



PERFORMANCE SERVICE INTERFACES AND DATA REPRESENTATION- DEVELOPER MANUAL

WP3

Document Filename:	KWF-WP3-UIBK-v1.0-DRDeveloperManual.doc
Work package:	WP3
Partner(s):	UIBK, CYFRONET, FIRST
Lead Partner:	UIBK
Document classification:	PUBLIC

Abstract: This manual describes in detail the design and implementation of the stable version of performance service interfaces and data representations for performance monitoring and analysis services in the K-WfGrid project.

Delivery Slip

	Name	Partner	Date	Signature
From	Hong-Linh Truong	UIBK	06/01/2006	
Verified by	Piotr Nowakowski	CYFRONET	07/10/2006	
Approved by	Steffen Unger	FIRST	09/10/2006	

Document Log

Version	Date	Summary of changes	Author
0.1	25/11/2005	First version	Hong-Linh Truong
0.2	01/12/2005	Update all chapters	Hong-Linh Truong
0.3	06/01/2006	Revise all chapters	Hong-Linh Truong
0.4	22/08/2006	Revise all chapters	Hong-Linh Truong
0.5	31/08/2006	Revise all chapters	Hong-Linh Truong
1.0	07/10/2006	QA check	Piotr Nowakowski

CONTENTS

1	COPYRIGHT NOTICE.....	4
2	INTRODUCTION.....	5
2.1	ABBREVIATIONS AND ACRONYMS.....	5
2.2	REFERENCES AND SOURCE CODE.....	5
3	IMPLEMENTATION STRUCTURE	8
3.1	PROTOTYPE USE CASES.....	8
3.1.1	<i>Instrumentation use case.....</i>	8
3.1.2	<i>Using data representations and PDQS use case.....</i>	8
3.1.3	<i>Using WARL use case</i>	9
3.1.4	<i>Using WfPerfOnto use case.....</i>	9
3.2	SOFTWARE COMPONENT MODEL.....	9
3.3	DETAILED IMPLEMENTATION MODEL	10
3.3.1	<i>XML-based requests and data representations for performance and monitoring data.....</i>	10
3.3.1.1	XML Representations for Performance and Monitoring Data.....	10
3.3.1.2	Performance Data Query and Subscription.....	10
3.3.1.3	Workflow Analysis Request Language (WARL)	11
3.3.1.4	Standardized Intermediate Representation	12
3.3.1.5	Workflow Instrumentation Request language (WIRL).....	13
3.3.2	<i>OWL-based ontologies for Performance data.....</i>	14
3.3.2.1	WfPerfOnto	14
3.3.2.2	WfMetricOnto	14
3.3.2.3	Mondata.....	15
4	SOFTWARE TESTING	16
5	CONTACT INFORMATION AND CREDITS.....	17
6	THE GPL LICENSE AGREEMENT.....	18
7	REFERENCES.....	19

1 COPYRIGHT NOTICE

Copyright (c) 2004-2006 Distributed and Parallel Systems Group, University of Innsbruck. All rights reserved.

Use of this product is subject to the terms and licenses stated in the GPL license agreement. Please refer to Section 6 for details.

Java™ is a registered trademark of Sun. All rights reserved.

Jena is copyrighted by Hewlett-Packard Development Company, LP.

Castor is copyrighted by Intalio Inc., and others.

Xerces is licensed under the Apache License, The Apache Software Foundation.

This research is partly funded by the European Commission IST-2002-511385 Project “K-WfGrid”.

2 INTRODUCTION

This document describes performance service interfaces and data representations (DR) implemented in WP3 in the framework of the K-WfGrid project (hence DR refers to the performance service interfaces and data representations in K-WfGrid.). DR contains XML-based data representations and languages, and OWL-based ontologies defined for

- Describing monitoring and performance data and specifying requests for controlling the performance data query and subscription.
- Describing the structure of workflow applications and specifying requests for controlling the instrumentation of workflows.
- Specifying performance analysis requests
- Specifying search for performance problems requests
- Describing information about performance data and services, representing ontological form for performance data associated with Grid workflows [WPPERFONTO].

DR provides (standardized) representations for data and requests and it is used by various services which involve in performance analysis and monitoring, e.g., DIPAS and GEMINI, as well services which utilize performance and monitoring data, e.g. KAA and Scheduler. DR helps simplifying the interaction and integration among various K-WfGrid Grid services.

2.1 ABBREVIATIONS AND ACRONYMS

Abbreviation	Description
DIPAS	Distributed Performance Analysis Service
DR	Performance Service Interfaces and Data Representation
GOM	Grid Organizational Memory
GWES	Grid Workflow Execution Service
KAA	Knowledge Assimilation Agent
OWL	Web Ontology Language
PDQS	Performance Data Query and Subscription
SIRWF	Standardized Intermediate Representation for Workflow
WARL	Workflow Analysis Request Language
WF	Workflow
WIRL	Workflow instrumentation request language

2.2 REFERENCES AND SOURCE CODE

All the source files of DR can be obtained from K-WfGrid CVS. DR consists of two parts

- XML-based data representations and requests: in \$KWFGRID/dr directory
- OWL-based ontologies for performance data: in \$KWFGRID/dipas/pmaonto directory

where \$KWFGRID refers to the root directory of the whole K-WfGrid CVS.

Fig. 2-1 depicts the directory structure of kwfgrid/dr which includes all the sources for XML-based data representations and requests

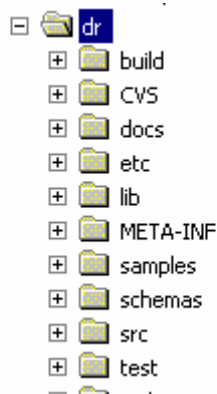


Fig. 2-1: Structure of source directory of XML-based data representations and requests.

Fig. 2-2 presents the structure of source directory of OWL-based ontologies for performance and monitoring data. The root directory pmaonto can be found under kwfgrid/dipas.

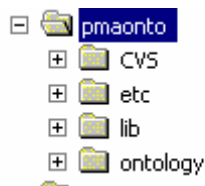


Fig. 2-2: Structure of source directory of OWL-based ontologies.

Table 2-1 shows the list of XML-based data representation and languages which have been defined and developed.

Category	Name	Description	Source File	Java package
Data Representation	host.cpu.used	CPU Usage of a computational node	schemas/cputime.xsd	net.kwfgrid.dr.hostcpuused
	service.available	Availability of services	schemas/ServiceAvailability.xsd	net.kwfgrid.dr.serviceavailable
	host.mem.used	Memory Usage of a computational node	schemas/memusage.xsd	net.kwfgrid.dr.hostmemused
	host.system.loadavg	Load average of a computational node	schemas/loadavg.xsd	net.kwfgrid.dr.hostsystemloadavg
	path.delay.roundtrip	Roundtrip delay of a network path measured at IP level	schemas/deplayroundtrip.xsd	net.kwfgrid.dr.pathdelayroundtrip
	path.bandwidth.capacity.TCP	TCP bandwidth of a network path	schemas/bandwidthtcp.xsd	net.kwfgrid.dr.bandwidthtcp
	host.sysinfo	System information of a computational node	schemas/hostsysinfo.xsd	net.kwfgrid.dr.hostsysinfo
	app.prof	Profiling data of applications	schemas/approfta.xsd	net.kwfgrid.dr.appprof
	app.event	Application events	schemas/appevent.xsd	net.kwfgrid.dr.appevent

	wfa.event	Workflow events	schemas/genericeventdata.xsd	net.kwfgrid.dr.genericevent
	SIRWF	Standardized intermediate representation for workflow-based application	schemas/sirwf.xsd	net.kwfgrid.dr.sirwf
Requests	PDQS	Performance data query and subscription	schemas/pdqs.xsd	net.kwfgrid.dr.pdqs
	WIRL	Workflow instrumentation request language	schemas/wirl.xsd	net.kwfgrid.dr.wirl
	WARL	Workflow analysis request language	schemas/warl.xsd	net.kwfgrid.dr.warl

Table 2-1: XML-based data representations and requests.

Table 2-2 shows the list of OWL-based ontologies which have been defined and developed.

Name	Description	Source File
WfPerfOnto	Ontology describing performance data associated with workflows	ontology/wfperfonto.owl
WfMetricOnto	Ontology describing performance metrics	ontology/wfmetric.owl
Mondata	Ontology describing information about available performance and monitoring data	ontology/mondata.owl

Table 2-2: OWL-based ontologies for performance and monitoring data.

3 IMPLEMENTATION STRUCTURE

3.1 PROTOTYPE USE CASES

Because the work on performance services interfaces and data representations defines a set of XML schemas and OWL-based ontologies used by various services which provide and/or use performance and monitoring data, the use cases described in the following sections will present how these services can utilize the defined XML schemas and ontologies to describe monitoring and performance data exchanged, to specify requests for retrieving data and controlling the instrumentation, etc.

3.1.1 Instrumentation use case

Fig. 3-1 presents the instrumentation use case in which

- DIPAS specifies instrumentation request by using WIRL
- Given a request from DIPAS, the instrumentation service can return the application structure describing in SIRWF or conduct the instrumentation based on information specified in the request.

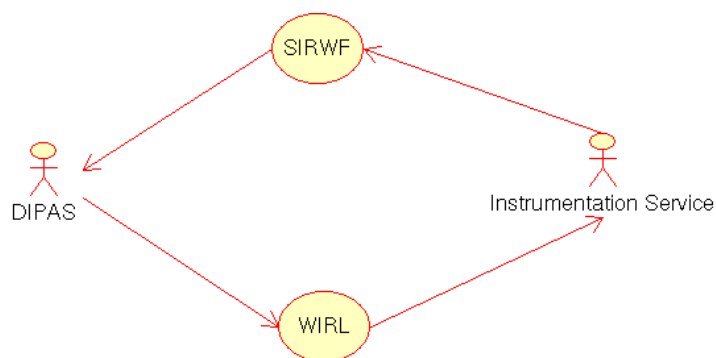


Fig. 3-1: Instrumentation use case.

3.1.2 Using data representations and PDQS use case

Fig. 3-2 presents a use case in which data representations describing performance and monitoring data and PDQS language are employed. Clients of the monitoring services, e.g., DIPAS, KAA and Scheduler specify requests for querying or subscribing data using PDQS schema. The PDQS request is then sent to the monitoring service which returns the resulting data described in well-defined data representations.

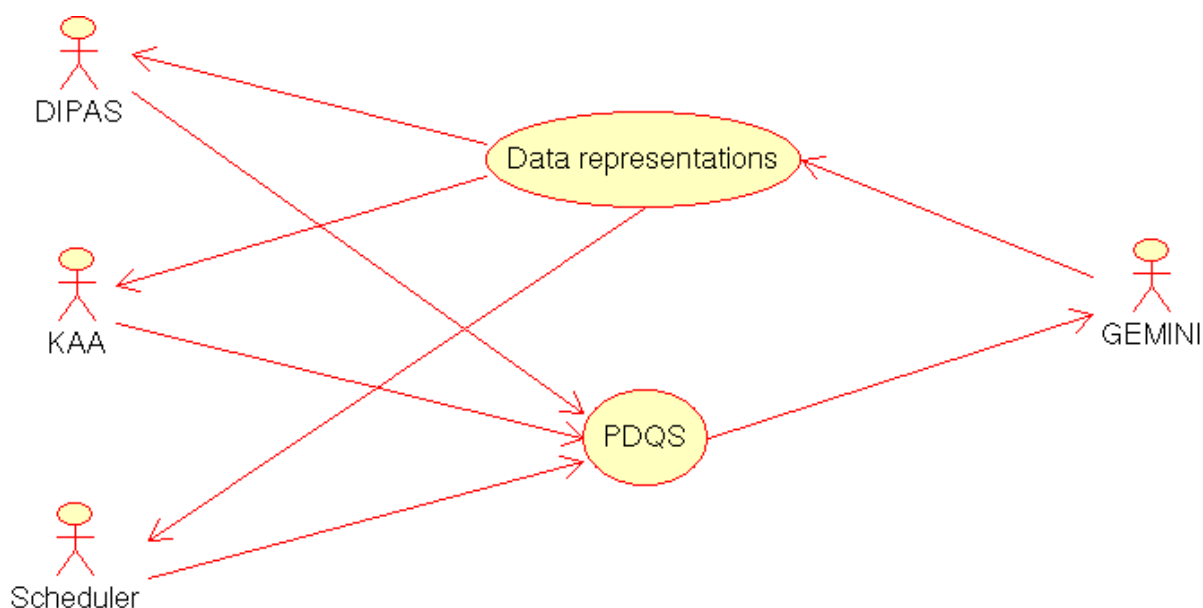


Fig. 3-2: Use case of using data representations and PDQS.

3.1.3 Using WARL use case



Fig. 3-3: Use case of using WARL.

Fig. 3-3 presents the use case in which WARL is used. From the portal, the user can specify a request for performance analysis by using WARL. The WARL request is sent to DIPAS Gateway which conducts performance analysis of workflows and searches for performance problems.

3.1.4 Using WfPerfOnto use case

Fig. 3-4 presents the use case in which WfPerfOnto is used. The performance analysis service, DIPAS, can export performance data associated with workflows into WfPerfOnto representation and store the exported performance result into GOM so that other services can utilize the ontological performance data.

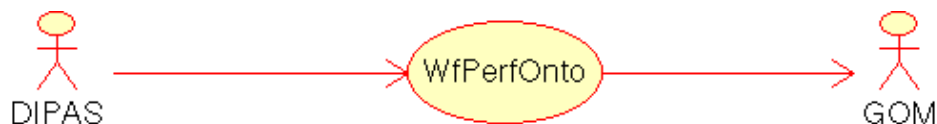


Fig. 3-4: Using WfPerfOnto use case.

3.2 SOFTWARE COMPONENT MODEL

The DR can be divided into two main parts

- XML-based data representations and languages.

- OWL-based performance ontologies.

3.3 DETAILED IMPLEMENTATION MODEL

3.3.1 XML-based requests and data representations for performance and monitoring data

This part includes

- Representations for monitoring and performance data
- Performance data query and subscription (PDQS) language
- Standardized intermediate representation for workflow (SIRWF)
- Workflow Instrumentation Request Language (WIRL)
- Workflow Analysis Request Language (WARL)

3.3.1.1 XML Representations for Performance and Monitoring Data

Each type of performance and monitoring data is specified by a schema. A message containing monitoring data of a monitored resource (e.g. machine, network path, code region) consists of the following information:

- Resource identifier: indicates information about the monitored resource or the experiment in which the monitoring and performance data is collected and measured.
- Data type identifier: indicates the type of the monitoring data.
- Performance measurements: are performance and monitoring data of a resource.

A generic structure of a message, which is used to describe the performance data, is outlined as follows:

```
<MonitoringData dataTypeID="datatypeid" resourceID="resourceid">
    <!-- XML describes experiment information -->
    <!-- XML describes performance measurements -->
    ....
</MonitoringData>
```

where `dataTypeID` and `resourceID` are data type identifier and resource identifier, respectively. The part expressing performance measurements is dependent on each type of data.

To add a new schema describing a new data type, the developer should define a schema which is based on the above-mentioned generic structure.

3.3.1.2 Performance Data Query and Subscription

Performance monitoring and analysis services support data query and subscription, and notification. Requests for data query and subscription will be expressed in a pre-defined XML schema named PDQS (Performance Data Query and Subscription). PDQS requests will be used in service interfaces for data query and subscription. PDQS requests can be constructed based on OWL [OWL] descriptions of monitoring data published in GOM.

The main elements of PDQS schema are presented in Fig. 3-5

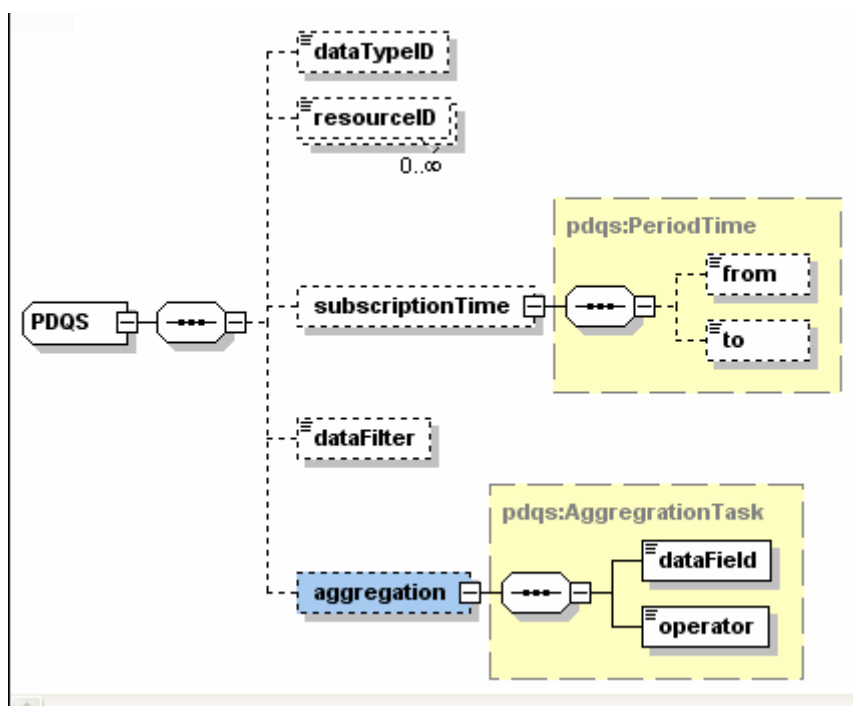


Fig. 3-5: PDQS schema visualized by XMLSpy

Data subscription and query requests will contain the following information

- `dataTypeID` and `resourceID`: are used to determine a type of the monitoring data associated with a monitored resource.
- `subscriptionTime`: specifies the duration during which the subscription is valid.
- `dataFilter`: used to filter the content of performance data. `dataFilter` is expressed in XPath/XQuery [XPATH,XQUERY]
- `aggregation`: supports the aggregation of requested data. Currently we define two aggregation operators: MIN and MAX.

3.3.1.3 Workflow Analysis Request Language (WARL)

A WARL request includes three parts

- Constraints on objects to be analyzed, `constraints`, defined by `WARLConstraint`
- Performance metrics to be analyzed, `analyze`, defined by `WARLAnalyze`, and
- Performance problem specifications, `perfProblemSpecs`, defined by `WARLPerfProblemSpec`.

Constraints include information about objects to be hierarchical workflow concepts. Constraints basically consist of a set of concepts and their properties. For example

- Concepts: such as `Workflow` and `Activity`. Concepts are identified by type and name. The name indicates the identifier of the concept in the workflow description, e.g. activity name, while the type determines whether the concept is a workflow or an activity or a code region, etc.

- Properties: properties of concepts. For example, a code region is belonged to an invoked application.

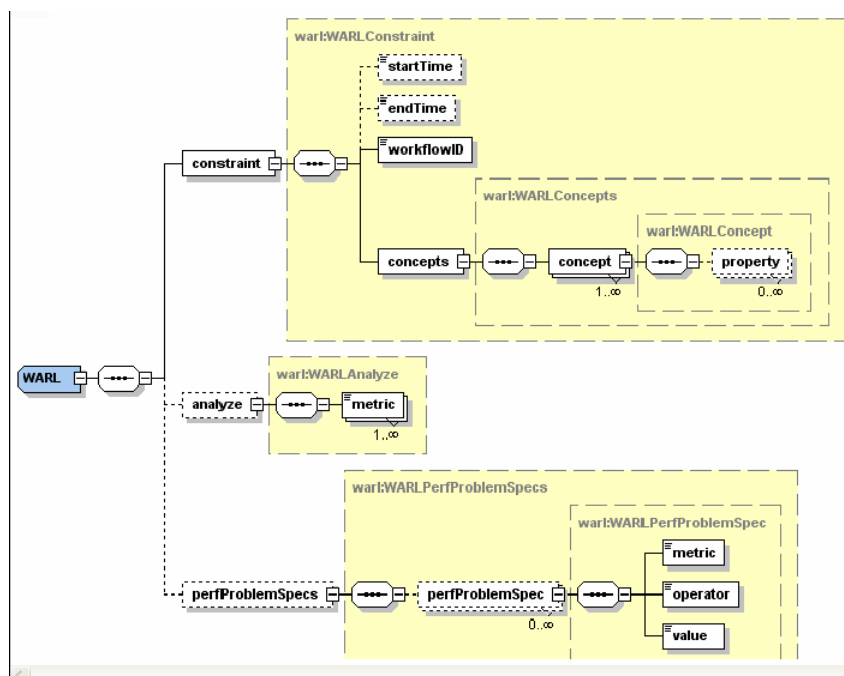


Fig. 3-6: WARL schema visualized with XMLSpy

A WARL request specifies a list of performance metrics that should be analyzed and provided. This list contains a set of metrics, each identified by a unique name. Performance problem specifications include a set of performance conditions, each condition includes a metric name, an operator (e.g., greater than or less than), and a value (e.g., indicating a threshold).

3.3.1.4 Standardized Intermediate Representation

SIRWF is used to describe the structure of invoked applications of workflow activities; SIRWF is developed based on SIR [SIR]. Based on SIRWF, the instrumentation requesters can select interesting code regions for instrumentation.

Fig. 3-7 presents main elements of SIRWF. SIRWF currently supports only levels of program unit and function call. An application process (SIRApp) is represented as a set of program units. Each program unit (SIRUnit) contains a set of code regions. A code region (SIRCodeRegion) is a function call that contains information (e.g. name, source information) about the calling function (SIRCallee). Each program unit or code region is associated with a unique identifier. The requester uses that identifier to specify a program unit or a code region. By using SIRWF, the instrumentation requester can understand the structure of applications and make instrumentation requests by specifying code regions together with performance metrics that should be instrumented.

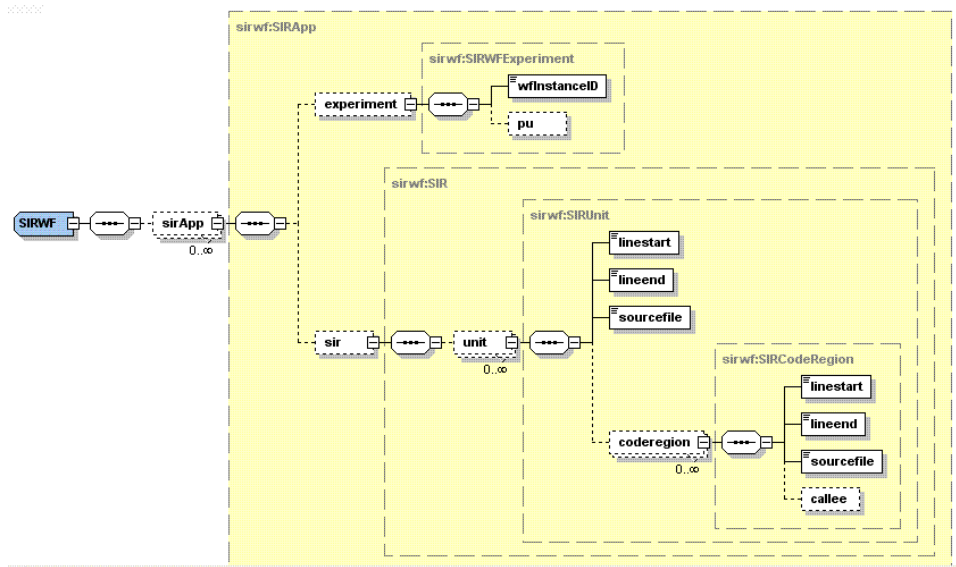


Fig. 3-7: Main elements of SIRWF visualized with XMLSpy.

3.3.1.5 Workflow Instrumentation Request language (WIRL)

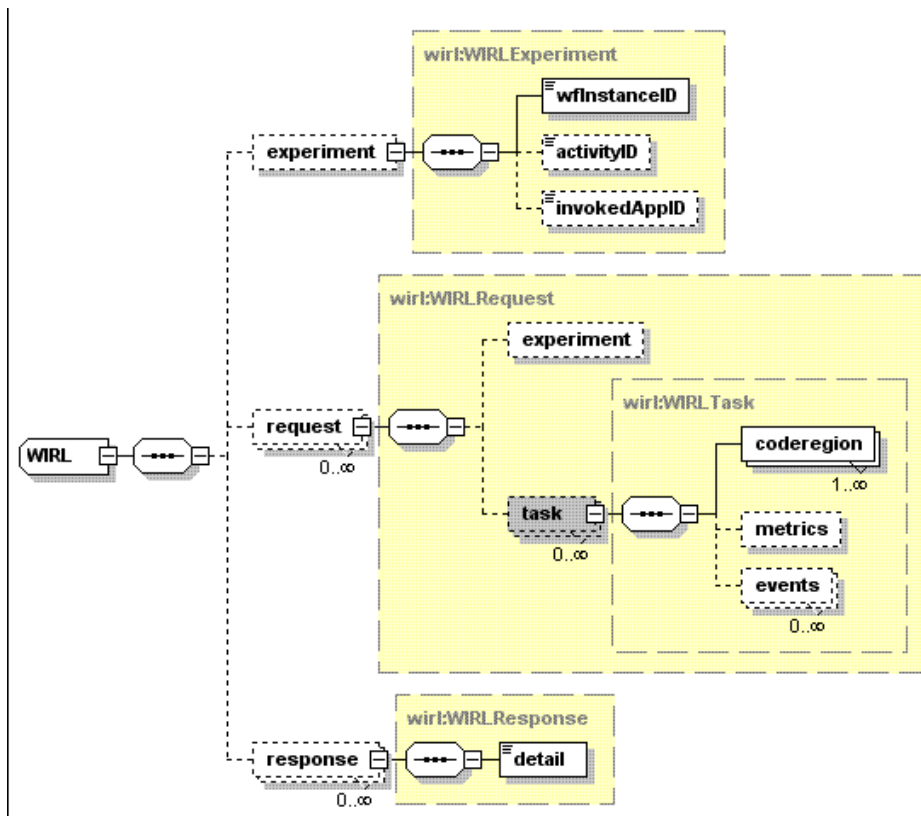


Fig. 3-8: Main elements of WIRL schema visualized with XMLSpy

WIRL is an XML-based request and response language. Main elements of WIRL schema are depicted in Fig. 3-8.

A *WIRL request* consists of *experiment information* and *instrumentation tasks*. Experiment information (e.g., activity identifier, application name, computational node, etc.) identifies applications to be instrumented. *Instrumentation tasks* specify instrumentation operations. Examples of instrumentation tasks can be a request for all instrumented functions within an application, to enable or disable an instrumented code, etc. The current request names are presented below

```
<xsd:simpleType name="RequestName">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ATTACH"/>
    <xsd:enumeration value="GETSIR"/>
    <xsd:enumeration value="ENABLE"/>
    <xsd:enumeration value="DISABLE"/>
    <xsd:enumeration value="FINALIZE"/>
  </xsd:restriction>
</xsd:simpleType>
```

Table 3-1 explains the instrumentation request names.

Request name	Description
ATTACH	To attach/prepare to instrument a given application
GETSIR	To get SIR of the application
ENABLE	To enable/instrument a code region
DISABLE	To disable/deinstrument a code region
FINALIZE	To finish the instrumentation of an application

Table 3-1: Instrumentation requests

Code region identifiers are obtained from the list of functions provided by the instrumentation. Metric names are pre-defined; performance metrics are specified by the metric ontology [WPPERFONTO]. Code region identifiers can be used to indicate code regions to be instrumented.

A *WIRL response* contains the name of a request, the status of the request (e.g. OK, FAIL) and possibly detailed result information encoded in `<![CDATA[...]>` tag.

3.3.2 OWL-based ontologies for Performance data

3.3.2.1 WfPerfOnto

WfPerfOnto is an ontology describing performance data associated with Grid workflow. Refer to [WPPERFONTO] for further information about WfPerfOnto.

3.3.2.2 WfMetricOnto

WfMetricOnto is an ontology describing performance metrics associated with Grid workflows. Refer to [WPPERFONTO] for further information about WfMetricOnto.

3.3.2.3 *Mondata*

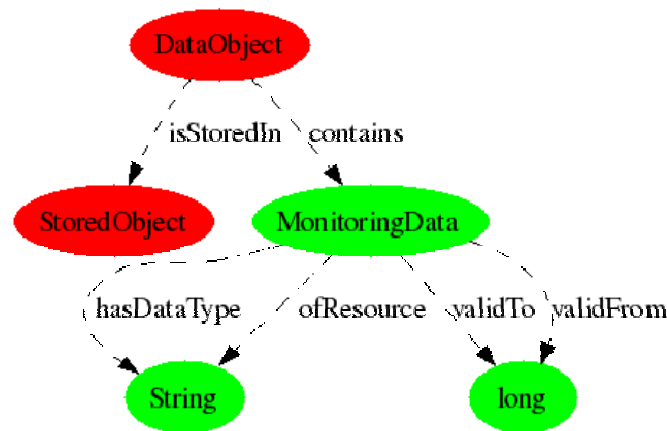


Fig. 3-9: Ontology describing information about monitoring data (modata).

mondata is an OWL description of information about monitoring data. Fig. 3-9 depicts main concepts and properties in mondata. The class `MonitoringData` describes the concept of information about monitoring data. `MonitoringData` has four properties:

- Property `hasDataType`: specifies the sensor which generates the monitoring data.
- Property `ofResource`: specifies the resource to be monitored.
- Property `isStoredIn`: specifies the handle of the Monitoring Service which provides the monitoring data.
- Properties `validFrom`, `validTo`: specify the duration in which the data is available.

This ontology has been integrated into GOM.

4 SOFTWARE TESTING

The current prototype contains a set of Java source, XML and ontology samples which can be used to test the prototype. An XML document can be tested whether it complies with the corresponding XML schema by using those samples. For example, if the XML document is not valid, the parsing sample will generate error messages.

5 CONTACT INFORMATION AND CREDITS

DR is developed at:

Distributed and Parallel Systems Group,
Institute of Computer Science, University of Innsbruck
Technikerstrasse 21A, A-6020, Innsbruck, Austria

Developers:

- Hong-Linh Truong

With contributions from

- Peter Brunner
- Robert Samborski

Part of DR code is based on previous developments in SCALEA-G framework [SCALEA-G].

Visit <http://www.dps.uibk.ac.at/projects/pma> for further information about our work on performance monitoring and analysis. Further comments, suggestions, and bug reporting, contact Hong-Linh Truong at truong@dps.uibk.ac.at.

6 THE GPL LICENSE AGREEMENT

Copyright (c) 2004-2006 Distributed and Parallel Systems Group, University of Innsbruck. All rights reserved.

This software includes voluntary contributions made to K-WfGrid. For more information on K-WfGrid, please see <http://www.kwfgrid.eu>.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

7 REFERENCES

- [CASTOR] <http://www.castor.org/>
- [DRUSER] K-WfGrid Consortium, "K-WfGrid Performance Service Interfaces and Data Representation - User Manual", August, 2006
- [JENA] Jena Semantic Web Toolkit –home page: <http://jena.sourceforge.net/>
- [JUNIT] <http://www.junit.org/>
- [OWL] "OWL Web Ontology Language Reference", <http://www.w3.org/TR/owl-ref/>
- [SCALEA-G] Hong-Linh Truong and Thomas Fahringer. "SCALEA-G: a Unified Monitoring and Performance Analysis System for the Grid", Scientific Programming, 12(4):225-237, IOS Press, 2004.
- [SIR] Clovis Seragiotto, Hong-Linh Truong, Thomas Fahringer, Bernd Mohr, Michael Gerndt, and Tianchao Li, "Standardized Intermediate Representation for Fortran, Java, C and C++ Programs", Technical Report AURORATR2004-18, Institute for Software Science, University of Vienna, October 2004. Also APART Working Group TR <http://www.kfa-juelich.de/apart/>).
- [WFPERFONTO] Hong-Linh Truong, Thomas Fahringer, Francesco Nerieri, Schahram Dustdar "Performance Metrics and Ontology for Describing Performance Data of Grid Workflows, IEEE International Symposium on Cluster Computing and Grid 2005 (CCGrid2005), 1st International Workshop on Grid Performability, IEEE Computer Society Press, Cardiff, UK, 9 - 12 May 2005.
- [XPATH] XML Path Language, <http://www.w3.org/TR/xpath.html>
- [XERCES] <http://xerces.apache.org/>
- [XMLAPACHE] <http://xml.apache.org/>
- [XQUERY] XQuery: an XML Query Language, <http://www.w3.org/TR/xquery/>