

Querying a Relational Database of SPASE Metadata

Cory Nathe
Augsburg College

What is it for?

- # Implement a site to insert SPASE records into a database.
 - # Implement a SPASE based interface for a client to query the database.
 - # The client is located on the *space.augsburg.edu* server while the database is on the *xspace.augsburg.edu* server.
-

How does it work?

- # Two servers communicate using SOAP to exchange XML-based messages.
 - # Simple Object Access Protocol (SOAP) is a protocol for exchanging XML-based messages over computer networks, using HTTP.
-

Query process: Step 1

SPASE Database Query System

FindResources: (Pass constraints and get a list of ResourceIDs for matching resources)

ResourceType:

ResourceID: *Example: spase://augzburg/RES_000001*

StartDate: - - to EndDate: - - *Example: 2005-01-01 to 2006-01-01*

ReleaseDate: - - *Example: < 2005-01-01*

Format:

GetDetails: (Pass a ResourceID and get a SPASE description XML file)

ResourceType:

ResourceID: *Example: spase://augzburg/RES_000001*

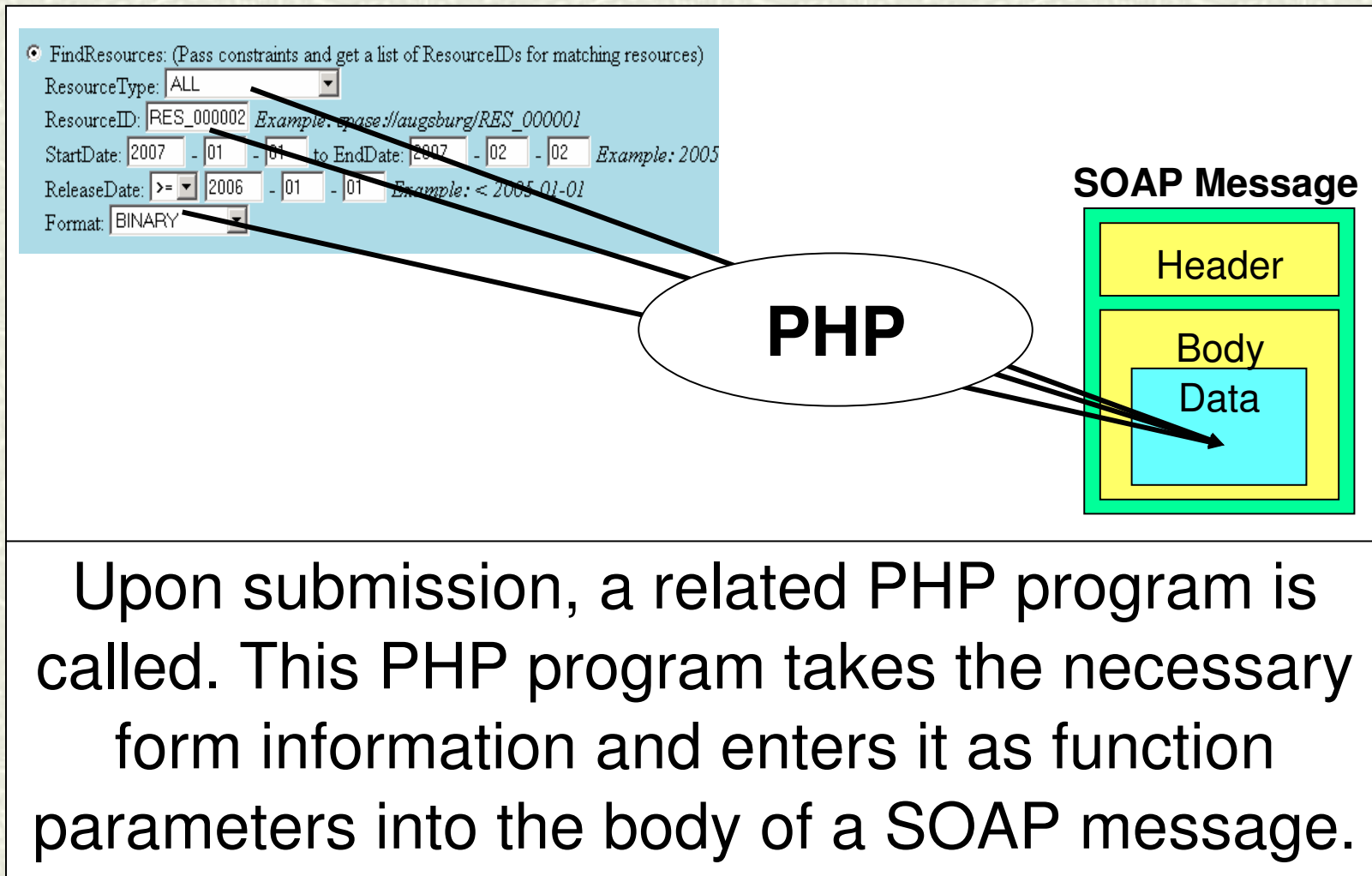
GetResource: (Pass a ResourceID and get an Access URL)

ResourceType:

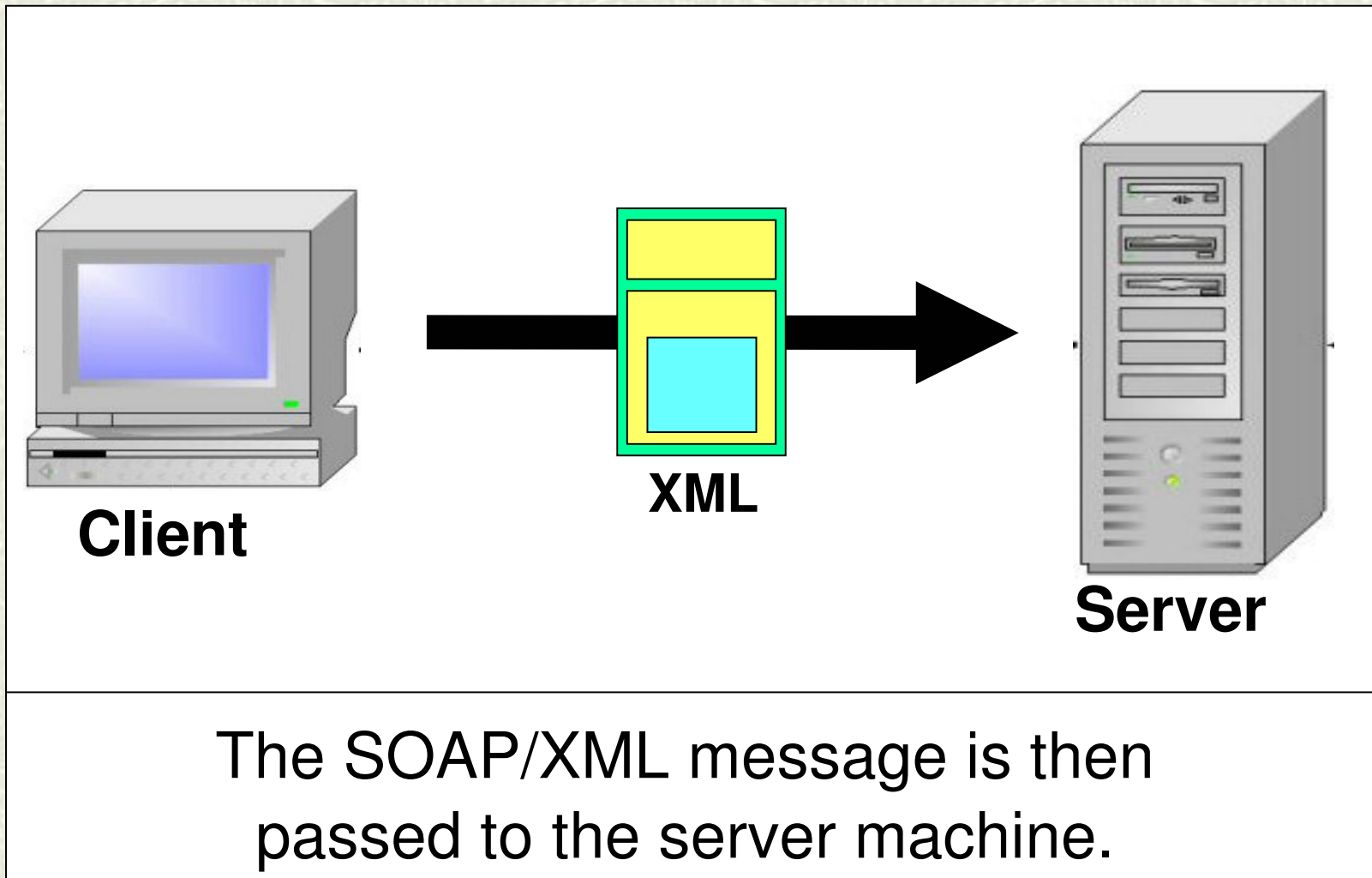
ResourceID: *Example: spase://augzburg/RES_000001*

At the client side, the query information is entered into an HTML form.

Query process: Step 2

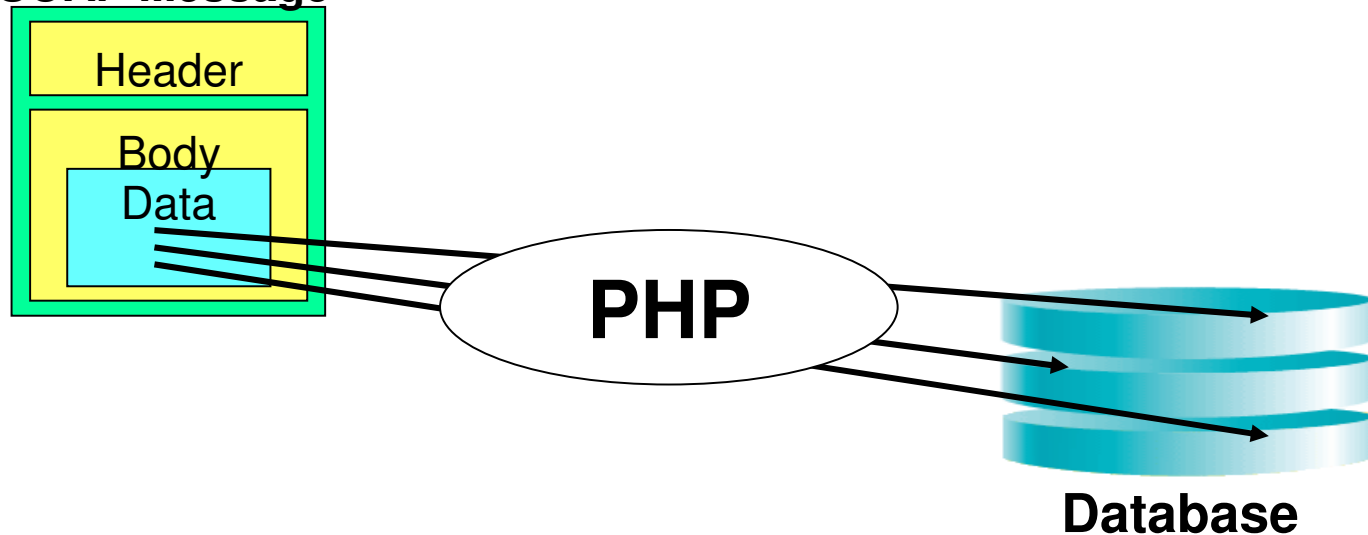


Query process: Step 3



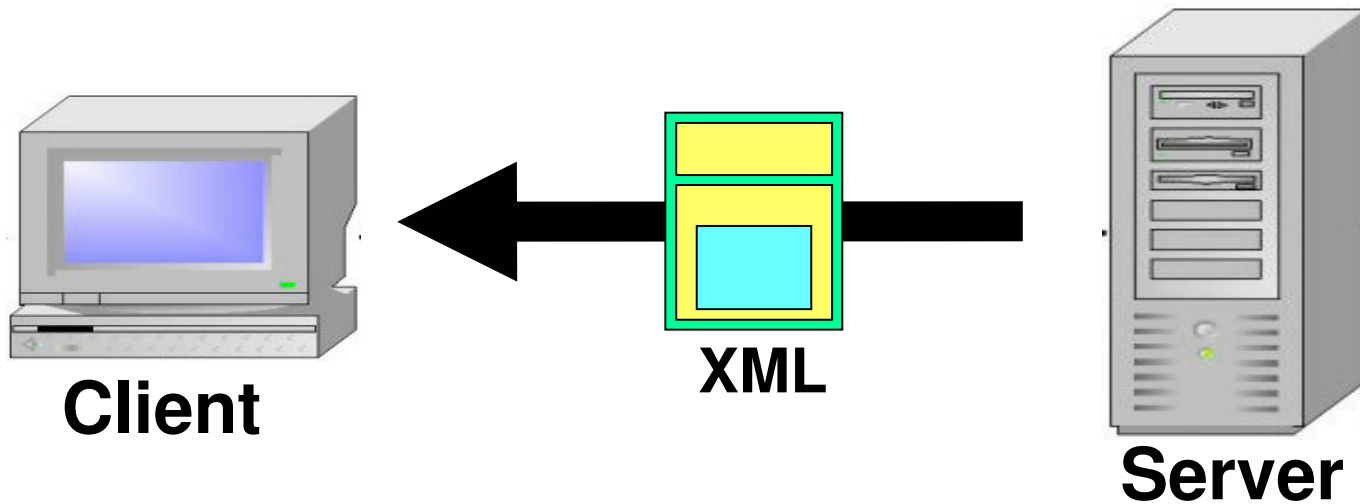
Query process: Step 4

SOAP Message



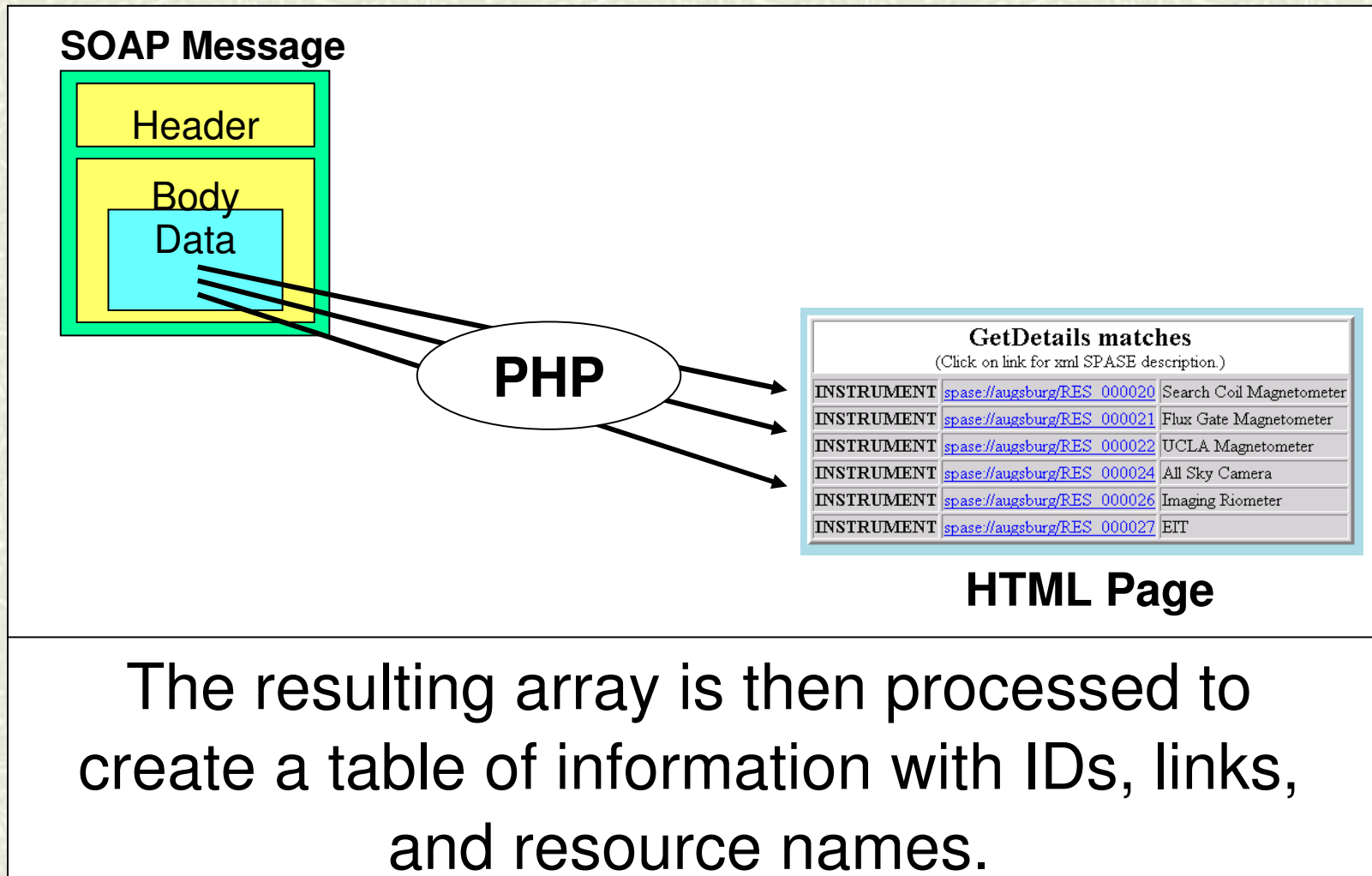
The server then takes the parameters from the message and performs the function call through another PHP program. This, in turn, queries the necessary SPASE database tables and fields.

Query process: Step 5



Upon completion of the query, the server machine returns the results to the client as an array of database field information in an XML message using SOAP.

Query process: Step 6



The resulting array is then processed to create a table of information with IDs, links, and resource names.

Demonstration

- # <http://xspace.augsburg.edu/~skar/spaseTest.html>
- # <http://space.augsburg.edu/~nathe/spase/query.html>

Example SOAP message

Client Request

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <getProductDetails xmlns="http://warehouse.example.com/ws">
      <productID>827635</productID>
    </getProductDetails>
  </soap:Body>
</soap:Envelope>
```

Server Response

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <getProductDetailsResponse xmlns="http://warehouse.example.com/ws">
      <getProductDetailsResult>
        <productName>Toptimate 3-Piece Set</productName>
        <productID>827635</productID>
        <description>3-Piece luggage set. Black Polyester.</description>
        <price currency="NIS">96.50</price>
        <inStock>true</inStock>
      </getProductDetailsResult>
    </getProductDetailsResponse>
  </soap:Body>
</soap:Envelope>
```

Client PHP program

```
$namespace = "http://xspace.augsburg.edu/~nathe/soapServer";
$xmlDir="http://xspace.augsburg.edu/~skar/xml/";
require_once('SOAP/Client.php');
$info = new SOAP_client("http://xspace.augsburg.edu/~nathe/soapServer/databaseserver.php");

print "<html><head><title>SPASE Query Results</title></head>";
print "<body bgcolor='lightblue'>";
print "<hr>";

if($resType!="")
{
    $params = array("resType"=>$resType,"residFind"=>$residFind,"format"=>$format,
        "startDate"=>$startDate,"stopDate"=>$stopDate,"releaseOp"=>$releaseOp,
        "releaseDate"=>$releaseDate);
    $response = $info->call("FindResources",$params,$namespace);
    if($response->message)
    {
        echo "ERROR message : " . $response->message;
    }
}
```

Server PHP program

<?

```
class database
{
    var $link;
    function database()
    {
        $link=mysql_connect("localhost:/tmp/mysql.sock", "skar", "qazwsx");
        mysql_select_db("Spase",$link);
    }

    function FindResources($resType,$residFind,$format,$startDate,
        $stopDate,$releaseOp,$releaseDate)
    {
        $test="RESOURCEID like '%".$residFind."%";
```