Provenance Support for Grid-Enabled Scientific Workflows

Fakhri Alam Khan, Yuzhang Han, Sabri Pllana, and Peter Brezany

Institute of Scientific Computing, University of Vienna
Nordbergstrasse 15/C/3, A-1090, Vienna Austria
{khan|han|pllana|brezany}@par.univie.ac.at

***Abstract*— The Grid is evolving and new concepts like Semantic Grid, Knwoledge Grid are rapidly emerging, where humans and distributed machines share, exchange, and manage data and resources intelligently. Computational scientists typically use workflows to describe and manage scientific discovery processes. However, the credibility of the obtained results in the scientific community is questionable if the computational experiment is not *reproducible*. This issue is being addressed in our research reported in this paper via development of workflow provenance system for Grid-enabled scientific workflows. Workflow provenance collects data on workflow activities, data flow and workflow clients. Provenance information can be used to trace and test workflows and the data produced. Our approach supports reproducibility (i.e. to support re-enactment of workflow by an independent user) and dataflow visualization (i.e. visualization of statistical characteristics of input/output data). We illustrate our approach on the Non-Invasive Glucose Measurement (NIGM) application.**

I. Introduction

The Grid computational infrastructure with its heterogeneous and geographically distributed resources has made e-Science a reality. E-Science [1] enables the scientists to perform complex computational experiments and share their results through workflows. Workflows specify the order of execution of Grid tasks (i.e. activities). A Grid task may take a data input, process data, and produce a data output. As the workflows on the real-world applications are getting more data intensive and the concept of *Knowledge Grid* i.e. a smart networked environment where researchers and machines can effectively share and manage knowledge resources [2], is getting mature, the scientists' concerns on information about the kind of activities that are executed and the corresponding data input(s)/output(s) are growing. Scientists use workflows to describe and manage scientific discovery processes, where combinations of various kinds of activities are tried until the desired result is obtained. However, the obtained results are not of particular help if we do not know exactly how they are produced. To ensure the *reproducibility* i.e. to enable scientists to execute, test and verify the workflow independently, it is needed to collect sufficient *provenance* information on workflow execution steps at abstract and operational levels. Basically, we may define the workflow provenance as "the process of collection of information on workflow execution". In this paper we address the issue of reproducibility of scientific workflows via a suitable workflow provenance system.

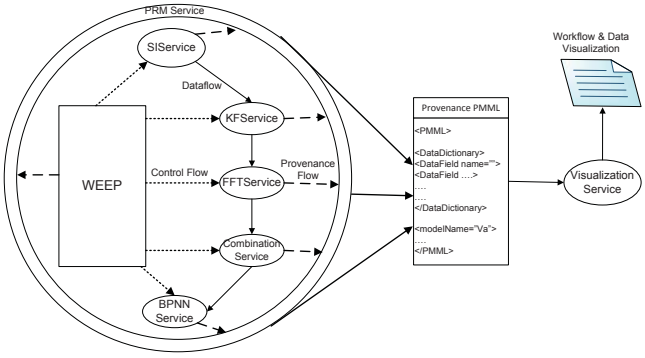

Fig. 1. Workflow provenance model for NIGM

Our workflow provenance system, (1) increases the user understanding on workflow activities and the data flow between activities through visualization, and (2) statistically analyzes the input/output data. Fig. 1 depicts our workflow provenance system for the Non-Invasive Glucose Measurement (NIGM) application [3]. NIGM efficiently computes the glucose value in patient blood by a method developed with the help of a Traditional Chinese Medicine (TCM) theory [4]. The individual components (SIService, KFService, FFTService, CombinationService, and BPNNService) are introduced in Section III. The NIGM workflow is enacted and executed by WEEP [5], the workflow enactment engine, designed and developed at the University of Vienna. The workflow provenance system has three main components,

1) Process recording and monitoring service (*PRM Service*). Its task is to continuously monitor the workflow for any invocation activity, control flow, or dataflow.
2) *Provenance PMML*. Provenance data on activities and data exchanged are stored in the PMML document and are combined with pre-defined statistical PMML models. Predictive Model Markup Language (PMML) [6] is an XML-based language for statistical and data mining models.
3) *Visualization Service*. It takes the PMML document as input and produces the visualization of workflow and statistical characteristics of the data used by workflow.

The major contributions of this paper include:

- Identification of the provenance information that should

978-0-7695-3401-5/08 $25.00 © 2008 IEEE
DOI 10.1109/SKG.2008.86

be collected to ensure the workflow reproducibility.

- Extension of the standard PMML schema (xsd) to support the complete description of the workflow provenance information.
- Proposal and implementation of new concepts for continuous workflow monitoring and visualization.

This paper is organized as follows. Section II provides the necessary background information and discusses the related work. Section III presents the NIGM application scenario in the context of the China Austria Data Grid (CADGrid) project [7]. The provenance recording technique and the PMML document structure used are described in Section IV. The provenance visualization application for scientific workflows is presented in Section V. Section VI introduces and gives insight into the reproducibility. In Section VII we give a list of our future work tasks. Section VIII concludes the paper.

II. Related Work

The need to collect provenance data and determine the ownership and/or pedigree of a particular object was first realized in the field of Arts. One such instance is International Foundation for Art Research [8], which was established in 1969 and since 1970 it offers Art authentication research services for works of Art. In last few decades, the emergence of databases and particularly curated databases and data warehouses, has enabled the researchers and scientists to have access to huge amount of geographically distributed shared data. This access and sharing led scientists to issues like:

1) Data Management: as the size of shared data increased dramatically with ever increasing growth rate.
2) Integration: because of the diverse nature of data sources.
3) Source Authentication: Sources are needed to be authenticated because they are modified frequently as well as new resources are added.
4) Data Quality: is a concern because of the curated databases, and modification in the source databases.

These issues triggered the research on provenance for databases. Cui et al. focused on tracing data lineage through transformations in a data warehousing environment [9]. Buneman et al. made contribution to the data provenance through 'Why Provenance' [10] i.e. from an output visualized view they determined all the tuples that contributed to the output, whereas Cui et al. focused on 'Where Provenance' [11] i.e. instead of determining which tuples in the source contributed to the output they focused and traced where an output tuple came from. In recent years there has been much work on provenance, but to the best of our knowledge there is no work focusing on provenance for data mining activities over scientific workflows and using PMML as a medium for storing and designing provenance application. Karma provenance framework [12] provides provenance collection of workflows for process and data items. Karma mainly focuses on low overhead provenance collection [13] with quality determination of data products and has not addressed model validation

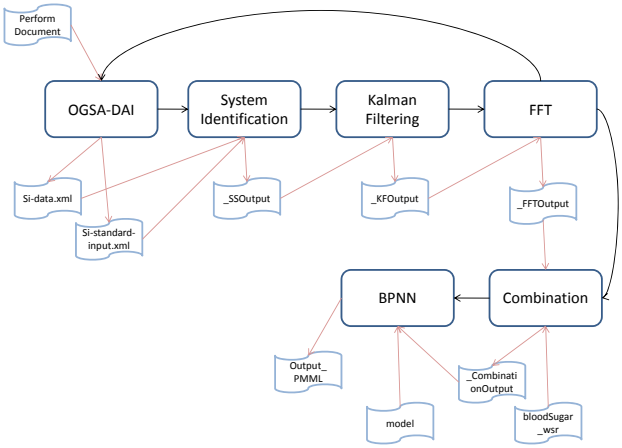

Fig. 2. NIGM workflow

of workflows, and reproducibility. Provenance Aware Service Oriented Architecture (PASOA) [14], is focusing to standardize the provenance collection framework over Service Oriented Architecture (SOA) [15]. The problem of inter-operating and integrating distributed resources in the myGrid project [16] has been addressed through the development of Taverna [17]. Taverna enables the bio-scientists to perform distributed complex computation and provides easy management of workflows and simple provenance collection.

III. Application Scenario

The application scenario is based on the CADGrid NIGM services workflow. The non-invasive method for measuring blood glucose values in the NIGM workflow is based on the meridian-theory. According to the meridian-theory, human body has 14 acupuncture meridians [18], which are longitudinally distributed lines. Special electro signal meridian measurement instruments are used, which sends an electric signal (white noise) into one of the meridian source point and measures the corresponding signal output. These measurements are stored and are used as input to the NIGM workflow. The NIGM workflow consist of the following algorithms, deployed as WS-I and WSRF-compliant CADGrid services: (1) System Identification, (2) Kalman Filtering, (3) Fast Fourier Transformation, (4) Combination Service, and finally (5) Back Propagation Neural Network. The first three Services are invoked iteratively within a loop, until all eigenvalues for a given set of input value pairs (meridian measurements) are computed. NIGM services are coordinated and invoked by WEEP. Fig 2 shows the NIGM workflow and all associated input(s)/output(s). A brief description of NIGM workflow components is given below.

- OGSA-DAI [19]: At the beginning of each iteration the OGSA-DAI *DataService* is invoked to retrieve input data for a particular experiment and delivers the data in the

WebRowSet XML format to the services performing the requested calculations.

- System Identification Service: The *SIService* implements a System Identification [20] algorithm. It receives the files delivered by the *DataService* and generates the input files used by the *KFService*.
- Kalman Filtering Service: The *KFService* implements the Kalman Filter [21] algorithm to extract a signal from a series of incomplete and noisy data stored in the file generated by the *SIService*, which serves as input for the *FFTService*.
- Fast Fourier Transformation Service: The *FFTService* implements the Fast Fourier Transformation (FFT) [22] algorithm. The service calculates the eigenvalues from the transformed waveform using FFT.
- Combination Service: The *CombinationService* is executed after all the iterations. The service combines all the output data files produced by the *FFTService* together with the file containing the measured blood sugar data. The newly created file is used as an input for the *BPNNService*. The file provides the input values for the training process of a neural network and supervisor values.
- BPNN Service: *BPNNService* has two functions:
  1) It builds an individual health model for each considered person. This model is patient specific and has the form of neural network. It is stored in a PMML file.
  2) The patient specific model is used to predict blood glucose value for the considered particular patient.

Although the NIGM workflow methodology for glucose value prediction is fairly accurate but there are certain frequent questions that may be posed by a doctor and a researcher; these questions include:

- Was the input to the model correct?
- How much accurate the output or model is?
- What services were involved in glucose measurement workflow?
- What transformations were performed to produce the output i.e. data passed through what services and what actually happened to the data?
- Can I fine tune the workflow and if yes, then is there any information available on models parameter significance?

To answer these queries, we need to have an efficient and effective workflow provenance system, that collects provenance data about services communication, data flow, intermediate results, data transformation, execution time of the services, perform model validation over provenance data, and provide visualization.

IV. Provenance System Architecture and Implementation

The main aim of this research is to enrich data mining and scientific workflows with provenance data. The main benefit is enhancement of the scientists' trust on workflows execution. Our main objective can be divided into sub-objectives (intermediate goals) characterized as follows:

- Support for data flow provenance collection i.e. recording provenance information about input and output of services.
- Support for service flow provenance collection i.e. recording provenance data about the number of services in a workflow and execution order of services.
- Providing visualization for service flow i.e. control flow visualization through directed graph (DG), where DG is a set of nodes representing activities connected via lines, which show the execution order of activities.
- Providing data flow visualization for input(s) and output(s) of workflow and visualization of data generated by individual activities within a workflow.
- Providing statistical description of the workflow data and their visualization, like mean, clustering of data, output quality, and divergence of data.
- Designing a reproducibility model. It aims at collecting enough provenance information to re-run, test, and verify workflows.

Fig. 1 in Section I visualizes the provenance system architecture we proposed to achieve the above mentioned objectives. The provenance system is comprised of three main components realized by two WSRF [23] services; 1) Process recording and monitoring service *(PRM Service)* and 2) *Visualization service*, and one 3) *Provenance PMML document*. The next sub-sections describe these components functionality, implementation and working in detail.

A. Process Recording and Monitoring Service (PRM Service)

The WEEP (Workflow Enactment Engine Project) aims to orchestrate WS-I [24] and WSRF services as specified by Web Services Business Process Execution Language (WS-BPEL 2.0) compliant document. WEEP provides a framework to process and validate WS-BPEL (version 2.0) documents, provide client toolkit for secure and reliable invocation of the engine, supports fault tolerant orchestration of distributed services, performs WS-I/WSRF service invocation dynamically by evaluating its WSDL definitions, control (start, restart, stop, resume, and suspend) and schedules the execution of WS-BPEL processes [25].

The WEEP engine takes WS-BPEL document as input, parses the BPEL and then invokes dynamically the services and hence orchestrates the workflow, as depicted by Fig. 1. For provenance collection, we have developed a new WSRF service which continuously monitors the WEEP as well as the workflow. Whenever there is an activity such as WEEP receives BPEL documents as input from the client, WEEP initiates a SOAP message to invoke a web service , and SOAP response message from the invoked service to the WEEP; the *PRM service* collects the SOAP message, parses it, transforms it into PMML format, and stores the information in a PMML document. The *PRM service* supports the parsing of the input data in the Webrowset, XML, and tables format. Fig. 1

depicts that the whole workflow orchestrated by the WEEP is covered and monitored by the *PRM service*. The existence of *PRM service* for the whole life span of the WEEP enables the complete provenance collection. In our architecture the provenance collection service is a standalone service and hence provenance collection can be switched off by undeploying the service without affecting the overall workflow orchestration.

B. Provenance Visualization Service

The *PRM service* runs continuously and for the whole life span of the WEEP engine, instructions and data are continuously sent and stored in PMML documents. These PMML documents serve as core input to the *visualization service*. The *visualization service* major task is to parse the PMML and then visualize the provenance data. The core logic of the *visualization service* is implemented as an Eclipse RCP application basically by using the GMF and GEF environment. The GMF and GEF are Eclipse based IDEs (Integrated Development Environments) for developing and customizing graphical and visual applications, especially diagram editors. The *visualization service* core includes:

- A BPEL document visualizer, which visualizes the invocation sequence of the orchestrated services.
- A PMML file parser, which navigates the structure and content of the PMML document files delivered by the PRM service. The parser result of single PMML files is stored for later use.
- A data file modifier, which can generate update information on certain data files, this information is to be sent back to the PRM service for data file modification.
- A dataflow chart composer, which generates the information on the topology of a dataflow chart using the parse result of single PMML files given by the PMML file parser.
- A graphical user interface. The information provided by the components listed above are visualized by graphical user interface.

C. PMML: A provenance information store

The use of PMML for provenance is a new approach. We decided to use PMML for reasons listed below:

- PMML is a language for defining statistical and data mining models. We compute and visualize the statistical characteristics of workflow and data, like mean, and divergence in data.
- PMML is a standard and commonly used in almost all data mining systems and is vendor independent. It will assist our provenance system in the future to be standardized and used in domain independent ways.
- In PMML we can have both provenance data (data dictionary) and data mining models in one document, which makes our task much more easier.
- The *data dictionary* component of a PMML document is suitable as a carrier of meta-data of the data used in the workflow.

```
<DataDictionary numberOfFields="56">

  <!-- pointers to data-level pmml documents-->
  <DataField name="SI_stdInput" optype="continuous" dataType="double">
   <Extension name="fileType" value="webRowSet"/>
   <Extension name="serviceName" value="SIService"/>
   <Extension name="IO" value="input"/>
   <Extension name="filename" value="SI_stdInput.pmml"/>
  </DataField>

  <DataField name="SI_input1" optype="continuous" dataType="double">
   <Extension name="fileType" value="webRowSet"/>
   <Extension name="serviceName" value="SIService"/>
   <Extension name="IO" value="input"/>
   <Extension name="filename" value="SI_input1.pmml"/>
  </DataField>
 ----------------
 ----------------
   <DataField name="BPNN_output" optype="continuous" dataType="double">
   <Extension name="fileType" value="webRowSet"/>
   <Extension name="serviceName" value="BPNNService"/>
   <Extension name="IO" value="output"/>
   <Extension name="filename" value="BPNN_output.pmml"/>
  </DataField>

  <!-- this is the BPEL file describing the invokation sequence-->
  <Extension name="controlFlow" value="NIGM.bpel"/>

  <!-- this is the pointer to the service-level PMML document -->
  <Extension name="SIService" value="SIService.pmml"/>
  <Extension name="KFService" value="KFService.pmml"/>
  <Extension name="FFTService" value="FFTService.pmml"/>
  <Extension name="CombinationService" value="CombinationService.pmml"/>
  <Extension name="BPNNService" value="BPNNService.pmml"/>

</DataDictionary>
```

Fig. 3. Workflow-level PMML document: NIGM.pmml

- PMML has it own set of models already well defined like transformation model, regression model, cluster model etc. We use these models to achieve our goal i.e. statistical description of the workflow.

In the provenance store, we store a hierarchy of PMML documents, the combination of which together provides a complete and systematic view of the workflow provenance. In this hierarchy, PMML documents are organized in three levels. Documents at a higher level are associated with those at the lower level through pointers. The levels are described below from the highest to the lowest:

- Level 1: workflow-level document. This single document contains the information about workflow input and output data. It also clarifies the service invocation sequence. For every point of information, it provides a pointer which points to other low-level documents which contain detailed information. This provides an overall insight into the workflow. Fig. 3 give a snapshot of the workflow level PMML document.
- Level 2: service-level documents. These are a group of PMML documents each of which contains knowledge about a single service. Information such as description of service parameters, input(s)/output(s) data and invocation interface are included. When it comes to input(s)/output(s) data, pointers to low-level documents are provided. This kind of documents establishes perception to the services. Fig. 4 give a snapshot of the service level PMML document.
- Level 3: data-level documents. Every one of this type of documents is associated to a single data file used by a service. They are the basic building stones in the structure of data flow. They provide detailed information for all the documents at higher levels. Fig. 5 give a snapshot of the

```
<DataDictionary numberOfFields="52">

 <!-- algorithm parameters -->
 <DataField name="parameter" optype="continuous" dataType="double">
  <Extension name="parName" value="coef0"/>
  <Extension name="actual" value="0.53"/>
  <Extension name="initial" value="0.3"/>
  <Extension name="significance" value="1.2"/>
 </DataField>
 ------------------
 ------------------
 <!-- invocation interface of the SIService -->
 <DataField name="invocation" optype="" dataType="">
  <Extension name="funcName" value="runSI"/>
  <Extension name="parName" value="numX"/>
  <Extension name="parName" value="numY"/>
  <Extension name="parName" value="inputFileName"/>
  <Extension name="parName" value="outputFileName"/>
  <Extension name="numX" value="integer:0,0xFFFFFFFF"/>
  <Extension name="numY" value="integer:0,0xFFFFFFFF"/>
  <Extension name="inputFileName" value="string:PATH"/>
  <Extension name="outputFileName" value="string:PATH"/>
 </DataField>

</DataDictionary>
```

Fig. 4. Service-level PMML document: SIService.pmml

```
<DataDictionary numberOfFields="3">

      <DataField name="SI_input" optype="continuous" dataType="double">
        <Extension name="no_rows" value="50050"/>
        <Extension name="mean" value="37.253"/>
        <Extension name="variance" value="0.873"/>
        <Extension name="stdDev" value="0.934"/>
        <Extension name="unit" value="mV"/>
      </DataField>

      <DataField name="SI_output" optype="continuous" dataType="double">
        <Extension name="no_rows" value="50050"/>
        <Extension name="mean" value="12.223"/>
        <Extension name="variance" value="0.493"/>
        <Extension name="stdDev" value="0.702"/>
        <Extension name="unit" value="mV"/>
      </DataField>

      <DataField name="physicalInfo" optype="" dataType="">
        <Extension name="fileName" value="input_SI1.xml"/>
        <Extension name="DB_name" value="130.136.112.044"/>
        <Extension name="DB_location" value="/home/khan/..."/>
        <Extension name="DB_tableName" value="SI_inputOutput"/>
      </DataField>

</DataDictionary >
```

Fig. 5. Data-level PMML file

data level PMML document.

Fig. 1 shows the interaction between the WEEP engine, NIGM services and storing of data in PMML document. Next two sub-sections, IV-C.1 and IV-C.2, explain PMML *data dictionary* and *models* in provenance perspective.

1) Provenance aware PMML data dictionary: In this part we give details and illustrate the data dictionary section of a provenance aware PMML. Data dictionary is generated after the execution of the workflow. Fig. 5 gives a snapshot of a data-level *provenance PMML data dictionary*, while Fig. 4 displays the *DataDictionary* of a service level PMML document. The data dictionary contains two types of information on workflow execution, activities and mathematical models used:

1) Data describing the physical characteristics. This provenance information is directly collected and written into the *provenance PMML document* by workflow activities, i.e. whenever an activity receives input(s) data or generate output(s), it puts into the document data type(s) of input(s)/output(s), number of rows in input(s)/output(s), number of input(s)/output(s) attributes, names of attributes, input source type, and information on missing values.
2) Data describing the statistical characteristics of the workflow. The statistical provenance data is not directly collected by activities of the workflow, they are computed from the physical provenance data and the data itself. Statistical values calculated and stored in the PMML data dictionary are; mean of numerical input(s), variance of data, and standard deviation.

```
<!--Regression Model, already exists in PMML specification /Schema-->


  <RegressionModel ModelName ="Sample for linear regression"
 functionName="regression" algorithmName="linearRegression"
 targetFieldName="outputData">
        <MiningSchema>
              <MiningField name="SI _ input"/>
               <MiningField name="SI _ output" usageType="predicted"/>
        </MiningSchema>
        <RegressionTable intercept="132.37">
        <NumericPredictor name="input _ NIGM" exponent="1" coefficient="7.1"/>
        </RegressionTable>
  </RegressionModel>

<!--MCC - Model Checking Criteria -->

 <MCCModel modelName="C:/DOCUME~1/Temp/CPVEDM9V.C3T"
functionName="residual" algorithmName="changeMeasurement"
modelType="linear" targetFieldName="MCC_ output" >
        <MiningSchema>
              <MiningField name="SI _ input" usageType="active" domain="time"/>
              <MiningField name="input _ length" usageType="active"/>
              <MiningField name="SI _ output" usageType="active"  domain="frequency"/>
              <MiningField name=output _ length" usageType="active"/>
              <MiningField name="MCC _ output" usageType="difference"/>
        < /MiningSchema >

< /MCCModel >
```

Fig. 6. Model section of provenance PMML

In the NIGM workflow, on every service execution, new data is inserted into the data dictionary section of the the PMML. The provenance information is collected and stored in data dictionary as described below.

Every service generates input provenance information and output provenance data. A service, when invoked by WEEP, generates input provenance, which includes the provenance information on input data (such as type, source, no. of rows, etc.), the invoker of the service and the time of invocation. When the service completes processing and produces the result then it generates output provenance, which includes information on the output generated by this service. The WEEP engine is itself a service and it adds the starting and ending time of its execution. This information is used by the PMML models to calculate to overall execution time of the workflow.

2) Design of PMML provenance models: PMML uses XML based models to represent data mining models. These models, either make use of the data in the data dictionary or the raw data in data files. There are certain built in data mining models like association rule, neural network, regression, naive Byes etc, available as well as PMML allows the extension and definition of new models. We propose several models to accomplish our goals. Fig. 6 gives a snapshot of the model section of a *provenance PMML document*, which belongs to the data level PMML document of the input data file for *SIService*. We propose following models: model for workflow

model validation/checking, model for finding the mean of numerical data, and model for finding divergence in input and output data.

V. Visualization and monitoring using provenance data

Workflow can be seen as a combination of control flow and dataflow. *Control flow* refers to the order in which individual activities are invoked and executed, whereas *dataflow* in this context, is tracing of the input(s) and output(s) consumed or produced by workflow activities. *Dataflow visualization* is critical to the *users* of the constructed workflows, whose concern is to monitor the status of the data, during the workflow execution. We propose the dataflow visualization and monitoring as a possibility of application for *workflow provenance system*. The status of data i.e. size and content of input(s)/output(s) data files, might change during the execution of the workflow. We propose the visualization application to be a service, which reads the PMML provenance documents generated at provenance recording phase, interpret the status and changes to the data and visualize it. In order to enable our *workflow provenance system* to assist in realizing data flow monitoring and visualization, the provenance record must contain sufficient and meaningful information about data. For this purpose, we add the following content in the PMML record files, each PMML file actually stores the metadata about one data file in the workflow: (1) Number of rows, (2) Number of attributes, (3) Maximal and minimal value of each attribute, (4) Mean value of each attribute (these are added into the *Data Dictionary* section). For visualization we propose eager approach i.e. as soon as the workflow is started and provenance records are generated, the visualization process starts and dynamically reflects the data status information. The visualization application is itself a service, that takes these PMML files as input and output graphical and textural representation. A snapshot of the prototype *workflow visualization application* is shown in Fig 7, highlighting the main features and sections. The visualization application consists of 5 work zones, described below:

- Zone 1 - *Control flow zone*: This zone depicts the control flow of a workflow as a directed graph. Each node represents one workflow activity and the edges represent the invocation sequence. A click event on any node triggers update in the Zone 3 and 4. Zone 3 displays the tree-form structure of the service-level PMML document, while Zone 4 displays the source code of this PMML document. As shown in the figure, the box right beneath the START represents an INVOKE activity associated with the OGSA-DAI *DataService*.
- Zone 2 - *Dataflow zone*: This zone is devoted to the dataflow visualization. Every rectangle represents a data file and the connecting lines represent the transformation of files. These data files are input(s)/output(s) to the services, however, these services are hidden in this zone. For example, the top level rectangle is the input file *SI_stdInputs* to *SIService*, which generates files *SI_I00, ..., SI_I03*, each of which is transformed again into *SI_stdOutput0*, *SI_stdOutput1*. Zone 2, acts as a master zone for zones 3, 4 and 5 i.e. clicking on any rectangle in Zone 2 causes a refresh event of the graphs in the zones, and the updated graphs are displayed.
- Zone 3 - *Provenance Navigator*: This zone contains a tree-formed navigator of the PMML document corresponding to the selected node in Zone 1 or 2. As in the Fig. 7, *provenance navigator* displays the structure of PMML document against the select node i.e. *SI_stdInput*. The navigator is analogous to a file navigation tree. Every leaf in this navigation tree stands for a tag in the PMML document. In the tree, the root represents the root XML-tag in the document (because PMML has XML based syntax). *DataDictionary* and *ClusteringModel* represents two child tags in this documents, and so on. User can perform three different operations on this zone. The allowed operations are displayed via buttons aligned to the left of Zone 3 as *structure*, *file*, and *properties*. The *structure* as shown in Fig 7, demonstrates a tree-mode structure of the PMML document, while *file* is a windows-fashioned file manager, where the user can browse the local file system and can choose to display the PMML file independent from the workflow visualization application. The *properties* option gives the user a comprehensive perspective to miscellaneous information on the PMML file, such as information about size, path, author etc.
- Zone 4 - *PMML viewer and modifier*: This zone displays the actual provenance PMML file source, of selected node in *Dataflow Zone* or *Control flow Zone*. This Zone provides the ability to view and directly modify the source PMML document of selected data file, e.g. information such as names and values of the tags. The access to source code facilitates some advanced usage by the user, such as commenting and partial duplicating. The button aligned left to the Zone 4 are function buttons, for editing in this zone. *Insert/Delete*, are to insert or delete comments from the source PMML document. *Apply*, confirms the modifications made into the Zone 4 and saves the changes. *Help*, calls the help document on the usage of the GUI, whereas *Cancel* allows to undo the recently made changes into the PMML source.
- Zone 5 - *Data File Editor*: Zone 5 displays the real data of data file *SI_stdInput*, the selected node in zone 2. Updating a value in the cell of the table, results in change of the corresponding value in the data file.

VI. Workflow Reproducibility

Reproducibility is the most vital aspect of the science, it enables an independent researcher to run an experiment accurately at a different location, using different tools. Without reproducibility information the work is vanished after some

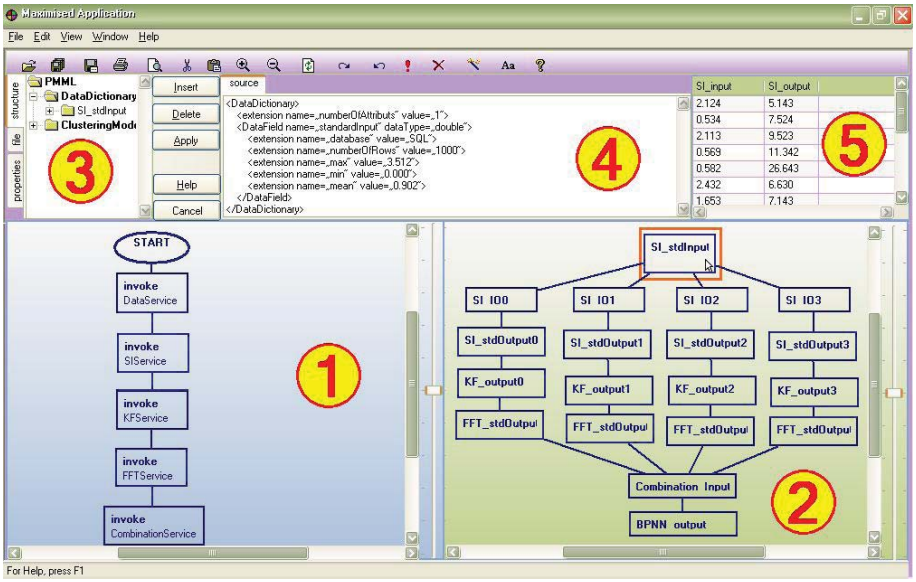

Fig. 7. Dynamic data and control flow visualization application

time and cannot be verified by independent researchers. Reproducibility is different from re-run as the later is usually executed by same researcher with the objective to measure the success rate. Reproducibility life cycle can be described as below:

- Perform experiment.
- Publish as all the information.
- Remote and/or independent researcher performs the experiment using published information.
- Evaluation of results.

To enable reproducibility over NIGM workflow, we collect and store complete information on sources used, services used, the sequence in which services interact, information on data transferred between services and input to the workflow i.e. their types, size, attributes, information on results (final output of the workflow) and statistical information on data such as mean, divergence, etc. We collect the information discussed, to make our workflow reproducible and store them in the PMML document for each instance of workflow run. We split our provenance information into two major types:

- *Workflow provenance*: It is more coarse grained information. The provenance information like name and type of data sources, name and sequence of activities, etc.
- *Data provenance*: This is finer grained provenance information set. It contains complete information on datasets and their transformation. The provenance data, like input(s)/output(s) names, types, size, parameters significance, etc, falls in this category.

Fig. 8 demonstrates the provenance information and their categorical division. Reproducibility can neither be measured nor quantified, but it is observed by the ability and ease of the scientist to run a workflow created by others. The provenance information collected and calculated, is sufficient

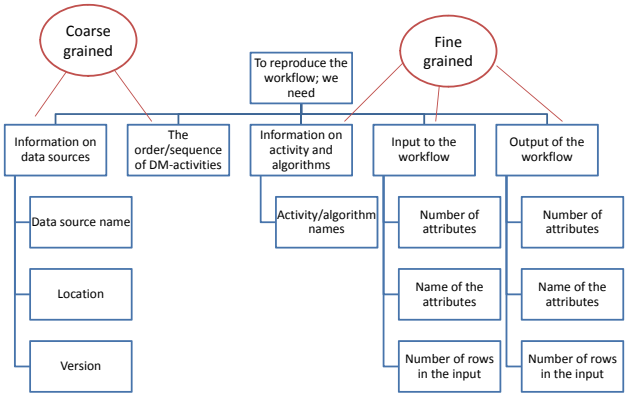

Fig. 8. Provenance reproducibility categories

for reproducibility and will bring benefits in:

1) Verification and validation of NIGM workflow results.
2) Avoid duplication of work. The provenance information will reduce efforts in understanding and execution of workflow.
3) Scientists' work remains live. The benefits discussed above will automatically keep the experiment alive, easily executable, understandable and testable, and hence will reduce the chance of probable loss of workflow.

VII. Future Work

This paper mainly presents the architectural concepts; other features, concepts, and performance evaluation will be presented in the upcoming papers. In the future we will extend our work from the specific NIGM workflow to more generalized

data mining workflows. Listed below are our other research plans.

- A mathematical model or algorithm can contain one or more parameters. The values of parameters influence the calculation carried out by the model and accordingly affects the final result given by using the model. Naturally, the user would want to optimize the result by modifying some of the model parameters. Unfortunately, there are raised two issues when approaching this problem. On one hand, in some models, such as the neural network one, the relation between each single parameter and the result achieved by using the model is not defined and can only be decided empirically. On the other hand, a workflow might contain tens of services, and the number of parameters in total can reach hundreds. We propose a method to determine the numeric weight of each parameter in a workflow, which reflects the influence the parameter exerts on the final result. The observation of weights can assist the user in finding the significant parameters to the result and accordingly determining which parameter to change if one wants to optimize the result. The provenance record of every service should contain the weight of every parameter so as to provide the user with information on how to adjust the model result as well as to show the most significant parameters.
- PMML is XML based, which makes it vendor independent and widely usable, but the PMML document structure lacks the features for mathematical models validation. In the future we will address this issue.
- We are presently working on the visualization of provenance data, standardization of provenance communication protocol, and on standardization of the PMML extensions.

VIII. Conclusions

This paper has introduced our workflow provenance system in the context of scientific workflows. One of the key contributions in this paper is the identification and classification of provenance information to enable reproducibility. For collecting and storing provenance data, we have adapted a novel PMML based approach and have extended PMML models and constructs. The visualization application of the workflow provenance system gives a detailed view of the workflow and dataflow. The provenance information collection and visualization is sufficient to support reproducibility, and will help the scientists to run, verify and optimize the NIGM workflow.

Acknowledgment

This research is done in the context of the GridMiner [26], [27] and ADMIRE [28] projects.

References

[1] E-Science, http://en.wikipedia.org/wiki/E-Science.
[2] H. Zhuge, *The Knowledge Grid*. Singapore: World Scientific Publishing Co., 2004.
[3] I. Elsayed, J. Han, T. Liu, A. Wohrer, F. A. Khan, and P. Brezany, "Grid-enabled non-invasive blood glucose measurement," in *Computational Science - ICCS 2008*. Krakow, Poland: LNCS 5101, June 2008, pp. 76–85.
[4] G. Maciocia, *The Foundations of Chinese Medicine: A Comprehensive Text for Acupuncturists and Herbalists*. Elsevier Churchill: Livingstone, 2005.
[5] WEEP, "Workflow enactment engine project, institute for scientific computing, university of vienna, austria," http://weep.gridminer.org/.
[6] PMML, "Predictive model markup language," http://dmg.org/pmml-v3-2.html.
[7] CADGrid, "China austria data grid project, institute for scientific computing, university of vienna, austria," http://www.par.univie.ac.at/project/cadgrid/,.
[8] IFAR, "International foundation for art research (ifar) online," http://www.ifar.org.
[9] Y. Cui and J. Widom, "Lineage tracing for general data warehouse transformations," in *VLDB 2001, Proceedings of 27th International Conference on Very Large Data Bases, September 11-14, 2001, Roma, Italy*, September 11-14 2001, pp. 471–480.
[10] P. Buneman, S. Khanna, and W. C. Tan, "Why and where: A characterization of data provenance," in *Proceedings of the 8th International Conference on Database Theory*. London, UK: Lecture Notes In Computer Science; Vol. 1973, 2001, pp. 316 – 330.
[11] Y. Cui and J. Widom, "Practical lineage tracing in data warehouses," in *ICDE '00: Proceedings of the 16th International Conference on Data Engineering*. Washington, DC, USA: IEEE Computer Society, 2000, p. 367.
[12] "Karma provenance framework," http://www.extreme.indiana.edu/karma/, Computer Science Department, Indiana University, USA.
[13] Y. L. Simmhan, B. Plale, D. Gannon, and S. Marru, "Performance evaluation of the karma provenance framework for scientific workflows," Lecture Notes in Computer Science, November 2006.
[14] PASOA, "Provenance aware service oriented architecture," http://www.pasoa.org, University of Southampton, UK.
[15] SOA, "Service oriented architecture," http://www.oasis-open.org/.
[16] myGrid, http://www.mygrid.org.uk/, Department of Computer Science, University of Manchester, UK.
[17] "Taverna workbench," http://taverna.sourceforge.net/, Department of Computer Science, University of Manchester, UK.
[18] W.-B. Zhang, D.-M. Jeong, Y.-H. Lee, and M. S. Lee, "Measurement of subcutaneous impedance by four-electrode method at acupoints located with single-power alternative current," The American Journal of Chinese Medicine (AJCM) 32(5), pp. 779 – 788, 2004.
[19] OGSA-DAI, http://www.ogsadai.org.uk/.
[20] L. Ljung, *System Identification - Theory For the Use*. Upper Saddle River, N.J.: PTR Prentice Hall, 1999.
[21] R. E. Kalman, "A new approach to linear filtering and prediction problems," ASME - Journal of Basic Engineering, 1960.
[22] S. Bochner and K. Chandrasekharan, *Fourier Transforms*. Princeton Book Comp., 2001.
[23] I. Foster, J. Frey, S. Graham, S. Tuecke, K. Czajkowski, D. Ferguson, F. Leymann, M. Nally, T. Storey, and S. Weerawaranna, "Modeling stateful resources with web services," v1.1. Technical report, Globus Alliance, 2004.
[24] K. Ballinger, D. Ehnebuske, C. Ferris, M. Gudgin, C. K., Liu, M. Nottingham, and P. Yendluri, "Ws-i basic profile version 1.1," http://www.ws-i.org/Profiles/BasicProfile-1.1.html, 2006.
[25] I. Janciak and P. Brezany, "Workflow enactment engine for wsrf-compliant services orchestration," in *The 9th IEEE/ACM International Conference on Grid Computing (Grid 2008)*. Tsukuba, Japan: IEE/ACM, September 29 - October 1 2008.
[26] GridMiner, http://www.gridminer.org/, Institute for Scientific Computing, University of Vienna, Austria.
[27] P. Brezany, I. Janciak, and A. M. Tjoa, "Gridminer: An advanced support for e-science," In: W. Dubiczky (ed.), Data Mining Techniques in Grid Computing Environments, Willy Blackwell, 2008.
[28] ADMIRE, "Advanced data mining and integration research for europe," http://www.admire-project.eu/.