

**Conceptual model for EIAO DW, R2**

Appendix A to Deliverable Number: 6.1.1.1-2



Version: 1.1

Date: 2007-01-06

Author: Torben Bach Pedersen and Christian Thomsen

Dissemination Level: Project

Status: FINAL

License:

This work is licensed under the Creative Commons Attribution-ShareAlike License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/2.5/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.

This document consists of 27 pages plus this cover

## **Abstract**

In this document, the conceptual model for the data warehouse EIAO DW is described in details. EIAO DW is a data warehouse that holds results from the European Internet Accessibility Observatory (EIAO) project. These results are mainly on the accessibility to disabled users of web resources that are automatically crawled and evaluated by other parts of the EIAO Observatory. However, the results also include statistics about technologies used by and linked to from the tested web resources.

Version Control

<i>Version</i>	<i>Status</i>	<i>Date</i>	<i>Change</i>	<i>Author</i>
0.1	DRAFT	2006-10-20	First version	Torben Bach Pedersen and Christian Thomsen
1.0	FINAL	2006-10-30	Comments from Terje Gjøsæter, Nils Ulltveit-Moe, Annika Nietzio and Morten Goodwin Olsen added	Christian Thomsen
1.1	FINAL	2007-01-06	Corrections and improvements based on comments from Jens Frøkjær and Tom Oddershede	Christian Thomsen

## Table of Contents

1 Introduction.....	3
1.1 Brief project description.....	3
1.2 Scope of this document .....	3
1.3 Related work and readers instructions.....	3
2 Conceptual model for EIAO DW.....	3

# 1 Introduction

## 1.1 Brief project description

The overall objective of the project is to contribute to better eAccessibility for all citizens and to increase use of standards for on line resources. The project will be carried out as part of the Web Accessibility Benchmarking cluster (WAB) together with the projects SupportEAM and BenToWeb.

The project will establish the technical basis for a possible European Internet Accessibility Observatory (EIAO) consisting of:

- A set of web accessibility metrics.
- An Internet robot for automatically and frequent collecting data on web accessibility and deviations from web standards (the WAI guidelines)
- A data warehouse providing on line access to collected accessibility data.

## 1.2 Scope of this document

This document covers the conceptual model for release 2.0 of the data warehouse in the EIAO project, the EIAO DW. The entire schema design follows the classic approach with conceptual, logical, and physical models as described, e.g, in R. Elmasri and S. Navathe: "Fundamentals of Database Systems", Addison Wesley, 2003. The logical and physical models are to be found in the documentation part of D6.3.3.1-2.

It should be noted that this is document version 1.1. Compared to version 1.0 of the document, the conceptual model described in the present document has been updated to correct found problems and to improve and simplify the model based on obtained experiences.

## 1.3 Related work and readers instructions

This document is related to the following documents:

- D6.1.1.1-2 is the functional specification for the EIAO DW release 2.
- D5.1.1.1-2 is the functional specification for the EIAO crawler and describes (together with D3.2.1) what data and how that data is collected.
- D3.2.1 "2<sup>nd</sup> version of ROBACC WAMs" describes the Web Accessibility Metrics (WAMs) that generate the data to store in the data warehouse.
- D6.6.1.2.2 "Collecting and maintaining a URL directory for the project" describes the URL directory used in the project.

# 2 Conceptual model for EIAO DW

The conceptual model is shown in Figure 2.1. The notation is based on UML 2.0 (see [www.uml.org](http://www.uml.org)). The model illustrates classes for which information should be stored in EIAO DW. In the model, attributes of the classes are shown as well as associations between different kinds of classes.

The dotted ellipses are strictly speaking not part of the conceptual model, but are included for ease. They show how the classes are grouped together as *dimensions* in the logical model (see D6.3.3.1-2). In ellipses, the hierarchy within each dimension is represented such that higher levels in the hierarchy are drawn above lower levels in the hierarchy. For example, in the Date dimension it is seen that days roll up into months.

In the following, we describe each of the shown classes, its attributes, and its associations to other classes. Whenever we refer to a class, its name will be capitalised. When we refer to an entity represented by an instance of a class, the entity name will not be capitalised.

To avoid repetitions, we describe the ID attributes that all the classes have once and for all here. A class named *Xyz* will have an ID attribute named *XyzID*. For each ID attribute, it holds that this attribute is an integer used for unique identification of instances of classes. A specific value for an ID attribute of a class *X* does not carry any special meaning apart from being a unique number among the ID attributes for instances of *X*. The reason for using these ID attributes is that they are *stable* in the sense that we can edit the values that carry meaning without changing the key (namely the ID attribute). Further, as the ID attributes are integers, they take up less space in the fact table and can be more efficiently handled.

Two classes (that become the fact tables in the logical model) are, however, exceptions to what is described above. The *TestResult* and *TechnologyFinding* classes do not have IDs. Instances of these class are special in the sense that their purpose is to “glue” other class instances together by means of associations, e.g. to represent that a specific result (i.e. *Result* instance) is associated to a specific test subject (i.e. *Subject* instance).

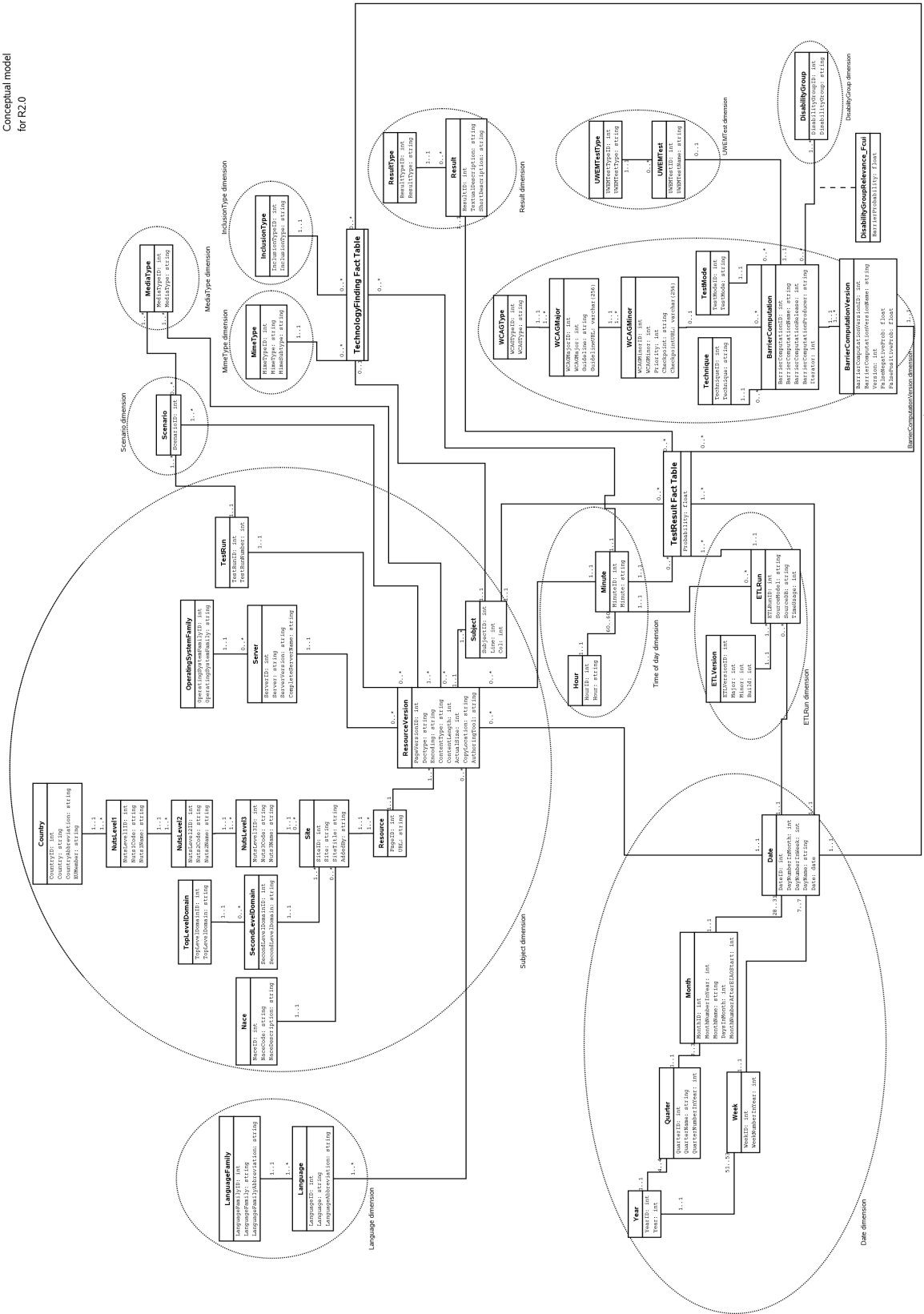


Figure 2.1 Conceptual model

### **TestResult Fact Table**

This is the class used to represent a single test of single subject with a single version of a specific barrier computation being used. For example, there will be an instance of TestResult for each time a specific `img` element on a given web page is tested with a version of a barrier computation that deals with `img` elements. The TestResult is associated with other classes. Specifically, TestResult is associated with the classes Subject, Result, and BarrierComputationVersion, describing what was tested, what the result was and what barrier computation version was used, respectively. TestResult is also associated with Minute and Day such that it can be represented when the test took place. Finally, TestResult is associated to ETLRun such that it can be represented in which ETL run a specific test result was created.

TestResult has one attribute.

<b>Name:</b> Probability
<b>Type:</b> Float
<b>Description:</b> Describes the probability with which a specific test failed. This attribute is needed for heuristic tests. For definite tests this probability is always 1 (see D3.2.1).
<b>Values:</b> Float values in the interval [0,1]

### **Subject**

The Subject class represents the tested subjects. The term *subject* is here used in the same sense as is D3.2.1. Thus, a subject is a CSS or (X)HTML element that is relevant for a specific test.

Subject is associated to PageVersion. The association shows that a specific subject belongs to a specific version of a specific page. Other page versions may very well have identical subjects as well as a page version may (and probably will) have many subjects. However, one subject belongs to one and only one page version.

The attributes of Subject are described below.

<b>Name:</b> Line
<b>Type:</b> Integer
<b>Description:</b> Describes the line number that the considered subject begins at in the HTML source. The first line has number 1. For special things that the WAMs have not extracted directly from the HTML (such as things specified in the HTTP header instead of in the HTML), line 0 is used
<b>Values:</b> Non-negative integers

<b>Name:</b> Col
<b>Type:</b> Integer
<b>Description:</b> Describes the column number where the subject begins. This column number is on the line given by Line. The first column on a line has number 1. For special things that the WAMs have not extracted directly from the HTML (such as things specified

<b>Name:</b> Col
in the HTTP header instead of in the HTML), column 0 is used
<b>Values:</b> Non-negative integers

### **ResourceVersion**

The ResourceVersion class represents specific versions of web resources (such as HTML and CSS resources). For example, a page like the front page of <http://news.bbc.co.uk/> is often updated and therefore different versions of this page will be considered in different surveys. Thus, ResourceVersion models the dynamic aspects of web resources (the static aspects are modeled by Resource described later).

A resource version is associated to a date and a timestamp. These represent when the resource version was last modified. A resource version has exactly one associated resource. A resource version also has exactly one associated test run. Thus, each time a resource is assessed, we consider a new version of the resource (which may or may not be identical to the previous version assessed). A resource version is associated to a number of media types that represent which media types the resource version contains explicit CSS rules for. A resource version is also associated with a number of scenarios. This is to represent in which scenarios the resource version was used. It is possible for, for example, a CSS file to be used in many different scenarios. Further, a resource version has exactly one server which represents the server hosting the resource. A resource version can have many associated subjects. A resource version is also associated with a number of languages to represent the natural languages used in the resource.

The attributes of PageVersion are described below.

<b>Name:</b> Doctype
<b>Type:</b> String
<b>Description:</b> Gives the name of the used HTML version for HTML pages and “CSS” for CSS pages
<b>Values:</b> The DOCTYPE from the HTML document or “CSS”

<b>Name:</b> Encoding
<b>Type:</b> String
<b>Description:</b> Gives the encoding used for the page version
<b>Values:</b> Legal encoding names such as utf-8, iso-8859-1, etc. The preferred alias (see <a href="http://www.iana.org/assignments/character-sets">http://www.iana.org/assignments/character-sets</a> ) from IANA is always used

<b>Name:</b> ContentType
<b>Type:</b> String
<b>Description:</b> Gives the text from the Content-Type field of the HTTP header
<b>Values:</b> Legal content types

<b>Name:</b> ContentLength
<b>Type:</b> Integer
<b>Description:</b> Gives the content length of the page version in bytes. The value originates from the Content-Length field in the HTTP header
<b>Values:</b> Positive integers.

<b>Name:</b> ActualSize
<b>Type:</b> Integer
<b>Description:</b> Gives the content length of the page version in bytes. The value originates from WAMs and represents the actual size of the resource version, not the (possibly incorrect or missing) content length reported by the HTTP header
<b>Values:</b> Positive integers.

<b>Name:</b> CopyLocation
<b>Type:</b> String
<b>Description:</b> Gives a URL to where the local copy used in the assessment is stored
<b>Values:</b> Valid URLs to cached web pages

### **TestRun**

The TestRun class represents the surveys performed by the EIAO project. One test run can consider many web sites with many page versions. A test run can consider many resource versions and many scenarios. This is represented by associations with the ResourceVersion and Scenario classes. TestRun has one attribute.

<b>Name:</b> TestRunNumber
<b>Type:</b> Integer
<b>Description:</b> Test run sequence number. These are numbers are assigned by the crawler.
<b>Values:</b> Positive integers

### **Server**

This class represents the different kinds of server software (like Apache, IIS, etc.) used to host the web resources. One server product can host many page versions. A server product is associated with the operating system it runs on such that Apache for Windows is different from Apache for Linux. Thus, the Server class is associated to the OperatingSystemFamily class.

The attributes of Server are described below.

<b>Name:</b> Server
<b>Type:</b> String
<b>Description:</b> This is the name of the server product

<b>Name:</b> Server
<b>Values:</b> The identified server product names, e.g. "Apache". Further, the value "Unknown" must be possible

<b>Name:</b> ServerVersion
<b>Type:</b> String
<b>Description:</b> The version of the considered server product
<b>Values:</b> Version numbers, e.g. 1.2.31. Further, the value "Unknown" must be possible

<b>Name:</b> CompleteServerName
<b>Type:</b> String
<b>Description:</b> The complete identification string from the server product
<b>Values:</b> For example "Apache/1.3.26 (Unix)". Further, the value "Unknown" must be possible

### **OperatingSystemFamily**

The OperatingSystemFamily class represents the different generic families of operating systems that the server products are running on. For example the Windows family covers both Windows XP and Windows 2000 as members. The Unix family covers all the different flavors of Unix, including Linux, FreeBSD, MacOS X etc.

OperatingSystemFamily has the attribute OperatingSystemFamily.

<b>Name:</b> OperatingSystemFamily
<b>Type:</b> String
<b>Description:</b> The families of operating systems.
<b>Values:</b> "Windows", "Unix", "Other", "Unknown"

### **Resource**

The Resource class represents web resources. However, the changing parts (such as the size of the content) of resources are represented by ResourceVersion. Resource only represents the static parts. Thus a resource can have several associated resource versions. A resource is considered as belonging to one site.

Resource has the attribute URL.

<b>Name:</b> URL
<b>Type:</b> String
<b>Description:</b> The complete URL for the represented resource
<b>Values:</b> Valid URLs

### **Site**

The Site class represents different web sites including their web addresses. A site is associated with a second level domain, a NUTS code (level 3) and a NACE category.

Site has the attributes Site, SiteTitle, and AddedBy.

<b>Name:</b> Site
<b>Type:</b> String
<b>Description:</b> The web address for the site (i.e. the host and domain name, but no path to a specific document).
<b>Values:</b> For example <a href="http://bundesbank.de">bundesbank.de</a> , <a href="http://www.dr.dk">www.dr.dk</a> , and <a href="http://www.cs.aau.dk">www.cs.aau.dk</a>

<b>Name:</b> SiteTitle
<b>Type:</b> String
<b>Description:</b> The <eiao:title> from the RDF.
<b>Values:</b> Titles for the web sites

<b>Name:</b> AddedBy
<b>Type:</b> String
<b>Description:</b> A string telling who added the URL to the observatory. If the URL was added because the crawler followed a link, the value will be "EIAO Observatory".
<b>Values:</b> Strings

### **NUTSLevel3**

The NUTSLevel3 class represents NUTS codes at level 3, i.e., the lowest level. The class is associated with the class NUTSLevel2 for representing NUTS codes at the next level. NUTSLevel3 has two attributes.

<b>Name:</b> Nuts3Code
<b>Type:</b> String
<b>Description:</b> The NUTS level 3 code
<b>Values:</b> Strings of length 5, e.g. "AT111", or NULL when not applicable.

<b>Name:</b> Nuts3Name
<b>Type:</b> String
<b>Description:</b> The name of the NUTS3 region represented
<b>Values:</b> Strings

### **NUTSLevel2**

The NUTSLevel2 class represents NUTS codes at level 2, i.e., the middle level. The class is associated with the class NUTSLevel1 for representing NUTS codes at the next level and with NUTSLevel3 for representing NUTS codes at the lower level. NUTSLevel3 has two attributes.

<b>Name:</b> Nuts2Code
<b>Type:</b> String
<b>Description:</b> The NUTS level 2 code
<b>Values:</b> Strings of length 4, e.g., "AT11" or NULL when not applicable

<b>Name:</b> Nuts2Name
<b>Type:</b> String
<b>Description:</b> The name of the NUTS2 region represented
<b>Values:</b> Strings

### **NUTSLevel1**

The NUTSLevel2 class represents NUTS codes at level 2, i.e., the highest level. The class is associated with the class NUTSLevel2 for representing NUTS codes at the middle level. Also, the NUTSLevel1 class is associated to the Country class. NUTSLevel1 has two attributes.

<b>Name:</b> Nuts1Code
<b>Type:</b> String
<b>Description:</b> The NUTS level 1 code
<b>Values:</b> Strings of length 3, e.g. "AT1" or NULL when not applicable

<b>Name:</b> Nuts1Name
<b>Type:</b> String
<b>Description:</b> The name of the NUTS1 region represented
<b>Values:</b> Strings

### **Country**

The class Country is used to represent countries. Country has three attributes.

<b>Name:</b> Country
<b>Type:</b> String
<b>Description:</b> The official short name of the represented country in English as given in ISO3166
<b>Values:</b> Strings. For example "Denmark" but not the long official name "Kingdom of

<b>Name:</b> Country
Denmark". "Unknown" is also allowed

<b>Name:</b> CountryAbbreviation
<b>Type:</b> String
<b>Description:</b> The two letter abbreviation for the represented country as given in ISO3166 and in the NUTS codes. Note, however, that ISO3166 uses "GB" for United Kingdom and the NUTS codes use "UK". In the EIAO DW the abbreviation "UK" is used.
<b>Values:</b> Strings of length 2. For example "DK", "DE", and "NO"

<b>Name:</b> EUStatus
<b>Type:</b> String
<b>Description:</b> Shows whether the represented country is a member of the EU, an applicant country or outside the EU
<b>Values:</b> "EU member", "EFTA member", "Candidate country", "Outside EU", and "Unknown" (the latter to be used when the country is also "Unknown")

### **SecondLevelDomain**

The class SecondLevelDomain represents second level domains. A second-level domain can have several associated sites, but only one top level domain. Note that for "short" addresses such as [relaxml.com](http://relaxml.com), both the SecondLevelDomain and Site instances represent the entire domain name. For [www.relaxml.com](http://www.relaxml.com), the Site class represents everything of this address (including the www part) whereas the SecondLevelDomain class only represents the relaxml.com part.

SecondLevelDomain has one attribute.

<b>Name:</b> SecondLevelDomain
<b>Type:</b> String
<b>Description:</b> The second level domain name represented
<b>Values:</b> Legal second level domain names

### **TopLevelDomain**

Top level domains are represented by the class TopLevelDomain. A top level domain can have several second level domains. It has one attribute.

<b>Name:</b> TopLevelDomain
<b>Type:</b> String
<b>Description:</b> The top level domain name represented
<b>Values:</b> Legal top level domain names

### **Nace**

The class Nace is used to represent a NACE category that the owner of a site belongs to. The Nace class is associated with the Site class.

Nace has two attributes apart from the ID attribute.

<b>Name:</b> NaceCode
<b>Type:</b> String
<b>Description:</b> A NACE code, not including the textual description.
<b>Values:</b> Legal NACE codes

<b>Name:</b> NaceDescription
<b>Type:</b> String
<b>Description:</b> The textual descriptions for the individual NACE codes.
<b>Values:</b> Descriptions from the definition of NACE codes

### **Language**

The Language class represents the different languages used on assessed web pages. A language may be spoken differently in different countries. For example, a language could be “German as spoken in Germany” and another “German as spoken in Austria”. Therefore, the Language class is associated with the LanguageFamily class that represent the “generic” languages, i.e. “German” in the previous example. A language belongs to exactly one language family. Note that sometimes it is only possible to detect that a resource version uses “German” as language and not if this is “German as in Austria”. For that reason, Language can also represent the generic language without any country information.

The attributes of Language are described below.

<b>Name:</b> Language
<b>Type:</b> String
<b>Description:</b> Represents the language as they are spoken in certain countries
<b>Values:</b> The name of the language followed by parenthesis giving the country or the name of the language without any country shown. For example “German (Germany)” and “German”. Further “Unknown” must be possible

<b>Name:</b> LanguageAbbreviation
<b>Type:</b> String
<b>Description:</b> This is the abbreviation used for languages when indicating a language for a web page. For example “en” means English and “en_US” means English as spoken in the US. See RFC3066, ISO639, ISO3166
<b>Values:</b> Legal values as defined by RFC3066

### **LanguageFamily**

The LanguageFamily class represents the language used when the “sub language” is ignored. For example for “en\_US” which means “English as spoken in the US”, the language family is “English”. A language family has at least one member, but can have many.

The attributes of LanguageFamily are given below.

<b>Name:</b> LanguageFamily
<b>Type:</b> String
<b>Description:</b> The name of the language family as defined by ISO639
<b>Values:</b> Names defined by ISO639 and “Unknown”

<b>Name:</b> LanguageFamilyAbbreviation
<b>Type:</b> String
<b>Description:</b> The short code assigned by ISO639
<b>Values:</b> Legal codes defined by ISO639

### **Date**

The class Date is used for representing the day part of a specific date. The values will be added to the data warehouse on an on demand basis. The class Date is participating to the Date dimension in the logical model. Classes to represent dates (or a date dimension in the logical model) are used instead of using attributes of type Date. This is beneficial for many reasons. Since this class is the first of these classes, the advantages will be briefly described here. One reason is that it is possible to represent the value “Unknown”. The integer IDs are also useful when precomputed aggregates are to be handled (e.g., the results for a specific year). Further, the use of classes (a dimension in the logical model) gives the possibility to represent domain specific knowledge that otherwise would have to be handled in the reporting layer. An example of the latter is the attribute MonthNumberAfterEIAOStart that for a given month gives the month number relatively to when the Observatory was launched. Instead of using calendar logic in the reporting layer, it is then immediately possible to see this. However, to provide a much flexibility as possible, an attribute of type Date is also added to the Date class.

Date is associated with Month and Week. It is also associated with TestResult such that the date for an assessment can be tracked. Further it has an association to ResourceVersion representing that a resource version has been modified at a specific date. Thus the Date dimension is used as an *outrigger* from the Subject dimension in the logical dimensional model. In the same way Date is associated with the ETLRun class (to represent when a specific ETL run was started) and the Date dimension is, thus, also used as an *outrigger* from the ETLRun dimension in the dimensional model. Finally, Date is associated with TechnologyFinding. This is to represent when the use of a specific technology was found by the EIAO crawler.

Date has four attributes.

<b>Name:</b> DayNumberInMonth
<b>Type:</b> Integer
<b>Description:</b> The day number in the month
<b>Values:</b> 1 to 31

<b>Name:</b> DayName
<b>Type:</b> String
<b>Description:</b> The English name of the week day
<b>Values:</b> Monday, Tuesday, ..., Sunday

<b>Name:</b> DayNumberInWeek
<b>Type:</b> Integer
<b>Description:</b> The day number in the week. 1 is Monday, 2 is Tuesday, ..., 7 is Sunday as specified by ISO8601
<b>Values:</b> 1 to 7

<b>Name:</b> Date
<b>Type:</b> Date
<b>Description:</b> Holds the date represented by this Date instance. Note that this also includes the associated month and so on.
<b>Values:</b> Legal dates, NULL to represent the unknown date

### **Month**

The class Month represents months. A month belongs to exactly one quarter and has between 28 and 31 days. The values will be added to the data warehouse on an on demand basis. The attributes of Month are described below.

<b>Name:</b> MonthNumberInYear
<b>Type:</b> Integer
<b>Description:</b> The month number in the year. 1 is January, 2 is February, ...,12 is December
<b>Values:</b> 1 to 12

<b>Name:</b> MonthName
<b>Type:</b> String
<b>Description:</b> The English name of the month
<b>Values:</b> January, February, ..., December

<b>Name:</b> DaysInMonth
<b>Type:</b> Integer
<b>Description:</b> The number of days in the month
<b>Values:</b> 28 to 31

<b>Name:</b> MonthNumberAfterEIAOStart
<b>Type:</b> Integer
<b>Description:</b> The month number after the Observatory was launched. First month is January 2006 and has number 0
<b>Values:</b> Integers. (Note that the integers do not have to be positive. If a checked web page was created before the Observatory was launched, the value will be negative in the representation of the relevant date)

### Quarter

The Quarter class represents calendar quarters. A quarter has 3 months, but belongs to exactly one year. The values will be added to the data warehouse on an on demand basis.

The attributes of Quarter are described below.

<b>Name:</b> QuarterName
<b>Type:</b> String
<b>Description:</b> A textual name for the quarter
<b>Values:</b> "Q1 yyyy", "Q2 yyyy", "Q3 yyyy", and "Q4 yyyy" where yyyy represents the year 1970 or a year after year 1970.

<b>Name:</b> QuarterNumberInYear
<b>Type:</b> Integer
<b>Description:</b> The quarter number in the year
<b>Values:</b> 1 to 4

### Year

The class Year represents a calendar year. The values will be added to the data warehouse on an on demand basis. A year has four quarters and a number of weeks. Except the ID attribute, Year only has one attribute.

<b>Name:</b> Year
<b>Type:</b> Integer
<b>Description:</b> The represented year
<b>Values:</b> Integers greater than or equal to 1970 to be able to represent dates since the UNIX standard epoch. The value indicates directly which year is represented. That means

<b>Name:</b> Year
-------------------

that the integer 2005 represents year 2005
--

### **Week**

Calendar weeks are represented by the class Week. A week has a week number which is relative to exactly one year (even though a week may start in one year and end in another). The values will be added to the data warehouse on an on demand basis.

Week has one attribute.

<b>Name:</b> WeekNumber
-------------------------

<b>Type:</b> Integer
----------------------

<b>Description:</b> The calendar week number
--

<b>Values:</b> 1 to 53
------------------------

### **Minute**

The class Minute represents minutes. Minute is associated with Hour and ResourceVersion (the latter to represent the last modification time for a page version). Minute is also associated with the ETLRun class to represent when an ETL run was started. Thus, the Time dimension is also used as an outrigger from the Subject and ETLRun dimensions in the dimensional model. Further, the Minute class is associated with TestResult to be able to represent the time where an assessment took place and with TechnologyFinding to represent the time where a specific technology was found by the EIAO crawler. Note that all represented times are to be interpreted as UTC times. Minute has one attribute.

<b>Name:</b> Minute
---------------------

<b>Type:</b> Integer
----------------------

<b>Description:</b> Represents the minute number in the associated hour
---

<b>Values:</b> 0 to 59
------------------------

### **Hour**

Hour represents hours. Note that all represented times are to be interpreted as UTC times. Hour has one attribute.

<b>Name:</b> Hour
-------------------

<b>Type:</b> Integer
----------------------

<b>Description:</b> Represents the hour number in the day
---

<b>Values:</b> 0 to 23
------------------------

### **ETLRun**

The ETLRun class represents specific runs of the ETL software. Each time the ETL is started for a given DW, an instance of ETLRun is created. This makes it easier to track the origin of data and to do statistics about the amount of added data and time usage in each ETL

run. The ETL program used has a version and exactly one ETL version is used by a specific ETL run. This means that ETLRun is associated with the ETLVersion class.

ETLRun has three attributes.

<b>Name:</b> SourceModel
<b>Type:</b> String
<b>Description:</b> The name of the RDF model used to extract assessment results from
<b>Values:</b> RDF model names

<b>Name:</b> SourceDB
<b>Type:</b> String
<b>Description:</b> Name of the MySQL database used for storing the source RDF data
<b>Values:</b> Database names

<b>Name:</b> TimeUsage
<b>Type:</b> Integer
<b>Description:</b> The number of seconds spent by the represented ETL run
<b>Values:</b> Non-negative integers

### **ETLVersion**

ETLVersion represents different versions of the ETL tool. This is useful if, for example, a bug is found in a specific version and all possible affected data should be located. ETLVersion has three attributes.

<b>Name:</b> Major
<b>Type:</b> Integer
<b>Description:</b> The major version number. For release 2 of EIAO DW, this is 2
<b>Values:</b> Non-negative integers

<b>Name:</b> Minor
<b>Type:</b> Integer
<b>Description:</b> The minor version number. The first ETL version for EIAO DW release 2 has minor version number 0. If updates are performed, this number will be incremented
<b>Values:</b> Non-negative integers

<b>Name:</b> Build
<b>Type:</b> Integer

<b>Name:</b> Build
<b>Description:</b> Used when smaller changes are made to the code
<b>Values:</b> Non-negative integers

### **BarrierComputationVersion**

The BarrierComputationVersion class is used to represent a specific version of an implementation of a barrier computation (represented by the BarrierComputation class described below) as defined in D3.2.1. BarrierComputationVersion is associated with BarrierComputation. BarrierComputationVersion has the following attributes.

<b>Name:</b> BarrierComputationVersionName
<b>Type:</b> String
<b>Description:</b> Gives the complete name for the represented barrier computation version (including the version number). Described in D3.2.1
<b>Values:</b> For example “UWEM.B.10.3.2.2.HTML.DEF.2.1”

<b>Name:</b> Version
<b>Type:</b> Integer
<b>Description:</b> Gives the version number for the represented barrier computation version Described in D3.2.1
<b>Values:</b> Positive integers

<b>Name:</b> FalseNegativeProb
<b>Type:</b> Float
<b>Description:</b> Gives the probability for a false negative as specified in D3.2.1
<b>Values:</b> Floats in [0,1]

<b>Name:</b> FalsePositiveProb
<b>Type:</b> Float
<b>Description:</b> Gives the probability for a false positive as specified in D3.2.1
<b>Values:</b> Floats in [0,1]

### **BarrierComputation**

The BarrierComputation class is used to represent a barrier computation as described in D3.2.1 (but disregarding the specific implementation version). The BarrierComputation class is associated to the UWEMTest class. This association represents which UWEM test the represented barrier computation is dealing with. BarrierComputation is also associated to the WCAGMinor class to represent which WCAG 1.0 checkpoint it is checking for. Further, it has associations to the TestMode and Technique classes to represent the mode of the barrier computation and what (CSS or HTML) it considers.

The BarrierComputation class is also associated to the DisabilityGroup class. This is to be able to represent how disability groups are influenced if a given barrier computation test does not pass. Therefore there is an association class for this association. This association class has the attribute BarrierProbability. That attribute is used to represent UWEM  $F_{up}$  values (i.e., values that show how severe a failed test is for a disability group).

BarrierComputation has the following attributes.

<b>Name:</b> BarrierComputationName
<b>Type:</b> String
<b>Description:</b> The name of the barrier computation
<b>Values:</b> Names as defined in D3.2.1

<b>Name:</b> BarrierComputationRelease
<b>Type:</b> Integer
<b>Description:</b> The release number of the specifications. 2 for the specifications in D3.2.1 (i.e., 2 for release 2 of the EIAO Observatory)
<b>Values:</b> Positive integers

<b>Name:</b> BarrierComputationProducer
<b>Type:</b> String
<b>Description:</b> Name of the producer of the barrier computation. Described in D3.2.1
<b>Values:</b> Strings

<b>Name:</b> Iterator
<b>Type:</b> String
<b>Description:</b> Gives an iterator for WAMs that deal with the same WCAG checkpoint
<b>Values:</b> Strings consisting of three digits as described in D3.2.1

### **Technique**

The Technique class represents techniques covered by barrier computation versions. Apart from the ID attribute, Technique only has one attribute.

<b>Name:</b> Technique
<b>Type:</b> String
<b>Description:</b> The name of the represented technique.
<b>Values:</b> "HTTP", "HTML", and "CSS"

### **TestMode**

The TestMode class represents test modes used by barrier computation versions. Apart from the ID attribute, TestMode only has one attribute.

<b>Name:</b> TestMode
<b>Type:</b> String
<b>Description:</b> The name of the represented test mode.
<b>Values:</b> “Definite”, “Heuristic”, and “Auxiliary”

### **WCAGMinor**

The WCAGMinor class represents a WCAG 1.0 checkpoint. A checkpoint is associated with (i.e. belongs to) a guideline (represented by the WCAGMajor class). The WCAGMinor class has the following attributes.

<b>Name:</b> WCAGMinor
<b>Type:</b> Integer
<b>Description:</b> The minor number of a WCAG checkpoint
<b>Values:</b> Integers. 0 is used to show that no applicable WCAG 1.0 checkpoint exists

<b>Name:</b> Priority
<b>Type:</b> Integer
<b>Description:</b> The WCAG priority for the represented checkpoint
<b>Values:</b> 1, 2, or 3 (and 0 if the WCAGMinor attribute is 0)

<b>Name:</b> Checkpoint
<b>Type:</b> String
<b>Description:</b> The WCAG checkpoint (i.e., the text describing what to do)
<b>Values:</b> Strings

<b>Name:</b> CheckpointURL
<b>Type:</b> String
<b>Description:</b> A URL to where the checkpoint is defined
<b>Values:</b> URLs

### **WCAGMajor**

The WCAGMajor class represents a WCAG 1.0 guideline. A guideline has a number of associated checkpoints (represented by the WCAGMinor class). WCAGMajor is associated with WCAGType to represent which type the WCAG checkpoint has (currently WCAG 1.0 or “None”). The WCAGMajor class has two attributes.

<b>Name:</b> WCAGMajor
<b>Type:</b> Integer
<b>Description:</b> The number of the represented WCAG guideline

<b>Name:</b> WCAGMajor
<b>Values:</b> Integers. (0 is used to show that no applicable WCAG 1.0 guideline exists)

<b>Name:</b> Guideline
<b>Type:</b> String
<b>Description:</b> The WCAG guideline (i.e., text describing what to do)
<b>Values:</b> Strings

<b>Name:</b> GuidelineURL
<b>Type:</b> String
<b>Description:</b> A URL to where the guideline is defined
<b>Values:</b> URLs

### **WCAGType**

WCAGVersion represents the version of the WCAG guidelines that a checkpoint belongs to. The WCAGType class has one attribute.

<b>Name:</b> WCAGType
<b>Type:</b> String
<b>Description:</b> The version of the WCAG guidelines
<b>Values:</b> Currently “WCAG 1.0” and “None” are the only allowed values

### **DisabilityGroup**

The DisabilityGroup class represents disability groups (e.g., blind people, deaf people, and people with dyslexia) for the EIAO observatory. DisabilityGroup is associated with the BarrierComputation class. This association has an association class with the attribute DisabilityGroupRelevance\_Fpu. This attribute holds the probability for that a subject introduces a barrier for the relevant disability group if the barrier computation fails when testing the subject. This will be used when computing aggregates by means of C-WAMs (see D3.2.1).

DisabilityGroup has only one attribute apart from the ID attribute.

<b>Name:</b> DisabilityGroup
<b>Type:</b> String
<b>Description:</b> The name of the user group
<b>Values:</b> Strings

### **UWEMTest**

The UWEMTest class represents UWEM tests. The class is associated with UWEMTestType that represents which types of documents the test handles. Further, UWEMTest is associated with the BarrierComputation class to represent which barrier

computation that incorporates the represented UWEM test. UWEMTest only has one attribute apart from the ID.

<b>Name:</b> UWEMTest
<b>Type:</b> String
<b>Description:</b> The name of the represented UWEM test
<b>Values:</b> Strings

### **UWEMTestType**

The UWEMTestType class represents types of UWEM tests represented by the UWEMTest class. UWEMTestType only has one attribute apart from the ID.

<b>Name:</b> UWEMTestType
<b>Type:</b> String
<b>Description:</b> The type of a represented UWEM test
<b>Values:</b> "HTML", "CSS", "External object"

### **Result**

The Result class is used to represent the results of an EIAO assessment. A result belongs to a specific type of results (see below). There are many results describing the fails that resulted in negative outcomes of the application of the barrier computation versions, but only one pass result which is used for all positive outcomes.

The attributes of Result are described below.

<b>Name:</b> TextualDescription
<b>Type:</b> String
<b>Description:</b> Textual description for a test
<b>Values:</b> Strings

<b>Name:</b> ShortDescription
<b>Type:</b> String
<b>Description:</b> A textual description of a test, but shorter than the corresponding TextualDescription
<b>Values:</b> Strings

### **ResultType**

The ResultType class is used to represent general types of results (tests can be passed or failed). Apart from the ID attribute, ResultType only has one attribute.

<b>Name:</b> ResultType
<b>Type:</b> String

<b>Name:</b> ResultType
<b>Description:</b> Describes the type of result
<b>Values:</b> "pass", "fail", "cannotTell" as in EARL. In R2, only these results types are used in the EIAO observatory. In later releases, other result types may be used in the entire observatory.

### **MediaType**

The class MediaType is used to represent the medias that can be supported explicitly by CSS. Thus the MediaType class is used to represent those media types and is associated with the Scenario class to represent which media types a specific page scenario found explicit support for. Similarly, MediaType is associated with ResourceVersion to represent which media types a specific resource versions supports explicitly. This is a many-to-many association as one media type may be supported in many page scenarios and one page scenario may support many media types. MediaType has one attribute.

<b>Name:</b> MediaType
<b>Type:</b> String
<b>Description:</b> Describes the type of media
<b>Values:</b> The media types defined in <a href="http://www.w3.org/TR/REC-CSS2/media.html">www.w3.org/TR/REC-CSS2/media.html</a> and the value "No media types explicitly stated".

### **Scenario**

The Scenario class is used to represent a scenario. It participates in a many-many relationship with the ResourceVersion class (as previously described this association represents which media types are supported within a single resource version). It is also associated with the TestRun class to represent which test run a specific scenario is used in. Finally, the Scenario class is associated with the MediaType class. This association is used to represent which media types are explicitly supported in a specific scenario (i.e. in the (X)HTML and the possible CSS files). Scenario has no attributes apart from the ID attribute.

### **TechnologyFinding Fact Table**

This is the class used to represent the finding of a single subject that uses (i.e., holds or links to) an object using a specific technology. For example, there will be an instance of TechnmologyFinding for each time a specific `img` element for a JPEG image is found on a given web page.

The TechnologyFinding class is associated with Subject (to represent the where the technology finding was done), MimeType (to represent the used technology), InclusionType (to represent how the technology is used.), Minute, and Date (the two latter to represent when a specific technology was found by the EIAO crawler).

Since TechnologyFinding has no measures, it is a so-called *factless fact table* (see R. Kimball: "The Data Warehouse Toolkit", Wiley, 2002) used to track events.

### **MimeType**

The MimeType class represents the different MIME types found in resource versions. It has two attributes.

<b>Name:</b> MimeType
<b>Type:</b> String
<b>Description:</b> The MIME type as reported by the web server
<b>Values:</b> Legal MIME types such as “text” and “image”

<b>Name:</b> MimeSubtype
<b>Type:</b> String
<b>Description:</b> The MIME subtype as reported by the web server
<b>Values:</b> Legal MIME subtypes as “xml” and “jpeg”

### **InclusionType**

The InclusionType class is used to represent how a given technology/object is included in a resource. It has one attribute to show this.

<b>Name:</b> InclusionType
<b>Type:</b> String
<b>Description:</b> Shows how a given technology/object is included in a resource
<b>Values:</b> “Internally linked” for a link to a resource on the same site, “Externally linked” for link to a resource on another site, “Embedded” for a resource embedded in the HTML