

Client Programming Interfaces and Examples

Greg Riccardi
riccardi@cs.fsu.edu
5 August 2003

1. Example of client processing with DAIS and JDBC

Method `executeQuery` of class `SqlDataResource` provides a simple way for client applications to execute the JDBC `executeQuery` method using a Grid Data Resource as the database interface. The following section lists the interfaces that support class `SqlDataResource`.

```
1      public class SqlDataResource implements dais.DataResource {
2          public RowSet executeQuery (String query)
3              throws dais.DAISException, SQLException {
4              // returns a cached row set obtained from the data resource
5              // local variables
6              RequestDataSet ds = null;
7              ResponseDataSet resp = null;
8              Activity req1 = null;
9              SqlQueryResponseElement elem = null;
10             // create a request and call performImmediate
11             ds = new OgsaDaiRequestDataSet();
12             req1 = new OgsaDaiSqlQuery("req1", query);
13             ds.addElement("req1", "Request.Query.SqlQuery", req1);
14             resp = performImmediate(ds);
15             if (! resp.successful("req1")) {
16                 throw resp.getException("req1");
17             }
18             // extract response element and convert to RowSet
19             elem = (SqlQueryResponseElement) resp.getElement("req1.resp2");
20             return elem.getRowSet();
21         }
```

The key to this method is the `getRowSet` method of line 20. This method must create a `RowSet` object from the XML version of the query response that is returned by the call to `performImmediate` in line 14.

Method `getRowSet` is simple to implement as long as query response is returned as the XML produced by the `WebRowSet` `writeXML` method.

```
1      class SqlQueryResponseElement {
2          public RowSet getRowSet() throws DAISException {
3              WebRowSet rs = null;
4              try {
5                  rs = new WebRowSet();
6                  java.io.Reader reader =
7                      new StringReader(getValue().toString());
8                  rs.readXml(reader);
```

```

9         return rs;
10    } catch (java.sql.SQLException e){
11        throw new DAISException
12            ("SQL problem with creating WebRowSet:"+e.getMessage());
13    }
14 }
15 }

```

2 Client libraries for dataset and data resource

A set of basic interfaces and methods for client library for DAIS

```

16 package dataset;
17 public interface DataSet {
18     public addElement(String name, Type type, Value value);
19     public void setValue(String name, Value value);
20     public Type getType(String Name);
21     public Value getValue(String Name);
22     public String[] getNames();
23 }
24
25 package dais;
26 public interface RequestDataSet extends DataSet {
27 }
28 public interface ResponseDataSet extends DataSet{
29 }
30 public interface Activity extends Value{
31 // base interface for type of a dataset element
32     public String getType();
33 }
34 public interface DataResource {
35     public DataSet performImmediate(DataSet req);
36 }
37
38 package dais;
39 public interface Request extends Activity{}
40 public interface Query extends Request {}
41 public interface SqlQuery extends Query {}
42 public interface Update extends Request {}
43 public interface Update extends Request {}
44 public interface Response extends Activity{}
45 public interface Query extends Response {}
46 public interface SqlQuery extends Activity {
47 }
48 public interface SqlQueryResponseElement {
49     RowSet getRowSet(ResponseDataSet ds);
50 }
51
52
53 package ogsadai;
54 public class OgsaDaiRequestDataSet implements dais.RequestDataSet {
55     public OgsaDaiRequestDataSet(String name, String query){ }
56 }
57 public class OgsaDaiSqlQuery implements SqlQuery {
58     public OgsaDaiSqlQuery(String name, String query){ }
59 }

```

```
60 public class OgsaDaiRequestDataSet implements RequestDataSet {
61     public OgsaDaiRequestDataSet (){}
62 }
```

```
63
```