



THE VISI PROTOTYPE
For image retrieval

Bjørge Næss
September 2007

Technical report

CAIM-TR-1

TABLE OF CONTENTS

1.	Introduction	1
2.	End-user functionality	2
2.1.	Getting started	2
2.1.1.	Query Specification – left-hand panel.....	3
2.1.2.	Using an existing image – Query by Example.....	3
2.1.3.	Drawing a seed image – Query by Sketch	4
2.1.4.	Setting weights – the center panel.....	5
2.2.	Querying the database	5
3.	System Implementation.....	7
3.1.	Technology	7
3.2.	The web application	7
3.2.1.	Overall architecture	7
3.2.2.	How-to extend the application	10
3.3.	The database backend requirements.....	11
3.4.	Requirements.....	12
3.5.	How-to.....	13
3.5.1.	Extend functionality	13
3.5.2.	Use VISI with other DBMSs.....	13
3.5.3.	Add a new DB to an existing set of databases	13
3.6.	Known compatibility issues	13
4.	References	14
5.	Appendix	15
5.1.	Error messages and what to do.....	15
5.1.1.	Upload error	15
5.1.2.	Error while performing search	15
5.1.3.	Error when trying to fetch imageinfo.....	15
5.1.4.	Unhandled error.....	15

1. INTRODUCTION

The [VISI \(Vortex Image Search Interface\)](#)¹ search engine is a prototype of a web-based Content Based Image Retrieval system, available at <http://bulmeurt.uib.no:8500/caim/VISI/>. It was developed as a basic test bed for evaluation and exploration of various aspects of visual image retrieval. Our main motivation for building VISI was to create an image retrieval system with basic CBIR functionality without the need for expert knowledge of CBIR functionality. This would allow researchers to explore and evaluate different aspects of visual image retrieval systems, such as usability, user interaction, user behaviour and new approaches to interfaces without an expert knowledge of CBIR fundamentals.

Based on this, we decided to create a system based on the following requirements:

- Support for both Query-By-Example and Query-By-Sketch.
- Visual queries based on colour, shape, texture and spatial distribution.
- Provide easy query refinement for non-expert CBIR users.

The system was developed using a ColdFusion² front-end interface to Oracle 9i (O9i) InterMedia CBIR software. Visual Query Specification is provided through a sketch interface, illustrated in Figure 3. This tool is based on a Java Applet, *J-Painter*³, and provides a set of basic drawing tools: Freehand drawing; basic geometric shapes such lines, rectangles, circles; colour selection and a limited palette of pens. We believe the sketch tool is representative of state-of-the art visual query specification interfaces. The system is currently operating on top of Oracle 10g, but other CBIR-enabled databases can be easily added.

¹ The original version of VISI was developed by Øyvind Osdal in 2005 within the [Virtual Exhibits on Demand, VED](#) project, funded by the Norwegian Research Council (NFR).

² <http://www.adobe.com/products/coldfusion/>

³ <http://www.izhuk.com/painter/>

2. END-USER FUNCTIONALITY

2.1. Getting started

The opening screen, shown in Figure 1 provides some general information about the VISI prototype. After pressing the Start button the user will see the main screen, shown in Figure 2.

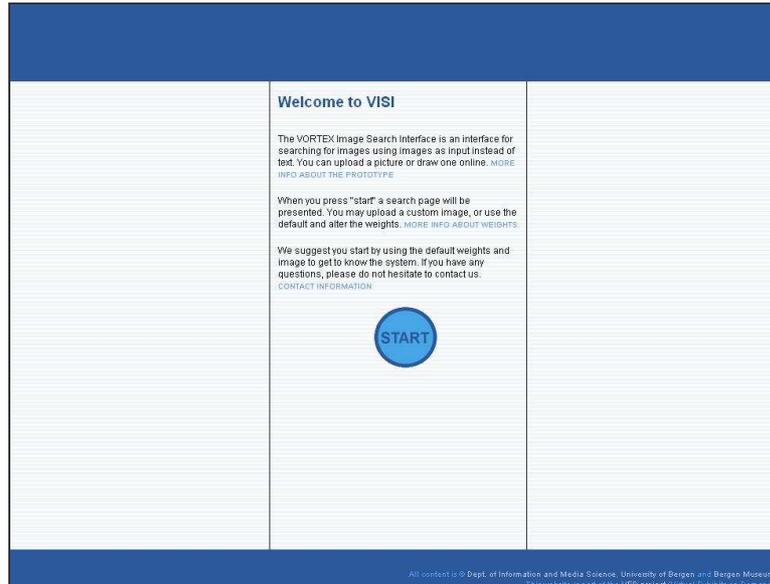


Figure 1: The VISI welcome screen

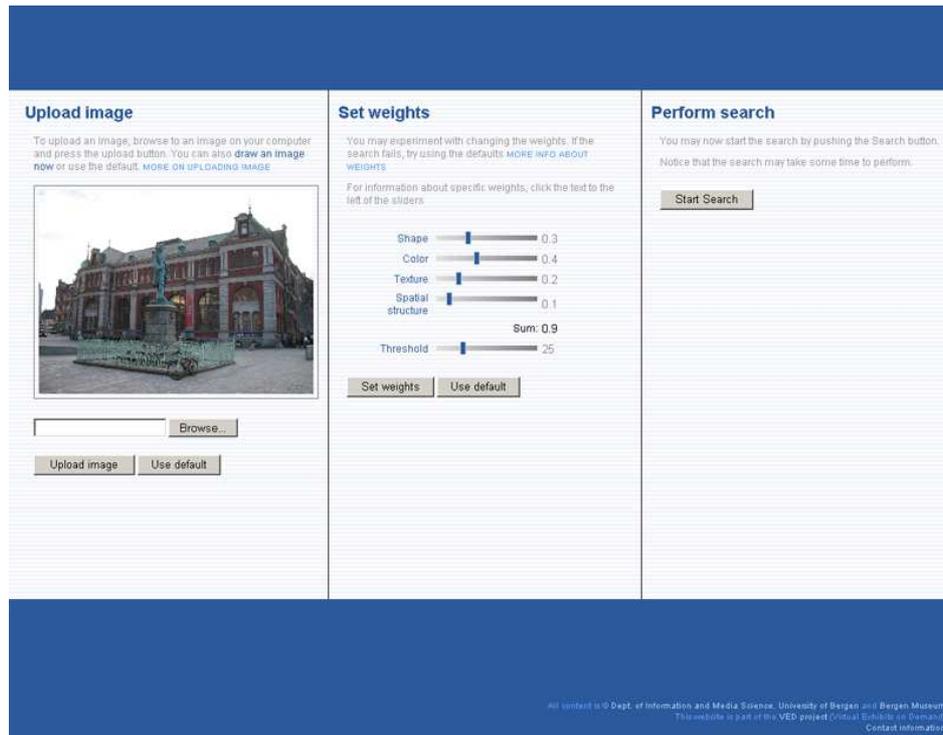


Figure 2: Specification of a visual query

The main VISI interface is divided into three sections, shown in Figure 2.

2.1.1. Query Specification – left-hand panel

There are two alternatives for specification of a search image:

1. Query-By-Example. The user can select an existing image from their hard drive to use as a query.
2. Query-By-Sketch. The user can create their own query image by a set of basic drawing tools.

2.1.2. Using an existing image – Query by Example

By default, an example image is presented in the left-hand panel of the query specification screen, as shown in figure 2. The image to be uploaded can be of any of the formats supported by Oracle InterMedia (currently: BMPF, CALS, FPIX, GIFF, JFIF, PBMF, PGMF, PPMF, PNMF, PCXF, PICT, PNGF, RPIX, RASF, TGAF, TIFF and WBMP)⁴.

⁴ To view the full list of supported formats in Oracle InterMedia, please refer to http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14297/ap_imgfmts.htm#g653513

To select an image, click the “Browse...” button and locate the image from the hard-drive. Once the image is selected, click the “Upload image” button. This might take a while depending on the internet connection speed. When the image is successfully uploaded, it will show up in this section by replacing the default image.

2.1.3. Drawing a seed image – Query by Sketch

In addition to being able to select a user-defined search image and retrieve images similar to it, the user can also sketch or draw an illustration of what she wants to return. This can be done by clicking the “draw an image now” link in the descriptive text in the section for uploading images in the left panel. A new screen appears, and an image editor is started. By default, this shows an example illustration of a whale. To clear and start a new drawing, click the icon at the end of the introductory sentence.

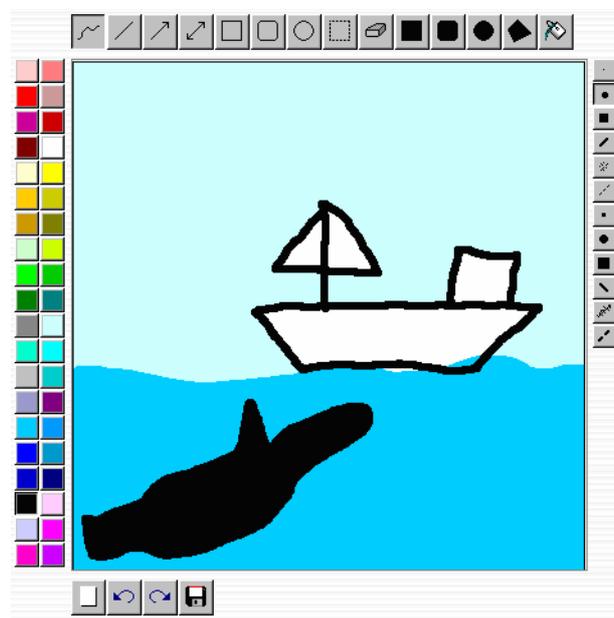


Figure 3: VISI Query-by-example sketch tool with example query

Basic sketches can be drawn by using the various tools in the toolbar surrounding the drawing pad. When complete, the drawing/sketch can be used as a query, indicating the visual characteristics that should be returned in the query result set.

The drawing interface (Figure 3) provides a set of basic drawing tools: Freehand drawing; basic geometric shapes such as lines, rectangles, circles; colour selection and a limited palette of pens. In addition, there are supporting tools such as an eraser, a colour filler, copy-and-paste, and undo/redo. The different tools can be used alone or in combination to create a visual query.

Once a seed image is selected or drawn and uploaded, it will remain throughout the user-session, or until it is replaced by another one.

2.1.4. *Setting weights – the center panel*

After the seed image has been selected/drawn and uploaded, it's time to specify a weighting scheme for the characteristics of the query using the center panel of the query specification interface, re. Figure 2. In most cases, the default values work well. However, if a search returns no results, it might be worthwhile adjusting the weights. The sum of “Shape”, “Color” and “Texture” values cannot exceed 1, while “Spatial Structure” and “Threshold” are independent weights.

The threshold-parameter denotes the maximum similarity-distance between the seed image and images that should be included in the result set. The scale goes from 0 – 100 where 0 meaning no distance in similarity (i.e. the images must be identical to the search image in order to be returned), and 100 meaning no similarity at all (will most likely return all images in the collection).

Note that after a weight is changed, the “Set weights” button must be clicked to apply the changes. For more information about weights, please refer to the “more info about weights” link or the Oracle InterMedia documentation⁵.

2.2. *Querying the database*

When the seed image is uploaded and weights are set, it is time to perform the actual search. This is done simply by hitting the “Start Search” button in the right-hand panel.

The query will then be processed against the database, and similar images will be returned and presented to the user as thumbnails in a (3 * n) grid. A summary of the current search parameters is also presented for convenience. Clicking on a thumbnail presents structural information about the particular image (width, height, file format, etc.), along with a selection of metadata-attributes extracted from the image (at the time writing, these attributes are caption, date and time taken, camera vendor, camera model, software, whether the flash was fired, when the photo was taken, and GPS information), as shown in Figure 4. The information shown depends on what kind of metadata that is stored with the image.

A menu with different options, among others, the possibility to use this image as a seed image is also provided. Choosing this option causes any previously uploaded images to be discarded and replaced by this image, and a new query with this as seed image is ready to take place.

⁵ http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14297/ch_imgref.htm#sthref962

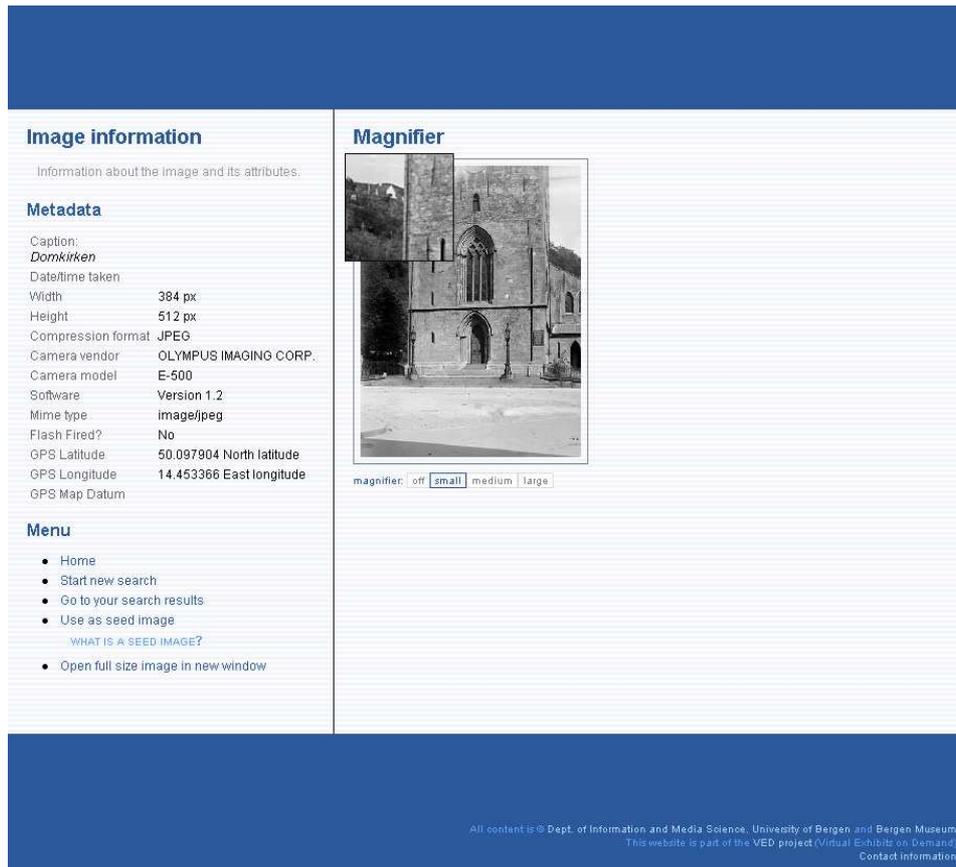


Figure 4: Viewing image details

The user might be interested in viewing image details. A possibility to magnify a subsection of the present image is provided by clicking on the image and dragging the magnifier-window to the area of interest.

3. SYSTEM IMPLEMENTATION

3.1. Technology

The test environment currently uses the following technologies: Oracle 10g⁶ w/intermedia, ColdFusion (currently version MX7), HTML, CSS, JavaScript Java Applet (the JPainter application).

3.2. The web application

The web application runs on a ColdFusion MX7-enabled server. The future developer should be familiar with ColdFusion (or any other server-side scripting-language), JavaScript, basic HTML and concepts like session variables and handling file uploads through the HTTP protocol.

3.2.1. Overall architecture

All files and scripts are subsequently referred to by using their relative path starting on the application root (i.e. where Application.cfm is located), as illustrated in Figure 5, below.

⁶ VISI with oracle 10g is compatible with earlier versions that used Oracle 9i

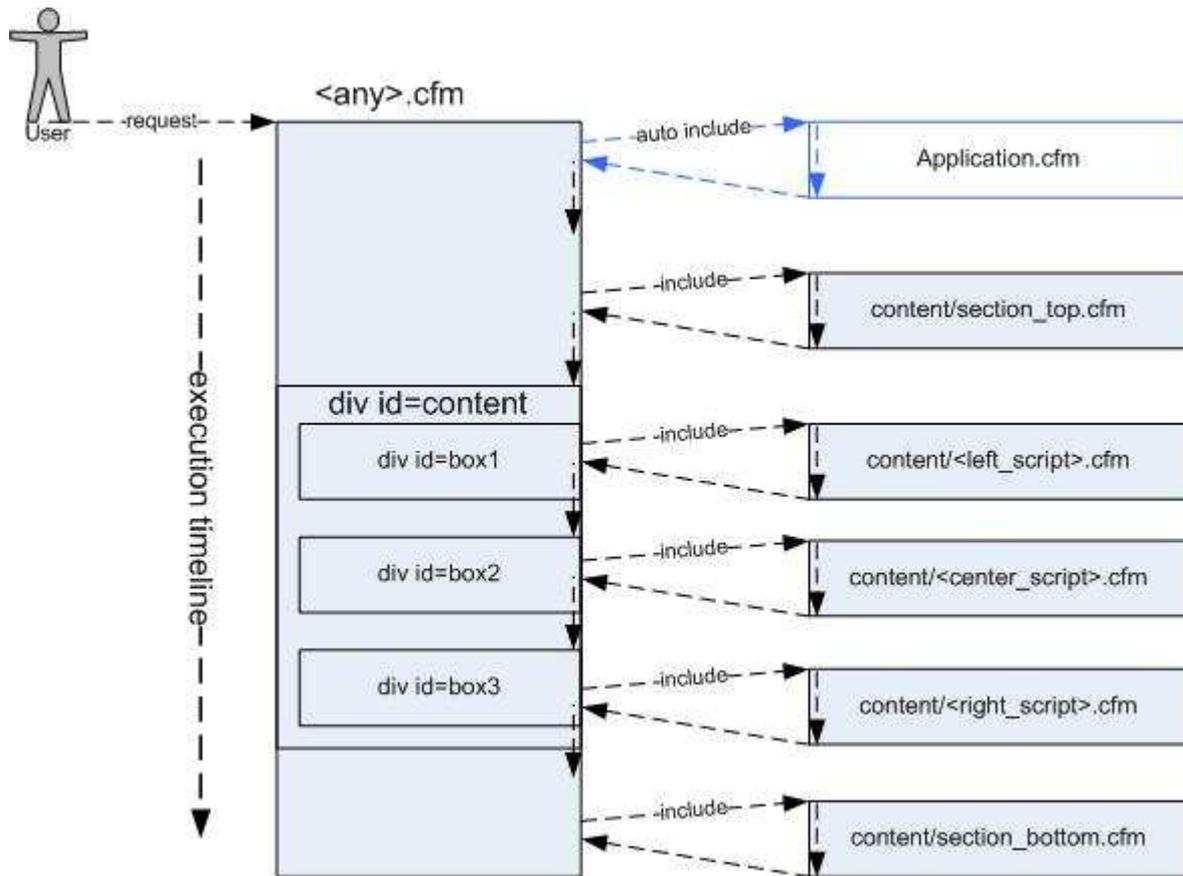


Figure 5: Overall VISI structure

The Application.cfm script

Here, the application-level settings are defined. This script is included implicitly by the ColdFusion server before any user-requests are processed (for more information, see the ColdFusion documentation). Among other configuration parameters, it defines a global default error-handler (errors/error_dispatcher.cfm).

Application.cfm also defines which data source (Oracle database link) to use (described in section 3.2.2) and defines the default values for weights (shape, color, texture and spatial structure) and the default threshold value to be used when issuing queries to Oracle. The remaining configuration parameters defined in this script are described in code-comments and should be fairly easy to grasp.

Other script files

The page layout as seen by the user can be divided into five sections, as shown in Figure 6.

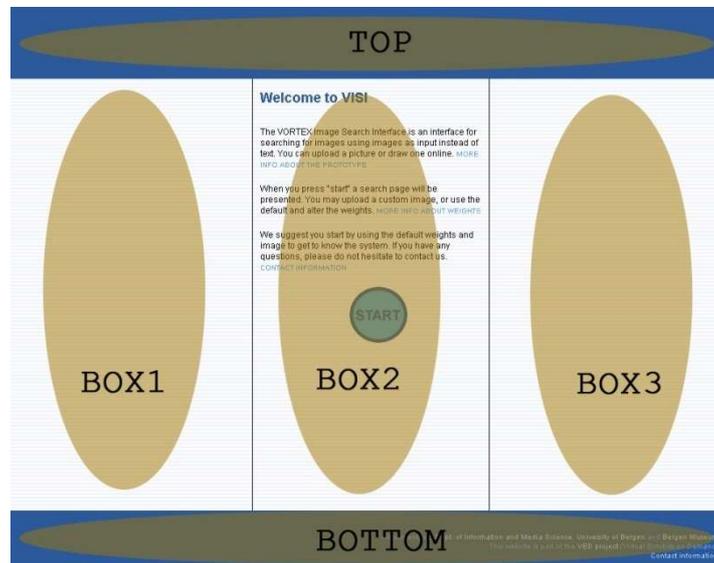


Figure 6: The different parts of a page

This structure corresponds to different parts of code in respective script files. The top-section is global and is included in each page by a `cinclude`-statement:

```
<cinclude template="content/section_top.cfm">
```

The Box1 section is what is contained in a `<div>` tag with the CSS-class 'Box1', i.e.:

```
<div id="box1">
  <!-- The content in the left section goes here --->
</div>
```

Similarly, the content of a `box2`-class `<div>` tag goes in the center-section and `box3` represents the right section.

The bottom-section works the same way as the top, by including a global file by a `cinclude` statement:

```
<cinclude template="content/section_bottom.cfm">
```

3.2.2. How-to extend the application

To create an Oracle database link (ODBC connection) for ColdFusion

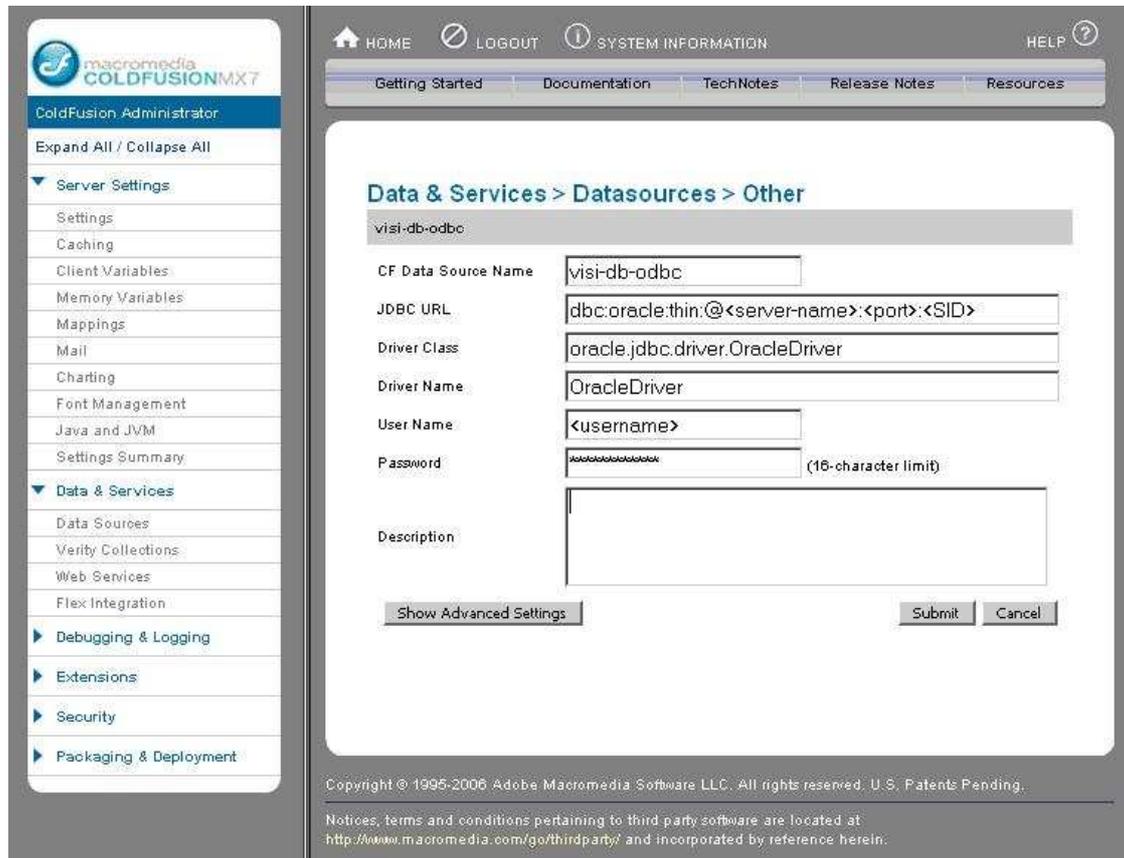


Figure 7: Adding a data source in ColdFusion Administrator

A data source must be created by the ColdFusion administrator giving access to an Oracle database. To do this, the Java-based JDBC driver for Oracle⁷ has to be downloaded and copied to the ColdFusion 'Class Path'. It is important that the data source is named '**visi-db-odbc**'. The creation of the data source is done by selecting "Data sources" under the "Data & Services" menu item in the ColdFusion administrator panel (see figure 7 above) and then filling in information on the following format (values in brackets <> must be replaced by actual values), as shown in the following table.

⁷ See http://www.oracle.com/technology/software/tech/java/sqlj_jdbc

CF Data Source Name:	visi-db-odbc
JDBC URL:	dbc:oracle:thin:@<server-name>:<port>:<SID>
Driver Class:	oracle.jdbc.driver.OracleDriver
Driver Name:	OracleDriver
User Name:	<username>
Password:	<password>
Description:	<any>

Table 1: Creation of the ColdFusion → Oracle link.

When done, choose “Show advanced parameters” and increase the BLOB buffer-size value (we used 64,000,000 bytes) in order to be able to fetch high-resolution images from the database.

To add a new script

Adding a new script to VISI should be fairly easy. It is done by creating the new script file, including the cinclude-statements for the top and bottom sections and creating the box1-3 class-div tags as described in section 3.2.1. The file ‘template.cfm’ provides an empty framework for creating new pages and should be used as a guideline.

3.3. The database backend requirements

VISI is dependent on some available features and structure in the underlying database system. It is important that the future developer has knowledge about the name of the image-table defined in the database. The database backend should also have a function or stored procedure that enables searching for an image using the URL of a seed image and specifying values for the different weights as parameters (the call to the search procedure is done in ‘handlers/performsearch.cfm’). Further, the stored procedure must return the matching results in the form of a comma-separated list of image-IDs. In Oracle the signature of this stored procedure is as follows:

```
PROCEDURE SEARCH (
  p_Url      IN VARCHAR2,
  p_Filename IN VARCHAR2,
  p_Shape    IN VARCHAR2,
  p_Color    IN VARCHAR2,
  p_Texture  IN VARCHAR2,
  p_Spatial  IN VARCHAR2,
  p_Threshold IN VARCHAR2,
  p_Results  OUT NOCOPY VARCHAR2 );
```

The resulting image-IDs (the result set) are passed to the OUT-parameter p_Results and further processed by VISI. This procedure can be renamed or contained in a package if desired. If so, the procedure call in ‘handlers/performsearch.cfm’ script must be updated to match the actual name. It should, however keep the basic functionality stated in the signature of the procedure above.

3.4. Requirements

The basic requirement is that the database backend has a table containing an ORDSYS.ORDIMAGE-column and a stored procedure that can receive the URL of an image, process a CBIR-query to the database system which in turn will return images that are evaluated to be similar.

In addition to calling the stored procedure described in the above section, VISI scripts also issues SQL-statements directly to the database. This means that VISI must have “knowledge” about the database structure. In particular, the table and attribute where the images are stored must be known to the application developer. The following scripts issue queries to the table containing the ORDSYS.ORDIMAGE-column:

```
lookcloser.cfm
content/showimage.cfm
content/downloadimage.cfm
uploads/showimage.cfm
```

Thus, modifying the database structure also implies reformulating the queries in these files.

To utilize the metadata capabilities supported by many digital cameras today, the database system should also provide the functionality to fetch this information. In the ORDSYS.ORDIMAGE-object type, methods support both reading and writing metadata that conform to different standards⁸.

⁸ http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14297/ap_xmlschms.htm

3.5. *How-to*

3.5.1. *Extend functionality*

Extending the functionality of VISI may involve both extending the database backend functionality (i.e. stored procedures) and the source code of VISI itself (i.e. support for both text and image queries by adding a text-field to the search form). The latter, however, may also lead to changes in the database backend (i.e. adding parameters to the search-procedure).

3.5.2. *Use VISI with other DBMSs*

Even though it is built on top of Oracle 10g, VISI is not dependent on a specific DBMS implementation. The only requirement is that the DBMS supports CBIR-queries. Using another DBMSs would, however, imply some level of customization. Of the most importance is adapting the list of scripts that communicates with the database to the new DBMS (as described in section 3.4).

3.5.3. *Add a new DB to an existing set of databases*

Using VISI with more than one database would require the new database(s) to provide a backend system that implements the requirements described in section 3.4. There is currently no functionality for letting the user select which database(s) to issue queries to. Nor is there functionality for merging result sets from the different databases.

3.6. *Known compatibility issues*

For best performance and stability, Internet Explorer should be used when accessing the VISI prototype. At the time writing, there are known compatibility issues with Mozilla-based browsers:

The weight-slider is fixed. It is possible to change the different weights, but the slider indicator is fixed at zero and provides no visual feedback as the values change. However, the actual weight (from 0 – 1) will change instantly as the slider is dragged by the user.

4. REFERENCES

Adobe ColdFusion 8, <http://www.adobe.com/products/coldfusion>
Retrieved 23.06.2007

J-Painter - online image editor, <http://www.izhuk.com/painter>
Retrieved 22.06.2007

Image File and Compression Formats, in *Oracle® interMedia Reference*, http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14297/ap_imgfmts.htm#g653513
Retrieved 31.08.2007

ORDImage and ORDImageSignature in *Oracle® interMedia Reference*, http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14297/ch_imgref.htm#sthref962
Retrieved 31.08.2007

JDBC, SQLJ, and Oracle JPublisher, in *Oracle Technology Network*,
http://www.oracle.com/technology/software/tech/java/sqlj_jdbc
Retrieved 31.08.2007

Metadata Schemas, in *Oracle® interMedia Reference*,
http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14297/ap_xmlschms.htm
Retrieved 22.06.2007

5. APPENDIX

5.1. Error messages and what to do

5.1.1. Upload error

Upload errors may be caused by invalid write permissions to the upload folder or wrong file-format on uploaded files.

What to do

First, make sure the file you are uploading is, in fact an image. Also verify that the full path of the upload folder is written correctly in the Application.cfm file and that the folder is writeable by the ColdFusion-user (if the server is on Linux, try typing 'chmod 777 uploads' in the application root directory).

5.1.2. Error while performing search

This error is probably caused by an invalid procedure call to the database. Either the issued procedure call is misspelled, has a wrong number of parameters or the parameter types are wrong.

What to do

First, make sure the procedure call goes to a procedure that actually exists in the target database. Second, verify that procedure call has the right number of parameters using the correct datatype. Further, verify that the parameter that will contain the returned items has the value 'out' set for the parameter 'cfproparam'.

5.1.3. Error when trying to fetch imageinfo

This error may be due to a misspelled SQL-query statement.

What to do

Make sure the SQL-statement in the originating file is syntactically correct and not resulting in an error when issued directly to the database. Also make sure the 'datasource' attribute of the 'cfquery' attribute is the same as defined in Application.cfm.

5.1.4. Unhandled error

This error is caused by an unknown situation. Read the error message carefully to identify its cause.