihood, or similar

If for performance reasons this is not sufficient, we could implement two tables; one for the individual scores and one for the detailed results.

For user surveys, external data sources, additional tables will be required. This will be added when we know more about which data should be included.

4. Add known materialised views

Add two additional tables for data we use often. This is similar to how matviews are implemented today.

Some of this data is already available in EIAODW datastaging.site and matviews.site.

The web site accessibility table should include:

- Time/date
- Testrun ID
- Web site
- Accessibility score (old CWAM).
- Smallest Accessibility Score (old min CWAM).
- Largest Accessibility Score (old max CWAM).
- Downloadcount
- Number of pages evaluated (old scenariocount)

The web page accessibility table should include:

- Time/date
- Testrun ID
- Web site
- URL
- Test ID
- Fail or Pass.

4 URL to the page where the content was found.

This latter, including an update from the sampler, should allow a fast extraction of data needed for the generated reports (instead of hours, seconds).

Changed from EIAO: New based on existing accessibility URL repository, accessibility data warehouse. Additionally, it contains transparency data.

Status: Ongoing

Priority: Very High

Estimated Effort: At most 5 days.

Work Package: WP4

Location: http://svn.egovmon.no/svn/eGovMon/experimental/eGovMonDB

Responsible: To be determined

4.23 Transparency Metrics

Applicable: Transparency

Short Description: Running the deterministic and heuristic (learning algorithm) tests. The deterministic tests will be described in D3.2.1 and learning algorithm tests will be described in D4.1.1.

The tests will be on web site and service level. There are no intentions of having tests on web page and element level, therefore there is no need for using relaxed. The implementation should follow the principal of any classification algorithm [Pattern].

The component should be a service similar to the web accessibility metric.

The outline of this can be seen in Figure 2.4.

Before startup manually collected pages and URLs should be added to training data. This should be files stored on the file system.

At startup, the stored flies should be read, the needed features extracted and the training should be kept in memory. If training turns out to be a performance issue we can add training batches, nightly builds or similar.

When a request is issued, the transparency metric downloads the needed URLs. If this turns out to be a performance issue, we can attempt at reusing some of the downloaded files from the crawler. Note that the download is not expected to be a bottle neck and it is easier to download the needed files.

For each page the needed features are extracted. Note that the features are extracted only once per page. All tests, deterministic and heuristic are applied to the features.

When the tests results are available, the data is committed to the database.

The work flow can be seen in figure Figure 3.2.

Changed from EIAO: New.

Status: Ongoing

Priority: Very High

Estimated Effort: -

Work Package: WP4

Location: Initial http://svn.egovmon.no/svn/eGovMon/experimental/FindPostListe

Responsible: Morten Goodwin Olsen and To be determined

5 Anatomy figure

In figure Figure 5.1 we present an anatomy graph of tasks to be developed. Anatomy chart is a development methodology where the intention is that the development is structured into anatoms where each anatom represent a task and a responsibility.

The anatomy graph is structured as following:

- Each anatom reflects an independent implementation task. Note that this is not the same as modules or components.
- The anatoms are coloured according to development status.
- An anatom is more important the further down in the graph the anatom is located. By developing the anatoms from bottom up in the graph, we ensure that the development is prioritised according to importance.
- Anatoms dependant on each other are connected with lines. Anatoms which can be developed independently are not connected.
- Each anatom has one or more responsible persons.

Details of each task, including responsibilities, are described in section 4.

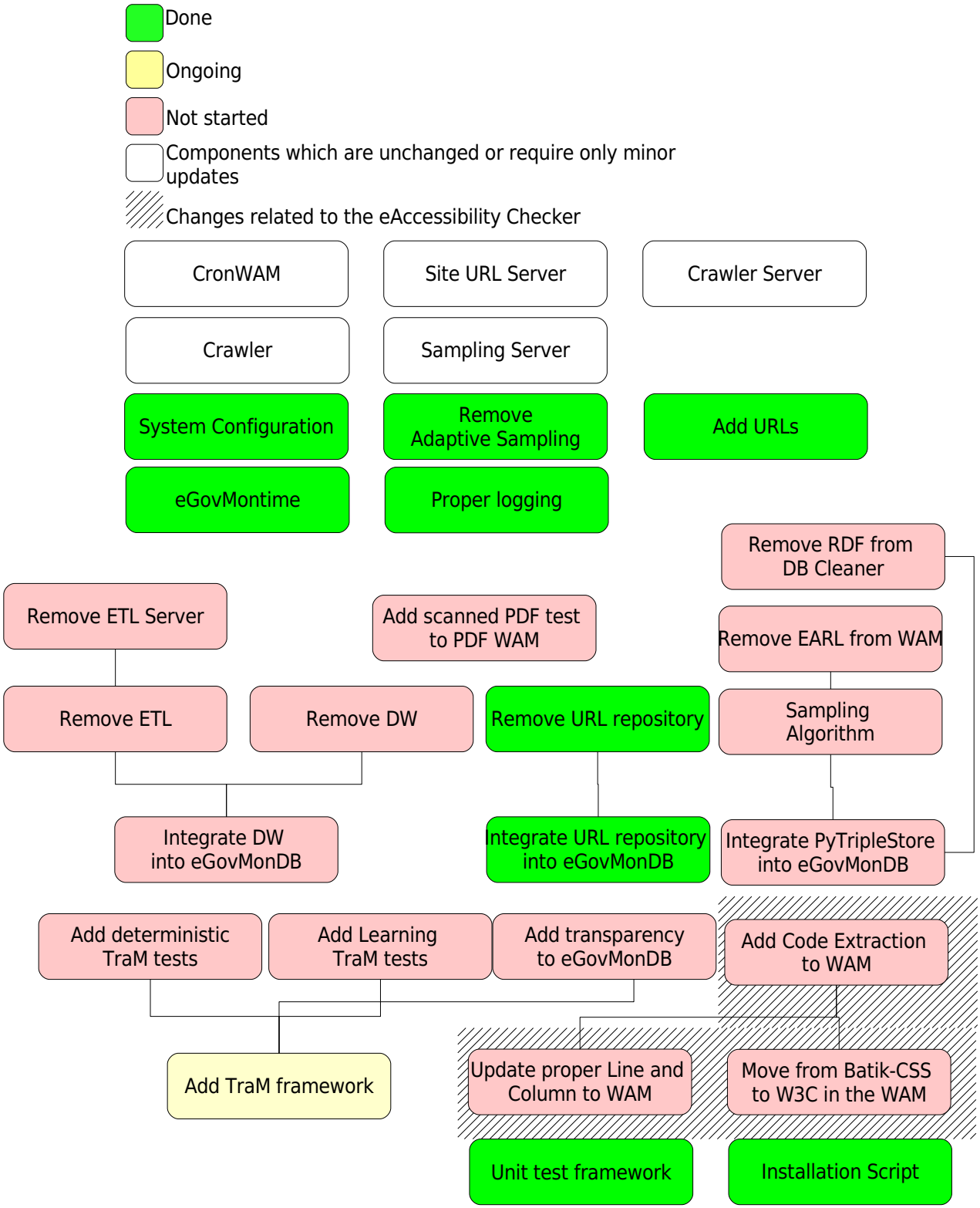

Figure 5.1: Anatomy Figure

References

D5.2.2.1-2: Nils Ulltveit-Moe, Morten Goodwin Olsen, Anand B. Pillai, Terje Gjøsæter, EIAO documented source code, 2008, http://www.eiao.net/publications/EIAO_D5.2.2.1-2.pdf

Pattern: Anil K. Jain, Robert P.W. Duin and Jianchang Mao, Statistical Pattern Recognition: A Review, 2000, Statistical Pattern Recognition: A Review