

2 USING A WEB SERVICE

The web services used in this project are the **Kulturmiljosok** web service which uses the **SRU** protocol and the web service created for the **ARK** project which uses a simple URL to get access to **ARK** data as **XML**.

2.1 KULTURMILJOSOK WEB SERVICE

The **SRU** (SearchRetrieve Request) is an **HTTP** URL which can be used to search the base URL (before the ?) for the request in the search part (after the ?) and retrieve the search as an **XML**. The search part consists of parameters separated by & with the structure “key=value”. An **SRU** could look like this:

```
http://www.kms.raa.se/cocoon/kulturmiljosok/sru?version=1.1&operation=searchRetrieve
&maximumRecords=0&recordSchema=http://raa.se/SRU/object/1.0&query=parish=880andprovin
ce=Sm
```

The part before the question mark is the client script which will call the web service. In this case the **SRU** will search **Kulturmiljosok** web service through the client script at:

```
http://www.kms.raa.se/cocoon/kulturmiljosok/sru
```

It is also possible to search for records in the public **FMIS** through the script:

```
http://www.kms.raa.se/cocoon/fmis-public/sru
```

After the question mark are the variables which are sent to the **SOAP** client to enable it to call the **SOAP** server. First is the version of the **SRU** call:

```
version=1.1
```

The operation the **SRU** will perform is called searchRetrieve. It is possible to use other operation but this one will retrieve the records called in the **SRU**.

```
operation=searchRetrieve
```

The third variable in the **SRU** URL is maximum records and if maximum records is 0 the search will return the number of records as an answer. If the maximum records is 1 the search of the **Kulturmiljosok** web service will return a record for each service it has searched with the number of records and a link to access these records.

If the maximum records is one or more on the **FMIS** public search it will return the first couple of records in the search together with the number of records total. However, **RAÄ** warns not to allow the maximum records to be too large.

maximumRecords=0

After this is the variable which determines the schema on which the retrieved **XML** is built:

recordSchema=http://raa.se/SRU/object/1.0

At the end of the URL is the query sent to the client with the parameters used for the **SOAP** call:

query=parish=880andprovince=Sm

The heritage web service (**Kulturmiljosok**) can use all the keys in the table below, while the **FMIS** public search can only use province, parish and county. The heritage web service can also use the Subsys key to decide which databases to search.

Key Name	Amount	Description (se)	Value Format
province	0-1	Landskapskod	Two letters – code
parish	0-1	Sockenkod	Integer – code
county	0-1	Länskod	Two digit – code
municipality	0-1	Kommunkod	Four digit – code
txt	0-1	Fritext	SRU-coded text
placeName	0-1	Ortnamn	SRU-coded text
Subsys	0-M	System att söka i	Name from catalog

The **SRU** URL above is actually written in **SRU** code like this:

http://www.kms.raa.se/cocoon/kulturmiljosok/sru?version=1.1&operation=searchRetrieve&maximumRecords=0&recordSchema=http%3A%2F%2Fraa.se%2FSRU%2Fobject%2F1.0&query=%28parish%3D%27880%27%29+and+%28province+%3D+%27Sm%27%29

The **SRU** code used in the URL makes use of special symbols through % code (see table below). These symbols must also be used in the placeName and txt keys and thus “Bronsålder” (Bronze Age) becomes “Brons%**C3**%**A5**lder”.

Symbol	& code
=	%3D
/	%2F
SPACE	%20
Å – translate to the UTF-8 code: C3A5	%C3%A5

2.2 TRANSFORMING THE XML OUTPUT

As we now have two datasets output as **XML** we can either display them as they are or transform them into **XML** with a different formatting or into a **HTML** webpage.

For this project several `my_webservice` scripts have been created in order to manage the transformation of the **XML**.

A) `my_ark.php`

Deals with the output of the **ARK** web service and create the right URL to send to the client.

B) `my_fmisp.php`

Deals with the output of the **FMIS** web service and create the right URL to send to the client.

C) `my_midasp.php`

Deals with the combined **Midas XML** output of the **FMIS** and **Sintana** web services.

These scripts are built in a similar fashion beginning with a series of variables being set through the GET action as above in the **SOAP** client. Some are set as defaults.

```
if ($_GET["method"]){
    $method = $_GET["method"];
}else{
    $method = 'getModule';
}
}
```

Then the **XML** name is created as a string which points to the **SOAP** client. This loads the **XML** file created by the web server. Then a `DOMDocument` object is created and loaded with the **XML** file.

```
$xml_name = $php_name . '?method=' . $method . '&ark_name=' . $ark_name .
    '&mod_key=' . $mod_key . '&field_name=' . $field_name . '&field_value=' .
    $field_value . '&item_value=' . $item_value;
$xml = new DOMDocument();
$xml->load($xml_name);
```

The files are then divided by a switch into different output types. The **HTML** type will take the **XML** file and transform it into **HTML** through predefined **XSLT** style-sheets.

```
case 'html':
    $xslt_name = 'xslt/' . $method . '_html.xslt';
    $xslt = new DOMDocument();
    $xslt->load($xslt_name);

    $proc = new XsltProcessor();
    $xslt = $proc->importStylesheet($xslt);
    $proc->setParameter(null, ' ', ' ');
    $newdom = $proc->transformToDoc($xml);
    include_once('../my_diss/css/header.php');
    print $newdom->saveXML();
    include_once('../my_diss/css/footer.php');
    break;
```

The original **XML** output methods just allows the **PHP** to output the original **XML** which is received from the web service:

```
case 'ark_xml':
header ("content-type: text/xml");
print $xml->saveXML();
break;
```

The `midas_xml` output takes the **XML** received from the web service and transforms through a predefined **XSLT** style-sheet into **Midas** format **XML** based on choices made by the project for parts of each dataset.

```
case 'midas_xml':
if($method=="getItemList"){
header ("content-type: text/xml");

$xml_name = 'xslt/' . $method . '_midas.xml';
$xml = new DOMDocument();
$xml->load($xml_name);

$proc = new XsltProcessor();
$xml = $proc->importStylesheet($xml);
$proc->setParameter(null, ' ', ' ');
$newdom = $proc->transformToDoc($xml);
print $newdom->saveXML();
}else{
print("The Midas XML is only available for the getItemList method");
}
break;
```

The **HTML** output uses the same CSS stylesheet as the main website which keeps a similar look throughout the web pages.

2.3 CREATING THE HERITAGE PORTAL

The Heritage Portal basically consists of four files:

1. `my_midas.php`: This script is described above and simple calls the script `my_midas_xml.php` with the parish name and number as a variable. It controls the output of either **HTML** or **XML**:

```
$xml_name=$dir_diss . "/my_midas/my_midas_xml.php?parish=" . $parishget;
$xml = new DOMDocument();
$xml->load($xml_name);
```

2. `my_midas_xml.php`: This script received the variable with the parish name and number and explodes it so that parish name becomes the field value sent to the **ARK** web service and parish number becomes parish which is sent to the **FMIS** web server:

```
// Retrieves a parish name and number
if ($_GET["parish"]){
$parishget=$_GET["parish"];
// Explode to get the number for FMIS and the name for ARK
$parishplace = explode("-",$_GET["parish"]);
```

```

    // Parish variable for FMIS
    $parish = $parishplace[0];
    // Field value for ARK
    $field_value = $parishplace[1];
}

//FMIS search
$xml_fmis_name = $dir_diss . '/my_fmis/my_fmis.php?output=midas_xml&parish=' .
$parish . '&max_rec=' . $max_rec;
$xml_fmis = new DOMDocument();
$xml_fmis->load($xml_fmis_name);

//Sintana ARK search
$xml_sintana_name = $dir_diss .
"/my_ark/my_ark.php?output=midas_xml&field_name=placename
&ark_name=sintana_ark&mod_key=sin_cd&method=getItemList
&field_value=" . $field_value;
$xml_sintana = new DOMDocument();
$xml_sintana->load($xml_sintana_name);

```

3. midas.xml: This script takes the two **Midas** formatted web service outputs retrieved above and combines the into one **XML** output:

```

<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" indent="yes" encoding="UTF-8" />
  <xsl:strip-space elements="*" />
  <xsl:variable name="fmis" select="document('fmis.xml')" />

  <xsl:template match="/">
    <xsl:apply-templates select="monuments" />
  </xsl:template>

  <xsl:template match="monuments">
    <monuments xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
xsi:noNamespaceSchemaLocation='http://www.roued.com/dissertation/my_diss/midas/schem
a/midas_monument.xsd'>
      <meta>
        <title>Combined Sintana and FMIS midas search</title>
      </meta>

      <xsl:apply-templates select="monument" />

      <xsl:apply-templates select="$fmis/monuments/monument" />

    </monuments>
  </xsl:template>

  <xsl:template match="monument">
    <xsl:copy-of select="." />
  </xsl:template>
</xsl:stylesheet>

```

4. midas_html.xml: This script simple takes the new **Midas** formatted single **XML** and outputs it as **HTML** styled like the rest of the web-site:

```

<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:srw="http://www.loc.gov/zing/srw/"
xmlns:object="http://raa.se/SRU/object/1.0">
  <xsl:output method="html" doctype-
system="http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd" doctype-public="--
//W3C//DTD XHTML 1.0 Strict//EN" indent="yes"/>
  <xsl:template match="/">
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <title>
          Midas Output
        </title>

```

```

        <link rel="stylesheet" type="text/css" media="screen, print"
href="../my_diss/css/diss_style.php"/>
    </head>
    <body>

        <div class="textbox">
            <h1 title="title">Combined FMIS and Sintana search</h1>
            <xsl:apply-templates select="//monument"/>
        </div>
    </body>
</html>
</xsl:template>

<xsl:template match="//monument">
    <div class="inbox">
        <h3> <xsl:value-of select="appellation/name"/> </h3>
        <p>ID: <xsl:value-of select="appellation/identifier"/></p>
        <p>FROM: <a>
            <xsl:choose>
                <xsl:when test="recordmeta/source='Sintana'">
                    <xsl:attribute name="href">../my_ark/my_ark.php?method=getItem
                    &ark_name=sintana_ark
                    &mod_key=sin_cd
                    &item_value=<xsl:value-of select="appellation/identifier"/>
                </xsl:attribute>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:attribute name="href">http://www.kms.raa.se/cocoon/fmis-
public/search.html
                </xsl:attribute>
                </xsl:otherwise>
            </xsl:choose>
            <xsl:value-of select="recordmeta/source"/>
        </a></p>
    </div>
</xsl:template>

</xsl:stylesheet>

```

These four scripts together with a web form as a search interface is all that is to it:

```

<h2>Heritage Portal: combined Midas search</h2>
    <form method="get" action="../my_midas/my_midas.php" >
        <label for="output" title="Heritage portal search">
            <h3>Output method: </h3>
            <p>Please choose an output method - for navigation to
            other pages choose HTML</p>
            <select name="output" tabindex="15" id="output"
accesskey="15" title="Choose output format">
                <option value="html">HTML</option>
                <option value="midas_xml">Midas XML</option>
            </select>
        </label>
        <hr />
        <label for="parish" title="Parishes">
            <h3>Parish:</h3>
            <p>Please choose a parish - only Lund and Borrby in
            Sweden are available in this service</p>
            <select name="parish" tabindex="16" id="parish"
accesskey="16" title="Choose parish">
                <option value="1298-Lund">Lund</option>
                <option value="1029-Borrby">Borrby</option>
            </select>
        </label>
        <hr />
        <br />
        <label for="submit" title="Submit">
            <input type="submit" value="View result" tabindex="17"
            id="submit" accesskey="17" title="View results"/>
        </label>
    </form>

```