Performance Analysis, Data Sharing and Tools Integration in Grids: New Approach based on Ontology*

Hong-Linh Truong[1] and Thomas Fahringer[2]

[1] Institute for Software Science, University of Vienna
truong@par.univie.ac.at
[2] Institute for Computer Science, University of Innsbruck
Thomas.Fahringer@uibk.ac.at

Abstract. In this paper, we propose a new approach to performance analysis, data sharing and tools integration in Grids that is based on ontology. We devise a novel ontology for describing the semantics of monitoring and performance data that can be used by performance monitoring and measurement tools. We introduce an architecture for an ontology-based model for performance analysis, data sharing and tools integration. At the core of this architecture is a Grid service which offers facilities for other services to archive and access ontology models along with collected performance data, and to conduct searches and perform reasoning on that data. Using an approach based on ontology, performance data will be easily shared and processed by automated tools, services and human users, thus helping to leverage the data sharing and tools integration, and increasing the degree of automation of performance analysis.
Key words: performance analysis, performance data model, Grid, ontologies.

1 Introduction

The recent emerging Grid computing raises many challenges in the domain of performance analysis. One of these challenges is how to understand and utilize performance data where the data is diversely collected and no central component manages and provides semantics of the data. Performance monitoring and analysis in Grids differ from that in conventional parallel systems in terms of no single tool providing performance data for all Grid sites and the need of conducting monitoring, measurement and analysis across multiple Grid sites at the same time. Normally users run their applications in multiple Grid sites, each is equipped with different computing capabilities, platforms, libraries that require various tools to conduct performance monitoring and measurement. Without the central component, performance monitoring and measurement tools have to provide a means for seamlessly utilizing the data they collect and provide, because many tools and services atop them need the data for specific purposes such as performance analysis, scheduling and resource matching. Current Grid performance tools focus on monitoring and measurement, but neglect data sharing and tools integration.

We take a new direction on describing the semantics of performance data and establishing performance data sharing and tools integration by investigating the use of

* This research is supported by the Austrian Science Fund as part of the Aurora Project under contract SFBF1104.

ontology in performance analysis domain. Basically, ontologies provide a *shared and common* understanding of a domain that can be communicated between people and heterogeneous and widely spread application systems; ontology is developed to facilitate *knowledge sharing and reuse* [6, 3]. Based on sharable and extensible ontologies in the domain of performance analysis, an analysis tool, service or user is able to access multiple sources of performance and monitoring data provided by a variety of performance monitoring and measurement tools, understanding the data and making use of that data. With the expressive power provided by ontology which can describe concepts, resources in sufficient detail, supporting automatic performance analysis will also be enhanced.

The rest of this paper is organized as follows: Section 2 discusses the motivation. In Section 3 we present our proposed ontology for performance data. We describe an architecture for an ontology-based model for performance analysis, data sharing and tools integration in Section 4. Section 5 overviews our prototype implementation. The related work is discussed in Section 6 followed by the Conclusion and Future work in Section 7.

2 Motivation

Currently, several data representations with different capabilities and expressiveness, e.g. XML, XML and relational database schema, are employed by Grid performance monitoring and measurement tools. However, little effort has been done to standardize the semantics of performance data as well as the way performance tools collaborate. In Grids, data is diversely collected and no central component manages and provides its semantics. Each Grid site may be equipped with its own performance monitoring and measurement tool. Thus, the end user or the high-level tool in Grids has to interact with a variety of tools offering monitoring and measurement service. Performance monitoring and measurement tools should not simply offer well-defined operations for other services calling them, e.g. based on Open Grid Services Architecture (OGSA) [4], but they have to provide means for adding semantics to the data as Grid users and services require seamless integration and utilization of the data provided by different tools.

Existing approaches on performance data sharing and tools integration which mostly focus on building wrapper libraries for directly converting data between different formats, making data available in relational database with specific data schema, or exporting data into XML, have several limitations. For example, building a wrapper requires high cost of implementation and maintenance; wrappers convert data between representations but not always between semantics. Although XML and XML schemas are sufficient for exchanging data between parties that have agreed in advance on definitions, their use and meaning, they mostly are suitable for one-to-one communication and impose no semantic constraints on the meaning of the data. Everyone can create his own XML vocabularies with his own definitions for describing his data. However, such vocabularies and definitions are not sharable and do not establish a common understanding about the data, thus preventing semantic interoperability between various parties which is an important issue that Grid monitoring and measurement tools have to support. Utilizing relational databases to store performance data [15, 13] simplifies sharing of data. However, data models represented in relational database are still very

tool-specific and inextensible. Notably, XML and relational database schemas do not explicitly express meanings of data they encode. Since all above-mentioned techniques do not provide enough capability to express the semantics of performance data and to support tools integration, they might not be applicable in Grids due to the autonomy and diversity of performance monitoring and measurement tools.

We investigate whether the use of ontology can help to solve the above-mentioned issues. Ontologies are a popular research topic in various communities such as knowledge engineering, cooperative information systems. An ontology is a formal, explicit specification of a shared conceptualization [6]. An ontology typically consists of a hierarchical description of important concepts in a domain, along with descriptions of the properties of each concept [7]. One of key features of ontology is that it provides a shared and common understanding of some domains that can be communicated between people and application systems. Another feature is that a set of ontology statements by itself can allow to conclude another facts, e.g. via description logics [7], while that can not be achieved with XML or database schema.

Ontology can help addressing the above-mentioned issues in many ways. Firstly, ontology can be used to directly *describe and model the data* collected, thus allowing to share a common understanding of performance data and easily correlating the data with the knowledge domain. Secondly, ontology can be used to *define mappings between different representations* employed by different Grid monitoring and measurement tools. This would allow a high level service to transparently access different types of data in a homogeneous way. This paper works on the first direction. Due to space limit, this paper discusses only main concepts and results of our approach. For more details, readers should refer to [16].

3 PERFONTO: Ontology for Describing Performance Data

While initial work on using ontology for system and network management has been introduced, e.g. in [1], to date we are not aware any ontology for describing performance data of applications in the field of performance analysis. Our starting point is that we try to propose an ontology for describing monitoring and performance data of both applications and systems. In this section, we describe PERFONTO (**ONTO**logy for **PERF**ormance data), an ontology based on OWL (Web Ontology Language) [11]. PERFONTO comprises two parts that describe *experiment-related concept* and *resource-related concept*. Here we briefly discuss main classes and properties of PERFONTO.

Experiment-related concept describes experiments and their associated performance data of applications. The structure of the concept is described as a set of definitions of *classes* and *properties*. Figure 1 demonstrates a part of classes and properties of experiment-related concept in PERFONTO. **Application** describes information about the application. **Version** describes information about versions of an application. **SourceFile** describes source file of a version. **CodeRegion** describes a static (instrumented) code region. Code regions are classified into subclasses that are programming paradigm-dependent and paradigm-independent. **Experiment** describes an experiment which refers to a sequential or parallel execution of a program. **RegionInstance** describes a region instance which is an execution of a static (instrumented) code region at runtime. A

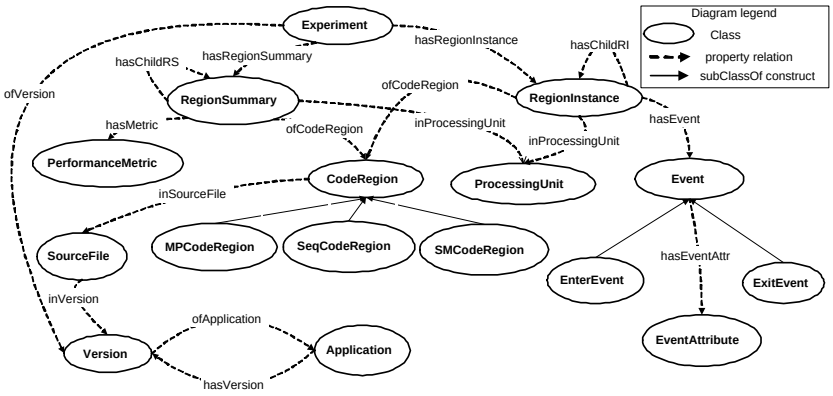

Fig. 1. Illustrative classes and properties of experiment-related concept.

code region instance is associated with a processing unit (this relationship is described by property *inProcessingUnit*) and has events (property *hasEvent*) and subregion instances (property *hasChildRI*). A processing unit, represented by class **ProcessingUnit**, describes the context in which the code region is executed; the context includes information about grid site, compute node, process, thread. **RegionSummary** describes the summary of code region instances of a static (instrumented) code region in a processing unit. A region summary has performance metrics (property *hasMetric*) and subregion summaries (property *hasChildRS*). **PerformanceMetric** describes a performance metric, each metric has a name and value. **Event** describes an event record. An event happens at a time and has event attributes (property *hasEventAttr*). **EventAttribute** describes an attribute of an event which has an attribute name and value.

Resource-related concept describes static, benchmarked, and dynamic (performance) information of computing and network systems. In the current version, resource-related concept provides classes to describe static and benchmarked data of computing and network resources. For example, **Site** describes information of (grid) computing site. **Cluster** describes a set of physical machines (compute nodes). Cluster has a subclass namely *SMPCluster* represented a cluster of SMP. **ComputeNode** describes information about physical machine. ComputeNode also has subclasses, e.g. *SMPComputeNode* represented an SMP machine. **Network** describes an available network. Subclasses of Network can be *EthernetNetwork*, *MyrinetNetwork*, etc. **NodeSharedMemoryPerf** describes performance characteristics of shared memory operations of a compute node. **NetworkMPColPef** and **NetworkMPP2PPerf** describe performance characteristics of collective and point-to-point message passing operations of a network, respectively.

The proposed ontology is largely based on our previous work on developing data schema for expressing experiment data in relational database [15] and on APART experiment model [2]. The development of PERFONTO should be considered as the investigation of using ontology for describing performance data, not establishing a standard for all tools. However, one of the main key advantages of ontology is that different

ontologies can be reconciled [10]. Therefore, one can employ or extend PERFONTO, others may develop their own ontologies. Finally, proposed ontologies can be merged.

4 An Architecture for An Ontology-based Model for Performance Analysis, Data Sharing and Tools Integration

Figure 2 presents a three layers architecture for an ontology-based model for performance analysis, data sharing and tools integration. At the core of this architecture is a performance data repository service which includes:

- *PERFONTO* is ontology for representing performance data discussed in Section 3.
- *Ontological database* is a relational database which is used to hold ontologies (e.g PERFONTO) and performance data (instance data).
- *ONTO APIs* are interfaces used to store and access data in ontological database.
- *Query Engine* provides searching and querying functions on ontological data.
- *Inference Engine* provides reasoning facilities to infer, discover and extract knowledge from ontological performance data.

The performance data repository service is designed as an OGSA Grid service [4]. The *Performance Data Collector* of performance monitoring and measurement tools can store collected data (instance data) along with corresponding ontology model (e.g. PERFONTO) into the ontological database. Via service operations, any clients needing performance data such as performance analysis tools, schedulers or users can easily request the ontology model and then retrieve instance data from ontological database. The key difference from approaches of using XML or relational database is that performance data is either described by a common ontology (e.g. PERFONTO) or by a tool-defined ontology, thus, with the presence of ontology model, these clients can easily understand and automatically process retrieved data. Via *Performance Data Wrapper*, data in tool-defined non-ontology format also can be extracted and transformed into ontological representation and vice versa. To implement this architecture, we select Jena [8] for processing ontology-related tasks.

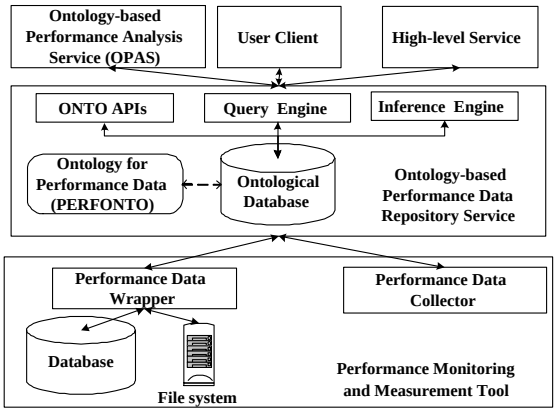

Fig. 2. Three layers architecture for an ontology-based model for performance analysis, data sharing and tools integration.

4.1 Search on Ontological Data

A search engine is developed to support clients finding interesting data in the ontological database. At the initial step, we use a search engine provided by Jena that supports RDQL query language [9]. The use of RDQL in combining with ontology can simplify and provide a high-level model for searches in performance analysis in which searching query is easily understood and defined by end-users, not only by tool developers.

```
l1:  SELECT ?regionsummary
     WHERE
l2:       (?regionsummary perfonto:inProcessingUnit ?processingunit)
l3:       (?processingunit perfonto:inNode "gsr410")
l4:       (?regionsummary perfonto:hasMetric ?metric)
l5:       (?metric perfonto:hasMetricName "wtime")
l6:       (?metric perfonto:hasMetricValue ?value)
l7:  AND (?value >=3E8)
l8:  USING perfonto FOR <http://www.par.univie.ac.at/project/scalea/perfonto#>
```

Fig. 3. An example of RDQL query based on PERFONTO.

Figure 3 presents an RDQL query, based on PERFONTO, which finds any region summary executed in compute node $gsr410$ that its wallclock time (denoted by metric name $wtime$) is greater than or equal to 3E8 microsecond. Line l_1 selects variable $regionsummary$ via $SELECT$ clause. In line l_2 information about processing unit of $regionsummary$, determined by property *perfonto:inProcessingUnit*, is stored in variable $processingunit$. The compute node of $processingunit$ must be *"gsr410"* as stated in line l_3. In line l_4, performance metric of $regionsummary$ is stored in variable $metric$ and line l_5 states that the name of $metric$ must be *"wtime"*. In line l_6, the value of $metric$ is stored in variable $value$ which must be greater than or equal to 3E8 as specified in line l_7. Line l_8 specifies the URI for the shortened name *perfonto*.

4.2 Reasoning on Ontological Data

The use of ontology for representing performance data allows additional facts to be inferred from instance data and ontology model by using axioms or rules. Based on ontology, we can employ inference engine to capture knowledge via rules.

Let us analyze a simple rule for detecting all MPI point-to-point communication code regions of which the average message length is greater than a predefined threshold [2]. As presented in Figure 4, line l_1 defines the name of the rule. In line l_2, a term of triple pattern specifies link between a region summary and its associated code region. Line l_3 states the code region is an instance of $MPCodeRegion$ (message passing code region) and is an MPI point-to-point communication region (denoted by mnemonic CR_MPIP2P) as specified in line l_4. Line l_5, l_6 and l_7 are used to access the average message length of the region summary. Line l_8 checks whether the average message length is greater than a predefined threshold (BIG_MESSAGES_THRESHOLD) by using a built-in function. In line l_9, the action of this rule concludes and prints the region

summary having big message. This example shows how using ontology helps simplifying the reasoning on performance data.

```
l1:  [rule_detect_bigmessages:
l2:       (?regionsummary perfonto:ofCodeRegion ?codeRegion),
l3:       (?codeRegion rdf:type perfonto:MPCodeRegion),
l4:       (?codeRegion perfonto:hasCrType "CR_MPIP2P"),
l5:       (?regionsummary perfonto:hasMetric ?metric),
l6:       (?metric perfonto:hasMetricName "AvgMessageLength"),
l7:       (?metric perfonto:hasMetricValue ?length),
l8:       greaterThan(?length, BIG_MESSAGES_THREADHOLD)
l9:  ->       print(?regionsummary,"Big message hold!")]
```

Fig. 4. An example of rule-based reasoning based on PERFONTO.

5 Prototype Implementation

We are currently implementing the proposed ontology and ontology-based service. The ontology-based performance data repository is an OGSA-based service of which the ontological database is based on PostgreSQL. However, in current prototype this service supports only operations for retrieving and storing ontology descriptions and instance data; searching and reasoning have to be done at client side. We are working on providing searching and reasoning operations.

We develop an Ontology-based Performance Analysis Service (OPAS) which supports ontology-based searching and reasoning. Figure 5 presents an user interface for performing searches in OPAS. In the top window the user can specify queries whereas the result will be shown in the bottom window. For example, we conducted a search with the query presented in Section 4.1 with a 3D Particle-In-Cell application. In the bottom window, under the subtree of variable $regionsummary$, list of region summaries met the condition will be shown. The user can examine performance metrics in details. Also other information such as source code and machine can be visualized as needed.

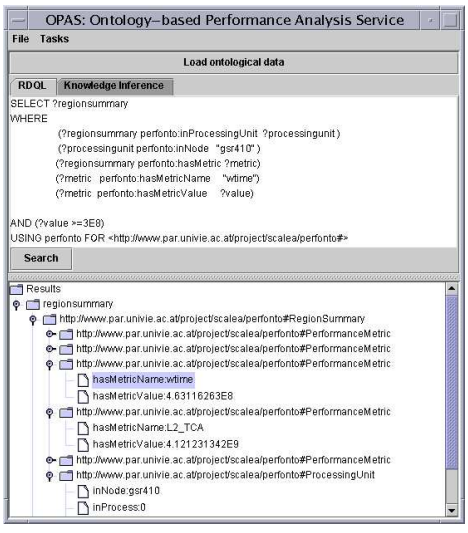

Fig. 5. GUI for conducting searches.

6 Related work

Database schemas are proposed for representing performance data, e.g. in SCALEA [15] and Prophesy [13]. However, these approaches are tool-specific rather than widely-accepted data representations. It is difficult to extend database schema structure to describe new resources. The relational database schema does not explicitly express semantics of data whereas the ontology does. As a result, building knowledge discovery via inference on ontological data is less intensive work, hard and costly.

The Global Grid Forum (GGF) Network Measurements working group has created an XML schema which provides a model for network measurement data [5]. Similarly, GLUE schema [14] defines a conceptual data model to describe computing and storage elements and networks. In [1], ontology has been applied for improving the semantic expressiveness of network management information and the integration of information definitions specified by different network managements. None of these schemas models concepts of application experiments. However, the objects modeled in GGF and GLUE schema are similar to that in our resource-related ontology. Thus vocabularies and terminologies of these schemas can be incorporated into our resource-related ontology.

Recent work in [12] describes how ontology can be used for resource matching in the Grid. Our framework can provide data for matching resources in Grids.

7 Conclusion and Future Work

In this paper, we have investigated how ontology can help overcome the lack of semantics description possessed by current techniques that are used in existing performance monitoring and measurement tools to describe performance data. Initial results show that ontology is a promising solution in the domain of performance analysis because it not only provides a means for seamless utilization and integration of monitoring and performance data but also increases the degree of automation of performance analysis.

Besides working toward the full prototype, we are currently enhancing and reevaluating our proposed ontology. We are extending resource-related concept to cover dynamic data of compute and network systems at runtime, and advancing the experiment-related ontology to describe performance properties, performance data of workflow applications, etc. In addition, we plan to study the use of ontology for mapping between different representations of performance data.

References

1. J.E. Lopez de Vergara, V.A. Villagra, J.I. Asensio, and J. Berrocal. Ontologies: Giving semantics to network management models. *IEEE Network*, 17(3):15–21, May-June 2003.
2. T. Fahringer, M. Gerndt, Bernd Mohr, Felix Wolf, G. Riley, and J. Träff. Knowledge Specifi cation for Automatic Performance Analysis, Revised Version. APART Technical Report, http://www.kfa-juelich.de/apart/result.html, August 2001.
3. D. Fensel. *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag, Berlin, 2001.
4. I. Foster, C. Kesselman, J. Nick, and S. Tuecke. Grid Services for Distributed System Integration. *IEEE Computer*, pages 37–46, June 2002.

5. GGF Network Measurements Working Group. http://forge.gridforum.org/projects/nm-wg/.
6. T. R. Gruber. A translation approach to portable ontology specifi cations. *Knowledge Acquisition*, 5(2):199–220, 1993.
7. Ian Horrocks. Reasoning with expressive description logics: Theory and practice. In Andrei Voronkov, editor, *Proc. of the 19th Int. Conf. on Automated Deduction (CADE 2002)*, number 2392 in Lecture Notes in Artifi cial Intelligence, pages 1–15. Springer, 2002.
8. Jena - A Semantic Web Framework for Java. http://jena.sourceforge.net.
9. RDQL: RDF Data Query Language. http://www.hpl.hp.com/semweb/rdql.htm.
10. Deborah L. McGuinness, Richard Fikes, James Rice, and Steve Wilder. An Environment for Merging and Testing Large Ontologies. In *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR2000)*, April 2000.
11. OWL Web Ontology Language Reference. http://www.w3.org/tr/owl-ref/.
12. H. Tangmunarunkit, S. Decker, and C. Kesselman. Ontology-based Resource Matching in the Grid—The Grid meets the Semantic Web. In *Proceedings of the Second International Semantic Web Conference*, Sanibel-Captiva Islands, Florida, USA, October 2003.
13. V. Taylor, X. Wu, J. Geisler, X. Li, Z. Lan, R. Stevens, M. Hereld, and Ivan R.Judson. Prophesy:An Infrastructure for Analyzing and Modeling the Performance of Parallel and Distributed Applications. In *Proc. of HPDC's 2000*. IEEE Computer Society Press, 2000.
14. The Grid Laboratory Uniform Environment (GLUE). http://www.cnaf.infn.it/ѕergio/datatag/glue/index.htm.
15. Hong-Linh Truong and Thomas Fahringer. On Utilizing Experiment Data Repository for Performance Analysis of Parallel Applications. In *9th International Europar Conference(EuroPar 2003)*, LNCS, Klagenfurt, Austria, August 2003. Springer-Verlag.
16. Hong-Linh Truong and Thomas Fahringer. An Ontology-based Approach To Performance Analysis, Data Sharing and Tools Integration in Grids. Technical Report AURORA TR2004-01, Institute for Software Science, University of Vienna, January 2004.