Integration of Live Access Server with Climate Data Analysis Tools (CDAT)

Velimir Mlaker
LLNL

30-May-2007

LAS CDAT for WCRP CMIP3 Multi-Model Data

About the data

World Climate Research Programme's Coupled Model Intercomparison Project Phase 3

- Used by IPCC scientists in preparation for AR4
- 35 TB
- 80,000 files
- 12 experiments, 23 models
- Institutions from 13 countries contributed model output
- Used in publishing of about 250 scientific papers
- Over 1000 registered users worldwide

About CDAT

Climate Data Analysis Tools

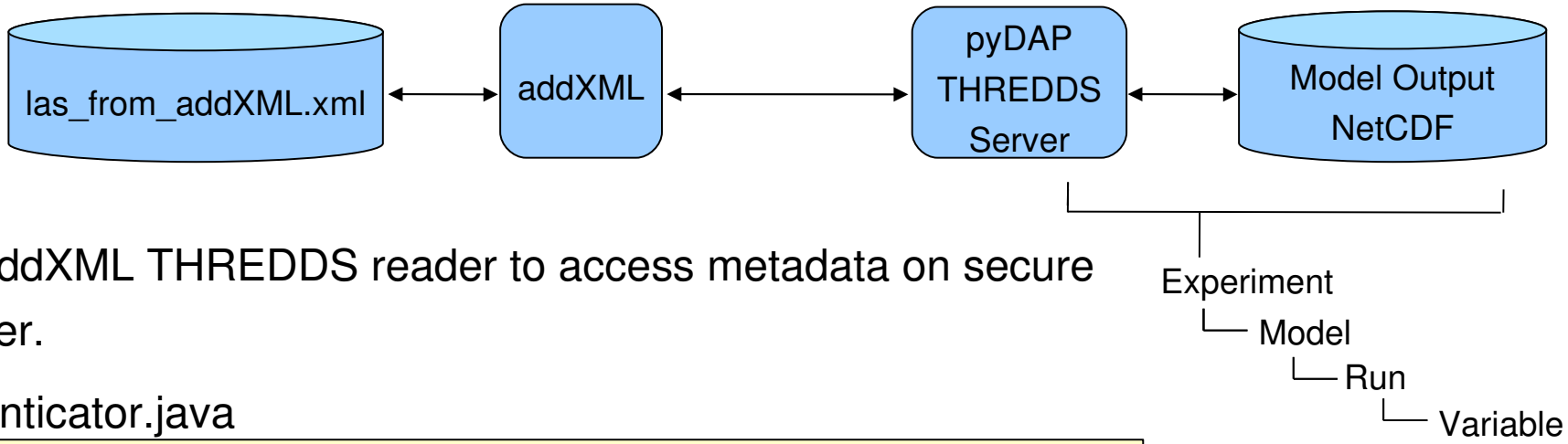
- Open source object-oriented data analysis in Python

LAS version

- armstrong-1.0
- addXML-1.2

LAS CDAT for WCRP CMIP3 Multi-Model Data

Changes to configuration: addXML1.2



Enhanced addXML THREDDS reader to access metadata on secure pyDAP server.

MyAuthenticator.java

```
public class MyAuthenticator extends java.net.Authenticator{
   private String username, password;
   public MyAuthenticator (String user, String pass){...}
   protected java.net.PasswordAuthentication
      getPasswordAuthentication (){...}
}
```

addXML.java

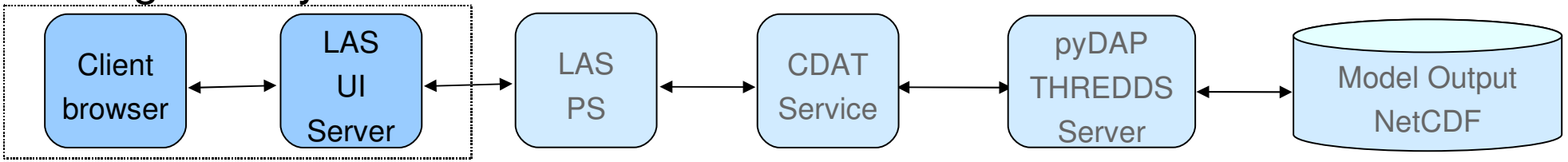
```
public static void main(String[] args) {
   ...
   java.net.Authenticator.setDefault(
      new MyAuthenticator(username, password));
   ...
}
```

The command:

```
addXML.sh -u <username> -p <password> -t http://esgcet.llnl.gov/dap/ipcc4/?thredds
```

LAS CDAT for WCRP CMIP3 Multi-Model Data

Changes to system: User Authentication Part 1



Tomcat security filter restricts website content to only allow registered users.

web.xml

```
<filter>
   <filter-name>SecurityFilter</filter-name>
   <filter-class>security.SecurityFilter</filter-class>
</filter>
<filter-mapping>
   <filter-name>SecurityFilter</filter-name>
   <url-pattern>/servlets/*</url-pattern>
</filter-mapping>
```

Filter configuration:

- Named object
- Path restriction

SecurityFilter.java

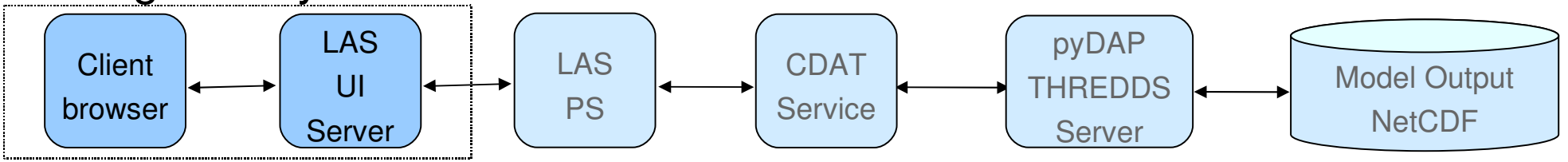
```
public class SecurityFilter implements javax.servlet.Filter{
   public void doFilter(...){
      String username = (String)session.getAttribute("username");
      if(username != null){
         chain.doFilter(hreq,hres);
      }else{
         hres.sendRedirect("/login.jsp");
      }
   }
}
```

Filter implementation:

1) Try to authorize session by lookup
2) Proceed if o.k.
3) Redirect to login page if failed

LAS CDAT for WCRP CMIP3 Multi-Model Data

Changes to system: User Authentication Part 2



Forwarding to login page is defined in the Struts framework.

login.jsp

```
<html:form action="/SubmitLogon" onsubmit="return=ValidateLogonForm(this);">
   <td><html:text  property="username"/></td>
   <td><html:password property="password"/></td>
   <td><input type="submit" name="Submit" value="Submit"/></td>
```

LogonAction.java

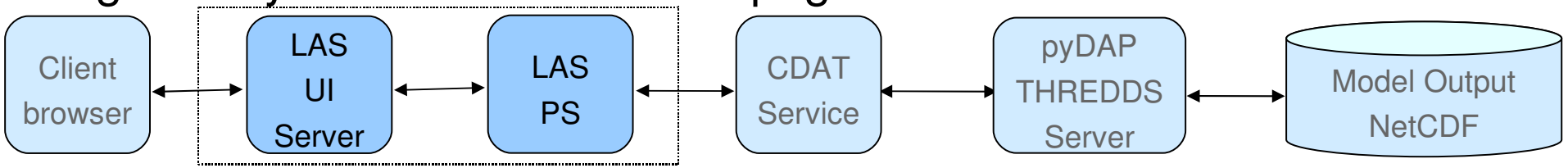
```
public class LogonAction extends org.apache.struts.action.Action {
   public ActionForward execute(...){
      String username = PropertyUtils.getSimpleProperty(form,"username");
      String password = PropertyUtils.getSimpleProperty(form,"password");
      if (authenticate(username, password)){
         session.setAttribute("username", username);
         session.setAttribute("password", password);
         return mapping.findForward("success");
      }else{
         return mapping.findForward("failure");
      }
```

struts-config.xml

```
<form-bean name="LogonForm" type="org.apache.struts.validator.DynaValidatorForm">
   <form-property name="username" type="java.lang.String"/>
   <form-property name="password" type="java.lang.String"/>
</form-bean>
<action path="/SubmitLogon" type="security.LogonAction"
        name="LogonForm" input="logon">
   <forward name="success" path="/" redirect="false"/>
   <forward name="failure" path="/login.jsp?action=fail" redirect="false"/>
</action>
```

LAS CDAT for WCRP CMIP3 Multi-Model Data

Changes to system: Credentials Propagation



Files affected:

- luis/XmlRequester.java
- luis/LogoutBean.java
- VM_global_library.vm

DataBean.java (UI)

```
username = session.getAttribute("username");
password = session.getAttribute("password");

xml.setCredentials(username, password);

sendRedirect(xml);
```

UI to PS

ProductRequest.java (PS)

```
username = request.getRootElement().getChild("username");
password = request.getRootElement().getChild("password");

backendRequest.addContent("username", username);
backendRequest.addContent("password", password);
```

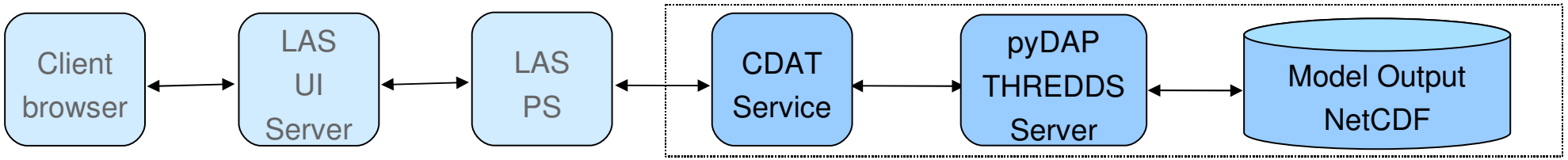
```
...

<backend_request>
  <region ID="region_0">
    <x_lo>-180</x_lo>
    <x_hi>180</x_hi>
    <y_lo>0</y_lo>
    <y_hi>0</y_hi>
  </region>
  <username>user1</username>
  <password>pass1</password>
  <properties>
```

TO BACKEND SERVICE

LAS CDAT for WCRP CMIP3 Multi-Model Data

CDAT Backend Service



Service implemented in Python

- Uses open-source SOAPpy library
- Direct inline calls to CDAT

Namespace from ProductWebService.java

Service details from productserver.xml

Start the backend SOAP service

```
SOAPpy.SOAPServer(("esgcet.llnl.gov", "6155"))
server.registerKWFunction (getProduct, "LASBackend")
server.serve_forever()
```

```
<backend_request>
  <region ID="region_0">
    <x_lo>-180</x_lo>
    <x_hi>180</x_hi>
    <y_lo>0</y_lo>
    <y_hi>0</y_hi>
  </region>
  <username>user1</username>
  <password>pass1</password>
  <properties>
```

FROM PS

XML parameter as indicated in ProductWebService.java

CDAT Callback

```
def getProduct(**kw)
   request = kw["lasBackendRequest"]
   doc = xml.dom.minidom.parseString(request).documentElement
   username = doc.getElementsByTagName("username")
   password = doc.getElementsByTagName("password")

   data = cdms.open(("http://%s:%s@"%(username,password) + url)

   vcs.canvas.plot(...)
```

CDAT reads netCDF via pyDAP server

CDAT generates plot

.gif

LAS CDAT for WCRP CMIP3 Multi-Model Data

http://esgcet.llnl.gov:8090/ipcclas