WS-DAI-DM: An Interface Specification for Data Mining in Grid Environments

Yan Zhang
School of Computer and Information Technology, Beijing Jiaotong University, Beijing, P.R. China
Email: 07112062@bjtu.edu.cn

Luoming Meng*, Honghui Li#, Alexander Woehrer†, and Peter Brezany†
* School of Computer Science and Technology, Beijing University of Posts and Telecommunications, Beijing, China
Email: lmmeng@bupt.edu.cn
#School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
Email: hhli@bjtu.edu.cn
†Institute of Scientific Computing, University of Vienna, Vienna, Austria
Email: {woehrer, brezany}@par.univie.ac.at

***Abstract* — Providing the appropriate access means for data mining services in Grid Environment is principal for combination of Grid and data mining. The transition from centralized data mining process as they are in traditional tools to Grid-compliant and Grid-based data mining services that can coordinate with each other is important to extract useful and potential knowledge/patterns from distributed data resources. But, data mining process has not been integrated with the current Grid architecture, and there are no related specifications or standards available within the Grid community for handling it. The work discusses the interface specification WS-DAI-DM and related issues of providing the consistent web service interfaces to mine meaningful and hidden information/patterns from distributed data resources in Grid environments. WS-DAI-DM can facilitate using data mining tools and developing data mining applications. The proposed specification idea is partially implemented in the feasibility study and we give an application scenario about how users can access data mining services in Grid environments conveniently via the proposed mechanism.**

***Index Terms* — WS-DAI-DM, data mining access service, Open Grid Service Architecture, data mining interface specification, OGSA-DAI WS-DAI-DM**

I. Introduction

With the accumulation and combination of data from different domains, the scale, complexity and diversity of these data are growing rapidly [1]. These data opens many opportunities for modern business, scientific research, improved decisions and better policies [2]. Grid technology provides an effective and integrated approach to share and coordinate the usage of geographically distributed heterogeneous large-scale data resources via dynamic and distributed Virtual Organizations [3]. Compared with conventional technologies, Grid computing is flexible, autonomous, dynamic and service-oriented and it can make the coordinated interaction with distributed data resources more systematic and more effective [4]. How to make better use of these distributed, heterogeneous and large size data resources managed by Grid platform and extract potential information from them is one of the most important problems of many applications in Grid environments. Providing the appropriate access means for data mining in Grid environments is the foundation for combination of Grid platform and data mining. But right now, the current Grid architecture does not make data mining services involved, and there are no related specifications or standards available within the Grid community for handling it. In this paper we present WS-DAI-DM - Web Service Data Access and Integration Data Mining, a WS-DAI-based data mining access mechanism in Grid architecture which is defined by the Open Grid Architecture (OGSA). In Fig.1, the gray box in layer II shows where we are and what we will complete in this work. It provides users and top-layer applications (layer I) with easy-to-use and simple web service interfaces, and hides the implementation of data mining process (layer III) and information about data resources

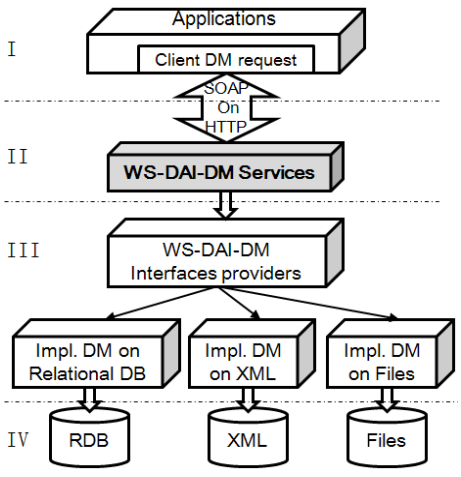


Figure 1. The function of WS-DAI-DM specification

doi:10.4304/jsw.6.6.953-960

(layer IV). Here, the distributed data resources can be relational databases, XML data and different types of files. Briefly, the proposed mechanism provides an approach for accessing distributed data mining resources in Grid environments and the paper discusses the related concepts and models.

The rest of paper is structured as follows: Section 2 introduces needed background and discusses related work. Section 3 represents the kernel part of this paper by describing WS-DAI-DM in detail, the proposed data mining access mechanism in Grid environments, including related concepts, access patterns and the design idea. The implementation of the proposed specification is discussed in Section 4 and then we give an application scenario to show how users can use this data mining access mechanism in Section 5. We finish in Section 6 with conclusions and an outlook on future work.

II. BACKGROUND AND RELATED WORK

The WS-DAI (Web Service Data Access and Integration) Family of specifications is shown as Fig.2 [5], and three sub-specifications extend the super-specification - WS-DAI. The specification WS-DAI is on the top level and it defines the core data access services which are web service interfaces to data resources [6]. Here, the main components of WS-DAI specifications are the following: data services, interfaces, properties, messages and data resources. Here, data services are web services that implement one or more WS-DAI-based interfaces to access distributed data resources; Interfaces are used to interact with data services; Properties are used to describe data services; Messages are sent to data services; Data resources represent any system that can act as a source of data [6]. The WS-DAI is a data model-independent specification which provides a basic and extensible framework for data access service interfaces [5]. We can define interfaces, properties and messages for accessing specific data resources according to the set of standard definition in WS-DAI. In fact, the WS-DAI specification does not provide fully transparent access to existing data resources, nor does it take responsibility for parsing language statements [5]. The underlying realizations describe the specific interfaces, messages and properties required to access specific data resources, such as relational data, RDF resources and XML sources. They implement similar message patterns and share the properties defined in WS-DAI, and different realization specifications could be used to access different types of data resources[5]. The WS-DAIR realization defines the access mechanism for relational databases [7]. The WS-DAIX realization describes a XML access mechanism [8]. The WS-DAI RDF(s) realization provides access to RDF(s) data [9][10][11]. These realizations are consistent with the framework defined by the WS-DAI core specification.

The OGSA-DAI team of the University of Edinburgh implemented the WS-DAIR and WS-DAIX specifications using OGSA-DAI 3.1, in which four-layer architecture is used, they are services, PortType providers, executors and resources, and executors implement operations either directly or by the execution of OGSA-DAI workflows [12] [13].

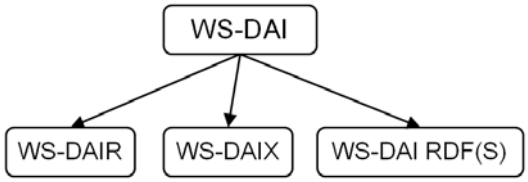


Figure 2. WS-DAI family of specification

Some expert groups focus on providing web service interfaces for data mining. Java Data Mining (JDM) was designed to provide standard-based interfaces both through Java and web services, and vendors should use the Technology Compatibility Kit (TCK) to implement the standard [14]. JDM conforms to SUN's Java Community Process listed as JSR-73 [15] and JSR-247 for JDM 2.0 [16], but it didn't take the distributed data resources into consideration. In the Weka4WS framework, WSRF-compliant web services are used to expose all the data mining algorithms provided by the Weka library, and the Weka4WS is implemented using the WSRF libraries and services provided by Globus Toolkit 4 [17]. The Weka4WS aims to provide web service interfaces for existing functions in Weka instead of providing consistent web service interfaces for all data mining tools/algorithms in Grid environments.

III. WS-DAI-DM: DATA MINING REALIZATION

Data mining access can plays a central role for many types of Grid applications. By data mining access we mean the ability to compose data and mine information from many data mining resources. Instead of simple datasets, what we can get through this process is data mining models which should be compliant with some standards and could be used by other applications easily and conveniently. WS-DAI-DM realization describes a collection of data mining access interfaces for data mining resources.

A. Possibilities For Data Mining Access in Grid Environments

There are two possibilities for integration of data mining access mechanism with web service-based Grid architecture. These mechanisms could be defined in two different ways as outlined below:

- To provide data mining access mechanism WS-DM (Web Service Data Mining) as an independent framework, which contain the identifying features of the data mining access services, which is totally different from existing data access services. This approach provides completely new access mechanism and it is at the same level with WS-DAI.
- To provide data mining access mechanism as a realization of WS-DAI specification, which is extended from WS-DAI and is at the same level with WS-DAIR, WS-DAIX and WS-DAI-RDF(S).

For the above two approaches, we use the latter for integration of data mining access in Grid environments.

Firstly, data mining access can be seen as an access to data resources through data mining query which is similar to relational query and XML query; Secondly, the whole data mining process is invisible to end-users, which is similar to the execution process of relational query. Thirdly, we need not to start from scratch since existing WS-DAI specification can be adopted.

B. Related Concepts

For the further discussion about WS-DAI-DM realization, the foundational concepts used in WS-DAI-DM are described as follows:

Data mining means data mining process which starts from receiving a data mining query/request and ends with responding a data mining result model, data pre-process and evaluation of data mining models are not included.

Data mining access service extends the definition of data access services defined in the core WS-DAI specification, it is the web service that implements one or more specified interfaces to extract useful and potential information/patterns from distributed data resources and then generate standard data mining result models.

Data mining query is composed of a list of parameters and it has nothing to do with any specific data mining languages and data mining algorithms, and it also does not define any data mining language.

Data mining model means the result of data mining process and it can be compliant with the data mining standards, such as PMML [18]. The generated data mining model can be used by other applications easily.

Data mining resources are data source/sink that can be queried to get meaningful information/patternts as the result of an underlying data mining processes, and it is same with data resources defined in the WS-DAI specification.

C. Properties

WS-DAI-DM realization extends a collection of properties defined in the WS-DAI specification [6]. We add the following properties:

SchemaDescription contains an XML document that describes the schema of the data mining resources, because the data mining resources are diverse and could be relational databases, XML data or plain files, etc. The element *SchemaFormatURI* is the URI of the schema which will be defined by the service implementers; the element *SchemaContent* is the content of the schema of data mining resources. For example, a relational database, *SchemaFormatURI* could be defined as part of the Common Information Model (CIM) of the DMTF [19], and corresponding *SchemaContent* will describe the database by CIM schema. The following is the structure of *SchemaDescriptionType*. The type of element *SchemaDescription* is *SchemaDescriptionType*

```
<SchemaDescriptionType>
    <wsdaidm:SchemaFormatURI/>
    <wsdaidm:SchemaContent/>
</SchemaDescriptionType>
<xsd:element name="SchemaDescription" type="wsdaidm:
SchemaDescriptionType"/>
```

DMAccessPropertyDocument collects all of the properties of a data mining service and provides description information about capabilities of a data mining resource. It extends the base properties defined by *PropertyDocument* in the WS-DAI specification to include the *SchemaDescription* element. The following is the structure of *DMAccessPropertyDocumentType*. The type of element *DMAccessPropertyDocument* is *DMAccessPropertyDocumentType.*

```
<DMAccessPropertyDocumentType>
    <wsdai:PropertyDocument/>
    <wsdaidm:SchemaDescription/>
</DMAccessPropertyDocumentType>
<xsd:element name=" DMAccessPropertyDocument"
 type="wsdaidm: DMAccessPropertyDocumentType "/>
```

D. Data Mining Access Interfaces

WS-DAI-DM extends the base interfaces and corresponding properties defined in the WS-DAI specification to exploit potential information/patterns and generate data mining models from data mining resources. In this subsection we will describe all the interfaces in detail.

1) ***DMAccess****:* Direct access allows the data mining models to be delivered to the consumer in the response message. It is one of the two access patterns described in the WS-DAI core specification. To cater for this mode of operation the specification defines *DMAccess* interface. It collects together messages that directly mine knowledge/patterns from the data mining resources represented by a data mining access service along with the properties that describe the behavior of these access messages. The *DMAccess* interface provides access to extract hidden information from underlying data mining resources and returns data mining result model to consumers. *DMAccess* interface provides two operations: (i) *GetDMAccessPropertyDocument* can expose all properties of the service, the input message contains *DataResourceAbstractName* - the abstract name of the data resource for which properties are required and the output is the properties described as *DMAccessPropertyDocument*; (ii) *DMExecute* can submit a data mining query and then get the result model. The associated *DMExecuteResponse* message is formatted according to the value specified in the *DatasetFormatURI*, as illustrated in Fig.3. The structures of the input request message and the output response message are as follows:

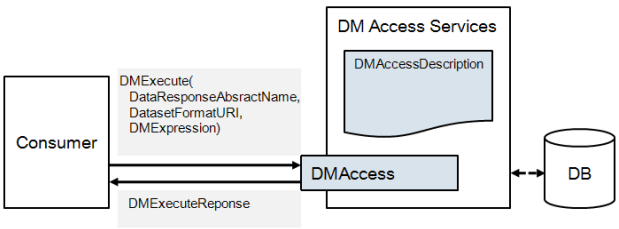


Figure 3. DMAccess- Direct data mining access

```
<DMExecuteRequest>
    <wsdai:DataResourceAbstractName/>
    <wsdai:DatasetFormatURI/>
    <wsdaidm:DMExpression/>
</DMExecuteRequest>

<DMExecuteResponse>
    <wsdai:DatasetFormatURI/>
    <wsdai:DatasetData/>
</DMExecuteResponse>
```

DMExpressionType is a new type defined in WS-DAI-DM, which is composed with parameters list. The following is the structure of *DMExpressionType*. A data mining query can be transformed as parameters list, such as for sequential pattern mining, the value of *AlgorithmType* is *SequentialPattern*, and table name, support value and confidence value, etc are stored in *DMParameter*.

```
<DMExpressionType>
    <wsdai:ExpressionType/>
    <AlgorithmType/>
    <wsdaidm:DMParameter/>+
</DMExpressionType>

<xsd:element name="DMExpression"
type="wsdaidm:DMExpressionType"/>
```

2) ***DMAccessFactory****:* Indirect access is another access pattern which the WS-DAI core specification supports. This pattern allows the data mining model, usually the result of a data mining query, to be accessed by way of a new service-managed data resource, and thus the result data - data mining model is not returned directly to the consumer. Indirect access is supported through the use of the factory pattern and it can minimize data movement and client-server communication by returning end-point reference in the response message instead of the result itself. *DMAccessFactory* interface is defined to cater for indirect access and it is used to create and provide access to the derived data mining model representing the result-model of a data mining query/request.

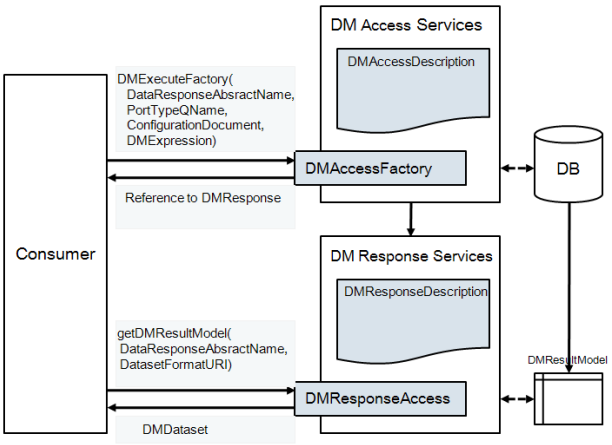


Figure 4. Indirect data mining access

DMAccessFactory interface provides *DMExecuteFactory* operation which is used to create a derived data mining model - the result of the data mining query and make this model available through, potentially, a separate data mining access service - *DMResponse* service, as shown in Fig.4. The structures of the input request message and the output response message are as follows:

```
<DMExecuteFactoryRequest>
    <wsdai:DataResourceAbstractName/>
    <wsdai:PortTypeQName/>
    <wsdai:ConfigurationDocument/>
    <wsdai:PreferredTargetService/>
    <wsdaidm:DMExpression/>
</DMExecuteFactoryRequest>

<DMExecuteFactoryResponse>
    <wsdai:DataResourceAddress/>
</DMExecuteFactoryResponse>
```

3) ***DMResponseAccess****:* In order to support access to the data mining models resulting from the use of the factory pattern, *DMResponseAccess* interface is defined to provide access to the *DMDataset* resulting from a data mining query over the data mining resources, here, *DMDataset* contains the data mining model. This allows access to the content in the *DMExecuteFactoryResponse* providing indirect access to a data mining model resource created via the *DMAccessFactory* interface, as shown in Fig.4. The interface provides two operations: (i) *GetDMResponsePropertyDocument* can expose the properties enumerated in the WS-DAI specification to provide information about a particular instance of a data mining model that a data mining access service represents, the request message is the element *wsdai:GetDataResourcePropertyDocumentRequest* and the response message is the element *wsdai:PropertyDocument*; (ii) *GetDMResultModel* gets a *DMResultModel* from *GetDMResultModelResponse* message, the structures of the input request message and the output response message are as follows:

```
<GetDMResultModelRequest>
    <wsdai:DataResourceAbstractName/>
    <wsdai:DatasetFormatURI/>
</GetDMResultModelRequest>

<GetDMResultModelResponse>
    <wsdai:DatasetFormatURI/>
    <wsdai:DatasetData/>
</GetDMResultModelResponse>
```

E. The Execution Process

This subsection demonstrates the execution process of using data mining access service to mine meaningful and potential knowledge/patterns from data mining resources. For this purpose, we use *DMAccess* and *DMResponse* services shown in Fig.5.

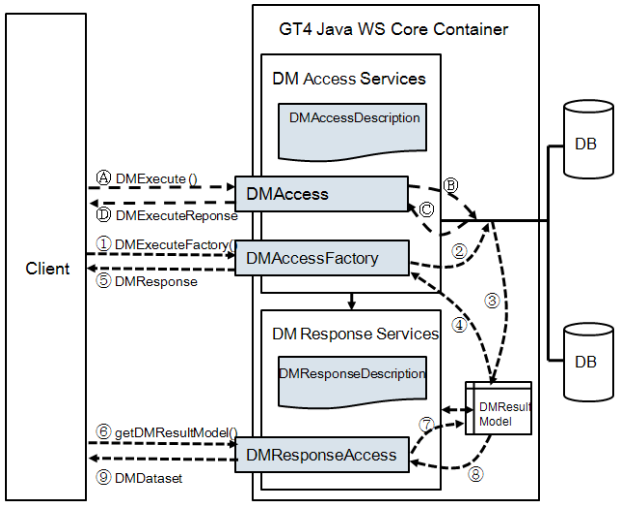


Figure 5. The execution process of DMAccess and DMResponse sevices

The execution processes of direct access are labeled with the capital letters. The first step is to submit a data mining request to *DMAccess* service using *DMAccess* interface (A). When receiving *DMExecuteRequest*, *DMAccess* service accesses the data mining resources according to the values of parameters (B) and then return the result to the interface (C). The result is returned to client side as message *DMExecuteReponse* (D).

For indirect access, as shown in Fig.5, the steps are labelled with numbers. Firstly, the client submits the data mining request to *DMAccess* service using *DMAccessFactory* interface (1). The service accesses data mining resources in the same way as it did in direct access (2), but in this case the data mining result model is created (3) and the endpoint reference of the model is returned to *DMAccessFactory* interface (4). The reference to *DMResponse* service is returned to the client as message *DMExecuteFactoryResponse* in order to access the result model (5). When the client needs to access the result model, the client sends message *GetDMResultModelRequest* to *DMResponse* service using *DMResponseAccess* interface (6). *DMResponse* service accesses the service-managed data mining model according to the values of parameters (7) and returns result to the interface (8). Finally, the dataset is send to the client as message *GetDMResultModelResponse* (9).

IV. IMPLEMENTATION

This section describes how we implement WS-DAI-DM with OGSA-DAI and we call this solution OGSA-DAI WS-DAI-DM. OGSA-DAI is a widely used Grid middleware that allows the sharing of data resources to enable collaboration and it supports data access, transformation, integration and delivery [20]. In OGSA-DAI, each activity is a well-defined functional unit and workflows are submitted to the execution engine via web services, which make it flexible and easy-use in both centralized and large-scale Grid environments [20]. We need not to start from the scratch by using OGSA-DAI in the implementation.

There are two possibilities about how to implement this specification with OGSA-DAI:

- Data mining process can be transformed as OGSA-DAI workflow in a class at client side and this class can be wrapped as web service according to WS-DAI-DM interface specification; clients submit data mining request to invoke the class and the workflow in the class is submitted to Data Request Execution Service (DRES) deployed on OGSA-DAI server.
- The 4-layer architecture is used in the implementation, which is similar with OGSA-DAI WS-DAIR [12] and OGSA-DAI WS-DAIX [13], where data mining process is transformed as OGSA-DAI workflow at server side and this workflow is executed by Data Request Execution Resource (DRER) which is OGSA-DAI's workflow execution component.

We use the second solution to implement OGSA-DAI WS-DAI-DM and the architecture is shown in Fig.6. WS-DAI-DM services delegate operations to portType providers. WS-DAI-DM portType providers get resources and delegates responsibility for execution of operations on the resources to executors. Executors implement WS-DAI-DM operations via operations on a resource directly, or via execution of OGSA-DAI workflows. OGSA-DAI components manage access to distributed data resources. The associated workflows use OGSA-DAI 3.2 activities plus a set of user defined activities developed for OGSA-DAI WS-DAI-DM.

According to the Section 3, OGSA-DAI WS-DAI-DM has two services and each of them is composed of portTypes, as shown the following:

- DMccess services:
  - CoreDataAccess, CoreResourceList
  - DMAccess, DMAccessFactory
- DMResponse services:
  - CoreDataAccess, CoreResourceList
  - DMResponseAccess

CoreDataAccess and *CoreResourceList* portTypes are inherited from the WS-DAI core specification [6]. Fig.7 shows the relationship between *DMAccess* service, portType providers and executors. *DMAccess* service delegates operations to *DMAccessProvider* and *DMAccessFactoryProvider*, and they get resources and delegate

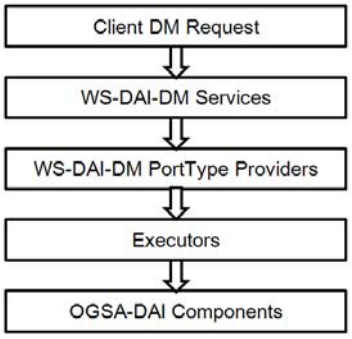


Figure. 6. The architecture of OGSA-DAI WS-DAI-DM

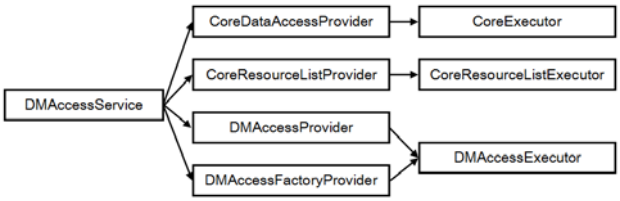


Figure. 7. DMAccess sevice , porType providers and executors

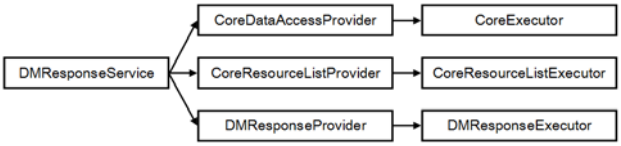


Figure. 8. DMResponse sevice , porType providers and executors

execution to *DMAccessExecutor*, in which *DMExecute*, *DMExecuteFactory* and *getSQLPropertyDocument* are implemented via the execution of OGSA-DAI workflows composed of user-defined activities. Fig.8 shows the relationship between *DMResponse* service, portType providers and executors, here, *DMResponse* service delegates operations to *DMResponseProvider*, and it delegates execution to *DMResponseExecutor*, in which *GetDMResponsePropertyDocument* and *GetDMResultModel* are implemented. We will describe OGSA-DAI WS-DAI-DM in detail in our other publications. Because the specification is independent of data mining applications/tools /algorithms, the execution time of access data mining services still come from the execution of data mining alrorithms wrapped inside, so the performance will not be the key issue in the research of specification.

This implementation method about WS-DAI-DM here described is just one of the solutions, which is used in our research and project. If other developers plan to use the specification, the concrete implementation and the data mining algorithms wrapped inside the data mining services should depend on the requirements of applications.

V. APPLICATION SCENARIO

In the section, we give an application scenario about mining the sequence of users' shopping transactions to show how users can use the data mining access service conveniently in a direct pattern at client side. It is a typical application about mining sequential patterns described in [21]. A large retail organization has many retail supermarkets distributed in different districts, and transaction data are stored in the relational databases distributed in different sites. Now the headquarter plans to develop an analysis software to analyze users' purchase habits based on all retail transaction data and the result should be submitted to the managers as reports. During the generation of the report, the data mining access service will be invoked according to the standard web service interfaces defined in WS-DAI-DM specification. At this moment, what the developers need to do is just to assign the values of parameters to invoke *DMAccess* service and the result sequences is returned and formatted as standard PMML model which can be parsed and embedded in the report easily.

The report is about analyzing qualified purchase sequences' extracted from the transaction data generated from 2009-08-01 to 2010-07-31, where 'qualified' means the large sequences with minimum support 20%. The table name is *tblPurchase*, in which the column *TRANS_ID* means the transaction id, the column *CUSTOMER_ID* means the customer id and the column *ITEM_ID* means the product id, the tables are stored in Oracle databases which have been registered as data resource group on central OGSA-DAI server and the server base URL is *http://localhost:9090/dai/services*. On OGSA-DAI server, data mining services have been deployed and related wsdl files and stub files can be generated with Java2wsdl and wsdl2Java tools. For *DMAccess* service, the service interface *DMIAccessService* and the locator *DMIAccessServiceLocator* which implement the interface are generated and deployed at client side. The following list shows the code embedded in report client application, and exceptions are skipped in order to keep the code simple.

```
1 DMAccessService service = new DMAccessServiceLocator();
2 DMAccess port = service.getDMAccess();
3 DMExecutionRequest deReq = new DMExecutionRequest();
4 DMExecutionResponse deResp = new MExecutionResponse();
5 HashMap pl = new HashMap();
6 pl.put("TableName", "tblPurchase");
7 pl.put("TidName", "TRANS_ID");
8 pl.put("CidName", "CUSTER_ID");
9 pl.put("ItemName", "ITEM_ID");
10 pl.put("setStartTime","2009-08-01")
11 pl.put("setEndTime","2010-07-31")
12 pl.put("SupValue", new double(0.2));
13 DMExpression ep = new DMExpression();
14 ep.setAlgorithmType("SequentialPattern");
15 ep.setParametersList(pl);
16 deReq.setServiceURL("http://localhost:9090/dai/services/");
17 deReq.setResourceName("TransData");
18 deReq.setDatasetFormatURI("http://www.dmg.org/PMML-
 4_0");
19 deReq.setDmExpression(ep);
20 deResp = port.DMExecute(deReq);
```

In the above code, the line 1 and line 2 are the typical usage of stub classes. Line 3 and line 4 create the instances of input and output messages. Line 5 to line 12 assign the values of parameters about data mining query to a hashmap and then *DMExpression* is set at line 14 and line 15. Afterwards, from line 16 to line 19, the input message *deReq* gets *DataResourceAbstractName*, *DatasetFormatURI* and *DMExpression*, and finally *deReq* is send to *DMExecute* operation and the result is formatted as standard PMML model and returned to the client in *deResp* message in line 20.

The input message *deReq* is send to *DMAccess* service and we can monitor the message with SOAPMonitor tool. The following is the structure of message *deReq*.

```
<DMExecuteRequest>
  <ServiceURL>
    http://localhost:9090/dai/services/
  </ServiceURL>
  <ResourceName>TransData</ResourceName>
  <DatasetFormatURI>
    http://www.dmg.org/PMML-4_0
  </DatasetFormatURI>
  <DMExpression>
    <AlgorithmType>SequentialPattern</AlgorithmType>
    <DMParameter>
      <name>TableName</name>
      <value>tblPurchase</value>
      <type>STRING</type>
    </DMParameter>
    <DMParameter>...</DMParameter>
  <DMExpression>
<DMExecuteRequest>
```

The output message deResp is returned to client from DMAccess service and the following is the structure of message deResp.

```
<DMExecuteResponse>
  <DMDataset>
  <DatasetFormatURI>
    http://www.dmg.org/PMML-4_0
  </datasetFormatURI>
  <DatasetData>
    <PMML version="4.0" xmlns="http://www.dmg.org/PMML-
    4_0">
        ...../data mining result model is represented as PMML
    </PMML>
  </DatasetData>
  <DMDataset>
<DMExecuteResponse>
```

In the report module, the output message deResp that contains the data mining result model formatted by PMML is returned and then is parsed. The parsed values are inserted in the report.

VI. CONCLUSION AND FUTURE WORK

In order to enhance the current Grid architecture with data mining access mechanism, it is very important to make data mining processes available though consistent service-based interfaces in Grid environments. In this paper we describe data mining access mechanism built on the top of WS-DAI specification, namely WS-DAI-DM, which provides a powerful interface specification to enable data mining applications through web services. WS-DAI-DM addresses the need of the web service interfaces to facilitate the development of data mining-enabled applications.

Our contribution in this work is twofold:

- To provide service-based interfaces and access patterns for data mining processes which can improve the reusability of existing data mining algorithms/tools and hide the detailed data mining workflows.
- Data mining access can seamlessly integrate with other capabilities of service-oriented Grid middlewares.

WS-DAI-DM specification is the first step of our research and it is the basis for embedding data mining in more applications. The work presented in this paper and the related implementations are still under development. In the future work, first, we will improve the specification, and next step, we plan to embed more data mining algorithms/tools in OGSA-DAI WS-DAI-DM. Additionally, we will implement the proposed WS-DAI-DM on China railway freight transport information grid [22][23] to help users and other subsystems extract useful and hidden information/patterns from distributed waybill data. At the same time, we are looking forward to further elaborate on the specification in the standardization process. We think, as a specification, WS-DAI-DM can be used by more projects and research groups.

ACKNOWLEDGMENT

The research has been supported by Project No. 2006AA01A121 of the National High-Technology Research and Development Program of China. Alexander Woehrer has been supported by the ADMIRE project which is financed by the European Commission via Framework Program 7 through contract No. FP7-ICT-215024.

REFERENCES

[1] J. Han and M. Kamber, *Data Mining: Concepts and Techniques, 2nd ed*. San Francisco, USA: Morgan Kaufmann Publishers, 2005.

[2] M. P. Atkinson, J. I. van Hemert, L. Han, A. Hume, and C. S. Liew, "A distributed architecture for data mining and integration," in *DADC '09: Proceedings of the second international workshop on Data-aware distributed computing*, New York, NY, USA, 2009, pp. 11–20, doi: 10.1145/1552280.1552282.

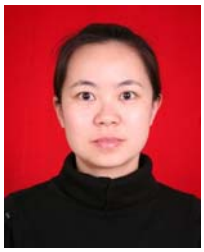
[3] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "Grid services for distributed system integration," *Computer*, vol. 35, no. 6, pp. 37-46, June 2002, doi:10.1109/MC.2002.1009167.

[4] C. Kesselman and I. Foster, *The Grid: Blueprint for a New Computing Infrastructure, 2nd ed*. San Francisco, USA: Morgan Kaufmann Publishers, 2004.

[5] M. Antonioletti, A. Krause, N. W. Paton, A. Eisenberg, S. Laws, S. Malaika, J. Melton, and D. Pearson, "The WS-DAI family of specifications for web service data access and integration," *SIGMOD Rec*., vol. 35, no. 1, pp. 48–55, 2006, doi:10.1145/1121995.1122006.

[6] M. Antonioletti, M. Atkinson, A. Krause, S. Laws, S. Malaika, N. W. Paton, D. Pearson, and G. Riccardi, "Web

Services Data Access and Integration - the core (WS-DAI) specification, version 1.0," Open Grid Forum, DAIS Working Group, Tech. Rep., July 2006.
[7] M. Antonioletti, B. Collins, A. Krause, S. Laws, J. Magowan, S. Malaika, and N. W. Paton, "Web Services Data Access and Integration - the Relational realization (WS-DAIR) specification, version 1.0," Open Grid Forum, DAIS Working Group, Tech. Rep., July 2006.
[8] M. Antonioletti, S. Hastings, A. Krause, S. Langella, S. Lynden, S. Laws, S. Malaika, and N. W. Paton, "Web Services Data Access and Integration - the XML realization (WS-DAIX) specification, version 1.0," Open Grid Forum, DAIS Working Group, Tech. Rep., August 2006.
[9] M. Antonioletti, C. B. Aranda, O. Corcho, M. Esteban-Gutierrez, A. Gomez-Perez, I. Kojima, S. Lynden, and S. M. Pahlevi, "WS-DAI RDF(S) realization: Introduction, motivational use cases and terminologies," Open Grid Forum, DAIS Working Group, Tech. Rep., October 2009.
[10] M. E. Gutierrez and A. Gomez-Perez, "Ontology access provisioning in grid environments," in *Semantic Grid: The Convergence of Technologies*. Dagstuhl, Germany: Internationales Begegnungs- und Forschungszentrum f¨ur Informatik (IBFI), Schloss Dagstuhl, Germany, 2005.
[11] M. E. Gutierrez, A. Gomez-Perez, O. Corcho, and O. M. Garcia, "WS-DAIOnt-RDF(S): Ontology access provision in grids," in *GRID '07: Proceedings of the 8th IEEE/ACM International Conference on Grid Computing*, Washington, DC, USA, 2007, pp. 89–96. doi:10.1109/GRID.2007.4354120.
[12] E. Theocharopoulos and M. Jackson, "OGSA-DAI WS-DAIR version 1.0," The OGSA-DAI Project Team, December 2008.
[13] M. Jackson and E. Theocharopoulos, "OGSA-DAI WS-DAIX version 1.0," The OGSA-DAI Project Team, December 2008.
[14] M. F. Hornick, H. Yoon, and S. Venkayala, "Java Data Mining (JSR-73): Status and overview."
[15] M. F. Hornick, "Java Specification Request 73: Java Data Mining (JDM)," JSR-73 Expert Group, July 2004.
[16] M. F. Hornick, "Java Specification Request 247: Java Data Mining (JDM) 2.0," JSR-247 Expert Group, September 2006.
[17] D. Talia, P. Trunfio, and O. Verta, "The Weka4WS framework for distributed data mining in service-oriented grids," *Concurr. Comput. : Pract. Exper*, vol. 20, no. 16, pp. 1933–1951, 2008.
[18] A. Guazzelli, M. Zeller, W.-C. Lin, and G. Williams, "PMML: An open standard for sharing models," *The R Journal*, vol. 1, no. 1, pp. 60–65, 2009.
[19] "Common Information Model (CIM) standards, version 2.23.0," Distributed Management Task Force, Inc, October 2009.
[20] "Open Grid Services Architecture - Database Access and Integration (OGSA-DAI)," http://www.ogsadai.org.uk/. (Access at Sep. 2010)
[21] R. Agrawal and R. Srikant, "Mining sequential patterns," in *ICDE '95: Proceedings of the Eleventh International Conference on Data Engineering*. Washington, DC, USA: IEEE Computer Society, 1995, pp. 3–14.
[22] Y. Zhang, A. Wöhrer, and P. Brezany, "Towards china's railway freight transportation information grid," in *MIPRO'09: Proceedings of the 32nd international Convention on Information and Communication Technology, Electronics and Microelectroincs*, Opatija, Croatia, 2009.
[23] Z. Xing, H. Li, G. Dai, Y. Zhang, "Research on the grid-based railway cargo information system". *Journal of Huazhong University of Science and Technology (Natural Science Edition)*, 2010. 38(s1), pp.115-119.



Yan ZHANG is a Ph.D. student in School of Computer and Information Technology at Beijing Jiaotong University, P.R.China. Yan ZHANG, born on Nov. 1979 in Xinjiang autonomous region, P.R.China, received her Bachelor degree in computer science from Institute of Computer Science, Lanzhou Jiaotong University, P.R.China on July 2000. Since Sep. 2005, she studied for her Master degree in School of Computer and Information Technology at Beijing Jiaotong University and the major is Grid computing. She studied for doctorate degree in advance in Sep. 2007 at School of Computer and Information Technology, Beijing Jiaotong University, and her major is Grid computing. From Oct. 2008 to Feb. 2010, she studied at University of Vienna, Austria as a joint-training Ph.D. student and her research focused on combination of Grid computing and data mining.

Yan ZHANG worked at Computer Center of Urumqi Railway Administration Department as a network maintenance engineer from Aug. 2000 to July. 2005 and her main duty were network maintenance, server maintenance, website design, user training and railway permanent way GIS (PWGIS) development. Since 2006, she participated in the project - Grid-based railway freight transportation management information system which is the National High-Technology Research and Development Program of China and the project number is 2006AA01A121, and her main duty is demand analysis, detailed design, outline design and coding. Her current research interests include Grid and cloud computing and distributed data mining.

Honghui LI is senior engineer, the deputy director of Network Management Research Center in School of Computer and Information Technology at Beijing Jiaotong University, P.R.China. She received Master degree from Zhongnan University of Technology, P.R.China in 1987. She has been finished many China national technology research projects and China National High-Technology Research and Development Programs.

Alexander Woehrer received his Master and PhD degree in business informatics from the University of Vienna, Austria in 2004 and 2010, respectively.

He is a PostDoc researcher at the Institute of Scientific Computing at the University of Vienna since 2004. His research interests include database access and integration in service-oriented environments, grid and cloud computing, data mining and integration. Dr. Wöhrer is a member of ACM, Open Grid Forum, IAENG and the Austrian Computer Society.