

Creating and Using Multi-protocol Bioinformatic Web Services

Bioinformatics Technology Conference (BioCon
2003)

February 06, 2003. San Diego

Jesus M. Castagnetto, Ph.D.
<jesusmc@scripps.edu>

<http://metallo.scripps.edu/talks>

Web Services

- o REST (aka HTTP methods), XML-RPC, and SOAP
- o Why do we need Web Services in Bioinformatics

The MDB Web Application

- o Background
- o Its Architecture
- o Web Services Implementation (with examples)
- o Access modes: synchronous, asynchronous

Summary and Acknowledgements

My old 'Web API' point of view ...

A web Application Program Interface, is a set of callable functions or methods that a web application exposes as a public interface for communication with external application.

... and the modern Web Services definition

"A Web service is a collection of functions that are packaged as a single entity and published to the network for use by other programs. Web services are building blocks for creating open distributed systems..." [1]

[1] from "The Web services (r)evolution, Part 1",
<http://www-106.ibm.com/developerworks/webservices/library/ws-peerl.html>

REST is the acronym for REpresentational State Transfer, and is a term coined by Roy Fielding [1] in a dissertation attempting to describe formally the Web architectural style.

The argument is that the HTTP methods (GET, POST, etc.) define a limited set of verbs that can be applied to an (in principle) infinite number of nouns (URIs).

This design has some beneficial effects:

- a. It allows simpler interoperation between otherwise uncoordinated agents (web clients, servers, spiders, etc.)
- b. The HTTP semantics constitutes a coordination language general enough to be of use in any desired communication pattern. Its methods make it akin to message passing or function calling protocols.
- c. Offering a service is as simple as publishing a URL.

[1] Roy Fielding's Ph.D. dissertation "Architectural Styles and the Design of Network-based Software Architectures"

"It's a spec and a set of implementations that allow software running on disparate operating systems, running in different environments to make procedure calls over the Internet." [1]

"It's remote procedure calling using HTTP as the transport and XML as the encoding. XML-RPC is designed to be as simple as possible, while allowing complex data structures to be transmitted, processed and returned." [1]

It supports server introspection. Making it easy for a client application to interrogate and discover the available methods and their parameters.

XML-RPC also supports "box-carrying" of requests, i.e. sending a series of RPC calls at one time. Uses the `system.multicall` method.

It does not support message passing/redirection, or complex data type definitions. But it does support mappings to simple and structured data types to/from many languages.

[1] from <http://www.xmlrpc.org/>

Simple Object Access Protocol

It is a "lightweight protocol for exchange of information in a decentralized, distributed environment" [1]

SOAP provides

- o XML based messaging
- o Support for message routing
- o Encoding rules for expressing data types (XML Schema)
- o Conventions for using SOAP for RPC
- o Transport Independence

Related specs and standards

WSDL, UDDI, MIME and DIME attachments, SOAP Authentication, ebXML, XML Security, etc.

[1] from the SOAP specs, <http://www.w3.org/TR/SOAP/>

Web Services Description Language

Used for describing interfaces for Web Services. It is a general specification independent of SOAP (i.e. WSDL for REST services) and defines data types, interface definitions, transport bindings, and service descriptions.

WSDL makes creating SOAP proxies simpler (either dynamic or static), it can also be used for interface documentation, and can be integrated into development systems (code editors, class generators, etc.).

Universal Description Discovery and Integration

Defines a system for automated publishing and discovery of Web Services, allowing registries to be maintained for different purposes and contexts.

UDDI defines an API specification for publish and inquire for businesses and web services, a data structure, and several XML Schemas for custody, subscription policy, replication, etc.

A (fuzzy) definition

Bioinformatics studies the information content and data flow in a biological system or process. It provides the link between the observations (data), the understanding of how a system works (information), and the subsequent application of these principles (knowledge).

Research areas (by "dimensionality")

1D,2D: Gene identification in Genome, sequence analysis (similarity, evolutionary distances, etc.), expression dynamics, etc.

3D: Structure similarities, dynamic changes, protein folding, structure-function relationships, etc.

nD: Protein ensembles in a cell, interaction networks, expression pathways, etc.

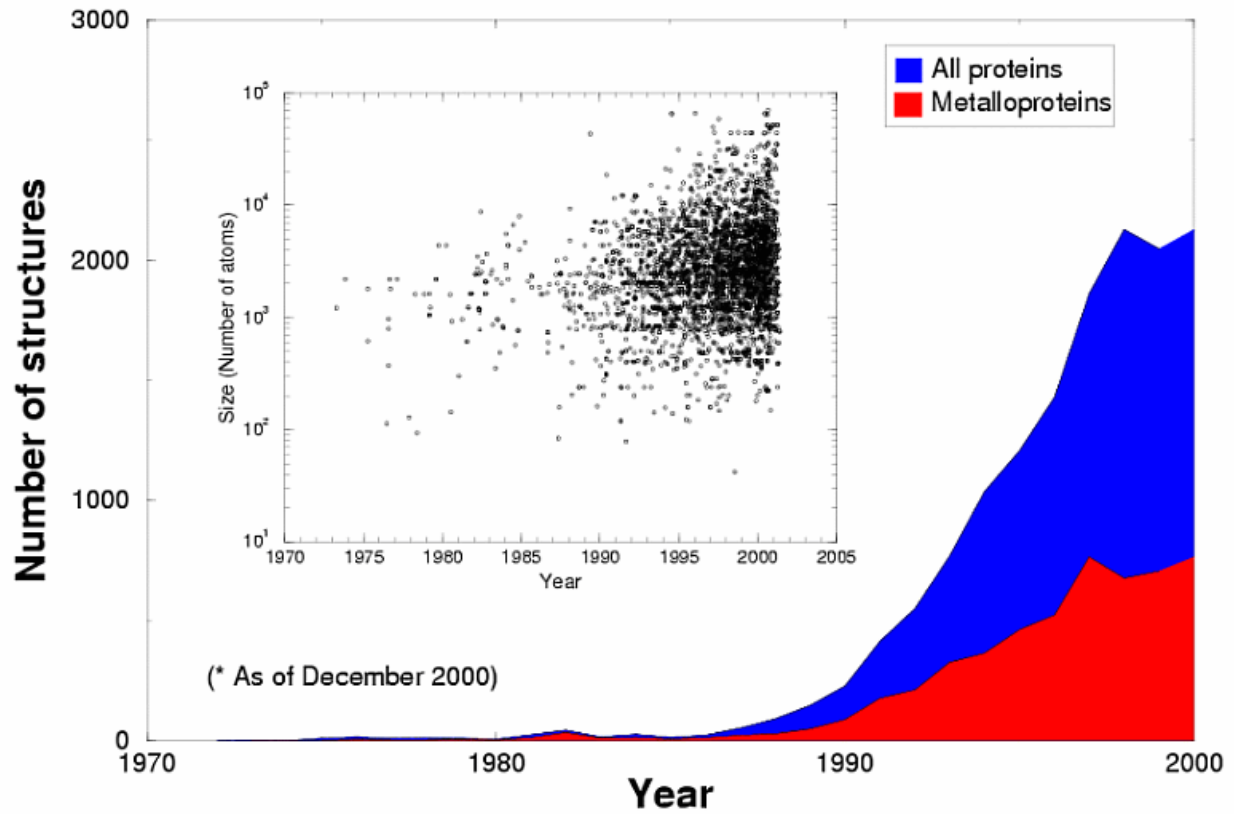
What is happening in bioinformatics

- o All that research is producing incredible amounts of data.
- o No single human being can hope to grok what all the data means without integration of the analysis from many sources.
- o A sizeable amount of that data is being made accessible via an interactive web interface, e.g.:
Protein Data Bank: <http://www.rcsb.org/>
GenBank : <http://www.ncbi.nlm.nih.gov/Genbank>

What web services bring to game

- o They make integration, collaboration, data sharing, and cross-domain data analysis possible and simpler.
- o No need to re-architecture existing resources, e.g. use a SOAP server to access data in a non-web aware system.

Increase in the number and size of new structures



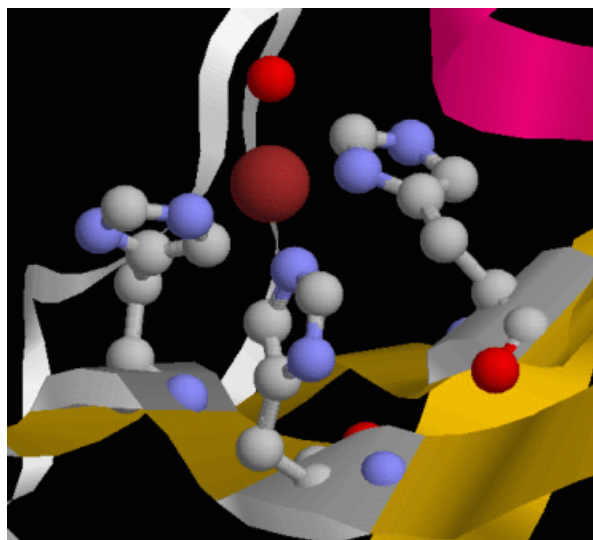
A quick definition

A protein that contains one or more metal ions which play a functional or structural role (sometimes is both).

Human Carbonic Anhydrase II (1hca), ribbon representation



A detail of the Zinc binding site



Our research program

The MDB is part of a bigger research program, which focuses on metalloprotein research and design:
The Metalloprotein Bioinformatics, Structure, and Design Program

The Objectives

- o Figure out what makes a metalloprotein tick
- o Rational design and construction of new metalloproteins

What do we need to do

1. Understand the geometrical requirements to bind a metal
2. Figure out the effect of the environmental constrains
3. Devise methods to design the sites
4. Analyze the possible candidates
5. Make the sites
6. Find out where we succeeded and where we failed
7. Go back to step 1

These are the reasons why we built the Metalloprotein-site Database and Browser ...

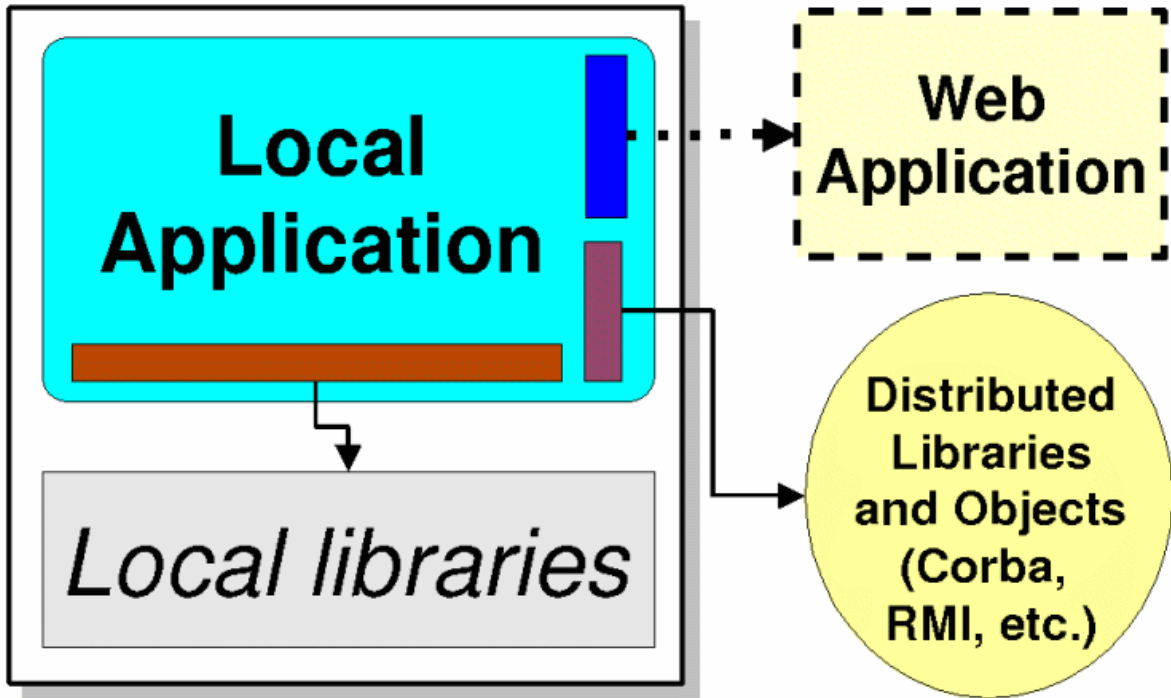
What is the MDB?

A database of quantitative data about metal-binding sites in proteins, created using automated tools and algorithms for metal site recognition and extraction.

And a Web Application that allows searching and analysis of this quantitative data.

... and what is becoming

All of the above, and a web based library of methods for other applications and web sites to use.



List of metal-containing proteins (circa 1999)

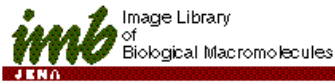
```
<?php
(...)
require("sql.php3.inc");
Header("Content-type: text/plain");

function mkCond($s) {
    $s = " metal=\"".$s."\"";
}

$metal = array("ag","au","ba","ca","cd","co",
              "cr","cu","fe","gd","hg","k","mg",
              "mn","mo","na","ni","pb","pt","v",
              "w","zn","sr");
array_walk($metal,"mkCond");

$condition = implode($metal," OR ");
if ($showmetal=="y") {
    $fields = "source_id,metal";
} else {
    $fields = "source_id";
}
$string = "select distinct $fields from site where ";
$string .= $condition." order by source_id";
$link = SQL_pconnect();
(...)
?>
```

<http://www.imb-jena.de/IMAGE.html>



Access to Metal Containing Biopolymer Structures by Molecule Type

- [Proteins](#)
- [Protein–Nucleic Acid Complexes](#)
- [Nucleic Acids](#)
- [Carbohydrates](#)

via the Periodic Table of Elements

The layout of the Periodic Table of Elements is from [WebElements \[http://www.shef.ac.uk/chemistry/web-elements/\]](http://www.shef.ac.uk/chemistry/web-elements/)

Jump start: select a metal from the Periodic Table and you will get a compilation of [PDB](#) structure entries, processed by the [MDB](#), with this particular metal listed by molecule type. Only metals given in bold do occur in biopolymer 3D structures currently included in the [PDB /MDB](#). For a more complete compilation of elements occurring in hetero components of either the [PDB](#) or the [NDB](#) use the [Periodic Table Hetero Components Database Element Browser](#).

Group	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Period																		
1	1 H																	2 He
2	3 Li	4 Be											5 B	6 C	7 N	8 O	9 F	10 Ne
3	11 Na	12 Mg											13 Al	14 Si	15 P	16 S	17 Cl	18 Ar
4	19 K	20 Ca	21 Sc	22 Ti	23 V	24 Cr	25 Mn	26 Fe	27 Co	28 Ni	29 Cu	30 Zn	31 Ga	32 Ge	33 As	34 Se	35 Br	36 Kr
5	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54

Listing Cu containing proteins

Copper (Cu) Containing Entries

This list was generated from the Metalloprotein Database at TSRI (MDB):
http://metallo.scripps.edu/
Contact: Jesus M. Castagnetto, metallodb@scripps.edu

Nucleic Acid

3 entries found

Code	Resolution	Description
<u>1d39</u>	1.20	/DNA\$ (Z, 5'-D\$(*CP*GP*CP*GP*CP*G)-3') COPPER(II) CHLORIDE SOAKED
<u>1d40</u>	1.30	/DNA\$ (Z, 5'-D(\$M==5==*CP*GP*UP*AP\$M==5==*CP*G)-3') COPPER(II) CHLORIDE
<u>231d</u>	2.40	DNA (5'-D(*CP*GP*AP*TP*CP*G)-3') DNA

Protein

239 entries found

Code	Resolution	Description
<u>1a2v</u>	2.40	COPPER AMINE OXIDASE FROM HANSENULA POLYMORPHA METHYLAMINE OXIDASE
<u>1a3z</u>	1.90	REDUCED RUSTICYANIN AT 1.9 Å RUSTICYANIN ELECTRON TRANSPORT
<u>1a4a</u>	1.89	AZURIN MUTANT WITH MET 121 REPLACED BY HIS, PH 6.5 CRYSTAL FORM, DATA (
<u>1a4b</u>	1.91	AZURIN MUTANT WITH MET 121 REPLACED BY HIS, PH 6.5 CRYSTAL FORM, DATA (
<u>1a4c</u>	2.45	AZURIN MUTANT WITH MET 121 REPLACED BY HIS, PH 3.5 CRYSTAL FORM, DATA (

We also have a lightweight molecular viewer (written in Java, approx. 70Kb in size) that is embeddable by using simple HTML code in a page, e.g.:

```
<!-- Display sites in lhca and lnpc -->
<div align='center'>
<form method="POST"
action="http://metallos.scripps.edu/services/remote/viewer.php"
name="Remote MDB Viewer" target="_blank">
<input type="hidden" name="source_id[]" value="lhca">
<input type="hidden" name="source_id[]" value="lnpc">
<input type="hidden" name="caption" value="lhca and lnpc - All sites">
<input type="submit" name="submit" value="Display sites in the MDB Viewer">
</form>
</div>
```

Output:

Using the remote viewer

MDB (Remote viewer) - Metalloprotein Site Database and Browser - Konqueror

Macromolecular Model Manager

Warning: Applet Window

Recent Query Cumulative Query

Show Hide Delete

Shown	Description	Metal
	pdb-1hca-_hg-1 Header: [lyase(oxo-acid) 02-apr-92 1hca] Title: [] ...	hg
	pdb-1hca-_zn-1 Header: [lyase(oxo-acid) 02-apr-92 1hca] Title: [] ...	zn
	pdb-1npc-_ca-1 Header: [hydrolase(metalloproteinase) 08-jan-92 ...	ca
	pdb-1npc-_ca-2 Header: [hydrolase(metalloproteinase) 08-jan-92 ...	ca
	pdb-1npc-_ca-3 Header: [hydrolase(metalloproteinase) 08-jan-92 ...	ca
	pdb-1npc-_ca-4 Header: [hydrolase(metalloproteinase) 08-jan-92 ...	ca
	pdb-1npc-_zn-1 Header: [hydrolase(metalloproteinase) 08-jan-92 ...	zn

wer and the

1hca and 1npc - All sites

View Picking Labels

Transform Box

Stereo display

Previous View

Window on model

Window on all

Magnify Region

Reinitialize view

Float viewer

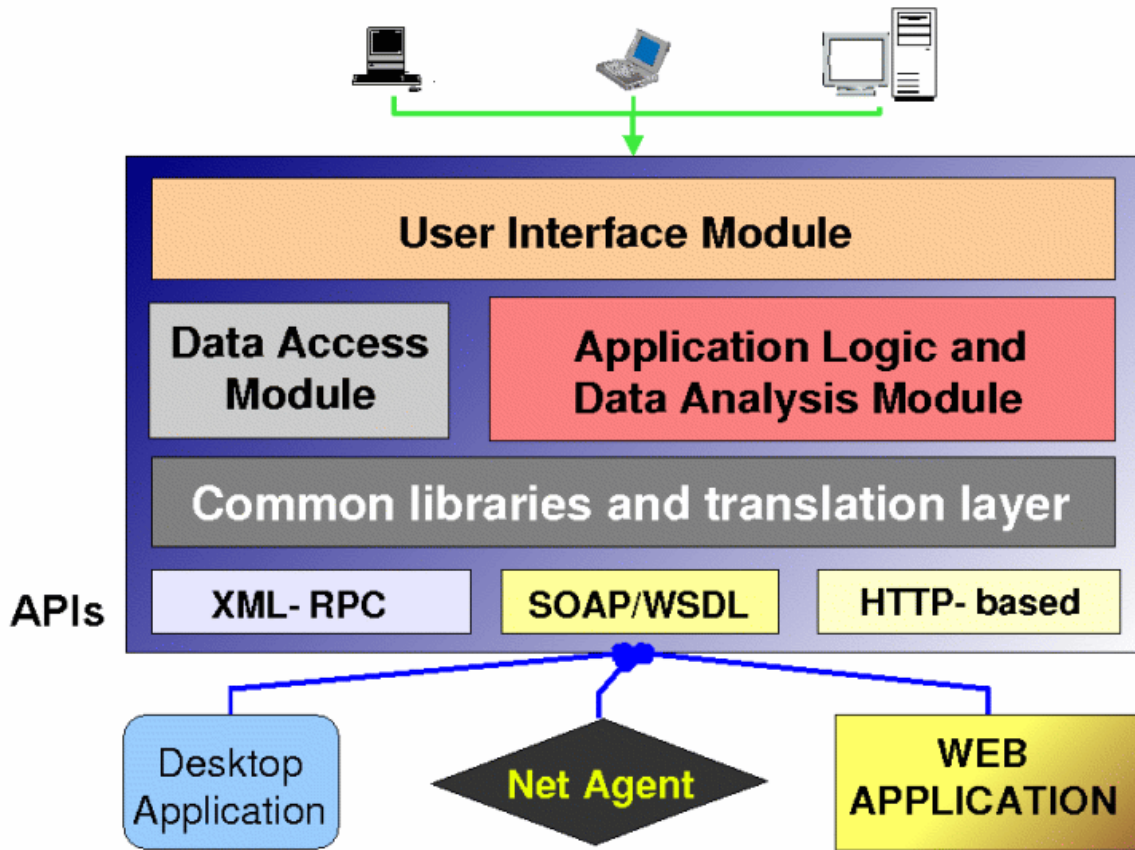
>>Dist<< >>Ang<< >>Dihed<<

The remote viewer is being used by several web sites, among them:

CaBP database: http://structbio.vanderbilt.edu/cabp_database/

IMB at Jena: <http://www.imb-jena.de/>

PROMISE: <http://bioinf.leeds.ac.uk/promise/>



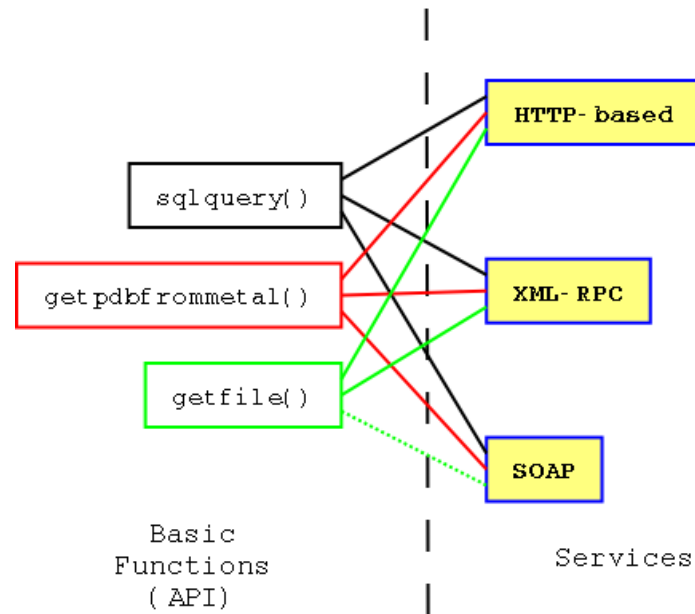
The initial plan

- o Refactor basic functions, so they can be used in the UI as well as the Web Services.
- o Implement wrappers for HTTP-based (REST) and XML-RPC interfaces.
- o Add a SOAP server and associated WSDL document.

... and the expanded plan

- o Add asynchronous modes of access the services.
- o Implement a tuple-space based access to query or analysis results and other generated documents.

A set of simple modular functions that can be wrapped in each of the services. Better than my original design of kludging everything in parallel code sets.



```
<?php
require_once "DB.php";

/**
 * Function to perform SQL queries
 *
 * @param   string   $q
 * @return  array    of results
 */

function &sqlquery($dsn, $query, $return_DB_Result=false) {
    $c = @DB::connect($dsn);
    if (is_object($c) && !DB::isError($c)) {
        $c->setFetchMode(DB_FETCHMODE_ASSOC);
        if ($return_DB_Result == true) {
            $result = $c->query($query);
        } else {
            $result = $c->getAll($query);
            if (!is_array($result))
                $result = array();
        }
        $c->disconnect();
        return $result;
    } else {
        return new DB_Error("Error while accessing database -- ".$c->getMessage());
    }
}
?>
```

```

<?php
require_once "globals.inc";
require_once "DB.php";

/**
 * Function to retrieve a list of PDB ids
 *
 * @param string $metal metal symbol
 * @param string $mode one of: first, last, random, new
 * @param int $count number of entries to retrieve
 * @return mixed array of results on success, false or a DB_Error otherwise
 */

function &getPDBFromMetal($metal, $mode, $count) {
    $fields = "distinct site.metal, protein.source_id, ";
    $fields .= "DATE_FORMAT(protein.rev_date, 'Y/m/%d') as revision_date,";
    $fields .= "DATE_FORMAT(protein.dep_date, 'Y/m/%d') as deposition_date,";
    $fields .= "protein.expdata, protein.r_value, protein.resolution,";
    $fields .= "protein.authors, protein.description";
    $from = "from site,protein";
    $where = "where site.metal = '$metal' and site.source_id = protein.source_id";
    switch ($mode) {
        case "last" :
            $ordlim = " order by protein.source_id DESC limit $count";
            break;
        case "new" :
            $ordlim = " order by protein.rev_date DESC limit $count";
            break;
        case "random" :
            $ordlim = "";
            break;
        case "first" :
            $ordlim = " order by protein.source_id ASC limit $count";
            break;
    }
    $query = "select $fields $from $where $ordlim";
    $c = @DB::connect(MDB_DATA_DSN);
    if (is_object($c) && !DB::isError($c)) {
        $c->setFetchMode(DB_FETCHMODE_ASSOC);
        $result = $c->getAll($query);
        $c->disconnect();
        if (!is_array($result))
            return false;
        if ($mode == "random") {
            $n = count($result);
            list($usec, $sec) = explode(' ', microtime());
            srand((float) $sec + ((float) $usec * 100000));
            $indexlist = array_rand(range(0, $n - 1), $count);
            $randomresult = array();
            foreach ($indexlist as $index)
                $randomresult[] = $result[$index];
            return $randomresult;
        } else {
            return $result;
        }
    } else {
        return new DB_Error("Error while accessing database -- ".$c->getMessage());
    }
}

?>

```

A simple HTTP-based API

```

<?php
require_once 'globals.inc';
require_once 'api/api_functions.php';

if (isset($func) && !empty($func)) {

    if (!in_array($func, array_keys($api))) {
        echo "Unknown function";
        exit;
    }

    // include the appropriate function
    require_once "{$api[$func]}";
    require_once "api/util_convert.php";

    // SQL query wrapper function
    function sql() {{{{/
        if (!$_REQUEST['query']) {
            echo "<b>ERROR: Missing parameters.</b>\n";
            return false;
        }
        // allow only SELECT, SHOW or DESCRIBE queries
        $allowed = "^(select|show|describe)";
        if (!eregi($allowed, trim($_REQUEST['query']))) {
            echo "<b>ERROR: Only 'SELECT', 'SHOW', or 'DESCRIBE' queries are
allowed.</b>\n";
            return false;
        }
        $result = sqlquery(MDB_DATA_DSN, $_REQUEST['query']);
        if (is_object($result) && DB::isError($result)) {
            echo "<b>ERROR ".$result->getCode().": ".$result->getMessage()."</b>\n";
            return false;
        }
        $format = isset($_REQUEST['format']) ? strtolower($_REQUEST['format']) : "csv";
        switch ($format) {
            case "wddx" :
                $out = wddx_serialize_value($result);
                break;
            case "serialize" :
                $out = serialize($result);
                break;
            case "table" :
                $out = toTable($result);
                break;
            case "csv" :
            default :
                $fields = array_keys($result[0]);
                $out = toCSV($fields);
                for ($i=0; $i < count($result); $i++) {
                    $out .= toCSV(array_values($result[$i]));
                }
                break;
        }
        echo $out;
        return true;
    }}}}/

// wrapper function to get PDB ids for a given metal
function metallopdb() {{{{/
    if (!$_REQUEST['metal']) {
        echo "<b>ERROR: Missing parameters.</b>\n";
        return false;
    }
}

```

```

$metal = strtolower($_REQUEST['metal']);
$mode = isset($_REQUEST['mode']) ? strtolower($_REQUEST['mode']) : 'first';
$count = isset($_REQUEST['count']) ? intval($_REQUEST['count']) : 5;
$format = isset($_REQUEST['format']) ? strtolower($_REQUEST['format']) : 'csv';

$result = getPDBFromMetal($metal, $mode, $count);
if (is_object($result) && DB::isError($result)) {
    echo "<b>ERROR " . $result->getCode() . ": " . $result->getMessage() . "</b>\n";
    return false;
}
if ($result == false) {
    echo "";
    return false;
}
switch ($format) {
    case "wddx" :
        header("Content-type: text/xml");
        echo wddx_serialize_value($result);
        break;
    case "serialize" :
        header("Content-type: text/plain");
        echo serialize($result);
        break;
    case "rss" :
        header("Content-type: text/xml");
        echo toRSS($result, $GLOBALS['server_remote']);
        break;
    case "csv" :
    default :
        header("Content-type: text/plain");
        $fields = array_keys($result[0]);
        $out = toCSV($fields);
        for ($i=0; $i < count($result); $i++) {
            $out .= toCSV(array_values($result[$i]));
        }
        echo $out;
        break;
}
return true;

}}}}/

// execute the api wrapper function
$output = $func();
} else {
    // show introspection

    require_once "general.lib";
    function get_content_ts() {/{}{/
        return -1;
    }/}}}/

    function get_content() {/{}{/
        $content = lib_content("__default");
        $content['name'] = $_SERVER['PHP_SELF'];
        $content['title'] = 'MDB - Web API (introspection)';
        $content['page_title'] = 'MDB - Web API (introspection)';
        $apidesc = "<div align='center'>\n";
        $apidesc .= "<table width='90%' border='0' cellpadding='5'>\n";
        foreach ($GLOBALS['apidoc'] as $func=>$doc) {
            if ($func == "get")
                continue;
            $apidesc .= "<tr><td>\n";
            $apidesc .= "<table border='0' width='100%' cellpadding='0' cellspacing='0'>\n";
            $apidesc .= "<tr><td bgcolor='#000000'>\n";
            $apidesc .= "<table border='0' width='100%' cellspacing='1' cellpadding='5'>\n";
            $apidesc .= "<tr bgcolor='#eeeeee'>\n";
            $apidesc .= "<th align='center'><i>Function</i>:<br /> $func</th>\n";

```

```

    $apidesc .= "<th align='left'>\n<i>Parameters</i>:<ul>\n";
    foreach ($GLOBALS['apiparams'][$func] as $param=>$desc)
        $apidesc .= "<li>$param: $desc</li>\n";
    $apidesc .= "</ul>\n</th>\n";
    $apidesc .= "</tr>\n<tr bgcolor='#ddffff'>\n<td colspan='2'>";
    $apidesc .= "<p><b>Description:</b><br />";
    $apidesc .= htmlspecialchars($doc)."</p>\n</td>\n</tr>\n";
    $apidesc .= "</table>\n</td></tr></table>\n</td></tr>\n";
}
$apidesc .= "</table></div>\n";
$content['body'] = $apidesc;
$content['timestamp'] = -1;
$content['page_img'] = 'MDB Web API:/images/api.png:/services/api/index.php';
return $content;
}}}}/

require_once "dohandler.lib";

}
?>

```

```

<?php
include 'slides/mdb/scripts/mdb_tests.cfg';
$service = MDB_SERVER.API_URI;
$fp = fopen($service,'r');
$out = '';
while($r = fread($fp,4096))
    $out .= $r;
fclose($fp);
echo $out;
?>

```

Output of the "Introspection" script

```

<table>
<tr><td>
<table border='0' width='50%' cellpadding='0' cellspacing='0'>
<tr><td bgcolor='#000000'>
<table border='0' width='100%' cellspacing='1' cellpadding='5'>
<tr bgcolor='#eeeeee'>
<th align='center'><i style='font-size: 1em;'>Function</i>:<br /> sql</th>
<th align='left'>
<i style='font-size: 1em;'>Parameters</i>:<ul>
<li style='font-size: 1em;'>query: urlencoded SQL query</li>
<li style='font-size: 1em;'>[format]: csv (default), wddx, serialized, table</li>
</ul>
</th>
</tr>
<tr bgcolor='#ddffff'>
<td colspan='2'><p style='font-size: 1em;'><b>Description:</b><br />General
function to query a database server.
Accepts a valid SQL query (&quot;SELECT&quot;;, &quot;SHOW&quot;;, or
&quot;DESCRIBE&quot;; statements only),
and an optional result format. Valid formats are &quot;csv&quot;; (default),
&quot;wddx&quot;;, &quot;serialized&quot;; (PHP serialization), &quot;table&quot;;
(HTML table).
Example of use:
/services/api.php?function=sql&amp;query=select+source_id+from+protein+limit+10
&amp;format=wddx</p>
</td>
</tr>
</table>
</td></tr></table>
<tr><td>
<table border='0' width='100%' cellpadding='0' cellspacing='0'>
<tr><td bgcolor='#000000'>
<table border='0' width='100%' cellspacing='1' cellpadding='5'>
<tr bgcolor='#eeeeee'>
<th align='center'><i>Function</i>:<br /> metallopdb</th>
<th align='left'>
<i style='font-size: 1em;'>Parameters</i>:<ul>
<li style='font-size: 1em;'>metal: atomic symbol for the metal</li>
<li style='font-size: 1em;'>[mode]: one of: first (default), last, random, new</li>
<li style='font-size: 1em;'>[count]: how many PDB entries to retrieve (default =
5)</li>
<li style='font-size: 1em;'>[format]: csv (default), wddx, rss, serialized</li>
</ul>
</th>
</tr>
<tr bgcolor='#ddffff'>
<td colspan='2'>
<p style='font-size: 1em;'><b>Description:</b><br />Specialized function to
retrieve a list of PDB structures
containing a particular metal ion, as indexed in the MDB. The retrieval
mode can be one of &quot;first&quot;; (default) to get the first &quot;n&quot;; PDB
ids (sorted

```

alphabetically), "last" to get the last "n" ids, "random" to get "n" random entries, and "new" to get the newest "n" entries released/indexed.

The number of entries to retrieve is set using "count".

The list can be formatted as "csv" (default), "wddx", "rss", or "serialized" (PHP serialization).

Example of use:

```
/services/api.php?function=metalopdb&metal=zn&mode=random&count=5&format=rss
```

```
</td>
```

```
</tr>
```

```
</table>
```

```
</td></tr></table>
```

```
</td></tr></table>
```

Using the sql() function

```
<?php
include 'slides/mdb/scripts/mdb_tests.cfg';
$query = "select source_id,expdata,description from protein ";
$query .= "where expdata not like 'nmr' limit 4";
$format = "table";
$q = urlencode($query);
$func = "sql";
$getstr = MDB_SERVER.API_URI;
$getstr .= "?func=$func&query=$q&format=$format";
$result = implode('',file($getstr));

echo "<h1 align='center'>Testing the HTTP based API</h1>
<b>Server used:</b> [ ".MDB_SERVER.
"]<br><b>Service:</b>[ ".API_URI." ]
<hr>
<h2>SQL query function</h2>
<b>Function vars:</b><br>
[query: $query]<br>
[format: $format]
<br><b>Using the URL: </b>".
wordwrap($getstr,45,"<br>\n",true).
"<br><b>RESULT</b><br>
$result\n";
?>
```

Output:

Using the metallopdb() function

```

<?php
include 'slides/mdb/scripts/mdb_tests.cfg';
$metal = 'zn';
$format = 'rss';
$mode = 'random';
$func = 'metallopdb';
$count = 5;
$getstr = MDB_SERVER.API_URI;
$getstr .= "?func=$func&metal=$metal";
$getstr .= "&mode=$mode&count=$count&format=$format";
$result = implode("",file($getstr));

echo "<h1 align='center'>Testing the HTTP based API</h1>
<b>Server used:</b> [ ".MDB_SERVER.
"]<br><b>Service:</b>[ ".API_URI."
<hr>
<h2>Retrieving $count random zinc containing proteins</h2>
<b>Function vars:</b><br>
[metal: $metal]
[count: $count]
[format: $format]
[mode: $mode]
<br><b>Using the URL: </b>".
wordwrap($getstr,45,"<br>\n",true).
"<br><b>RESULT</b><br>
<pre>\n";
echo htmlspecialchars($result)."\n</pre>\n";
?>

```

Output:

```

<?php
require_once "XML/RPC.php";
require_once "slides/mdb/scripts/xmlrpc_util.php";
require_once "slides/mdb/scripts/mdb_tests.cfg";

$c=new XML_RPC_Client(XMLRPC_URI, MDB_SERVER_NAME, MDB_SERVER_PORT);
$f=new XML_RPC_Message('system.listMethods');

$v=rpc_call($c, $f);
print "<h2>Methods available at http://" . $c->server . ":" .
$c->port . $c->path . "</h2>\n";
if ($v) {

    for($i=0; $i<$v->arraysize(); $i++) {
        $mname=$v->arraymem($i);
        print "<hr noshade size=1><H3>" . $mname->scalarval() . "</H3>\n";
        $f=new XML_RPC_Message('system.methodHelp');
        $f->addParam(new XML_RPC_Value($mname->scalarval(), "string"));
        $w=rpc_call($c, $f);
        if ($w) {
            $txt=$w->scalarval();
            if ($txt!="") {
                print "<H4>Documentation</H4><P style='font-size: 1.0em;'>${txt}</P>\n";
            } else {
                print "<P>No documentation available.</P>\n";
            }
        }
        $f=new XML_RPC_Message('system.methodSignature');
        $f->addParam(new XML_RPC_Value($mname->scalarval(), "string"));
        $w=rpc_call($c, $f);
        if ($w) {
            print "<H4>Signature</H4><P style='font-size: 1.0em;'>\n";
            if ($w->kindOf()=="array") {
                for($j=0; $j<$w->arraysize(); $j++) {
                    $x=$w->arraymem($j);
                    $ret=$x->arraymem(0);
                    print "<CODE style='font-size: 1.0em;'>" . $ret->scalarval() . " " .
                    $mname->scalarval() . "(";
                    if ($x->arraysize()>1) {
                        for($k=1; $k<$x->arraysize(); $k++) {
                            $y=$x->arraymem($k);
                            print $y->scalarval();
                            if ($k<$x->arraysize()-1) {
                                print ", ";
                            }
                        }
                    }
                    print ")</CODE><BR>\n";
                }
            } else {
                print "Signature unknown\n";
            }
            print "</P>\n";
        }
    }
}
?>

```

Output:

Methods available at <http://metallo.scripps.edu:80/services/xmlrpc.php>

method.sql

DocumentationGeneral function to query the MDB database server. Accepts a valid SQL query ("SELECT", "SHOW", or "DESCRIBE" statements only), and returns the resulting rows as a structure.

Signature

```
struct method.sql(string)
```

method.metallopdb

DocumentationSpecialized function to retrieve a list of PDB structures containing a particular metal ion, as indexed in the MDB. The retrieval mode can be one of "first" (default) to get the first "n" PDB ids (sorted alphabetically), "last" to get the last "n" ids, "random" to get "n" random entries, and "new" to get the newest "n" entries released/indexed. The number of entries to retrieve is set using "count".

The result can be returned as XML-RPC structure, WDDX packet or RSS document

Signature

```
struct method.metallopdb(string, string, int, string)
```

method.get

DocumentationReturns the requested file. It accepts 2 parameters: the name of the file, and (optionally) the whether the base64 encoding will be done on the "raw" (default) file, or the file after being compressed with gzip ("gzip").

Signature

```
base64 method.get(string, string)
```

system.listMethods

DocumentationThis method lists all the methods that the XML-RPC server knows how to dispatch

Signature

```
array system.listMethods(string)
```

```
array system.listMethods()
```

system.methodHelp

DocumentationReturns help text if defined for the method passed, otherwise returns an empty string

Signature

```
string system.methodHelp(string)
```

system.methodSignature

DocumentationReturns an array of known signatures (an array of arrays) for the method name passed. If no signatures are known, returns a none-array (test for type != array to detect missing signature)

Signature

```
array system.methodSignature(string)
```

Using method.sql from the XML-RPC service

```

<?php
require_once "XML/RPC.php";
require_once "slides/mdb/scripts/xmlrpc_util.php";
require_once "slides/mdb/scripts/mdb_tests.cfg";
?>
<b>method.sql:</b>
<small>"select source_id,expdata,rev_date,resolution,description
from protein limit 10"</small><br>
<?php
$method = "method.sql";
$query = "select source_id,expdata,resolution,description ";
$query .= "from protein where expdata like 'x-ray' and ";
$query .= "resolution >= 0.7 order by resolution limit 5";
$q = new XML_RPC_Value($query, $GLOBALS['XML_RPC_String']);
$msg = new XML_RPC_Message($method,array($q));

$c=new XML_RPC_Client(XMLRPC_URI, MDB_SERVER_NAME, MDB_SERVER_PORT);
$r = $c->send($msg);
if (!$r->faultCode()) {
    echo toTable(XML_RPC_decode($r->value()));
} else {
    echo "<pre>\n";
    echo "ERROR: ".$r->faultCode()." -- ".$r->faultString();
    echo "</pre>\n";
}
?>

```

Output:

```

method.sql:
"select source_id,expdata,rev_date,resolution,description
from protein limit 10"

```

```

source_id
expdata
resolution
description

```

```

1d8g
x-ray diffraction
0.74
Header: [deoxyribonucleic acid 23-oct-99 1d8g]
Title: [ultrahigh resolution crystal structure of b-dna decamer d(ccagtactgg)]
Compound: [mol_id: 1; molecule: 5'-d(cpccpapgtpapcptpgppg*)-3'; chain: a;
engineered: yes]
Source: [mol_id: 1; synthetic: yes]
Keywords: [calcium, alternate conformation, ultrahigh resolution, trp repressor,
polyamide]

```

```

1gci
x-ray diffraction
0.78
Header: [serine protease 02-sep-98 1gci]
Title: [the 0.78 angstroms structure of a serine protease - bacillus lentus
subtilisin]
Compound: [mol_id: 1; molecule: subtilisin; chain: null; ec: 3.4.21.62; engineered:
yes; biological_unit: monomer]

```

Source: [mol_id: 1; organism_scientific: bacillus lentus; expression_system: bacillus subtilis]
Keywords: [bacillus lentus, hydrolase, serine protease, ultra-high resolution]

liua
x-ray diffraction
0.80
Header: [electron transport 01-mar-02 liua]
Title: [ultra-high resolution structure of hipip from thermochromatium tepidum]
Compound: [mol_id: 1; molecule: high-potential iron-sulfur protein; chain: a]
Source: [mol_id: 1; organism_scientific: thermochromatium tepidum; organism_common: bacteria]
Keywords: [ultra-high resolution, crystal structure]

ldpl
x-ray diffraction
0.83
Header: [deoxyribonucleic acid 27-dec-99 ldpl]
Title: [a-dna decamer gcgta(t23)tacgc with incorporated 2'-methoxy-3'-methylenephosphate-thymidine]
Compound: [mol_id: 1; molecule: 5'-d(gpccpgtppap(t23)papccpgpc)-3'; chain: a, b; engineered: yes]
Source: [mol_id: 1; synthetic: yes]
Keywords: [deoxyribonucleic acid]

lm40
x-ray diffraction
0.85
Header: [hydrolase 01-jul-02 lm40]
Title: [ultra high resolution crystal structure of tem-1]
Compound: [mol_id: 1; molecule: beta-lactamase tem; chain: a; synonym: tem-1; ec: 3.5.2.6; engineered: yes; mutation: yes]
Source: [mol_id: 1; organism_scientific: escherichia coli; organism_common: bacteria; gene: bla; expression_system: escherichia coli; expression_system_common: bacteria; expression_system_strain: sfl20; expression_system_vector_type: plasmid; expression_system_plasmid: palter ex ii - tem-1]
Keywords: [acylation mechanism, x-ray structure, ultra- high resolution]

Using method.metallopdb from the XML-RPC service

```

<?php
error_reporting(1);
require_once "XML/RPC.php";
require_once "slides/mdb/scripts/xmlrpc_util.php";
require_once "slides/mdb/scripts/mdb_tests.cfg";

$c=new XML_RPC_Client(XMLRPC_URI, MDB_SERVER_NAME, MDB_SERVER_PORT);

$metal = new XML_RPC_Value('zn');
$mode = new XML_RPC_Value('random');
$count = new XML_RPC_Value(5, $GLOBALS['XML_RPC_Int']);
$format = new XML_RPC_Value('array');
$method = 'method.metallopdb';
$fmt = "<h3>Parameters: metal='s', mode='s', count=d, format='s'</h3>";
printf($fmt, $metal->scalarval(), $mode->scalarval(),
        $count->scalarval(), $format->scalarval());

$msg = new XML_RPC_Message($method, array($metal, $mode, $count, $format));
$r = $c->send($msg);
$v = $r->value();
if (!$r->faultCode()) {
    echo toTable2(XML_RPC_decode($v));
} else {
    echo "<pre>\n";
    echo "ERROR: ".$r->faultCode()." -- ".$r->faultString();
    echo "</pre>\n";
}
?>

```

Output:

```
Parameters: metal='zn', mode='random', count=5, format='array'
```

```
ROW: 1
```

```
metal
zn
```

```
source_id
1b2z
```

```
revision_date
1999/12/22
```

```
deposition_date
1998/12/03
```

```
expdata
x-ray diffraction
```

```
r_value
0.17
```

```
resolution
```

2.03

authors

c.k.vaughan,p.harryson,a.m.buckle,m.oliveberg,a.r.fersht

description

Header: [hydrolase 03-dec-98 1b2z]

Title: [deletion of a buried salt bridge in barnase]

Compound: [mol_id: 1; molecule: barnase; chain: a, b, c; ec: 3.1.27.3; engineered: yes; mutation: yes]

Source: [mol_id: 1; organism_scientific: bacillus amyloliquefaciens; cellular_location: extracellular; expression_system: escherichia coli; expression_system_vector: puc19; expression_system_plasmid: pmt410]

Keywords: [alpha/beta protein, hydrolase]

ROW: 2

metal

zn

source_id

1du3

revision_date

2000/09/27

deposition_date

2000/01/14

expdata

x-ray diffraction

r_value

0.29

resolution

2.2

authors

s.-s.cha,b.-j.sung,b.-h.oh

description

Header: [apoptosis 14-jan-00 1du3]

Title: [crystal structure of trail-sdr5]

Compound: [mol_id: 1; molecule: death receptor 5; chain: a, b, c, g, h, i; fragment: extracellular domain; engineered: yes; mol_id: 2; molecule: tnfr-related apoptosis inducing ligand; chain: d, e, f, j, k, l; engineered: yes]

Source: [mol_id: 1; organism_scientific: homo sapiens; organism_common: human; expression_system: escherichia coli; expression_system_common: bacteria; mol_id: 2; organism_scientific: homo sapiens; organism_common: human; expression_system_common: bacteria]

Keywords: [dr5, complex]

ROW: 3

metal

zn

source_id
1e4u

revision_date
2002/04/25

deposition_date
2000/07/12

expdata
nmr, 30 structures

r_value
ERROR_NON_NUMERIC_FOUND

resolution
ERROR_NON_NUMERIC_FOUND

authors
h.hanzawa,m.j.de ruwe,t.k.albert,p.c.van der vliet, h.t.timmers,r.boelens

description
Header: [gene regulation 12-jul-00 1e4u]
Title: [n-terminal ring finger domain of human not-4]
Compound: [mol_id: 1; molecule: transcriptional repressor not4; chain: a; fragment:
ring finger domain; engineered: yes]
Source: [mol_id: 1; organism_scientific: homo sapiens; expression_system:
escherichia coli]
Keywords: []

ROW: 4

metal
zn

source_id
1efq

revision_date
2001/02/09

deposition_date
2000/02/09

expdata
x-ray diffraction

r_value
0.23

resolution

1.6

authors

p.r.pokkuluri,x.cai,m.gu,f.j.stevens,m.schiffer

description

Header: [immune system 09-feb-00 1efq]

Title: [q38d mutant of len]

Compound: [mol_id: 1; molecule: kappa-4 immunoglobulin (light chain); chain: a; engineered: yes; mutation: yes]

Source: [mol_id: 1; organism_scientific: homo sapiens; organism_common: human; expression_system: escherichia coli; expression_system_common: bacteria; expression_system_plasmid: pask40]

Keywords: [mutant, monomer, uranyl ion in crystal contact, aspartic acid in beta-sheet, protein stability]

ROW: 5

metal

zn

source_id

1fop

revision_date

2001/07/20

deposition_date

2000/08/28

expdata

x-ray diffraction

r_value

0.22

resolution

2.3

authors

c.s.raman,h.li,p.martasek,b.s.s.masters,t.l.poulos

description

Header: [oxidoreductase 28-aug-00 1fop]

Title: [bovine endothelial nitric oxide synthase heme domain complexed with l-arg and no(h4b-bound)]

Compound: [mol_id: 1; molecule: nitric-oxide synthase; chain: a, b; synonym: endothelial ec-nos, nos type iii, nosiii, endothelial nos, enos, constitutive nos, cnos; ec: 1.14.13.39; engineered: yes; other_details: cacodylate (residue cac) binds to sg cys 384 of both chains]

Source: [mol_id: 1; organism_scientific: bos taurus; organism_common: bovine; cell: endothelial cells; expression_system: escherichia coli; expression_system_common: bacteria]

Keywords: [nitric oxide synthase]

Server created using PEAR::SOAP

```

<?php
require_once 'globals.inc';
require_once 'SOAP/Server.php';
require_once 'api/function.sqlquery.php';
require_once 'api/function.getpdbfrommetal.php';
require_once 'api/util_convert.php';

$SOAP_OBJECT_STRUCT=false;

class MDB_SOAP_Services {

    var $method_namespace = '';
    var $dispatch_map = array();

    function MDB_SOAP_Services() {
        $this->method_namespace = 'urn:MDB_SOAP_Server';
    }

    function sql($query) {
        if (!eregi("^(\select|show|describe)", $query)) {
            trigger_error('Invalid query. Only SELECT, SHOW '.
                'and DESCRIBE allowed', E_USER_ERROR);
        }
        $db = $GLOBALS['dsn'].$GLOBALS['datadb'];
        $result = @sqlquery($db, $query, false);
        if (is_object($result) && PEAR::isError($result)) {
            trigger_error($result->getMessage(), E_USER_ERROR);
        } else {
            return new SOAP_Value('result','Struct', $result);
        }
    }

    function metallopdb($metal, $mode, $count) {
        return $this->_dometallopdb($metal, $mode, $count);
    }

    function rssmetallopdb($metal, $mode, $count) {
        return $this->_dometallopdb($metal, $mode, $count, true);
    }

    function _dometallopdb($metal, $mode, $count, $rss=false) {
        $count = intval($count);
        if ($count <= 0) {
            trigger_error("Parameter count must be an ".
                "integer > 0, value sent: $count", E_USER_ERROR);
        }
        $result = @getPDBFromMetal($metal, $mode, $count);
        if (is_object($result) && DB::isError($result)) {
            trigger_error($result->getMessage(), E_USER_ERROR);
        } elseif ($result == false) {
            trigger_error('No results were found', E_USER_WARNING);
        } else {
            if ($rss) {
                $pkt = toRSS(&$result);
                return new SOAP_Value('result','base64', base64_encode($pkt));
            } else {
                return new SOAP_Value('result','Struct', $result);
            }
        }
    }
}
}

```

```
$server = new SOAP_Server();  
$mdb_services = new MDB_SOAP_Services();  
$server->addObjectMap($mdb_services);  
$server->service($HTTP_RAW_POST_DATA);  
?>
```

Using the methods sql, metallopdb and rssmetallopdb

```

<?php
require_once 'SOAP/Client.php';
require_once "slides/mdb/scripts/mdb_tests.cfg";
$SOAP_OBJECT_STRUCT=false;
$url = MDB_SERVER.SOAP_URI;
$ns = 'urn:MDB_SOAP_Server';
$c = new SOAP_Client($url);

function niceOut($obj) {
    echo "<table border='1'>\n<tr>\n<td bgcolor='#eeeeee'>\n";
    print_rec($obj);
    echo "\n</td>\n</tr>\n</table>\n";
}

function print_rec($obj) {
    if (is_string($obj)) {
        echo "<pre>\n$obj\n</pre>\n";
    } else {
        foreach ($obj as $key=>$val) {
            if (is_array($val) || is_object($val)) {
                echo "<b>$key</b>:\n".print_rec($val)."<br>\n";
            } else {
                echo "<b>$key</b>: $val<br>\n";
            }
        }
    }
}

?>
<h2>Excercizing the sql method</h2>
[query: 'select source_id,description from protein limit 3']
<?php
$ret = $c->call('sql',array('query'=>'select source_id,description from protein
limit 3'),$ns);
niceOut($ret);
?>
<hr>
<h2>Making a deliberate error using the metallopdb method</h2>
[metal: zn][mode: first][count: -3]
<?php
$ret = $c->call('metallopdb',array('metal'=>'zn',
'mode'=>'first','count'=>-3),$ns);
if (PEAR::isError($ret))
    echo "<br><b>OOPS we got an error</b>\n";
niceOut($ret);
?>
<hr>
<h2>Using the metallopdb method correctly</h2>
[metal:zn][mode: random][count: 3]
<?php
$ret = $c->call('metallopdb', array('metal'=>'zn', 'mode'=>'random',
'count'=>3), $ns);
niceOut($ret);
?>
<hr>
<h2>Using the rssmetallopdb method</h2>
[metal: zn][mode: last][count: 3]
<?php
$ret = $c->call('rssmetallopdb', array('metal'=>'zn', 'mode'=>'last', 'count'=>3),
$ns);
niceOut(htmlspecialchars(base64_decode($ret)));

```

?>

Output:

```
Exercizing the sql method
[query: 'select source_id,description from protein limit 3']

source_id: laoo
description: Header: [metallothionein 08-jul-97 laoo]
Title: [ag-substituted metallothionein from saccharomyces cerevisiae, nmr,
minimized average structure]
Compound: [mol_id: 1; molecule: ag-metallothionein; chain: null; synonym: ag-mt;
biological_unit: monomer; other_details: ag(i) substituted]
Source: [mol_id: 1; organism_scientific: saccharomyces cerevisiae; organism_common:
baker's yeast; strain: 30ln]
Keywords: [copper detoxification, metal-thiolate cluster]
0:

source_id: laqq
description: Header: [metallothionein 31-jul-97 laqq]
Title: [ag-substituted metallothionein from saccharomyces cerevisiae, nmr, 10
structures]
Compound: [mol_id: 1; molecule: ag-metallothionein; chain: null; synonym: ag-mt;
biological_unit: monomer; other_details: ag(i) substituted]
Source: [mol_id: 1; organism_scientific: saccharomyces cerevisiae; organism_common:
baker's yeast; strain: 30ln]
Keywords: [copper detoxification, metal-thiolate cluster]
1:

source_id: 2aw0
description: Header: [copper transport 08-oct-97 2aw0]
Title: [fourth metal-binding domain of the menkes copper-transporting atpase, nmr,
20 structures]
Compound: [mol_id: 1; molecule: menkes copper-transporting atpase; chain: null;
fragment: fourth metal-binding domain; ec: 3.6.1.36; engineered: yes;
biological_unit: monomer; other_details: ag(i)-bound state]
Source: [mol_id: 1; organism_scientific: homo sapiens; organism_common: human;
expression_system: escherichia coli]
Keywords: [copper-binding domain, hydrolase, copper transport]
2:
```

```
Making a deliberate error using the metallopdb method
[metal: zn][mode: first][count: -3]
OOPS we got an error
```

```
error_message_prefix: PHP
mode: Errno: 256
Filename: /export/asd/metallo2/metallopdb/webdocs/services/soap.php
Lineno: 46
level: 1024
code: SOAP-ENV:Server
message: Parameter count must be an integer > 0, value sent: -3
userinfo: 1024
callback:
```

```
Using the metallopdb method correctly
```

[metal:zn][mode: random][count: 3]

metal: zn
source_id: let5
revision_date: 2000/08/24
deposition_date: 2000/04/12
expdata: x-ray diffraction
r_value: 0.19
resolution: 1.90
authors: m.j.boullanger,m.kukimoto,m.nishiyama,s.horinouchi, m.e.p.murphy
description: Header: [oxidoreductase 12-apr-00 let5]
Title: [crystal structure of nitrite reductase asp98asn mutant from alcaligenes faecalis s-6]
Compound: [mol_id: 1; molecule: nitrite reductase; chain: a; fragment: 40 - 376;
synonym: cu-nir; ec: 1.7.99.3; engineered: yes; mutation: yes]
Source: [mol_id: 1; organism_scientific: alcaligenes faecalis; organism_common:
bacteria; expression_system: escherichia coli; expression_system_common: bacteria;
expression_system_vector_type: plasmid; expression_system_plasmid: pet28]
Keywords: []
0:

metal: zn
source_id: libh
revision_date: 2001/05/09
deposition_date: 2001/03/28
expdata: x-ray diffraction
r_value: 0.23
resolution: 2.00
authors: m.e.stroppolo,a.pesce,m.d'orazio,p.o'neill,d.bordo,c.rosano,
m.milani,a.battistoni,m.bolognesi,a.desideri
description: Header: [oxidoreductase 28-mar-01 libh]
Title: [x-ray 3d structure of p.leiognathi cu,zn sod mutant m4li]
Compound: [mol_id: 1; molecule: cu,zn superoxide dismutase; chain: a; ec: 1.15.1.1;
engineered: yes; mutation: yes]
Source: [mol_id: 1; organism_scientific: photobacterium leiognathi;
organism_common: bacteria; expression_system: escherichia coli;
expression_system_common: bacteria]
Keywords: [subunit interaction]
1:

metal: zn
source_id: lkwg
revision_date: 2002/09/18
deposition_date: 2002/01/29
expdata: x-ray diffraction
r_value: 0.17
resolution: 1.60
authors: m.hidaka,s.fushinobu,n.ohtsu,h.motoshima,h.matsuzawa, h.shoun,t.wakagi
description: Header: [hydrolase 29-jan-02 lkwg]
Title: [crystal structure of thermus thermophilus a4 beta- galactosidase]
Compound: [mol_id: 1; molecule: beta-galactosidase; chain: a; ec: 3.2.1.23;
engineered: yes]
Source: [mol_id: 1; organism_scientific: thermus thermophilus; organism_common:
bacteria; strain: a4; expression_system: escherichia coli;
expression_system_common: bacteria; expression_system_strain: jml05;
expression_system_vector_type: plasmid; expression_system_plasmid: pexm9]
Keywords: [glycoside hydrolase family 42, trimer]
2:

Using the rssmetallopdb method
[metal:zn][mode: last][count: 3]

```

<rdf:RDF
  xmlns:rdf=&quot;http://www.w3.org/1999/02/22-rdf-syntax-ns#&quot;;
  xmlns=&quot;http://purl.org/rss/1.0/&quot;;
  xmlns:mn=&quot;http://usefulinc.com/rss/manifest/&quot;;
  xmlns:dc=&quot;http://purl.org/dc/elements/1.1/&quot;;
&gt;

  <channel
rdf:about=&quot;http://metallo.scripps.edu/services/soap.php&quot;;&gt;
  <title&gt;MDB - The Metalloprotein Database and Browser</title&gt;
  <link&gt;http://metallo.scripps.edu</link&gt;
  <description&gt;Data retrieved using the MDB API</description&gt;
  <items&gt;
    <rdf:Seq&gt;
      <rdf:li
rdf:resource=&quot;http://metallo.scripps.edu/remote/remote.php3?source_id[]=9nse&amp;meta
/&gt;
      <rdf:li
rdf:resource=&quot;http://metallo.scripps.edu/remote/remote.php3?source_id[]=9icv&amp;meta
/&gt;
      <rdf:li
rdf:resource=&quot;http://metallo.scripps.edu/remote/remote.php3?source_id[]=9ici&amp;meta
/&gt;
    </rdf:Seq&gt;
  </items&gt;
</channel&gt;

  <item
rdf:about=&quot;http://metallo.scripps.edu/remote/remote.php3?source_id[]=9nse&amp;meta
&link&gt;http://metallo.scripps.edu/remote/remote.php3?source_id[]=9nse&amp;meta
  <title&gt;(Zn) 9nse [deposited on: 1999/01/13, revised on:
2000/10/25]</title&gt;
  <description&gt;Header: [oxidoreductase 13-jan-99 9nse]
Title: [bovine endothelial nitric oxide synthase, ethyl- isoselenourea complex]
Compound: [mol_id: 1; molecule: nitric oxide synthase; chain: a, b; fragment: heme
domain; synonym: nos, enos; ec: 1.14.13.39; engineered: yes]
Source: [mol_id: 1; organism_scientific: bos taurus; organism_common: bovine; cell:
endothelial; expression_system: escherichia coli; expression_system_common:
bacteria; expression_system_strain: bl21]
Keywords: [heme protein, tetrahydrobiopterin]
Authors[h.li,c.s.raman,p.martasek,v.kral,b.s.s.masters,t.l.poulos] ExpData[x-ray
diffraction] R-value[0.20] Resolution[2.24]</description&gt;
  </item&gt;

  <item
rdf:about=&quot;http://metallo.scripps.edu/remote/remote.php3?source_id[]=9icv&amp;meta
&link&gt;http://metallo.scripps.edu/remote/remote.php3?source_id[]=9icv&amp;meta
  <title&gt;(Zn) 9icv [deposited on: 1995/12/16, revised on:
1996/11/15]</title&gt;
  <description&gt;Header: [complex (nucleotidyltransferase/dna) 16-dec-95
9icv]
Title: [dna polymerase beta (pol b) (e.c.2.7.7.7) complexed with seven base pairs
of dna; soaked in the presence of datp (0.1 millimolar) and zncl2 (0.2
millimolar)]
Compound: [mol_id: 1; molecule: dna polymerase beta; chain: a; ec: 2.7.7.7;
engineered: yes; mol_id: 2; molecule: dna (5'-d(cpaptptpapgpa)-3') (dot)
(5'-d(tpcptpapaptpg)-3'); chain: t, p; engineered: yes; other_details: soaked in
the presence of datp (0.1 millimolar) and zncl2 (0.2 millimolar)]
Source: [mol_id: 1; organism_scientific: homo sapiens; organism_common: human;
expression_system: escherichia coli; mol_id: 2; synthetic: yes]
Keywords: [dna replication, dna repair, nucleotidyltransferase, complex
(nucleotidyltransferase/dna)] Authors[h.pelletier,m.r.sawaya] ExpData[x-ray
diffraction] R-value[0.18] Resolution[2.70]</description&gt;
  </item&gt;

  <item
rdf:about=&quot;http://metallo.scripps.edu/remote/remote.php3?source_id[]=9ici&amp;meta

```

<link>http://metallo.scripps.edu/remote/remote.php3?source_id[]=9ici&metal[]</link>
<title>(Zn) 9ici [deposited on: 1995/12/15, revised on: 1996/11/15]</title>
<description>Header: [complex (nucleotidyltransferase/dna) 15-dec-95 9ici]
Title: [dna polymerase beta (pol b) (e.c.2.7.7.7) complexed with seven base pairs of dna; soaked in the presence of dttp (1 millimolar) and zncl2 (1 millimolar)]
Compound: [mol_id: 1; molecule: dna polymerase beta; chain: a; ec: 2.7.7.7; engineered: yes; mol_id: 2; molecule: dna (5'-d(cpaptptpagpa)-3') (dot) (5'-d(tpcptpapaptpg)-3'); chain: t, p; engineered: yes; other_details: soaked in the presence of dttp (1 millimolar) and zncl2 (1 millimolar)]
Source: [mol_id: 1; organism_scientific: homo sapiens; organism_common: human; expression_system: escherichia coli; mol_id: 2; synthetic: yes]
Keywords: [dna replication, dna repair, nucleotidyltransferase, complex (nucleotidyltransferase/dna)] Authors[h.pelletier,m.r.sawaya] ExpData[x-ray diffraction] R-value[0.17] Resolution[3.10]</description>
</item>

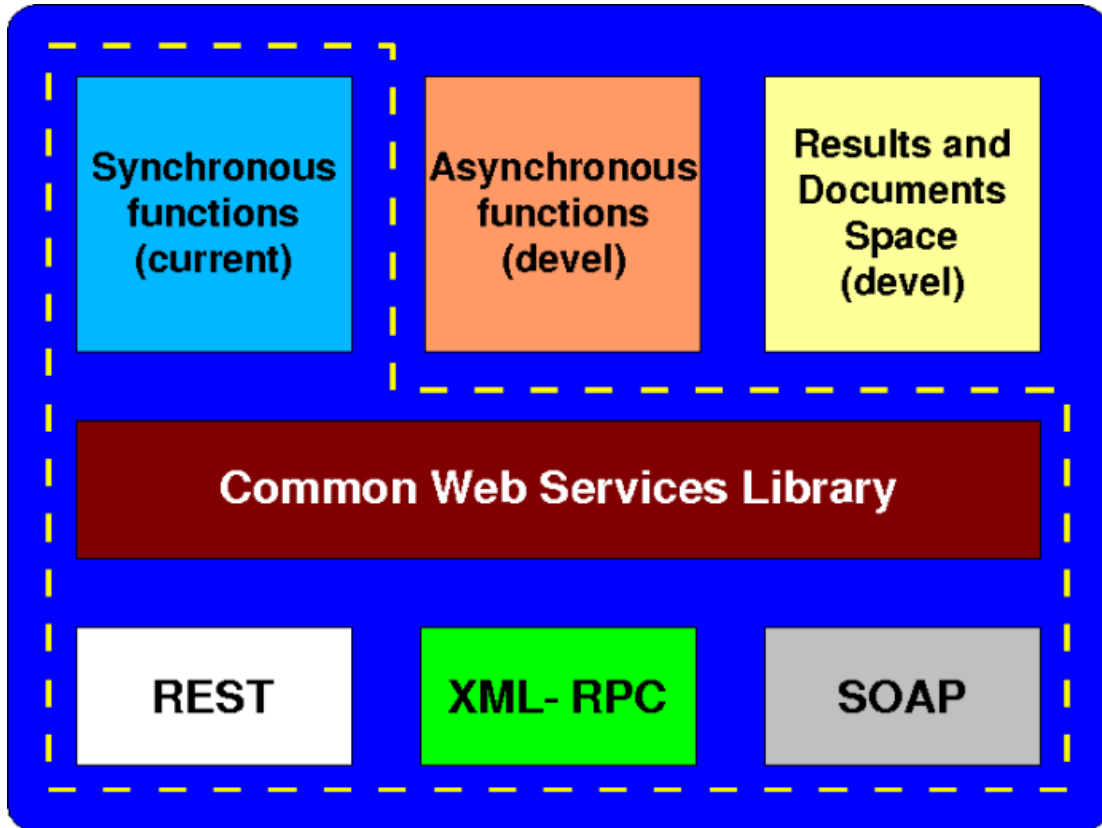
<rdf:Description rdf:ID="manifest">
<mn:channels>
<rdf:Seq>
<rdf:li
rdf:resource="http://metallo.scripps.edu/services/soap.php" />
</rdf:Seq>
</mn:channels>
</rdf:Description>
</rdf:RDF>

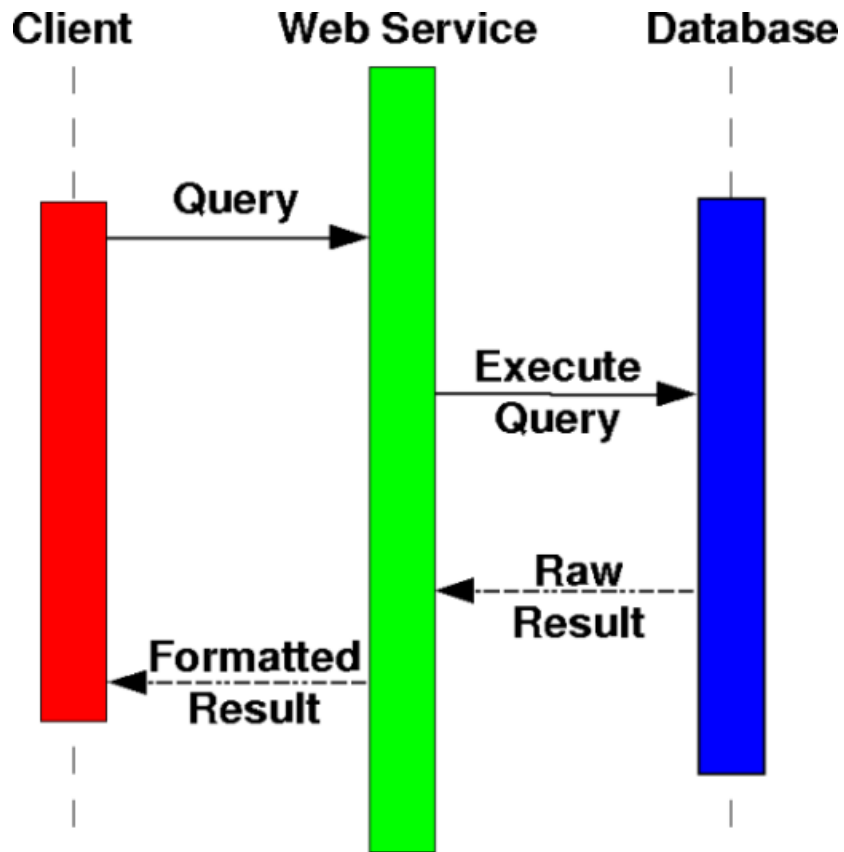
A possible problem

The client accessing the MDB web services has to wait for as long as it takes to get the results. This can be very inefficient if complex queries or analyses are requested.

... and a solution: asynchronous access

- o Decouple the request and the retrieval of the results, to allow scheduling of query execution to periods of decreased server load.
- o Query result caching: if the same query is requested by different clients, it will be executed only once.
- o Flexible result/documentation delivery. For example, an external web application can register interest in knowing the latest metalloproteins indexed in the MDB, for that it will register a callback service, a method/protocol, and a result format.





What is a tuplespace? [1]

"... (A) tuplespace is a shared datastore (the space) for simple list data structures (tuples). A very simple model is used to access the tuplespace, usually consisting of the operations write (out), take (in), read (rd) ..."

XML Spaces [2]

"... An XML-space is a tuple-space made up only of XML-tuples--that is, a public repository or buffer that can contain XML-tuples. An XML-space has a name, by which it is referenced, and it is hosted on a server, along with any number of other (and other-named) XML-spaces ..."

A Sample XML Tuple

```
<pdb_id>lhca</pdb_id>
<mdb_id>lhca_sl</mdb_id>
<metal_site>
  <metal_center>
    <metal>Zn</metal>
    <geometry>Tetrahedral</geometry>
  </metal_center>
</metal_site>
```

[1] <http://earl.strain.at/space/Tuple%20Spaces>

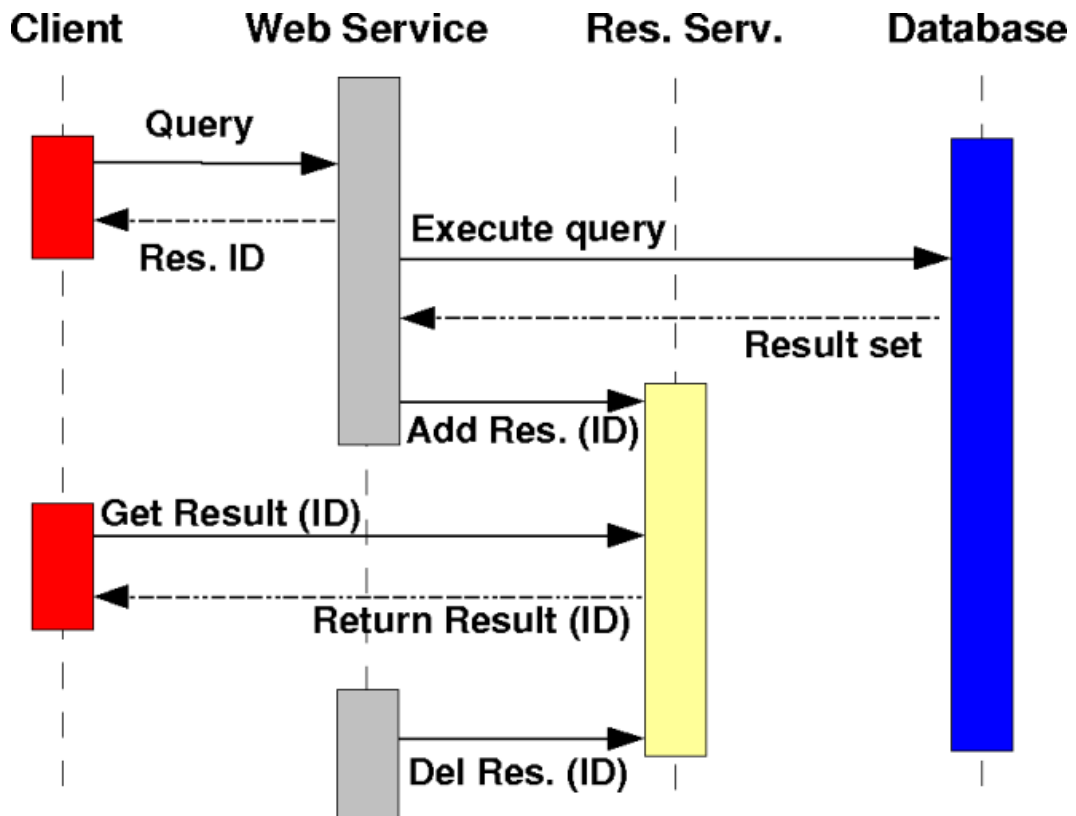
[2] "XML-Tuples and XML-Spaces, V0.7" (1999) <http://uncled.oit.unc.edu/XML/XMLSpaces.html>

Async. access

Example of asynchronous access

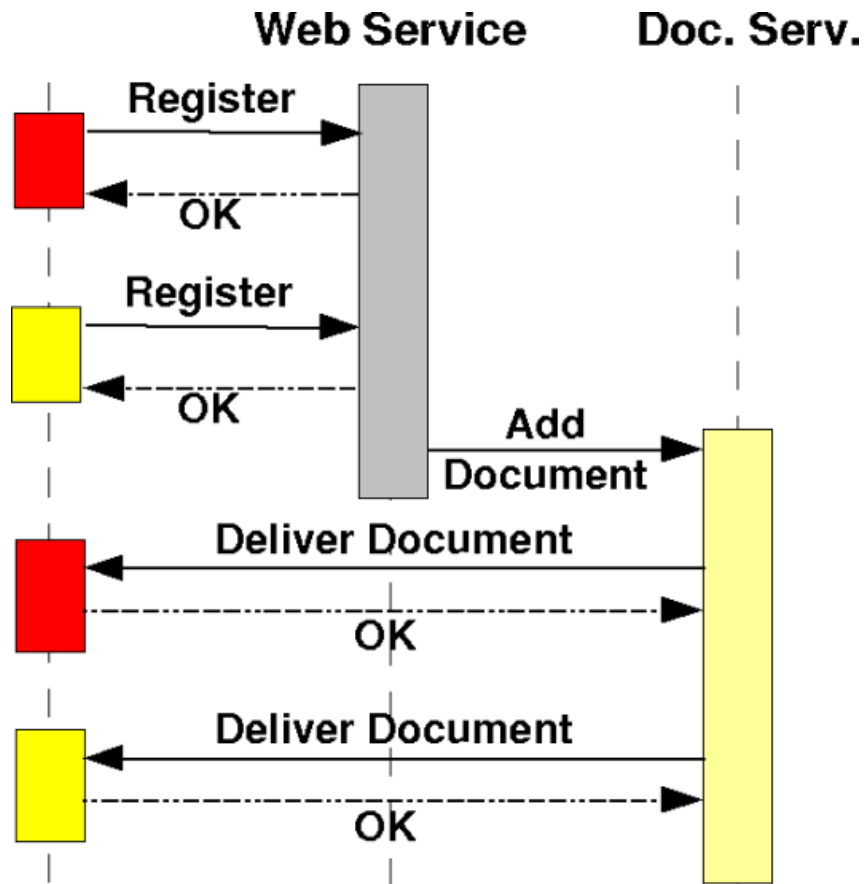
We will assume that the client connecting to the MDB web services is capable of using different protocols.

1. Web client asks for all Zn-containing proteins using the `sql()` method of the XML-RPC server.
2. XML-RPC server returns a result ID to client, then schedules query to be executed and results stored for later retrieval.
3. Query is executed and result set stored using the assigned ID.
4. Client uses SOAP to retrieve result. Server retrieves stored result set, and formats it as a complex structure using the appropriate XML Schema.



Example of callback registration

1. Web client tells the SOAP server that it wants to be informed whenever new Cu-containing proteins are indexed in the MDB. Gives as callback the URL: <http://www.example.com/newcu.jsp>, and wants to receive data in RSS format, using the HTTP POST method.
2. MDB Web services know how to use POST and format data in RSS, so a registration ID is returned. ID can be used to unregister the callback, or modify it.
3. The following week, new protein structures are indexed. Five new Cu-containing proteins are found.
4. MDB uses the registered URL and POSTs a RSS formatted document with information on the new Cu-containing proteins.
5. Callback is repeated next week unless unregistered, modified, or if an error condition was encountered the previous week (e.g. a critical HTTP error).



- o MSE: The Metal-binding Site Evaluator. Performs a full analysis on a user submitted PDB structure, generating a complete report of the metal-binding sites, their geometry, first and second shell ligands, hydrogen-bond interactions, etc. Uses our MSIT java application.

Analysis of 2sod.pdb

```

END OF PROCESS: Thu Jan 30 16:53:41 PST 2003

2sod_s1.desc

Site: 2sod_s1

[general]
number_of_metal_centers;2
number_of_first_shell_ligands;8
number_of_metal_ligand-atom_bonds;9
number_of_ligand-atom_metal_ligand-atom_angles;16
number_of_first_shell_h-bonds;7
number_of_second_shell_ligands;34
number_of_second_shell_h-bonds;19
number_of_intershell_contacts;0
number_of_intershell_h-bonds;16
number_of_disulfide-bonds;0

[metal_centers]
metal_center_1;CU.1.O;pentacoordinated;CU binding site
  metal_center_1_atom_1;CU.CU.1.O
metal_center_2;ZN.2.O;distorted tetrahedral;ZN binding site
  metal_center_2_atom_1;ZN.ZN.2.O

[first_shell_ligands]
# full residue name; denticity
HIS.44.O;1
HIS.46.O;1
HIS.61.O;1
HIS.118.O;1
HOH.3.O;1
HIS.69.O;1
HIS.78.O;1
ASP.81.O;1

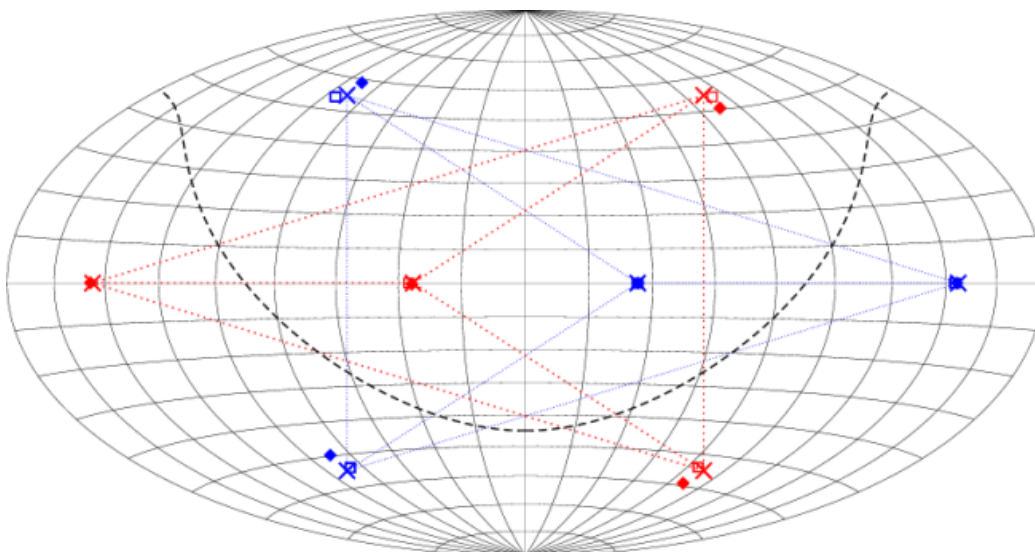
[metal_ligand-atom_bonds]
# metal atom; ligand atom; distance; type
CU.CU.1.O,ND1.HIS.44.O,2.0074546,M-L bond
CU.CU.1.O,NE2.HIS.46.O,2.1103191,M-L bond
CU.CU.1.O,NE2.HIS.61.O,2.2131639,M-L bond
CU.CU.1.O,NE2.HIS.118.O,2.0960212,M-L bond
CU.CU.1.O,O.HOH.3.O,3.1981292,M-Water bond
ZN.ZN.2.O,ND1.HIS.61.O,2.0946991,M-L bond
ZN.ZN.2.O,ND1.HIS.69.O,2.140547,M-L bond
ZN.ZN.2.O,ND1.HIS.78.O,2.0375605,M-L bond
ZN.ZN.2.O,OD1.ASP.81.O,1.9098946,M-O bond

[ligand-atom_metal_ligand-atom_angles]
# atom1; atom2; atom3; angle; dist(1-2); dist(2-3); dist(1-3);type
ND1.HIS.44.O, CU.CU.1.O, NE2.HIS.46.O, 130.22617, 2.0074546, 2.1103191, 3.7356489, L-M-L PROPER
ND1.HIS.44.O, CU.CU.1.O, NE2.HIS.61.O, 74.24425, 2.0074546, 2.2131639, 2.5630897, L-M-L PROPER

```

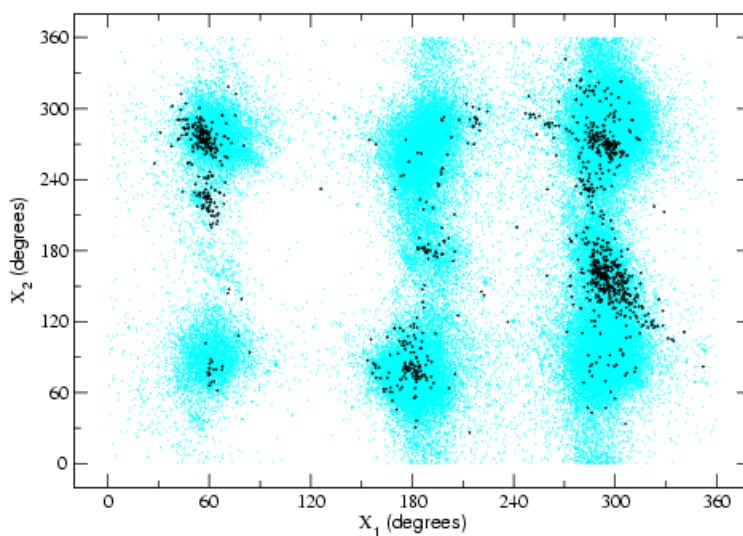
- o ClusterGeom: Java application that performs a geometrical analysis of iron-sulfur clusters ([4Fe4S] and [3Fe4S]). Generates parameters and projections that can be correlated with cluster distortion from an ideal geometry.

[4Fe4S] 2D projection



- o Expansion of the analytical tools, to include: Chi angle distributions and 2D comparisons, Bond angle distributions, Liganding pattern and parameter distribution correlations, etc.

Chi@1@-Chi@2@ distribution. All Histidines vs Zn(His)@3@X



Summary

- o Bioinformatic Web Applications must implement Web Services. Small effort to implement them, and high payoffs in terms of sharing, collaboration, and knowledge integration.
- o Use REST, XML-RPC, SOAP, or the [RPC flavor of the month]. Just keep them nice and simple, your fellow researchers will love you for that.
- o REST is the simplest way to share. We've been doing it since the CGI spec appeared. No spec on how to represent the returned data.
- o XML-RPC is simple, but data with complex structure might not be easy to represent.
- o SOAP is a more complex, but it gives you more flexibility (XML Schemas, RPC and Document Literal, WSDL, Service redirection, etc.)
- o Plan on synchronous and asynchronous access to improve performance. Consider XML Spaces and similar Tuple/Data Spaces.
- o Use the MDB Web Services. Use the MDB Web Services. Use the MDB Web Services...

Acknowledgments

People at the Metalloprotein Bioinformatics, Structure and Design Program



The "vast" MDB team (circa 1999)



\$\$ from NIH.

Users of the MDB.

Talks as Biocon 2003

- o "Integrating Distributed Bioinformatics Data Using Data Webs", Robert Grossman.
- o "Bioinformatics Data Integration Approaches via SOAP Web Services and XML", Jurgen Kaljuvee.
- o "caCORE: Infrastructure for Communal Biomedical Informatics", Peter Covitz.
- o "Best Practices for Designing XML Schemas for Bioinformatics", Ayesha Malik.

Elsewhere

- o MDB talks at <http://metallo.scripps.edu/talks/>
- o SOAP talks at <http://talks.php.net/> (general issues and implementations such as PEAR::SOAP and php-soap)
- o REST Wiki at <http://internet.conveyor.com/RESTwiki/moin.cgi/FrontPage>
- o Article: "Do Web Services Need RPC?", Benoit Marchal, <http://www.developer.com/xml/article.php/1383941>
- o XML-RPC specs at <http://www.xmlrpc.org/>
- o SOAP specs at <http://www.w3.org/TR/SOAP/>
- o SOAP implementations at <http://www.soapware.com>
- o SOAP interoperability: <http://www.soapbuilders.com/>, <http://www.whitemesa.com/interop.htm>, <http://www.whitemesa.com/r3/interop3.html>
- o Tuple Spaces page at the XML Cover Pages <http://xml.coverpages.org/tupleSpaces.html>

This and other MDB related talks at:

<http://metallo.scripps.edu/talks>

Index

Agenda	2
Web Services	3
REST (HTTP methods)	4
XML-RPC	5
SOAP	6
WSDL & UDDI	7
Bioinformatics	8
Why Web Services	9
PDB Stats	10
Metalloprotein	11
MDB: Background	12
MDB: Description	13
MDB as Library	14
A (kludgy) first API	15
IMB @ Jena	16
MDB Remote Viewer	18
MDB: Architecture	20
Implementing the services	21
The Base Library	22
function sqlquery	23
function getPDBFromMetal	24
HTTP-based API	25
(REST) Introspection	28
Testing: sql()	30
Testing: metallopdb()	31
XML-RPC Instrospection	32
method.sql	34
method.metallopdb	36
SOAP service	40
Test SOAP	42
Accessing web services	47
Access modes	48
Sync. access	49
Tuplespaces	50
Async. access	51
Async. access	52
Register interest	53
Register interest	54
Future services	55
Summary	57
Acknowledgments	58
Further reading	59
Where to find this talk	60