Melka Kunture Virtual Museum

Report on the design & construction of the Website & WebGIS

Wim Feringa
Barend Köbben
International Institute for Geo-Information Science and Earth Observation (ITC)

Culture 2000 project "From the Past to the Present in Ethiopian Prehistory. An Interactive Museum for the Archaeological Park of the Early Palaeolithic site of Melka Kunture" |
Agreement n. 2006 – 1033/001 -001 CLT CA12.

Education and Culture
Culture 2000
European Cummunity Culture 2000

Università di Roma "La Sapienza"

CGT
Centro di GeoTecnologie
Centro di Geotecnologie

BORDEAUX 1
Université de Bordeaux

ITC
International Institute for Geo-Information Science and Earth Observation

Addis Ababa University

Document history

Version	Author	Content	Date
1.0	Wim Feringa	Main structure	13-04-2007
1.1	Wim Feringa	Details	06-07-2007
1.1.1	Barend Köbben	WebGIS parts & additional details	15-06-2007
1.2	Wim Feringa	Add timeline specs	28-08-2007
1.3	Wim Feringa	final version site design & construction	13-09-2007
1.4	Barend Köbben	final version WebGIS & CMS	05-10-2007

Table of Contents

1. REQUIREMENTS ANALYSIS

Goal:

30 years of research on the Melka Kunture site (Ethiopia) resulted in huge amount of data.
The EU funded a project on publication of studies on the site of Melka Kunture and on the construction of a museum on-site.

A second EU project was granted on making the data on studies and excavations available for a wide public of Scientists and more general public.
The project is concerned with promoting of the Melka Kunture archaeological site in Ethiopia. The University of Roma and the University of Bordeaux will organize a archaeological museum in Ethiopa on the archaeological site. ITC's task is to develop a website and a geo-website in order to spread the knowledge achieved by the archaeologists and encourage tourism. To achieve this ITC will
a) perform conceptual sturcturing and organization of the web GIS and Internet website.
B) develop of an internet website and a web GIS for the distribution of the cultural heritage relevant to the Melka Kunture site and Archaeological Park.

Website requirements:

Functional requirements

- database driven for the Museum pages and the Scientists pages
- webgis for the scientists site
- updates via a Content Management System

Non-functional requirements

- Accessability
  In the EC adopted the "Web Content Accessibility Guidelines 1.0", but it is not yet compulsory.
  - follow guidelines, if possible if not too time consuming
    - http://www.w3.org/TR/WAI-WEBCONTENT/
  - focus on the option for the user to change the font size
- Audit and control
  The main participants of the project are responsible for maintenance of the database and website Responsibilities need to be described by these main participants.
  ITC only designs and produces the first version of the website but is NOT responsible for further updates and/or maintenance.
- Availability
  - Full time service
    - Expert hosting advised
    - Down time has to be limited
    - Unlimited time length
    - Hosting of the website
      - Taken care of by: Università di Roma "La Sapienza", contact: Prof. Marcello Piperno
    - Domain registry
      - Preferred under the "eu" domain.
      - Taken care of by: Università di Roma "La Sapienza", contact: Prof. Marcello Piperno
- Copyright
  - Ownership
    - Statement appears on the website on the page "Terms of use" on the "contact" page.
  - Copyright statements on website
    - Statement appears on the website on the bottom of every page.
- Dependency on other parties
  Delivery of material through FTP and e-mail.
  - Delivery of material
    - Images/video/audio/animation/VR/3D

        - Formats
            - Images: JPEG, TIFF, GIF, PSD, PNG
            - Video: MP3, AVI
            - Audio: WAV, MP3
            - Animation: SWF, Flash, QuickTime, AVI
            - VR: Quicktime, SWF, Flash
            - 3D: Google SketchUp, ArcGis
        - Resolution
            - Images: preferred 300 dpi on final size
            - Video: size minimal 300x400 pixels.
            - Audio
                - Voice: 8 bit, mono, 11.025 Hz sample rate
                - Music: 16 bit, stereo, 44.100 Hz sample rate
            - Animation: size minimal 300x400 pixels
            - VR: size minimal 300x400 pixels
        - Length
            - Video: try to limit to 60 seconds
            - Audio:
                - Voice: limit to 30 seconds
                - Music: depending on track
            - Animation: depends on issue
    - Text
        - Length indicated per element if restrictions are applicable
        - ASCII, MsWord *.doc, RTF
- Development constraints
    - Time path - planning
        - Start - June 12: design
        - June 13 - June 22: commenting design + delivery material
        - June 25 - July 12: production of version 1.0
        - July 15 - August 3: testing
        - August 6 - September 14: debugging and correcting
        - September 18: Final version of CD-Rom and web
- Documentation
  All design files and specifications for the website are included in this report.
  A user guide is available in the "museum" pages, where the user is introduced to the "virtual museum".
- Efficiency
  Efficient use of server space and bandwith imply minimization of filesizes and textlength. Especially for the public site the general rule can be applied that only few take the effort to scroll on longer pages, so keep text short. Images and other media need to be minimal in file size and length to avoid long downloads that make users to quit before all is visible. It should be avoided that (other than standard used) extra software such as pdf-reader and ShockWave (plug-ins) have to be installed on the public site.
    - File sizes as small as possible
    - Text length, avoid scrolling, offer longer text as external files (PDF) to be downloaded
    - Standard Plug-ins as the Acrobat Reader and ShockWave (flash)
    - Meta information and "Alt" tags give users more information on the use of images
- Extensibility
    - The public website is designed as final product, meaning that no automatic interface changes (eg. Menu extension on adding a new entry in the database or addition of new pages) will be included in the maintenance functionality. The Expert website will be updated automatically on addingor changing elements in the database.
    - For the Expert website it will be possible to add new elements.
- Maintainability
  A limited group of persons from the partners have direct access to the databas, others will be able to access the database through a CMS. All is username/pwd protected.
    - The content of the website is updatable, existing material can be renewed and edited.
    - The database can be changed and updated by making use of the Data Base Management System or FTP

  - The site requires maintenance, this will be done by Daniele Simoncini, Centro di GeoTecnologie, San Giovanni Valdarno, Italy.
- Network bandwidth
  - Depending on expected number of requests, but the expectations are not too high.
- Performance constraints
  - Internet accessible
  - General public site also on CD-Rom
    - No installers for a DataBase on the user's PC
    - An exact copy of the Public website is produced at the moment of the official release (September 20 2007) exept for the Scientists pages.
    - The copy runs as an off-line version on CD-Rom, without the option of using the database.
- Performance / Response time
  - high
- Platform compatibility
  - Multi platform
- Browser compatibility
  - Multi browser support
  - Recent version and one version down
  - Controlled by Cascading Style Sheets
- Quality of content.
  All delivered content needs to be checked on all kind of errors and needs to be corrected before uploading. Responsible are those who deliver the data.
  - In the Planning for production a two week period is reserved for testing the website. This testing includes a detailed check on the content.
  - The testers are the main data deliverants.
- Reliability.
  - Allowed failures: there is no indication on this, but since it is not a commercial website a certain number of failures (connection to the DB, performance of the Map Server …) is acceptable, if the problems are not consistent.
  - Allowed unavailability: The website should in principle always be available, in case of unavailability problems need to be solved within hours. The Host needs to have procedures that guarantee the "up" stage of the website.
- Resource constraints
  As understood this website alsoneeds to be very good accessible in Ethiopia and neighboring countries. This implies that the systems on the users side not always are of latest state of the art. A distinction is made between the Public and the Expert site, where the Experts will be better equipped than the general public.
  The public website (the Museum) makes an extensive use of images, which makes that a lot of data (images) need to be downloaded. The different branches of the main website all have different elements. The museum itself shows many artefacts, all of them are images. Image quality of the shown objects has to be such that details can be distinguished.
  - The probable small network bandwidth is not taken into concideration
  - The monitor resolution: the website is designed for 1024 x 768 pixels
- Robustness
  Systems can always fail, in those cases where the failure is caused by design and/or implementation mistakes feedback is important. The hosting company plays an important role in this, failures mostly will be on the server side, since the website is once developed and build and will not be subject of change.
  - Feedback form for failures will go to one of the partners, they have to contact the hosting company
  - The sites maintenance will be done by Daniele Simoncini, Centro di GeoTecnologie, San Giovanni Valdarno, Italy. He is the most obvious person to take care of this also.
- Scalability
  Expansion is normal for a website, the design and production of this website however did not focus on a probable growth.The project is fixed.
- Security
  The database needs to be protected for unwanted access but also for changes in the structure that has a negative influence on performance.

- Direct access to database and Site by one person
- Updating content through a CMS (Content Management System) for the partners

- Software, tools, standards etc.
  - Public website: make use of accepted standards
    - XHtml
    - Css
    - Asp
    - Plug-in (Acrobat Reader, ShockWave)
  - Experts
    - XHtml
    - Css
    - Asp
    - Webgis
    - Plug-in (Acrobat Reader, ShockWave
- Support issues
  Not discussed
- Usability by target user
  - Profile
    - Who is the user
      - Scientists
      - Public
        - Students (Ethiopian)
        - hobbyist
        - Tourists
    - Background
      - Scientists
        - High educated
        - Visit the website with scientific interest
      - Public
        - Ethiopian student
          - Visit the website as task
        - Hobbyist
          - Highly interested in the topic
        - Tourist
          - As preparation for a visit
          - Has visited the museum before and likes to know more
    - Knowledge level
      - Scientists
      - Student can be of all levels
      - Hobbyist level will be various
      - For the Tourists this is not defined
  - Needs
    - Scientists
      - Detailed information on excavation
      - Reports
      - Research publications
      - Location and time
    - Public
      - General info on general history and excavations
  - Context visit
    - Scientists
      - Part of research
      - Adding information or thoughts
    - Public
      - By accident, via search engine
      - On purpose, planning or deepening
      - Forced as part of education
      -

User requirements

There were no discussions with potential users.

Usability requirements , system acceptability

- Learnability
  User interface for the Public site is designed in such a way that it is fully intuitive, for the Expert site some help and instruction could be required.
- Efficiency of use
  Fast downloads, waiting times are minimal
- Memorability
  Not discussed on importancy of re-visit, probably not too many things will change on the Public site, the Expert user will find the way back
- Few and non-catastrophic errors, error handling / help support
  System should not be too many times "unavailable", specially not for the Expert users. Important is a good error handling. Monitoring the site on daily is important. The sites maintenance will be done by Daniele Simoncini, Centro di GeoTecnologie, San Giovanni Valdarno, Italy. He is the most obvious person to take care of this also.
- Subjective satisfaction
  In this project this is not budgeted. No user enquiry is performed to get this information.

Content

- Public
  - Museum content
- Scientists
  - Excavation objects
  - Meta information
  - Scientific notes
  - Locational information

2. CONCEPTUAL DESIGN

Design objectives

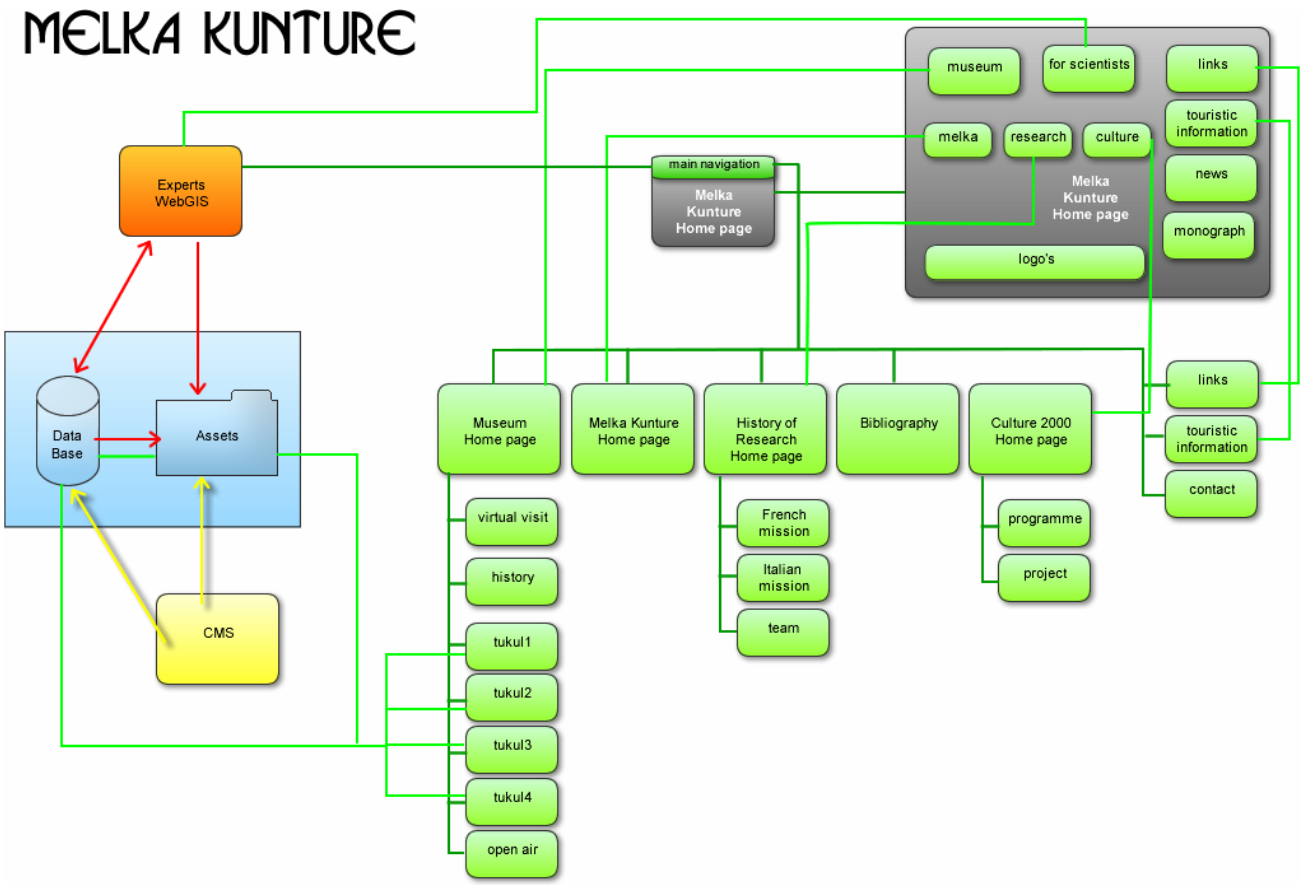
- ability to create an off-line CD-Rom version of the public site
- ability for lower profile systems (internet connection, processor) to view the websites
- ability for visual less able persons to view content by larger font sizes
    - controll by two versions of css
- ability for "screen readers" to access correct structure
    - structure of css important
        - always navigation on top
        - avoid frames
        - use standard tags for hierarchy (H1 – H7, P..)
- Base graphics look and feel on the Melka Kunture folder
    - general public
        - metaphor is the Museum
        - buildings and site can be entered by clicking
        - objects can be queried by clicking
        - user is visually guided through the museum by using maps of museum and buildings
        - a swap from general to Scientists (and vice versa) is possible, taking the selected query along
    - Scientists
        - map as interface
        - query the data by making use of forms

Functional design

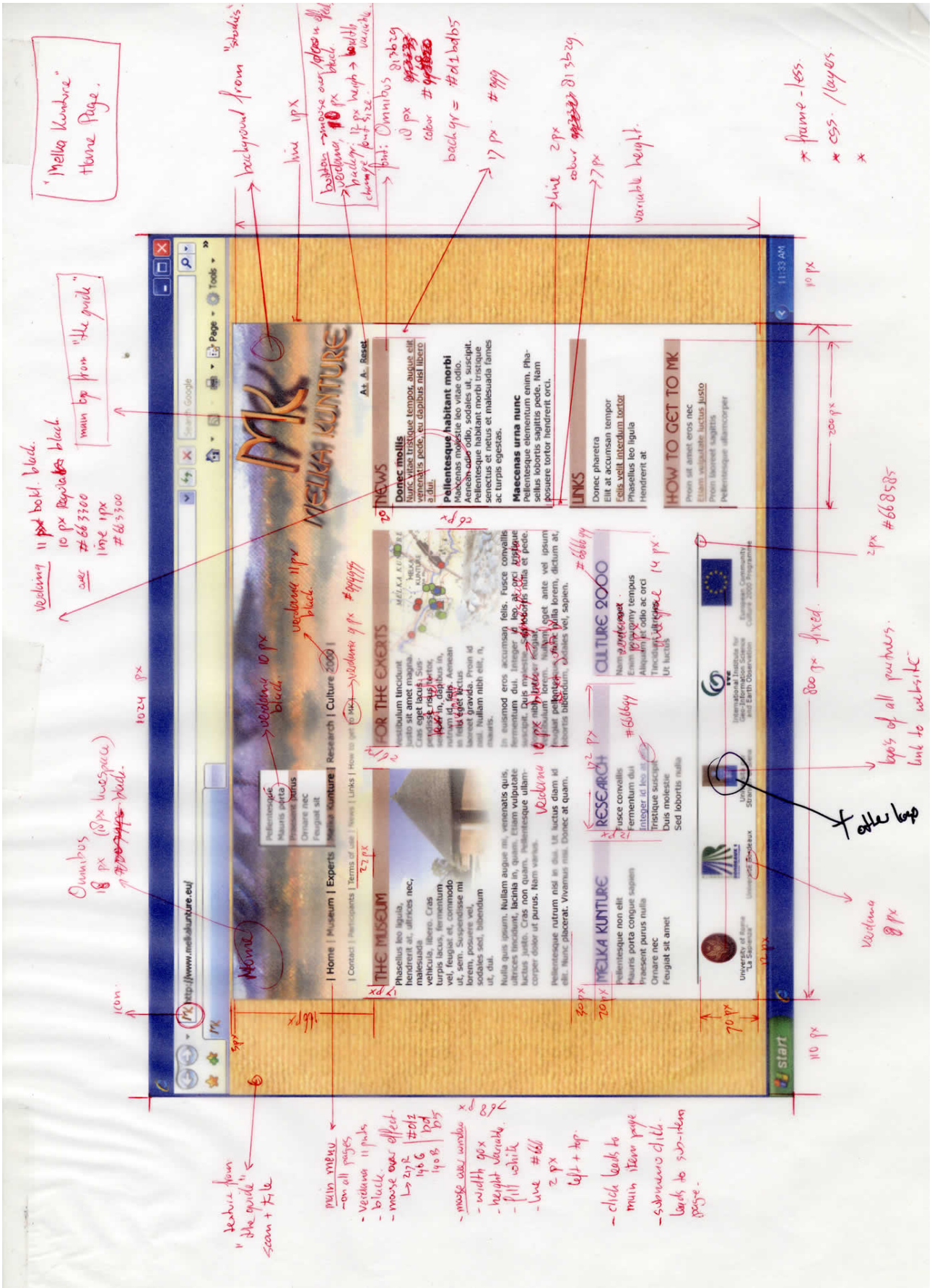
- Website
    - Home page
        - Main URL
        - Museum
            - Short abstract of the Museum pages
            - Link to the Museum main page
        - For Scientists
            - Short abstract of the For Scientists pages
            - Link to the For Scientists main page
        - News
            - Links to news page (updated regular)
        - Monograph
            - Link to the Monograph page for download of pdf files of the separate chapters
        - Touristic information
            - Links to the Touristic infora
        - Melka Kunture
            - Short abstract of the Melka Kunture page
            - Link to the Melka Kunture page
        - History of Research
            - Short abstract of the History of Research pages
            - Link to the History of Research main page
        - Culture 2000 project
            - Short abstract of the Culture 2000 project pages
            - Link to the Culture 2000 project main page
        - Logo's participants
            - Link to resp. homepages
    - Melka Kunture
        - One page with the history of Melka Kunture
    - History of Research
        - General History
        - Italian History
        - French History

        - Listing of participants
    - Museum
        - Entry page
        - Virtual museum
            - Explain relation to real museum
            - Explain how to use the virtual museum
        - Tukul 1-4
            - Navigation per Tukul
            - Make use of a mpa of the Tukul
            - Click on panels and showcases give detailed view on panel
            - Click on item on panel give more detailed view on item
            - Timeline shows position of listed panel in time
    - For Scientists
        - Start with an overview map of all Melka Kunture sites
            - Get attributes of:
                - Survey points
                - Geological points
                - Main archeological sites
            - Get maps and Data of the different sites
                - Garba IVC, IVD, IVE, Ik
                - Karre IM
                - Gombore IB, II OAM
                - Balchit Atelier Gaggia
                - Simbiro III5
            - Get maps of the site:
                - Gombore II Butchering site, II1, II3, II4, II5, Iy
                - Simbiro III1, III2a, III2b
                - Garba I, IIIB, IIIC, XII
    - Bibliography
        - Listing of publications
    - Culture 2000
        - Introduction page
        - Culture 2000 programme
        - Culture 2000 project
    - Contact
        - Page with a listing of the Participants and Terms of use for the website
    - Links
        - Page with links to web sites with related content
    - Touristic information
        - Page with adresses and maps
- Content Management System
    - Separate URL
    - Username/PWD protected
    - Fixed procedure
        - Database needs to be filled correctly and complete
        - Data entry according specification
        - Images, audio, video, animation, illustrations in correct formats, dimensions and resolution
- Data Base Management System
        - On the server

- Site map/flow



3. GRAPHIC DESIGN



Font

Per page the specificatons for the fonts will be given. On all pages after the "home" page only the changed styles and new styles are listed.

Home page

Top Title left:	Omnibus 18 pixels / 18 pixels linespace / black
Titles:	Omnibus 18 pixels / 18 pixels linespace / #813b29 / background #d1bdb5
Titles:	Omnibus 18 pixels / 18 pixels linespace / #666699 / background #e5dde3
Main menu:	Verdana 11 pixels / black
	Mouse over background #d1bdb5
Main menu pull-down:	Verdana 10 pixels / black
	Mouse over background #d1bdb5
Sub menu:	Verdana 9 pixels / #999999
	Mouse over effect underline
Text size button:	Verdana 10 pixels / black
Text head:	Verdana 11 pixels / bold / black
Text:	Verdana 10 pixels / regular / black / linespace 14 pixels / justify width
	Mouse over – underline / #663300
	Mouse over – underline / #666699
Text logo's:	Verdana 8 pixels / linespace 9 pixels / align centre

Museum Home page

Top Title left:	Omnibus 18 pixels / 18 pixels linespace / white

Museum Detail page

Top Title left:	Omnibus 18 pixels / 18 pixels linespace / white
Poster title:	Omnibus 13 pixels / 13 pixels linespace / / #813b29 / background #d1bdb5
Poster heading:	Verdana 12 pixels / bold / black
Poster text:	Goudy / 11 pixels / linespace 12 pixels / black
Detailed heading:	Goudy / 11 pixels / bold / linespace 12 pixels / black
Detailed text:	Goudy / 11 pixels / linespace 12 pixels / black / justify width

Melka Kunture page

Titles:	Omnibus 18 pixels / 18 pixels linespace / #666699 / background #c7c4d3

Research page

Culture 2000 page

News page

Links page

How to get to Melka Kunture

Partners and sponsors

Contact

Terms of use

Backgrounds

The backgrounds refer to the main background on alle pages and the top-background

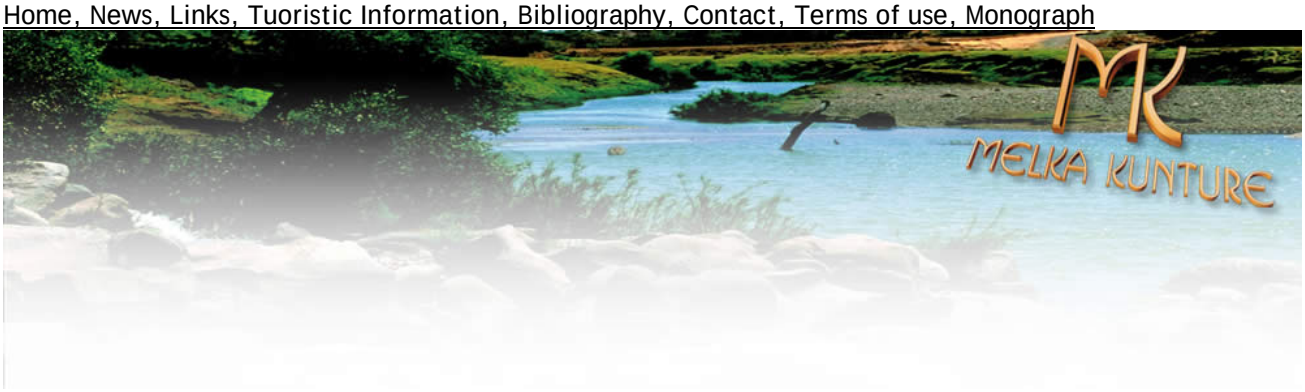
Main background

file name "baseBackground" 193 x 192 pixels

This is a background tile, which will be repeated such that the whole page is filled. This background is extracted from "the Guide" and the folder.

Top Backgrounds

Home, News, Links, Tuoristic Information, Bibliography, Contact, Terms of use, Monograph



Filename: homeBack, 998 x 275 pixels

This background is extracted from "the photo "Awash-River"

Museum

Filename: museumBack, 998 x 275 pixels

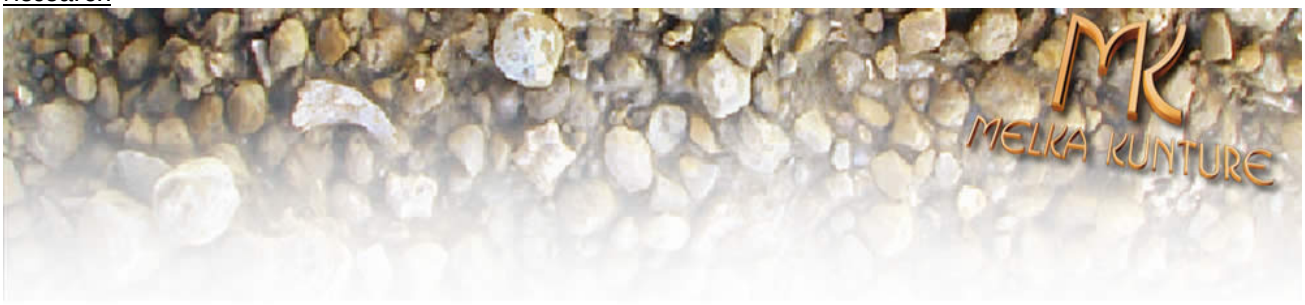
This background is taken from "ArticoloArcheomatica2007" photo1

Melka Kunture

Filename: melkaBack, 998 x 275 pixels

This background is an extract from "The Guide"

Research



Filename: researchBack, 998 x 275 pixels

This background is taken from a vertical photo from the "Butchering-site"

Culture 2000

Filename: cultureBack, 998 x 275 pixels

This background is a compilation of elements of the official Culture 2000 Home page and the Melka Kunture logo.
Some extra image editing is done on the lower part of this background, in order to lengthen the background to fit nicely.
It is advised to ask permission to the EC for use.

Dimensions

All measurements are in pixels.

Screen size
The website is designed for a screen size of: 1024 x 768 pixels.
In practice a smaller area will be available (browser-window consumes space).
Effictive size: 1017 x 628 pixels.
To make the website also accessible for users with smaller screensizes or for users that do not use "full screen" a smaller width of 998 pixels is chosen to show the content.
The background of the "unused" space is filled with a backgroundimage.

Content window
The content window has a fixed width of 800 pixels, and is centered on screen.
At the top a space of 3 pixels and at the bottom a space of 12 pixels is used to set the window "free" in the browser window.
A thin grey line of 1 pixel is used as frame for this window.

Top area (title, menu)
The total heigth of this is 166 pixels.
The baseline of the "Main menu" is 125 pixels from the top of the content window.
The height of the "pull-down" window depends on the number of subpages.
The baseline of the "Secondary menu" is 149 pixels from the top of the content window.
The "page title" is positioned on 122 pixels from the left en 32 pixels from the top of the content window.
The Melka Kunture logo is positioned on the right hand top, 7 pixels from the top and 672 pixels from the left.
The logo itself measures 238 x 132 pixels
The buttongroup for text-resising are positioned 825 pixels from the left and 156 pixels from the top.
The group itself measures 12 x 69 pixels

Content columns
Width of the columns vary, depending on lay-out per theme, but are evenly spread over available total width
Space between the columns is 26 pixels
Space between column and content-frame is 17 pixels
Space between vertical positioned columns (bottom – top) is 30 pixels
Title-backgrounds, height 20 pixels, length depending on used column width
Vertical lines (if used) positioned on left side of column, thickness is 2 pixels, length depending on length theme, colour #813b29

Logo's
(only on Home page)
Visual same volume
Total height (from bottom content line) is 70 pixels
Horizontal line at top of logo's, thickness 2 pixels, #668585, width same as columns

4. FUNCTIONAL DESIGN

In this chapter the functionality, how the user can interact with the website, is described.
The main functionlity is described for the Home page. All other pages follow the same procedure. Only there where a page has additional functionality, it will be described.

	Page	Element	Functionality	Remarks	Method
OVERALL		Content area	Fixed width of 998 pixels, centralised Layers Menu and page elements rebuilt on every page, no fixed content that will be repeated on all pages. Variable height, depending on content		CSS
HOME	Home	Page title	Display		Hard Code
		Fontsize	Buttons A give large font A gives small font	Has also effect on menu, Will not be implemented on Scientist and Museum pages because of the complex structure of these pages	JavaScript, CSS
		Main menu	On mouse-over brown area under text On mouse-click a pull-down (pop-up) menu opens		Css Layers visibility
		Main menu pop-up	On mouse-over brown area under text On mouse-click the required page will open and the pop-up menu will close		Css
		Secondary menu	On mouse-over a line will appear under text On mouse-click the required page will open		Css
		Active links in text	On mouse-over text will be coloured and underlined On mouse-click will open required page		Css
		Logo	hyperlinked		Static image
MUSEUM	Museum home page	Heading of structures	Hyperlinked, will change colour on mouse-over and be underlined. Will link to the Museum detail page of the required structure		Css
		3D image of museum	Clickable map, click on one of the buildings will lead to the Museum detail page of the required structure and give an update on the time-line	Has to be clear that the image map is clickable	Html
	Museum detail page	Location map lower left corner	Clickable map, the active structure is of darker colour and is not clickable. Other structures are clickable. When clicking on one of the other structures, this will lead to the required detail page.	Has to be clear that the image map is clickable	Html

		The name of the TUKUL will change to the name of the active structure	
	Plan of the active structure	Interactive map of the structure, the panels as well as display-tables or objects will be clickable. On mouse-over will give a change in colour and on click will open the required information in the poster-panel on the right. On click will send request through an *.asp page to quiry database and extract correct content to load in the poster-panel	Asp
	Poster-panel	This will be an extract of the corresponding poster in the real museum. Content comes from the database. All elements in the poster panel are clickable. On click will send request through an *.asp page to quiry database and extract correct content to load in the detail-panel on the right.	asp
	Detail-panel	Text, images as result of the request on the poster-panel are shown. A link to the Experts website is possible from this panel. A click will open in a new browserwindow the experts website with detailed information on the clicked element	asp
	Time-line	Shows the time span in which the active panel can be positioned. Time line is updated on every request.	Flash, asp

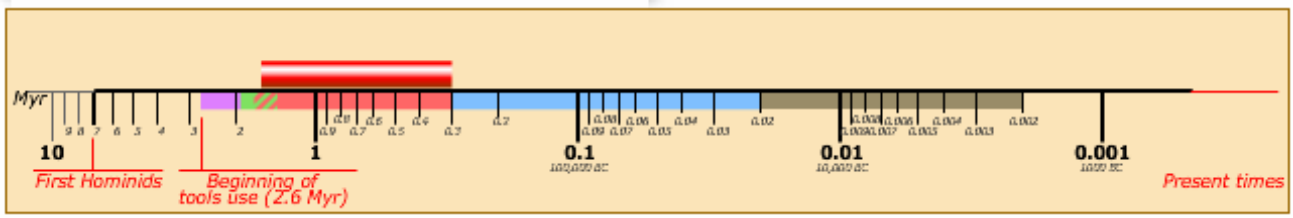
Flash Timeline

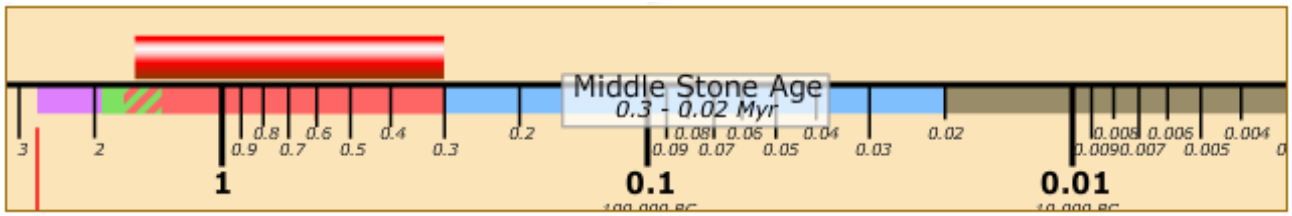
Introduction
The timeline is positioned on the museum pages, at the bottom. This timeline positions the content of a certain panel in time, so the user has a good idea of the age of the content.

The functionality

- Time line is shown on start in the panel, showing the main time periods
- Because of density of information the user can get extra information as pop-up window about the different time periods
- The user can zoom-in approx 200% by clicking on the timeline, another click will zoom-out to 100%
- When clicking on a panel, the dates for the "time-bar" will be passed to the flash file, and the time bar appears (if a Panel has a time indicator)

Design





Specifications

Flash Script

```
//****************************************
//  Wim Feringa, August 2007  //
//  www.itc.n                  //
//  feringa@itc.nl          //
//****************************************


//Get input from external file
InputOld = Old;                    //Activate the Old variable that is send by the html movie embed string
InputYoung = Young;                //Activate the Young variable that is send by the html movie embed string

//if panel has no chronology, the bar should NOT be visible
//the default value of the variables is always 0 when a tukul has no data
if (InputOld == 0) {
        setProperty(Mc_Chronology, _visible, 0);
} else {
        setProperty(Mc_Chronology, _visible, 1);
}
```

```
//on stage are three empty movieclips, for indication of the main control points
//get the x positions of the main calculation points
var nu = getProperty(Mc_nu, _x);                    //this movieclip is located on the point 0.001 (1000 bc), standing for the current time
var toen = getProperty(Mc_toen, _x);                //this movie is located on the point 10, oldest time in this scale
var NulPunt = getProperty(Mc_NulPunt, _x);          //since I work with log-calculations, the position 1 (giving 0) is my main calculation point

//calculate the main pixeldostance for a full-log number period
var Periode = (nu - toen)/4;                        //four main persiods, calculating amount of pixels per Log period, which have same length

//calculate the startingpoint of a certain chronology (oldest time)
var MyrOld = NulPunt - ((Math.log(InputOld) * Math.LOG10E) * Periode);

//calculate the endpoint of a certain chronology (youngest time)
var MyrYoung = NulPunt - ((Math.log(InputYoung) * Math.LOG10E) * Periode);

//Calculate length of the period
var lengte = (MyrYoung - MyrOld);

//set the x position and the length of Mc_Chronology
setProperty("Mc_Chronology", _x, MyrOld);
setProperty("Mc_Chronology", _width, lengte);
```

HTML script

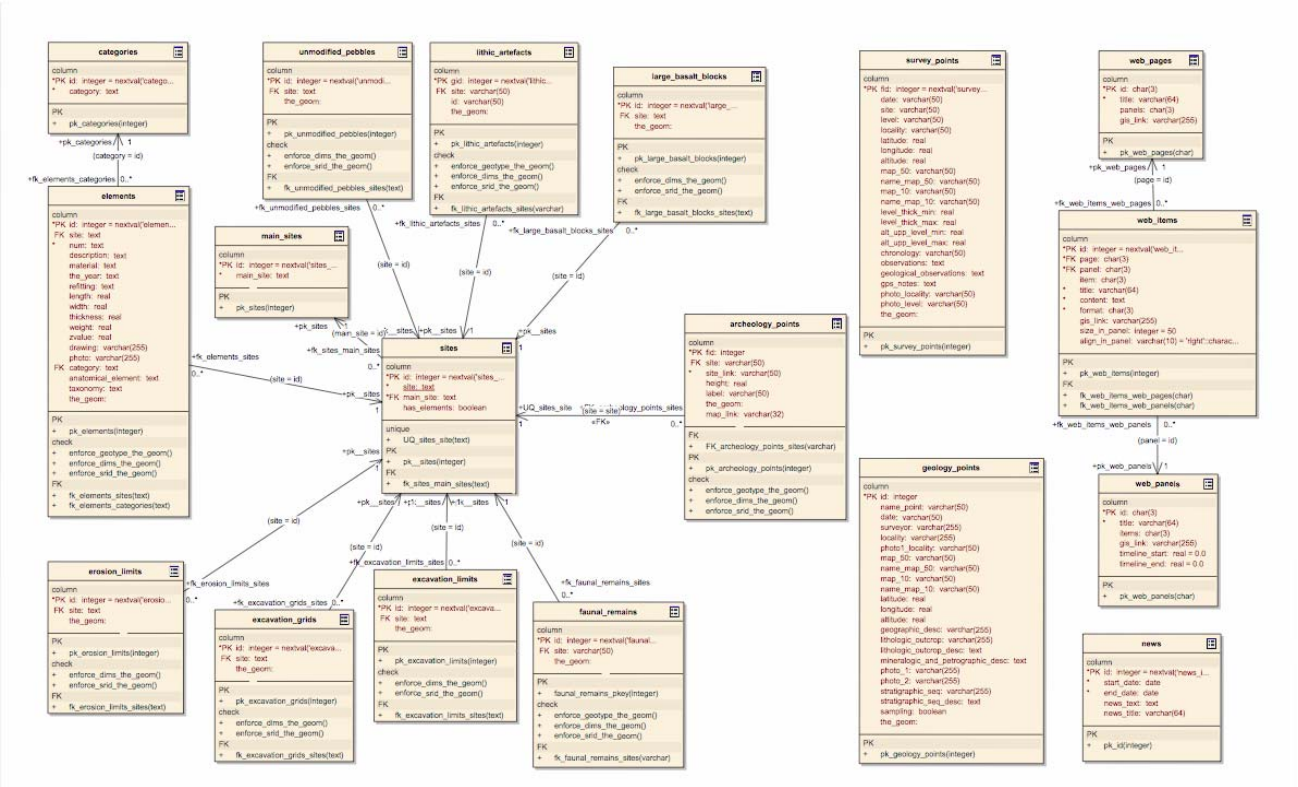
The variables and the values for the correct postioning and length of the "time-bar" are passed on to the flash file by html code. As well in the <parameter name> (used by IE Explorer) as in the <embed source> used by the other browsers the variable names (Old and Young) and their values are included.
This will be done at the moment that the user clicks on a panel. The asp page will generate the code and will replace the default values.

```
<div id="timeline">
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" code-base="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=7,0,19,0" width="635" height="100">
 <param name="movie" value="img/timeline8.swf?Old=0&amp;Young=0" />
 <param name="quality" value="high" />
 <embed src="img/timeline8.swf?Old=0&amp;Young=0" quality="high" plugin-spage="http://www.macromedia.com/go/getflashplayer" type="application/x-shockwave-flash" width="635" height="100"></embed>
</object>
```

5. THE DATABASE

The database used in the project is named "melka". A detailed overview of its design and implementation, as referenced in the next sections, can be found in *appendix A: Data Model Documentation*. The DB serves several purposes:

- holds the structure of the dynamic part of the Virtual Museum (the information panels and their contents). Its contents can be changed using the Content Management System (CMS). The setup of all this is explained in more detail in chapter 6: Virtual Museum Panels;
- holds the information for the dynamic News item on the Home page. . Its contents can be changed using the Content Management System (CMS). The setup of all this is also explained in more detail in chapter 6;
- holds the data and structure for the WebGIS, that is interfaced from the web pages "For Scientists". The setup of this is explained in more detail in chapter 7: WebGIS.



Database schema (see Appendix A fro details and "DBmodel.pdf" file on DVD for full-size version).

Implementation

The implementation of the Database was done in the in PostgreSQL/PostGIS database Management System. The PostgreSQL database server (`http://www.postgresql.org/`) is an object-relational database management system (ORDBMS). It's Open-source and supports most of the SQL standards, such as complex queries, foreign keys, triggers, views, transactional integrity and multi version concurrency control. Because of the liberal license, PostgreSQL can be used, modified, and distributed by everyone free of charge for any purpose, be it private, commercial, or academic.
PostGIS (`http://postgis.refractions.net/`)is an extension to PostgreSQL ORDBMS that allows OGC Simple features objects to be stored in the database. PostGIS includes support for GiST-based R-Tree spatial indexes, functions for analysis of OGC geometries and functions for processing of OGC geometries.

The database schema and contents could in theory be implemented in or transferred to any database server that supports the OGC Simple features for SQL 1.1 specification. Examples of such would be Oracle, DB2, MSsqlSpatial and MySQL (the latter only has limited support and does specifically miss spatial reference system support). For a complete and up to date overview see the OpenGeoSpatial consortium implementation pages at `http://www.opengeospatial.org/resource/products/byspec`.

The database has been built up and the data content inserted in three ways:

- schema built from scratch and contents filled by using the CMS. This is the case for the Virtual Museum and News parts; The content filling through the CMS has been largely undertaken by the archaeological experts (primarily dr. Rosalia Galotti);
- schema and contents by import from ESRI shape files (using the shp2pgsql command line utility to generate SQL insert queries). This is the case for the overview (point) maps in the WebGIS.
- schema built from scratch and contents filled by import from MS Access database files and further SQL querying. This is the case for the parts of the WebGIS that hold the excavation maps and their attribute tables; Things to note here are:
    - the separation of the original data tables as defined in the original MS Access files into a more complex object-relational model. This is necessary to ensure flexible and efficient data access of both the attributes and geometry, as well as to enable Constraints and Foreign Keys to uphold data integrity and atomicity;
    - all `elements` have been collected in one table, whereas the MS Access data had separate databases and tables for each of the excavations. A lot of unification has been done to enable searches on elements regardless of find site and level;
    - Drawings and photos are in this elements table are stored as links to files on the site (in the directory "panel_assets"), not in the DB itself. This is done because of data efficiency (if stored in BLOBS, a large overhead will be created for a column of which most tuples are empty) and because the same links can now be used in the general museum site, without the need for constant DB access. This is important for making as much as possible of the site functionality available in the off-line version on CD.

(re)build and backup of the DB

Backups of the database schema and it's contents on the date of delivery, as well as an XMI schema version, are available on the backup DVD (see appendix B: DVD contents).

To (re)build the DB, several options exists:

1. If you use PostgreSQL and PostGIS whose versions match the ones used in the current test server (PostgreSQL 8.2.3, PostGIS 1.2.1, the last digit of version numbers can be ignored), one can simply use the most recent DB dump "melkaBD.backup" and restore it to the database, using the `pg_restore` command-line application or the "restore…" command in pgAdminIII.
2. A more controlled solution that also works in other version databases is to use SQL query files:
    a. first load and run the commands in the file "melkaDB_schema_only.sql" to create the DB schema. The SQL queries in this file can be edited for other environments, eg. delete the queries that create the PostGIS projection table `spatial_ref_sys` if you already have one, change the database owner, or even rebuild the DB for use in another DBMS;
    b. then load the data by running the SQL commands in "melkaDB_data_only.sql".

Note that the database should be accessible for any DB client (most importantly the ASP pages mentioned in chapter 6 an 7) logged in under the login role "melka" (with currently the password "sopresso"). This role has to be defined in the PostGIS (or other DBMS) as having ALL rights on the "public" schema of the "melka" database.
This can be done in PostgreSQL using the query:

```
CREATE ROLE melka LOGIN PASSWORD 'sopresso'
      NOSUPERUSER NOINHERIT NOCREATEDB NOCREATEROLE;
GRANT ALL ON SCHEMA public TO melka;
```

6. VIRTUAL MUSEUM PANELS AND NEWS ITEM

Database Structure

This part of the DB holds the structure of the dynamic part of the Virtual Museum: The 4 Tukuls and the 2 open air features, the information of which is comprised of 80 information panels holding a total of 603 panel assets. These assets are either text fragments or images.

The `web_pages` table holds the list of dynamic web pages. These are currently only the pages that have panels on them: "TUKUL 1: African Prehistory", TUKUL 2: Geology and Volcanology", "TUKUL 3: Paleoanthropology", "TUKUL 4: Melka Kunture Archeology" and "Open Air Museum". The important columns of these have the following use:

- `panels` (of type `character(3)[]`, that means an array of strings of maximum 3 characters) refers to the `panel` column in the table `web_panels`. The three character code is in the form "pXX" or "sXX", where "p" denotes a virtual panel that refers to a panel in the real museum, while "s" denotes a virtual panel that refers to a showcase in the real museum, and where XX is a number from 0 - 99. Eg. a content of {p62,p63} in web-Page "w07: Open Air Museum" means that the web page contains the panels p62 and p63.

The `web_panels` table holds the list of dynamic web panels. These are currently 76 panels visualising real museum panels and 14 ones visualising showcases. The important columns of these have the following use:

- `title (character varying(64))`: This is used as the panel title in the web page middle section where the panels are shown.
- `items (character(3)[])`: refers to the `item` column in the table `web_items`. The column holds an array of three character codes. These codes are a free identifier, as entered by the editor in the web_items page in the CMS.
- `timeline_start(real)`: A real number that denotes the start time in MYA (Million Years Ago) relevant for this panel, to be shown in the time-line item (see chapter 5).
- `timeline_end (real)`: A real number that denotes the end time in MYA (Million Years Ago) relevant for this panel, to be shown in the time-line item (see chapter 5).

The `web_items` table holds the list of dynamic web items. These are currently 603 items used to put together the panels. The important columns of these have the following use:

- `page (character(3))`: Foreign key to the table `web_pages`;
- `panel (character(3))`: Foreign key to the table `web_panels`;
- `item (character(3))`: item identifier, used in the items arrays in the table `web_panels`;
- `title (character varying(64))`: This is used as the item title in the web page right section, where the item is shown if it's clicked in the panel.
- `format (character(3))`: either "txt" to denote a text fragment, or "img" to denote an image
- `content (text)`: If format is "txt", then this holds a HTML encoded text fragment. Special content is the "#" character, which will be used as a point to truncate the text (and construct a "read more…" link) when shown in the middle (panel) section of the page, while it is ignored (and thus the full text appears) when shown in the right (item) section of the page.
  If format is "img", then this holds a relative or absolute URL to the location of the image file.
- `size_in_panel (integer)`: If format is "img", denotes the width in pixels the image should take up in the middle (panel) section of the page. Ignored if format is "txt".
- `align_in_panel (character varying(10))`: If format is "img", denotes the alignment (left, right or middle) of the image in the middle (panel) section of the page. Ignored if format is "txt".

The `news` table is a small utility table that is used by the home page (`index.asp`) to show the latest (or last) news items. The important columns of these have the following use:

- `start_date (date)`: date when the news item should start appearing on the site; The dat itself also appears in the news list on the home page;
- `end_date (date)`: date after which the news item should disappear from the site;
- `news_title (character varying(64))`: short caption. This appears in the news list on the home page, with a "read more…" link next to it.;

- `news_text (text)`: short caption. Full text of the news item. Appears in a pop-up alert when the user clicks the "read more…" link in the news list on the home page.

Content Management System (CMS)

The CMS is used to change the panel setup of the Virtual Museum Panels and the News items, and to edit the contents of selected DB tables. It can be reached only by the URL (relative to the root of the site) `http:[site domain]/cms/`, not directly from the web site menus. The CMS is protected by a user-name/password combination. You can change the username or password by editing these lines in the file `/cms/cms.asp` :

```
var USERNAME = "melka_panel_assets";
var PASSWORD = "sopresso";
```

The CMS is made up of web pages that include server side script to connect to the database (using the "melka" login role, see chapter 5), containing HTML Forms that are used to show the current contents and request changes, deletions or new items. These pages are:

- `edit_news.asp` - to see the list of current news items and request deletions or additions to that list;
- `edit_panel.asp` - to see the list of current pages, their panels and their items, to request deletions as well as changes in the order of items in a panel and in the properties of an item (content, alignment, size, etc.); Includes a button to request an addition to the item lists of panels, which in turn will transfer the user to the page `add_item.asp`;
- `add_item.asp` - to request the addition of an item to a panel item list, either a text fragment or an image, plus their properties.
- `edit_table.asp` - to see the current contents of DB tables; Includes links in selected table cells to request a change of their value. Protected and complex data columns (eg. the primary key and OGC geometry columns) are not editable. The link will transfer the user to the page `edit_attributes.asp`;
- `edit_attributes.asp` - lets you change the value of a table cell; This is limited in functionality and has some *caveats*: Database constraints will be upheld, but otherwise no type checking or back-ups are done. And as the web DB connection is not aware of column *type*, it is up to the user to make sure the type of the value you insert matches that of the column, eg. you have to make sure to surround non-numeric values with single quotes. Wrong types will normally result in a DB error, and no harm done, but it's still important to check the result after changing.

The `item` editors include a mini-HTML editor to allow HTML formatting of text fragments. The Open Source JavaScript library "tiny_mce" was used for this (see `http://tinymce.moxiecode.com/`).

The forms on these pages trigger "headless" ASP pages, pages that only contain script but have no Web content. The script on these pages execute the requested SQL query, and if the database returns no errors, will silently redirect the user back to the CMS forms. These are:

- `change_item.asp`
- `change_panel.asp`
- `delete_item.asp`
- `delete_news.asp`
- `do_add_item.asp`
- `do_edit_attributes.asp`

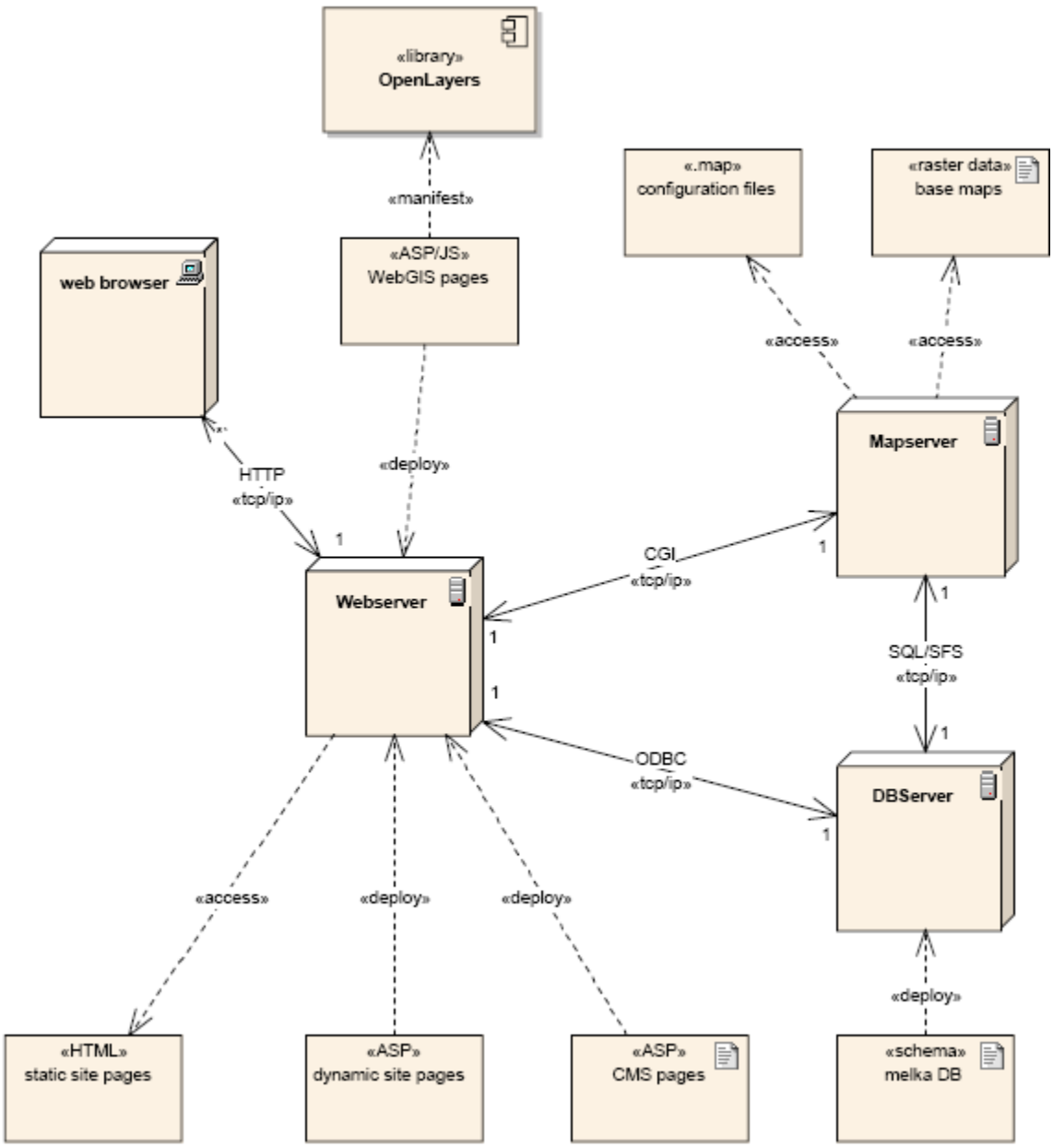
All the server side script is programmed in ASP JavaScript. ASP (Active Server Pages) is the server-side scripting technology of Microsoft's Internet Information Server.

7. WEBGIS

The WebGIS is the portion of the site that interfaces the website user (through a web browser) to the geometric and attribute data that is available in the "melka" database. The interface is available as a separated section of the website under the "For Scientists" menu. This part of the website is highly interactive and dynamic, and thus can not be represented on the off-line CD-ROM version of the site.

Architecture

A general overview of the architecture of the WebGIS is in the UML deployment model below (as printed from the Enterprise Architect model in file "DBmodel.eap", also available separately as "WebGIS-model.pdf"):



The WebGIS architecture has four main components:

Webserver

The webserver used is Microsoft Internet Information Server 6. Apart from standard static HTML pages this server can also run ASP JavaScript server-side scripts, allowing it to act as an application server that can

run the ASP scripts (as mentioned in chapter 6) to provide dynamic pages based on the database content as well as CMS functionality.

OpenLayers
The WebGIS pages that have map functionality on them, employ the library "OpenLayers". This is a JavaScript Library based on AJAX-principles (Asynchronous JavaScript And XML). OpenLayers can be used to build general WebMapping clients, among others for OGC Web Map Services.

OpenLayers makes it easy to put a dynamic map in any web page. It can display map tiles and vector data loaded from any source. It is a pure JavaScript library for displaying map data in most modern web browsers, with no server-side dependencies. OpenLayers implements a JavaScript API (Advanced Programming Interface) for building rich web–based geographic applications, similar to the Google Maps, with one important difference – OpenLayers is Free Software, developed for and by the Open Source software community. OpenLayers implements industry-standard methods for geographic data access, such as the OpenGIS Consortium's Web Mapping Service (WMS) and Web Feature Service (WFS) protocols. Under the hood, OpenLayers is written in object-oriented JavaScript, using components from `Prototype.js` and the `Rico` library. OpenLayers is still undergoing rapid development, update information can be found at `http://openlayers.org/` and examples at `http://openlayers.org/dev/examples/`

The latest version of the OpenLayers script library is always available on `http://openlayers.org/api/OpenLayers.js`. From here, you can include it in your own (X)HTML webpages and then call them using a simple JavaScript function. In our website, in order to not be dependent on the internet connection to the openlayers.org server, we link to a version on the same webserver that hosts the site and link to that in all pages employing the library by including the script call:

```
<script src=" [site root]/OpenLayers/OpenLayers.js">
</script>
```

To provide some basic understanding of the principles, find below the commented example of the OpenLayers Script for the overview map (as found in the file `gis/index.asp`):

```
base URL for Mapserver WMS ->  var baseURL = "http://geoserver.itc.nl/cgi-
                               bin/mapserv.exe?map=D:/Inetpub/geoserver/melkakunture/gis/mapserver/
                               config.map&";
            starting bounds ->  var bounds = new OpenLayers.Bounds(450000,958000,462000,971000);

 triggered by onload event of page
                            ->  function init(){

     make a new 'map' object ->  map = new OpenLayers.Map( "mapDiv", {maxExtent: bounds, projec-
'mapDiv' is the map placeholder in  tion:"EPSG:20137", minScale:250000, units: "m" } );
   the HTML, 20137 is the code for
         the Ethiopian projection

add raster baselayer of type 'WMS'  ikonosImage = new OpenLayers.Layer.WMS.Untiled("ikonos satellite im-
                            ->  age", baseURL, {layers: "ikonos"} );

                                 (…etc…)

       add vector overlay of type  geology_points = new OpenLayers.Layer.WMS.Untiled("geological
                     'WMS' ->  points", baseURL,
                               {layers: "geology_points",
                                transparent: "true",
       used later to filter data ->  mywhere: sql_search,
                                format: "image/png"}
                                );

                                 (…etc…)

   add layers to the map object ->  map.addLayer(ikonosImage);
                               map.addLayer(topoMap);
```

```
                                map.addLayer(topoMapGray);
                                map.addLayer(geology_points);
                                map.addLayer(survey_points);
  make some visible, some invisible map.addLayer(archeology_points);
                      at start -> survey_points.setVisibility(false);
                                geology_points.setVisibility(false);
                                archeology_points.setVisibility(true);

   set this layer as the Baselayer -> map.setBaseLayer(topoMapGray);
              add layerswitcher -> map.addControl(new OpenLayers.Control.LayerSwitcher());
            add coordinate show -> map.addControl(new OpenLayers.Control.MousePosition());
             zoom to full extent -> map.zoomToExtent(bounds);

      set one layer to be 'active' -> queryLayer = archeology_points;

    forward clicks in 'active' layer to map.events.register("click", map, queryResults);
                  query function ->
                                } //function init()
```

Adding Layers

You add layers using the `OpenLayers.Layer` object:

```
myNewLayer = OpenLayers.Layer.WMS(nameOfLayer,
   URLtoWMS,{listOfOptions} );
```

The `nameOfLayer` is a string that will be used in the layercontrol to name the layer. The `URLtoWMS` is the **base URL** to the service, that is the unique identifier of the WMS, **without** any added parameters (these will be supplied by OpenLayers). For MapServer WMS layers, this will be the URL to MapServer CGI + the MAP file defining the WMS, eg.
`[site root]/cgi-bin/mapserv.exe?map=[path_to_mapfiles]/[mapfile].map&`

`{listOfOptions}` is an array of `name:value` pairs that are used to add a multitude of special settings. The only one always needed is the `layers:` option, that should tell the WMS what layers are requested. Note that one OpenLayers layer can be made up of several WMS layers!

There is an important difference between what OpenLayers calls **Base Layers** and **Overlays**. There is always one **Base Layer** needed. It's the one that sets the projection, extent and units of the map. You can define several base layers if they share the same parameters (eg. several layers of one WMS, or the various types of Google maps), but the user can always have only one turned on, the choice will be made using a radio-button list.

Overlays are layers that can be fitted on top of the base layer. Because they can be transparent, many overlays can be used, and they each can individually be turned on or off (using a checkbox list).

Normally, WMS layers will become base layers, unless you have made sure they are transparent, in that case they can be overlays. To achieve that, you need to add the option `transparent: "true"` to the option list and also make sure you request the WMS for a suitable format, one that supports transparency, eg. `format: "image/png"`.

Querying attributes from WMS layers

There are several ways to request additional (attribute) data from the mapserver. Here, we use the OGC `GetFeatureInfo` request. In the HTML file there is an additional placeholder to retrieve the results of the request.:

```
<iframe id="resultsDiv" name="resultsDiv" frameborder="0" frameborder="no" >
   empty...
</iframe>
```

We use an <iframe> element, because a simple cannot retrieve new content based on a so-called asynchronous request (which is what this URL request in fact is). In our earlier code-view you found the lines:

```
map.events.register("click", map, queryResults);
```

This registers a so-called event-listener to the map that will be triggered every time anyone makes a "click" in the map object. The trigger will start a call-back function named queryResults, giving it information about the event (such as where the location of the click was, etc.):

```
                                  function queryResults(evt) {
                                   if (queryLayer != null) {
                get original URL for the layer ->   var url = queryLayer.getFullRequestString({
            add a WMS GetFeatureInfo request ->     REQUEST: "GetFeatureInfo",
                    specify format for errors ->    EXCEPTIONS: "text/html",
                  set BBOX to current extent ->     BBOX: queryLayer.map.getExtent().toBBOX(),
                       X location of the event ->   X: evt.xy.x,
                       Y location of the event ->   Y: evt.xy.y,
                          ask for HTML output->     INFO_FORMAT: "text/html",
                  set current layers for query->    QUERY_LAYERS: queryLayer.params.LAYERS,
                         width of current map->     WIDTH: queryLayer.map.size.w,
                        height of current map->     HEIGHT: queryLayer.map.size.h
                                                    }
                                                   );
                  get the iframe as an object ->    resultsDiv = top.document.getElementById("resultsDiv");
                 'feed' the iframe with the url ->  resultsDiv.src = url;
                  no further processing of evt ->   Event.stop(evt); //prevent 'bubbling up' of event...
                                                   } // if not null
                                                  } // function(evt)
```

The other pages that employ OpenLayers work in a similar fashion:

- siteSimple.asp - loads the vector data for one of the sites that has no additional attributes by calling the WMS with a filter (in the sql_search parameter of the new layer command) for the site requested. The baseLayer is always the excavation_grid, and there is no 'active' layer (or queryLayer), as there are no additional attribute data to be retrieved.
- site.asp - loads the vector data for one of the sites that has additional attributes by calling the WMS with a filter (in the sql_search parameter of the new layer command) for the site requested. The baseLayer is always the excavation_grid, and the 'active' layer (or queryLayer) is the elements layer, as this table holds the additional attribute data.

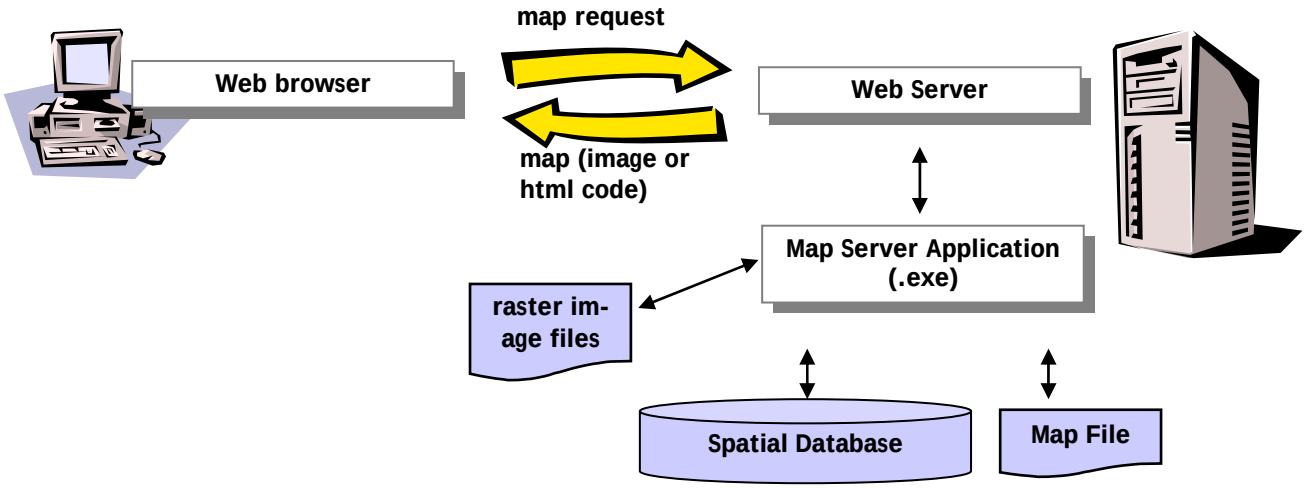
Mapserver

The Mapserver used is University of Minnesota Mapserver 4.10.0 (see http://mapserver.gis.umn.edu), an Open Source web application for mapping and data retrieval that can be configured to deliver OGC compatible WMS services (among many other things). We use UMN Mapserver as a CGI application (Common Gateway Interface - a standard for interfacing external applications with web servers). This means that the web browser client and the ASP pages running on the webserver will generate URL requests to the Mapserver CGI that are executed on request, so that the web server is able to provide dynamic/ real time information coming from the Mapserver. Mapserver is configured by so-called Map file, text-based configuration files that can be dynamically altered by URL parameter substitution.

The data used by MapServer to deliver maps and attribute info can come from a great many different data stores. In our setup, the majority (all vector geometry and all attributes) come from the Spatial Database "melka", and only the raster base layers for the overview map (ikonos_degradeRGB, topomap and topomap_grayed) come from stand-alone image files (in the JPEG format, georeferenced by using appropriate world files with .jgw extensions). This external data is stored on the webserver in [site root]/gis/mapserver/data/

The configuration (.map) files are stored in `[site root]/gis/mapserver/` and are:

- `config.map` - the configuration for the overview map.
- `configSite.map` - the configuration for the site maps that have additional attribute data.
- `configSiteSimple.map` - the configuration for the simple site maps that do not have additional attribute data.



Here we do not fully explain all details of these map files (please refer to the Mapserver documentation for that), but below are some comments of features specific to the MelkaKunture WebGIS setup:

```
start of MAP file ->    MAP
                        NAME MelkaKunture

                        (…etc…)

20137 is the EPSG code for the ->   PROJECTION
Ethiopian projection                 "init=epsg:20137" # UTM 37N on Adindan datum
                                    END

                        (…etc…)

One of the base layers ->   LAYER
                             NAME topomap #Image layer , baselayer for OpenLayers
this is a raster ->          TYPE RASTER
                             STATUS ON
data comes from image file ->    DATA "data/topomap.jpg"
                             METADATA
                              "wms_title" "topomap"
                             END
                             PROJECTION
                              "init=epsg:20137"
                             END
                            END #layer

                        (…etc…)

One of the overlay layers ->  LAYER
                               NAME geology_points
these are vector points ->     TYPE POINT
                               STATUS ON
data comes from DB ->          CONNECTIONTYPE postgis
connection parameters ->       CONNECTION "user=melka password=sopresso dbname=melka
                              host=localhost port=5432 options='-c client_encoding=UTF8' "
this is a “pseudo-SQL” query used by    DATA "the_geom FROM public.geology_points using unique id using
```

```
        MapServer (see ** below)->   srid=20137"
                                       METADATA
                                              "wms_title" "geology_points"
    attribute queries should return all ->    "wms_include_items" "all"
                                       END
            clicking is still registered if it   TOLERANCE 8
                      is <8 pixels 'off' ->      TOLERANCEUNITS pixels
      DB attribute to use for labelling ->       LABELITEM "name_point"
                                                 CLASS
     template for returning attributes ->         TEMPLATE "templates/query_geology_points.html"
       used in the queryResult <iframe>
                                                (…etc…)

                                                 END # class geol
                                                END #layer geology_points

                                               END #map
```

****** For connections to a PostGIS datasource, you use a LAYER with CONNECTIONTYPE postgis. The general form of the connection parameters is:

```
CONNECTIONTYPE postgis
CONNECTION "user=<username> password=<password> dbname=<database> host=<dbhost>
port=<port> options='-c client_encoding=<encoding>'"
DATA "<geometry column> from <schema>.<table> using unique <PKcolumn> using
srid=<srid>"
```

The <PKcolumn> should be the name of a column in your table that is unique for all rows, usually the Primary Key column. The <encoding> should match the encoding of the PostGIS database, usually this would be UTF8. So for the archeology_points data in the "melka" database this becomes:

```
CONNECTIONTYPE postgis
CONNECTION "user=melka password=sopresso dbname=melka host=localhost port=5432
options='-c client_encoding=UTF8' "
DATA "the_geom FROM public.archeology_points using unique fid using srid=20137"
```

The nice thing is, that you can change the DATA string, because it basically is a SQL select string, to any valid SQL statement that makes a more complex query. This is used in the Site and SiteSimple maps to get out only elements within the given site:

```
DATA "the_geom FROM (SELECT * FROM public.excavation_grids %MYWHERE%) AS foo
using unique id using srid=20137"
```

Note that the MapServer SQL parser is a bit peculiar: It needs the query always to result in only a geometry column, this is done by making a sub-query with an alias (hence the AS foo statement). Note also the use of the "parameter substitution" clause %MYWHERE%. If the WHERE clause in this case would be fixed, that would require a separate .map configuration file for each excavation site. This can be avoided by use a feature of Mapserver, called "parameter substitution". This makes sure that selected parameters, that in the .map file are surrounded by %-signs, will be substituted by the value that is found in the URL request using that same name. The site will be requested by the webserver/ASP pages with a URL that contains the parameter &MYWHERE=<name_of_site>, eg. in the siteSimple.asp you will find the following ASP and JavaScript code :

```
      start of server-side ASP code ->   <%
        get site from URL parameter->    var currSite = Request.QueryString("site").Item;
                                         if (typeof(currSite) == "undefined") {currSite = "none"}
                    end of ASP code ->   %>
  put ASP var into client-side JS var->  var currSite = "<%=currSite%>";
                                         if (currSite == "none") {
                                                sql_search = "";//finds all sites
```

```
                                    alert("Error: no site chosen!");
                                } else {
construct WHERE statement->             sql_search = " WHERE site ILIKE '" + currSite + "' ";
                                };
```

This way, by combined efforts of the ASP filling in the currSite parameter and the OpenLayers Javascript requesting a URL with the site name in the MYWHERE parameter, the final DATA statement requested from the DB will for example become:

```
DATA "the_geom FROM (SELECT * FROM public.excavation_grids WHERE site ILIKE
'Garba I') AS foo using unique id using srid=20137"
```

These WHERE statements can be further extended if necessary, as is implemented in the *category filtering* on the Maps and Element Data page (see below in "functionalities").

Database server

The general database server setup and functionality was described already in some detail in chapter 5. Here we further explain the fashion in which data can be updated or added to the WebGIS tables.

Updating existing data attributes is the simplest. If any attribute value should be changed, or an attribute value should be added where there currently is no value (or a NULL value), the attribute can just be changed or updated. For the Museum panels and the News items, this can be done using the CMS (see ch. 6). For selected tables, the CMS also includes a simple table editor (see ch. 6). These tables are:

- archeology_points
- geology_points
- survey_points
- elements

For other attributes, and if editing is more complex then just changing single column values, a direct connection to the Database Server is used. Just connect to the appropriate DB table and update the values. This can be done using SQL queries in the command line client (`psql`) of PostgreSQL, but it's much more convenient to use the GUI interface of pgAdminIII (delivered with the PostgreSQL installation, see also `http://pgadmin.org`).

For addition of information, also the DB connection through pgAdminIII can be used. Adding single entities to eg. the overview map points or excavation site elements can be done by simply using SQL INSERTs to the appropriate tables. Below is an example of adding a excavation site to the `archeology_points` table:

```
INSERT INTO archeology_points
(fid,site, site_link, height, label, the_geom, map_link)
VALUES
(999, 'test', 'w=w06&p=p99', 123.4567, 'test label',
GEOMFROMTEXT('SRID=20137;POINT(456000.00 962000.00)'),
'none');
```

For larger amounts of new entities, mass uploads from other sources are better suited. For data with OGC geometry content, the command-line utility `shp2pgsql` has been used (part of the PostGIS distribution). Eg. find below the command-line for adding the survey_points geometry:

```
shp2pgsql -s 20137 -i -I -S -W "UTF-8" survey.shp tmp_geom
> survey_points.sql
```

The resulting SQL file can be executed in psql command line or using `pgAdminIII`.

For attribute data from MS Access, we found exporting from Access to CSV format and then using the SQL COPY command worked best. Below is an example a CSV COPY used in setting up the DB:

```
COPY survey_points (
date , site , level, locality , latitude , longitude , altitude , map_50 ,
name_map_50 , map_10 , name_map_10 , level_thick_min ,  level_thick_max ,
```

```
alt_upp_level_min , alt_upp_level_max , chronology , observations , geologi-
cal_observations , gps_notes , photo_locality, photo_level)
FROM 'D:/Database survey.csv' WITH CSV HEADER DELIMITER AS ',';
```

Afterwards the data can be added to the imported geometric content by a DB JOIN:

```
UPDATE survey_points SET the_geom =
(SELECT tmp_geom.the_geom from tmp_geom,survey_points
WHERE tmp_geom.site ILIKE survey_points.site);
```

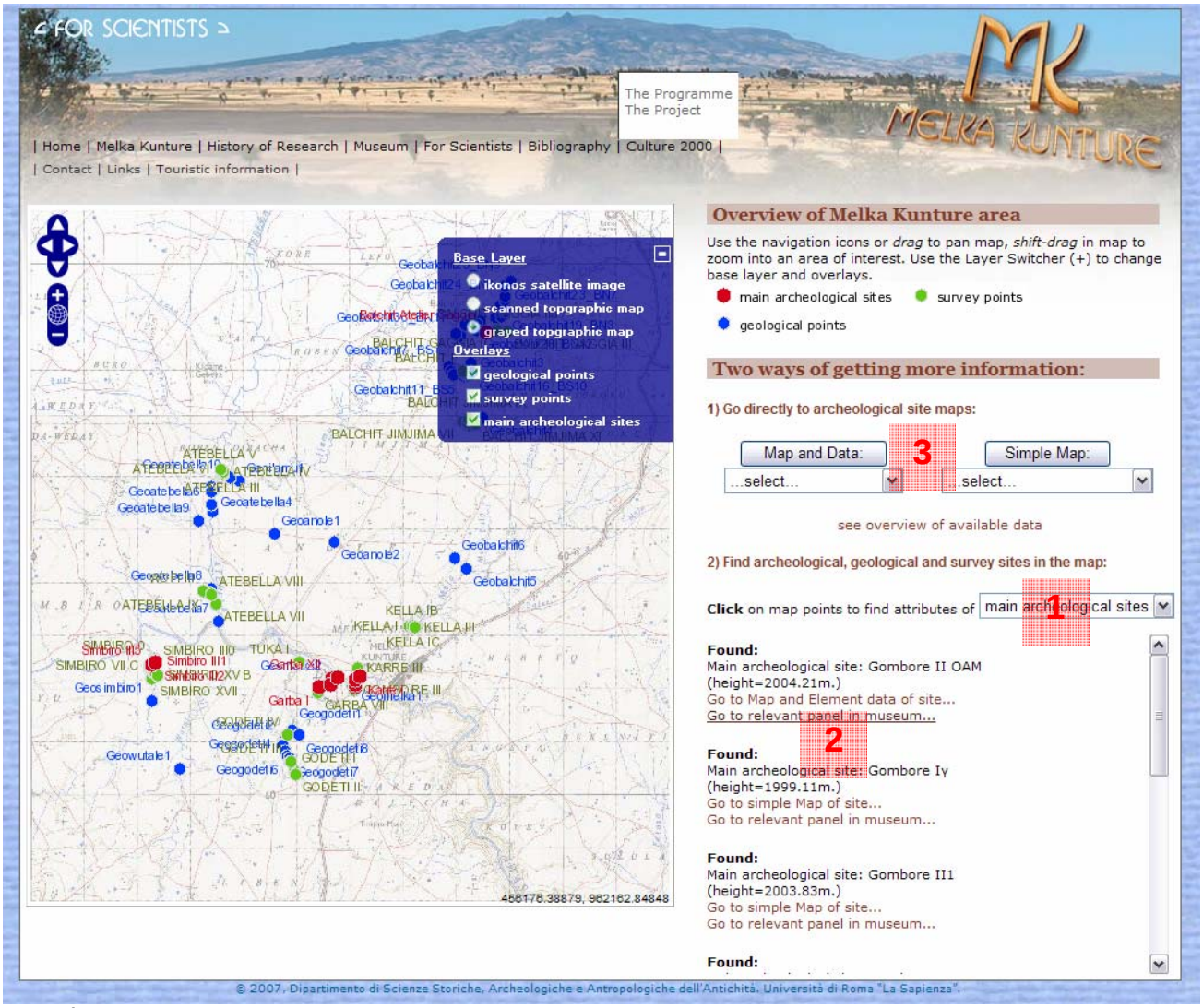
Adding WebGIS layers is basically just the same as adding any table with OGC geometry content.
The additional step to have them appear in the maps involves adding layers to the OpenLayers client-side JavaScript (see the section about OpenLayers above).

Functionalities

Below is an overview (with screen dumps) of the different functionalities of the WebGIS pages ("For Scientists"), and a short explanation of how that functionality has been implemented using the combinations of OpenLayers, general Javascript, ASP and Mapserver technologies.

Overview map

The start of the "For Scientists" section is the Overview page: `[site root]/expert/index.asp`.

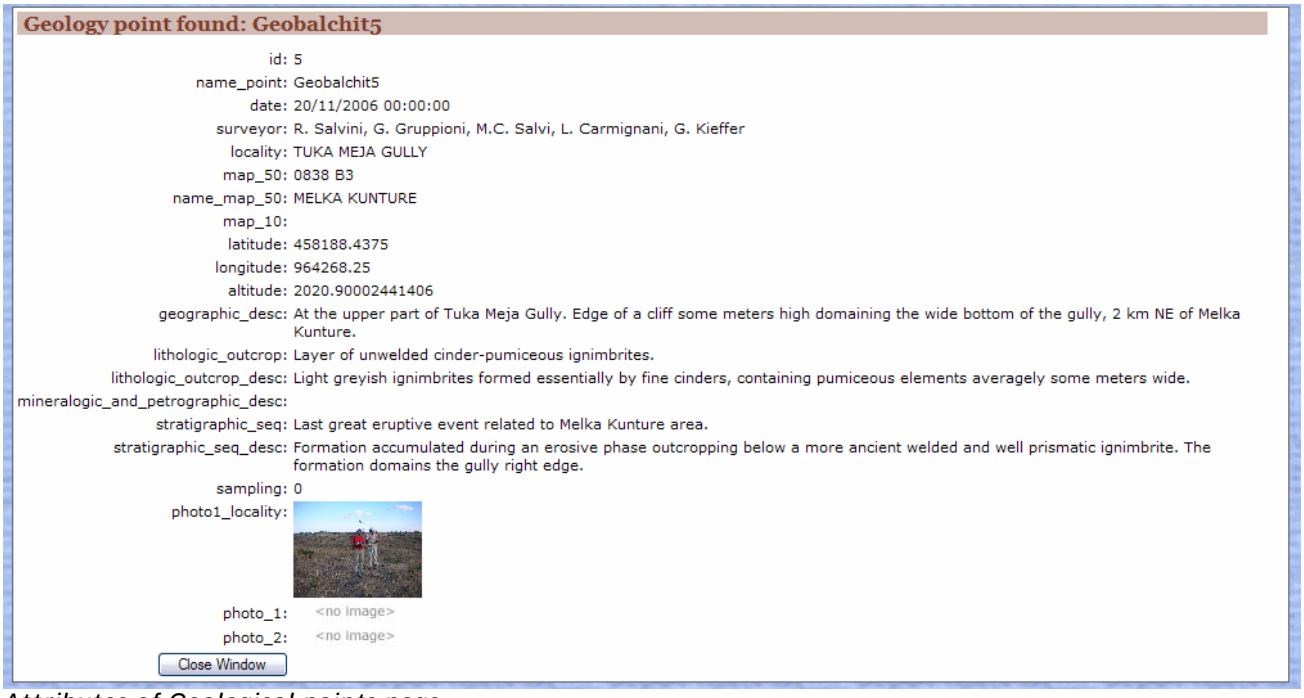


Overview map page

Here the overview base maps can be viewed (the ikonos satellite image, the scanned topographic map and a greyed version of that last map, which is the default). They can be overlaid with maximum 3 overlay layers:

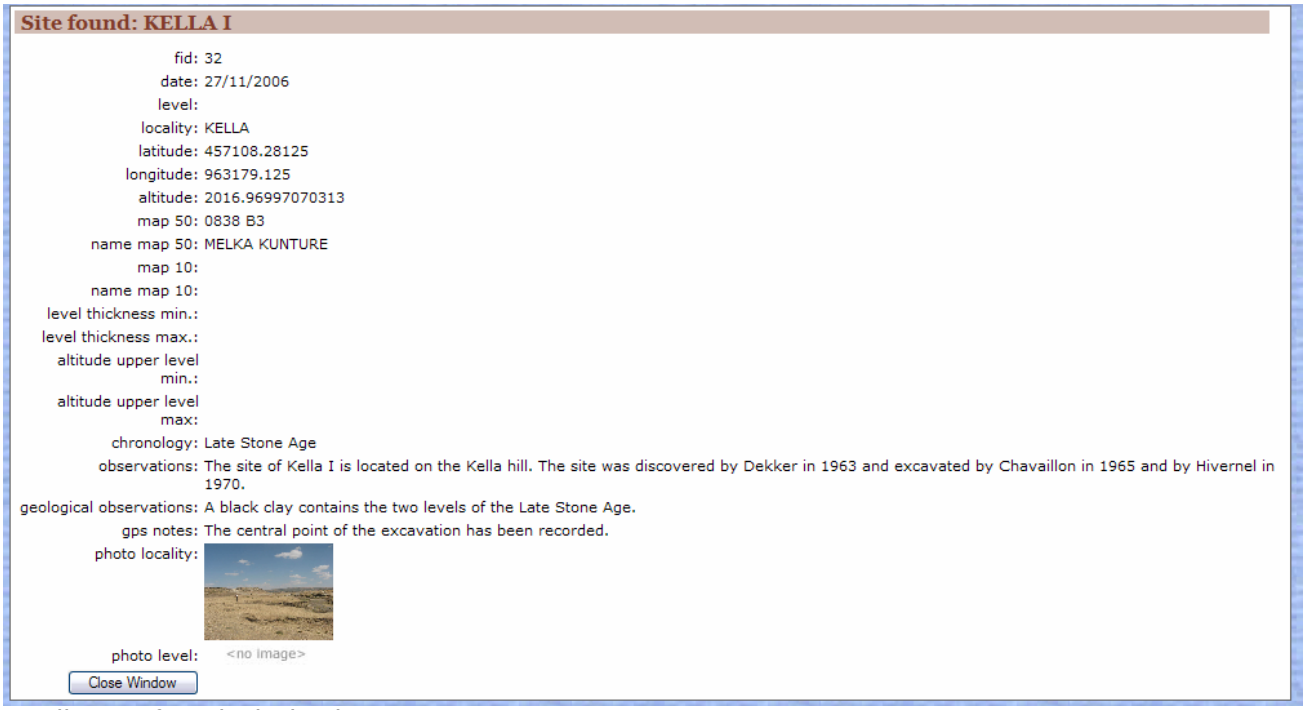
- **Geological points** - data coming from table `geology_points` (see ch. 5). If this layer has been made 'clickable' (by setting the pull down menu [1]), the OpenLayers script (see the OpenLayers section above) will assure any data occurrences within the click surroundings (tolerance is set in the .map file, currently at 8 pixels) will be retrieved from the DB by Mapserver and parsed in to the appropriate HTML template `[site root]/gis/mapserver/templates/query_geology_points.html` and shown in the resultsDiv `<div>` element [2]. The template will show the point name, the locality attribute and a link to show all attributes (see below). These attributes are shown using the page `[site root]/expert/find_geology.asp` (see below). This ASP page is a

simple DB lookup page, using the `name_point` attribute to find the appropriate tuples in the `geology_points` table.



Attributes of Geological points page

- **Survey points**- data coming from table `survey_points` (see ch. 5). If this layer has been made 'clickable' (by setting the pull down menu [1]), data occurrences will be parsed in to the appropriate HTML template `[site root]/gis/mapserver/templates/query_survey_points.html` and shown in the resultsDiv `<div>` element [2]. The template will show the point name and a link to show all attributes. These attributes are shown using the page `[site root]/expert/find_survey.asp` (see below). This ASP page is a simple DB lookup page, using the `site` attribute to find the appropriate tuple in the `survey_points` table.



Attributes of Geological points page

- **Main archaeological sites**- data coming from table `archeology_points` (see ch. 5). If this layer has been made 'clickable' (by setting the pull down menu [1], it is by default set to this layer), any data occurrences will be parsed in to the appropriate HTML template `[site root]/gis/mapserver/templates/query_archeology_points.html` and shown in the resultsDiv `<div>` element [2]. The template will show the site name and two links :
  - a link to the appropriate site map (Simple Map or Map and Element Data); This results in the same maps as using the *site chooser menus* [3].
  - a link to the relevant panel in the Museum site. This is achieved by having Mapserver substitute the `[site_link]` part of the HTML code `<a href="/melkakunture/museum/tukul4.asp?[site_link]">` in the template file with the appropriate column value found in the DB. Thus the `tukul4.asp` file will retrieve the appropriate `site_link` parameter (eg. `w=06,p=39`) and load panel 39 into its panel `<div>`.

You can also directly go to any of the Excavation Site maps by using the *site chooser menus* [3]. These will present either a Simple Map page or the Map and Element Data page.

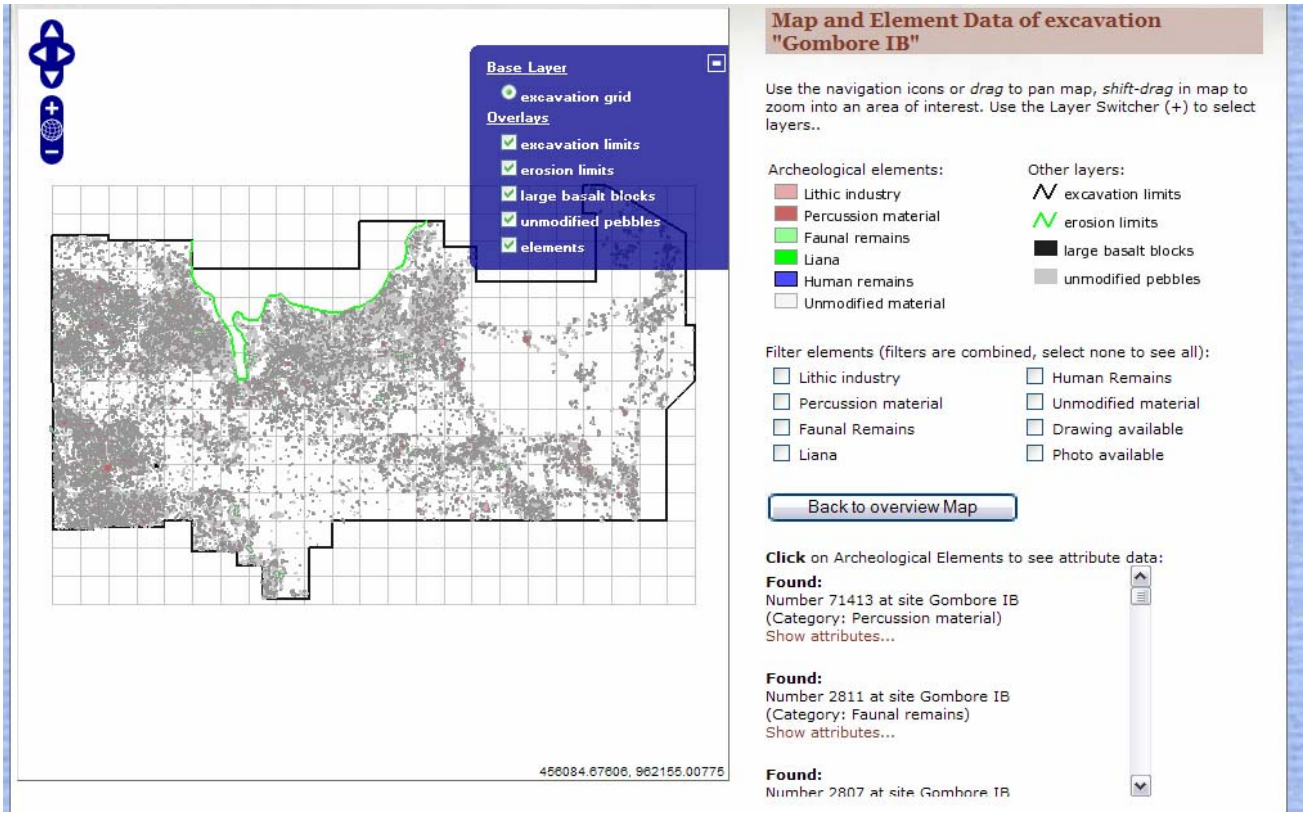
Simple Map page

The Simple Map page `[site root]/expert/siteSimple.asp` will be used for excavation sites that have geometric data, but no additional attribute data (the original data had no accompanying MS Access data file). It will show the available base map (always the excavation grid) and overlay maps (excavation limits, erosion limits, unmodified pebbles, lithic artefacts and faunal remains).

Simple Map page

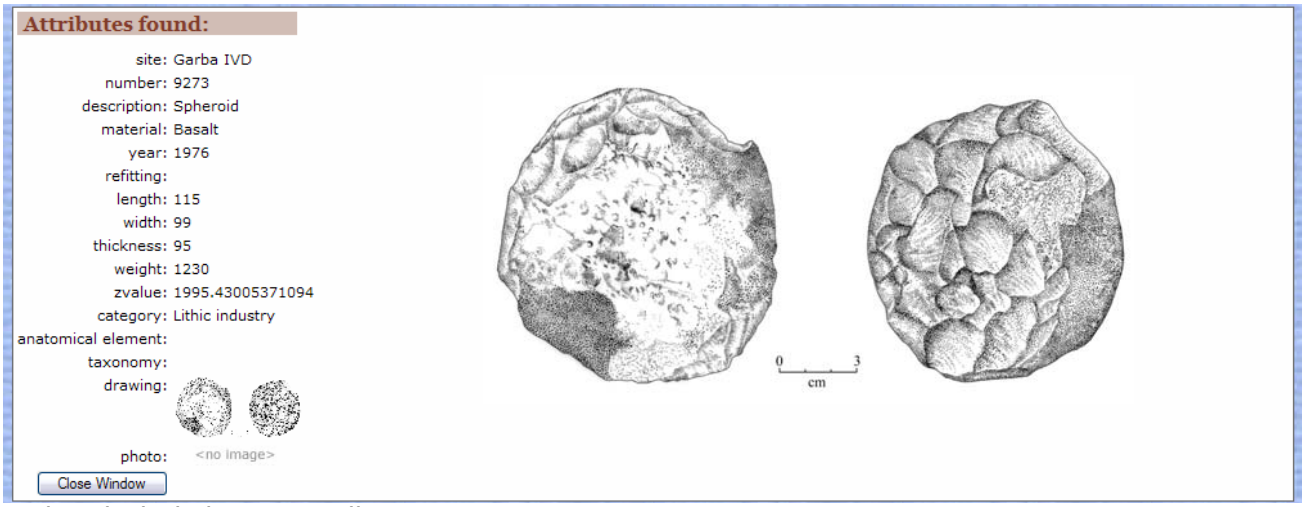
Map and Element Data page

The Map and Element Data page `[site root]/expert/site.asp` will be used for excavation sites that have geometric data as well as additional attribute data (the original data had accompanying MS Access data files). It will show the available base map (always the excavation grid) and overlay maps (excavation limits, erosion limits, large basalt blocks, unmodified pebbles and the elements layer).



Map and Element Data page

The elements overlay will be 'clickable' , and data occurrences will be parsed in to the appropriate HTML template `[site root]/gis/mapserver/templates/query_elements.html` and shown in the resultsDiv `<div>` element [2]. The template will show the point number and site, the category of the element ("Faunal remains", "Lithic industry", "Percussion material", "Human remains", "Unmodified material", "Liana" or "Lithic artefacts") and a link to show all attributes. These attributes are shown using the page `[site root]/expert/find_element.asp`. This ASP page is a simple DB lookup page, using the `site` and `number` attribute to find the appropriate tuple in the `elements` table. Clicking on drawings and photos in the file will result in an enlarged version of that image being shown (see below).



Archaeological Element Attributes page

An additional possibility in the Map and Element Data page is filtering the elements by category. Any of the categories mentioned above, plus two additional ones ("drawing available" and "photo available"), can be selected. The OpenLayers Javascript will trigger an update of the map, now using and additional WHERE

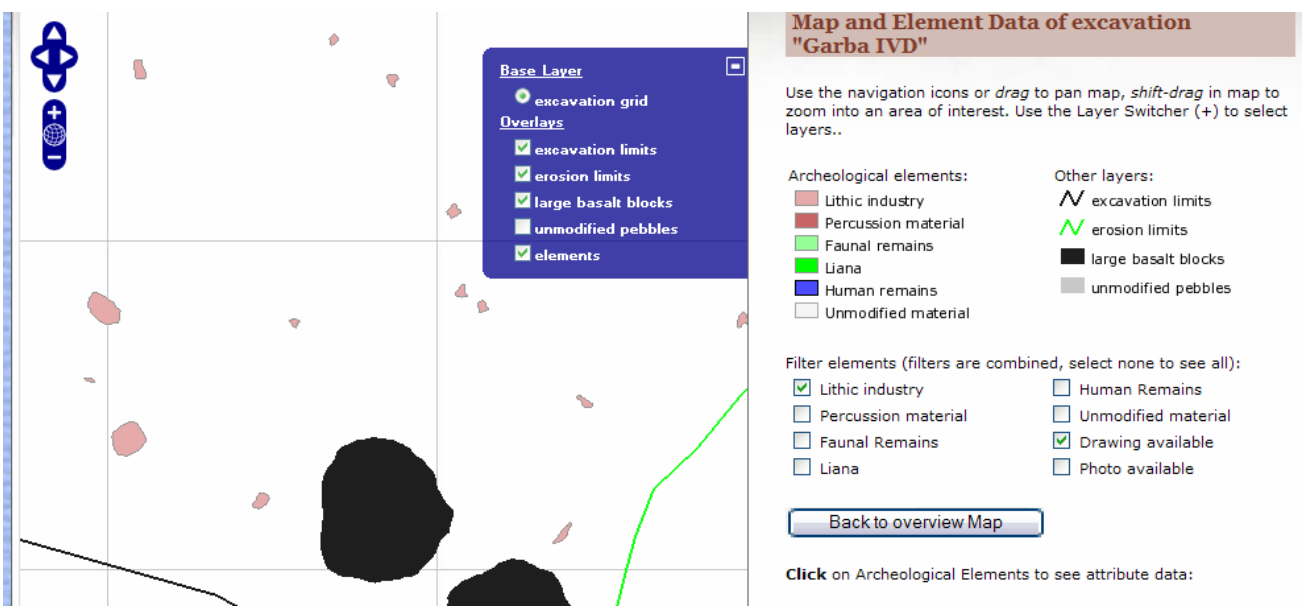
statement to narrow down the selection from the DB. Each of the filters adds a statement to the general WHERE clause (See above in OpenLayers section), eg. :

```
value=" AND category='Percussion material' "
```

When clicked, the `doFilter` JavaScript will be executed:

```
                                function doFilter() {
                                       currFilter="";
loop through checkboxes->                      for (var i=1; i<9;i++) {
                                              currFilterCheck =
                                top.document.getElementById("filter" + i).checked;
see if they are checked ->                    if (currFilterCheck == true) currFilter +=
if so, add filter ->            top.document.getElementById("filter" + i).value;
                                              }
remove layer->                        map.removeLayer(elements);
make new layer, now with filter->     elements = new OpenLayers.Layer.WMS("elements", baseURL,
                                              {layers: "elements",
                                               transparent: "true",
                                                mywhere: sql_search,
                                                filter : currFilter,
                                                format: "image/png"}
                                                );
add filtered layer ->              map.addLayer(elements);
                                       queryLayer = elements;
                                } // doFilter()
```

The result will be a map where only part of the elements (or none) will be visible:



Zoomed in Map and Element Data page with filter that only shows elements of category "Lithic Industry" of which a drawing is available.

8. PRODUCTION, LAUNCH AND MAINTENANCE

Functionality was tested continuously during production. The website has been on-line and the participating organisations were asked to give feedback during the actual production, on the ITC in-house server at `http://geoserver.itc.nl/melkakunture`.
Then, from July 15 - August 3 the participating organisations were asked to test the website and report bugs, failures and missing elements.

From August 6 – September 14 the so called debugging took place, where remarks were evaluated and errors corrected. On September 20 the final version of the website was released, and the website was presented to the EU and Ethiopian representatives at Addis Ababa, by the Italian project partners. It was planned that on or before that date the site had been transferred to its production server, under the domain of `www.melkakunture.eu`. Unfortunately, the domain and the server (to be installed by the project lead at the Universita La Sapienza in Rome) were not available up to the time of writing, therefore the site has, for the time being, been kept running at the before mentioned test server at ITC. Note that this test server is in no way guaranteed to be running 24/7, nor is it backed up in a regular schedule, etcetera, in short it is not suitable as a production system!

Note that maintenance of the website, WebGIS and server is **not** part of the sub-contract of ITC.

Remaining issues and further developing

At the time of writing, there still were a couple of things that were originally planned to be part of the Website and WebGIS, but were not realised at time of launching. Most of these are because of data not being available or having errors.

Mismatch between data of sites in DB and main_archeological sites map
Although all sites (in the current DB, see left column) can be reached through the pull-down menus, the ones that are 'empty' or ' mismatching' in the right column cannot be found through having selected them in the map and then click on the link in the "found:.." section. The reason for this is the mismatch between the `archeology_points` table (constructed from the original file `Main Archaeological Sites.shp` and the excavation sites as coming from the seperate site shape and MS Access-files:

Excavation list in current DB:	**main archeological sites.shp:**
24;"Balchit Atelier Gaggia";"Balchit";"TRUE"	"Balchit Atelier Gaggia"
11;"Garba I";"Garba I";"FALSE"	"Garba I"
12;"Garba IIIB";"Garba III";"FALSE"	"Garba III"
13;"Garba IIIC";"Garba III";"FALSE"	
5;"Garba IVC";"Garba IV";"TRUE"	"Garba IV"
6;"Garba IVD";"Garba IV";"TRUE"	
7;"Garba IVE";"Garba IV";"TRUE"	
10;"Garba XII";"Garba XII";"FALSE"	"Garba XII"
9;"Gombore Iγ";"Gombore I";"FALSE"	"Gombore I gamma"
3;"Gombore IB";"Gombore I";"TRUE"	"Gombore I"
19;"Gombore II Butchering site";"Gombore II";"FALSE"	"Gombore II Butchering Site"
14;"Gombore II OAM";"Gombore II";"TRUE"	"Gombore II OAM"
15;"Gombore II1";"Gombore II";"FALSE"	"Gombore II1"
16;"Gombore II3";"Gombore II";"FALSE"	"Gombore II3"
17;"Gombore II4";"Gombore II";"FALSE"	"Gombore II4"
18;"Gombore II5";"Gombore II";"FALSE"	"Gombore II5"
2;"Karre IM";"Karre I";"TRUE"	"Karre I"
1;"Karre Ik";"Karre I";"TRUE"	
20;"Simbiro III1";"Simbiro III";"FALSE"	"Simbiro III1"
21;"Simbiro III2a";"Simbiro III";"FALSE"	"Simbiro III2"
22;"Simbiro III2b";"Simbiro III";"FALSE"	
23;"Simbiro III5";"Simbiro III";"TRUE"	"Simbiro III5"

The solution is extending the `archeology_points` table at a later date with the 'missing' points.

Mismatch between photo/drawing names and links in data
In the original Access data files, photos and drawings were embedded OLE objects. The links to those were apparently lost in translation to a newer Access version and thus in the data as received by ITC, the links could not be extracted from the Access files to the DB. For most elements the photo link could be reconstructed from the elements data using the following logic:
`name photo = site + '-' + num + '-' + material + ' ' + description. + .'jpg'`
This goes wrong in a limited number of individual cases where the logioc was not maintained in the naming of the photo. This is mostly found in Faunal Remains. Eg. we found two photos in Garba IVE that are respectively named:

```
Garba IVE 2005-1-Hippopotamus bone.jpg
Garba IVE 2005-77-Phalanx Hippopotamus.jpg
```

So no logic here… There is no other solution then checking all elements with photos (filter them in the site map) and check if the photo is there. If it says <no image> it's correct, if it is empty or shows a box with an X in it, the photo should be there but is named incorrectly. One should then rename the photo, found in the server path `[site root]\gis\photos`, to the name it should have according to the logic described above.

Corrupt and missing data
The shape file `Karre_IK_unmodified_pebbles` was found to be corrupt (.shx file = 0 kB). At implementation time, there was no data available for the site "Gombore II OAM". Therefore both are not present in the DB. Furthermore, it was planned to have *vector* topographic background layers in the overview map. Unfortunately, the student project producing this data was not finished at implementation time, so this data could not be added.
Alll of these should be added at a later time by the maintenance persons.

Enschede, 11 October 2007,

Wim Feringa
Barend Köbben

APPENDIX A: DATA MODEL DOCUMENTATION

Generated by Enterprise Architect from the DBmodel (available on backup DVD as "DBmodel.eap" as well as standardised XMI outputfile "DBmodel.xmi").

Data Model

Type: **Package**
Package: Model
Detail: *Created on* 12-9-2007 10:45:10. *Last modified on* 12-9-2007 10:45:10
Notes:

Data Model
Created By: Kobben
Last Modified: 4-10-2007, *Version:*1.0

archeology_points

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	fid	integer	True	False	0	0	0		
False	site	varchar	False	False	50	0	0		
False	site_link	varchar	True	False	50	0	0		
False	height	real	False	False	0	0	0		
False	label	varchar	False	False	50	0	0		
False	the_geom		False	False	0	0	0		
False	map_link	varchar	False	False	32	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
FK_archeology_points_sites	FK	site		
pk_archeology_points	PK	fid		
enforce_geotype_the_geom	check		((geome-trytype(the_geom) = 'POINT'::text) OR (the_geom IS NULL))	
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	

Relationships

Columns	Association		Notes
(site = site)	**0..***	archeology_points**.**FK_archeology_points_sites	
	1	sites**.** UQ_sites_site	

categories

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.

Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval ('categories_id_seq' ::regclass)	
False	category	text	True	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_categories	PK	id		

Relationships

Columns	Association		Notes
(category = id)	**0..***	elements.fk_elements_categories	
	1	categories. pk_categories	

elements

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval ('elements_id_seq' ::regclass)	
False	site	text	False	False	0	0	0		
False	num	text	True	False	0	0	0		
False	description	text	False	False	0	0	0		
False	material	text	False	False	0	0	0		
False	the_year	text	False	False	0	0	0		
False	refitting	text	False	False	0	0	0		
False	length	real	False	False	0	0	0		
False	width	real	False	False	0	0	0		
False	thickness	real	False	False	0	0	0		
False	weight	real	False	False	0	0	0		
False	zvalue	real	False	False	0	0	0		
False	drawing	varchar	False	False	255	0	0		
False	photo	varchar	False	False	255	0	0		
False	category	text	False	False	0	0	0		
False	anatomical_element	text	False	False	0	0	0		
False	taxonomy	text	False	False	0	0	0		
False	the_geom		False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_elements	PK	id		
enforce_geotype_the_geom	check		((geometrytype(the_geom) =	

Name	Type	Columns	Initial Code	Notes
			'MULTIPOLY-GON'::text) OR (the_geom IS NULL))	
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	
fk_elements_sites	FK	site		
fk_elements_categories	FK	category		

Relationships

Columns	Association	Notes
(category = id)	**0..*** elements**.**fk_elements_categories **1** categories**.** pk_categories	
(site = id)	**0..*** elements**.**fk_elements_sites **1** sites**.** pk__sites	

erosion_limits

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval ('ero-sion_li mits_i d_seq': :regcla ss)	
False	site	text	False	False	0	0	0		
False	the_geom		False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_erosion_limits	PK	id		
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	
fk_erosion_limits_sites	FK	site		

Relationships

Columns	Association	Notes
(site = id)	**0..*** erosion_limits**.**fk_erosion_limits_sites **1** sites**.** pk__sites	

excavation_grids

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval	

								('excava-tion_grids_id_seq'::regclass)	
False	site	text	False	False	0	0	0		
False	the_geom		False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_excavation_grids	PK	id		
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	
fk_excavation_grids_sites	FK	site		

Relationships

Columns	Association	Notes
(site = id)	**0..*** excavation_grids**.**fk_excavation_grids_sites **1** sites**.** pk__sites	

excavation_limits

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval('excavation_limits_id_seq'::regclass)	
False	site	text	False	False	0	0	0		
False	the_geom		False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_excavation_limits	PK	id		
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	
fk_excavation_limits_sites	FK	site		

Relationships

Columns	Association	Notes
(site = id)	**0..*** excavation_limits**.**fk_excavation_limits_sites **1** sites**.** pk__sites	

faunal_remains

Database: PostgreSQL, *Package:* Data Model

Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval ('fau-nal_re mains_ gid_se q'::regc lass)	
False	site	varchar	False	False	50	0	0		
False	the_geom		False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
faunal_remains_pkey	PK	id		
enforce_geotype_the_geom	check		((geome-trytype(the_geom) = 'POLYGON'::text) OR (the_geom IS NULL))	
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	
fk_faunal_remains_sites	FK	site		

Relationships

Columns	Association		Notes
(site = id)	**0..***	faunal_remains**.**fk_faunal_remains_sites	
	1	sites**.** pk__sites	

geology_points

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes: =geometry=point

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0		
False	name_point	varchar	False	False	50	0	0		
False	date	varchar	False	False	50	0	0		
False	surveyor	varchar	False	False	255	0	0		
False	locality	varchar	False	False	255	0	0		
False	photo1_locality	varchar	False	False	50	0	0		
False	map_50	varchar	False	False	50	0	0		
False	name_map_50	varchar	False	False	50	0	0		
False	map_10	varchar	False	False	50	0	0		
False	name_map_10	varchar	False	False	50	0	0		
False	latitude	real	False	False	0	0	0		
False	longitude	real	False	False	0	0	0		
False	altitude	real	False	False	0	0	0		
False	geographic_desc	varchar	False	False	255	0	0		
False	lithologic_outcrop	varchar	False	False	255	0	0		
False	lithologic_outcrop_desc	text	False	False	0	0	0		
False	miner-	text	False	False	0	0	0		

	alogic_and_petrographic_desc								
False	photo_1	varchar	False	False	255	0	0		
False	photo_2	varchar	False	False	255	0	0		
False	stratigraphic_seq	varchar	False	False	255	0	0		
False	strati-graphic_seq_desc	text	False	False	0	0	0		
False	sampling	boolean	False	False	0	0	0		
False	the_geom		False	False	0	0	0		=geometry=point

Constraints

Name	Type	Columns	Initial Code	Notes
pk_geology_points	PK	id		

geometry_columns

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	f_table_catalog	varchar	True	False	256	0	0		
True	f_table_schema	varchar	True	False	256	0	0		
True	f_table_name	varchar	True	False	256	0	0		
True	f_geometry_column	varchar	True	False	256	0	0		
False	coord_dimension	integer	True	False	0	0	0		
False	srid	integer	True	False	0	0	0		
False	type	varchar	True	False	30	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
geometry_columns_pk	PK	f_table_catalogf_table_schemaf_table_namef_geometry_column		

large_basalt_blocks

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval('large_basalt_blocks_id_seq'::regclass)	
False	site	text	False	False	0	0	0		
False	the_geom		False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes

Name	Type	Columns	Initial Code	Notes
pk_large_basalt_blocks	PK	id		
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	
fk_large_basalt_blocks_sites	FK	site		

Relationships

Columns	Association	Notes
(site = id)	**0..*** large_basalt_blocks**.**fk_large_basalt_blocks_sites **1** sites**.** pk__sites	

lithic_artefacts

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	gid	integer	True	False	0	0	0	nextval ('lithic _artefa cts_gid _seq'::r eg-class)	
False	site	varchar	False	False	50	0	0		
False	id	varchar	False	False	50	0	0		
False	the_geom		False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_lithic_artefacts	PK	gid		
enforce_geotype_the_geom	check		((geome-trytype(the_geom) = 'POLYGON'::text) OR (the_geom IS NULL))	
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	
fk_lithic_artefacts_sites	FK	site		

Relationships

Columns	Association	Notes
(site = id)	**0..*** lithic_artefacts**.**fk_lithic_artefacts_sites **1** sites**.** pk__sites	

main_sites

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval	

								('sites_id_seq'::regclass)	
False	main_site	text	True	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_sites	PK	id		

Relationships

Columns	Association	Notes
(main_site = id)	**0..*** sites.fk_sites_main_sites **1** main_sites. pk_sites	

news

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 4-10-2007. *Last modified on*.4-10-2007.
Notes: Holds news items that appear on the home page while current date >= start_date && <= end_date

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval('news_id_seq'::regclass)	
False	start_date	date	True	False	0	0	0		
False	end_date	date	True	False	0	0	0		
False	news_text	text	False	False	0	0	0		
False	news_title	varchar	False	False	64	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_id	PK	id		

sites

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval('sites_id_seq1'::regclass)	
False	site	text	True	True					
False	main_site	text	True	False	0	0	0		
False	has_elements	boolean	False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
UQ_sites_site	unique	site		

Name	Type	Columns	Initial Code	Notes
pk__sites	PK	id		
fk_sites_main_sites	FK	main_site		

Relationships

Columns	Association	Notes
(site = id)	**0..*** erosion_limits.fk_erosion_limits_sites **1** sites. pk__sites	
(site = id)	**0..*** excavation_grids.fk_excavation_grids_sites **1** sites. pk__sites	
(site = id)	**0..*** excavation_limits.fk_excavation_limits_sites **1** sites. pk__sites	
(site = id)	**0..*** large_basalt_blocks.fk_large_basalt_blocks_sites **1** sites. pk__sites	
(site = id)	**0..*** lithic_artefacts.fk_lithic_artefacts_sites **1** sites. pk__sites	
(site = id)	**0..*** elements.fk_elements_sites **1** sites. pk__sites	
(main_site = id)	**0..*** sites.fk_sites_main_sites **1** main_sites. pk_sites	
(site = id)	**0..*** unmodified_pebbles.fk_unmodified_pebbles_sites **1** sites. pk__sites	
(site = id)	**0..*** faunal_remains.fk_faunal_remains_sites **1** sites. pk__sites	
(site = site)	**0..*** archeology_points.FK_archeology_points_sites **1** sites. UQ_sites_site	

spatial_ref_sys

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	srid	integer	True	False	0	0	0		
False	auth_name	varchar	False	False	256	0	0		
False	auth_srid	integer	False	False	0	0	0		
False	srtext	varchar	False	False	2048	0	0		
False	proj4text	varchar	False	False	2048	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
spatial_ref_sys_pkey	PK	srid		

survey_points

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes: =geometry=point

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	fid	integer	True	False	0	0	0	nextval ('sur-vey_po ints_fi	

								d_seq'::regclass)	
False	date	varchar	False	False	50	0	0		
False	site	varchar	False	False	50	0	0		
False	level	varchar	False	False	50	0	0		
False	locality	varchar	False	False	50	0	0		
False	latitude	real	False	False	0	0	0		
False	longitude	real	False	False	0	0	0		
False	altitude	real	False	False	0	0	0		
False	map_50	varchar	False	False	50	0	0		
False	name_map_50	varchar	False	False	50	0	0		
False	map_10	varchar	False	False	50	0	0		
False	name_map_10	varchar	False	False	50	0	0		
False	level_thick_min	real	False	False	0	0	0		
False	level_thick_max	real	False	False	0	0	0		
False	alt_upp_level_min	real	False	False	0	0	0		
False	alt_upp_level_max	real	False	False	0	0	0		
False	chronology	varchar	False	False	50	0	0		
False	observations	text	False	False	0	0	0		
False	geologi-cal_observations	text	False	False	0	0	0		
False	gps_notes	text	False	False	0	0	0		
False	photo_locality	varchar	False	False	50	0	0		
False	photo_level	varchar	False	False	50	0	0		
False	the_geom		False	False	0	0	0		=geometry=point

Constraints

Name	Type	Columns	Initial Code	Notes
pk_survey_points	PK	fid		

unmodified_pebbles

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval('unmodified_pebbles_id_seq'::regclass)	
False	site	text	False	False	0	0	0		
False	the_geom		False	False	0	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_unmodified_pebbles	PK	id		
enforce_dims_the_geom	check		(ndims(the_geom) = 2)	
enforce_srid_the_geom	check		(srid(the_geom) = 20137)	
fk_unmodified_pebbles_sites	FK	site		

Relationships

Columns	Association	Notes
(site = id)	**0..*** unmodified_pebbles.fk_unmodified_pebbles_sites **1** sites. pk__sites	

web_items

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.4-10-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	integer	True	False	0	0	0	nextval ('web_items_id_seq1 '::regclass)	
False	page	char	True	False	3	0	0		
False	panel	char	True	False	3	0	0		
False	item	char	False	False	3	0	0		
False	title	varchar	True	False	64	0	0		
False	content	text	True	False	0	0	0		
False	format	char	True	False	3	0	0		
False	gis_link	varchar	False	False	255	0	0		
False	size_in_panel	integer	False	False	0	0	0	50	
False	align_in_panel	varchar	False	False	10	0	0	'right':: character varying	

Constraints

Name	Type	Columns	Initial Code	Notes
pk_web_items	PK	id		
fk_web_items_web_pages	FK	page		
fk_web_items_web_panels	FK	panel		

Relationships

Columns	Association	Notes
(panel = id)	**0..*** web_items.fk_web_items_web_panels **1** web_panels. pk_web_panels	
(page = id)	**0..*** web_items.fk_web_items_web_pages **1** web_pages. pk_web_pages	

web_pages

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	char	True	False	3	0	0		
False	title	varchar	True	False	64	0	0		
False	panels	char	False	False	3	0	0		

False	gis_link	varchar	False	False	255	0	0		

Constraints

Name	Type	Columns	Initial Code	Notes
pk_web_pages	PK	id		

Relationships

Columns	Association		Notes
(page = id)	**0..***	web_items.fk_web_items_web_pages	
	1	web_pages. pk_web_pages	

web_panels

Database: PostgreSQL, *Package:* Data Model
Detail: *Created on* 12-9-2007. *Last modified on*.12-9-2007.
Notes:

Columns

PK	Name	Type	Not Null	Unique	Len	Prec	Scale	Init	Notes
True	id	char	True	False	3	0	0		
False	title	varchar	True	False	64	0	0		
False	items	char	False	False	3	0	0		
False	gis_link	varchar	False	False	255	0	0		
False	timeline_start	real	False	False	0	0	0	0.0	
False	timeline_end	real	False	False	0	0	0	0.0	

Constraints

Name	Type	Columns	Initial Code	Notes
pk_web_panels	PK	id		

Relationships

Columns	Association		Notes
(panel = id)	**0..***	web_items.fk_web_items_web_panels	
	1	web_panels. pk_web_panels	