GWD-I (Informational)

B. Tierney, Lawrence Berkeley National Laboratory
R. Aydt, University of Illinois at Urbana-Champaign
D. Gunter, Lawrence Berkeley National Laboratory
W. Smith, NASA Ames
M. Swany, University of California, Santa Barbara
V. Taylor, Northwestern University
R. Wolski, University of California, Santa Barbara

GGF Performance Working Group

March 2000
Revised 11-December-2001

A Grid Monitoring Architecture

Status of this Memo

This memo provides information to the Grid community regarding a Grid Monitoring Architecture (GMA) being developed by the Global Grid Forum Performance Working Group. The goal of the architecture is to provide a minimal specification that will support required functionality and allow interoperability. Distribution is unlimited.

1. Abstract

Large distributed systems such as Computational and Data Grids require a substantial amount of monitoring data be collected for a variety of tasks such as fault detection, performance analysis, performance tuning, performance prediction, and scheduling. Some tools are currently available and others are being developed for collecting and forwarding this data. The goal of this paper is to describe the major components of a ~~common gG~~rid monitoring architecture ~~with all the major components~~ and their essential interactions. By adopting standard terminology and describing the minimal specification to support required functionality, we hope to encourage the development of interoperable high-quality performance tools for the grid. To motivate the Grid Monitoring Architecture (GMA) design, and to guide implementation, a discussion of the characteristics ~~aid implementation, we also discuss the performance characteristics of a Grid Monitoring system and identify areas~~ that are critical to proper functioning of ~~the system.~~a performance monitoring system for the Grid are also presented.

B. Tierney, Lawrence Berkeley National Laboratory
R. Aydt, University of Illinois at Urbana-Champaign
D. Gunter, Lawrence Berkeley National Laboratory
W. Smith, NASA Ames
M. Swany, University of California, Santa Barbara
V. Taylor, Northwestern University
R. Wolski, University of California, Santa Barbara

GGF Performance Working Group

March 2000
Revised 11-December-2001

Table of Contents

2. Introduction

Performance monitoring of distributed ~~computing~~ components is critical for enabling high-performance distributed computing. Monitoring data is needed to determine the source of performance problems and to tune the system and application ~~for better performance~~. Fault detection and recovery mechanisms need monitoring data to determine if a server is down, and to decide whether to restart the server or to redirect service requests elsewhere [1][2] ~~[10][14]~~. A performance prediction service ~~might use~~takes monitoring data as inputs ~~for~~to a prediction model [3]~~[16]~~, which ~~would~~ is in turn ~~be~~ used by a scheduler to determine which resources to ~~use.~~assign to a job.

There are several groups ~~that are~~ developing Grid monitoring systems ~~to address this problem~~ [2][3][4][5]~~[9] [11] [14][16]~~ and these groups, along with others in the Global Grid Forum community, recognize ~~have recently seen~~ a need to interoperate. In order to facilitate this, we have developed an architecture ~~specific to the needs of a Grid monitoring system~~ of monitoring components that specifically addresses the characteristics of Grid platforms. A Grid monitoring system is differentiated from a general monitoring system in that it must be scalable across wide-area networks~~,~~ and ~~include~~ encompass a large number of heterogeneous resources. ~~Its~~ The monitoring system's naming and security mechanisms must also be integrated with other Grid middleware.

We believe the Grid Monitoring Architecture (GMA) described here addresses these concerns, and is ~~sufficiently general that it could be adapted~~also suitable for use in distributed environments other than the Grid. For example, ~~it~~ the monitoring sytem could be used with large compute farms or clusters that require constant monitoring to ensure all nodes are running correctly. [RAA1]

~~2.1~~3. Design Considerations [RAA2]

With the potential for thousands of resources at geographically ~~different~~ distant sites and tens-of-thousands of simultaneous Grid users, it is ~~important~~ critical that~~for~~ the data ~~management and~~ collection and distribution mechanisms ~~facilities to~~ scale, while~~,~~ at the same [RAA3]time~~,~~ protecting the data from spoiling. ~~.~~ To this end, two design principles guiding the GMA are first, that data discovery should be separate from data transfer and second, that there should be mechanisms for establishing long-lived "streams" of data, allowing `O(1)`, instead of `O(N)`, communications overhead for transferring `N` related data. A corollary of the second principle is that the efficiency and scalability considerations of the mechanism to establish a data "stream" will be amortized over `N` data, and thus may be separable from the efficiency and scalability considerations of the data transfer itself for large values of `N`.

In order to allow scalability in both the administration and in the performance impact of such a Grid monitoring system, decisions ~~about~~ concerning what is monitored, measurement frequency, and ~~how the data is made available~~accessibility to collected data must be distributed throughout system, with dynamic control at site of the local resources. ~~made locally to the monitoring activity.~~ [RAA4]Thus, instead of a centralized management component, multiple independent components coordinate their state through metadata entries in a directory service, which may itself be distributed. Distributing management in this fashion ~~monitoring server, there are many independent monitoring components. To bind the system together, these components place metadata describing their state in a ce ntral directory service, which may itself be physically distributed. Localizing the monitoring responsibilities~~ also helps minimize the effects of host and network failure, making the system more robust under precisely the kinds of conditions it is trying to detect. [DG5]

In order to separate data discovery from data transfer, an unchanging subset of metadata must be abstracted and placed in a universally accessible location, called here a "directory service", along with enough information to bootstrap the communication between the data's source and sink. Scalability results from restricting and organizing the metadata so that the directory service itself may be distributed, and so that the rate of communication between distributed nodes increases slowly relative to the total amount of data transferred.

This model is different from the "event channel" model of the ~~In some models, such as the~~ CORBA Event Service[6], which conflates the mechanism for finding the data that should be transferred with the mechanism for starting the transfer itself into a single "searchable" channel of information. ~~all communication flows through a central component, which represents a potential bottleneck.~~ In contrast, ~~we propose that~~ in our design ~~-~~performance event data, which makes up the majority of the communication traffic, ~~should~~ travels ~~-~~directly from the producers of the data to the consumers of the data. In this way, individual producer/consumer pairs can do "impedance matching" based on negotiated requirements, and the amount of data flowing through the system can be controlled in a precise and ~~localized~~ distributed fashion based on current local load considerations. The design also allows for replication and reduction of event data at intermediate components acting as consumer/producer caches or filters. Use of these intermediate components lessens the load on producers of event data that is of interest to many consumers, with subsequent reductions in the network traffic, as the intermediaries can be placed "near" the data consumers[RAA6]. ~~Because the directory service contains only metadata, with careful design it will~~ should not be a bottleneck.

~~We also considered a purely SNMP[RAA7] based solution for monitoring, but rejected it because we felt that the SNMP simple GET/SET model is not rich enough, as there is no support for subscription. Also, it is not clear that the security model maps well to the Grid Security Infrastructure. However, we definitely envision the use of SNMP-based tools as a source of monitoring data.~~

~~3.~~4. Architecture and Terminology

The Grid Monitoring Architecture consists of three types of components, shown in Figure ~~-~~1:

- Directory Service: supports information publication and discovery
- ~~oConsumer: receives performance data (performance event sink)~~
- Producer: makes performance data available (performance event source)
- Consumer: receives performance data (performance event sink)

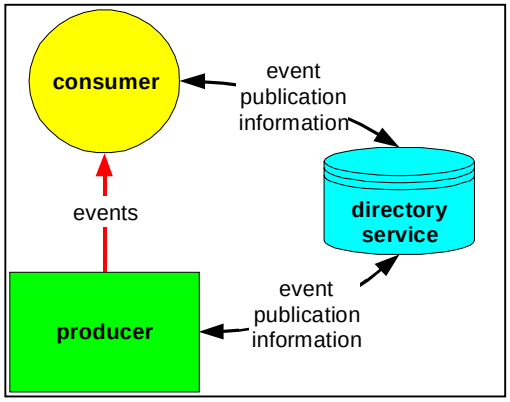

Figure 1: Grid Monitoring Architecture Components

The GMA is designed to handle performance data transmitted as timestamped (*performance*) *events*. An event is a typed collection of data with a specific structure that is defined by an *event schema*. Performance event data is always sent directly from a producer to a consumer.

The GMA architecture supports both a streaming publish/subscribe model ~~, similar to several existing Event Service systems such as the CORBA Event Service [1]~~, and a single transfer query/response model[RAA8]. For both models, producers or consumers that accept connections publish their existence in a directory service. Consumers can use the directory service to discover producers of interest and producers can use the directory service to discover consumers of