

Command and Control for Grid Infrastructures



A Thesis

Submitted to the Office of Graduate Studies

of

University of Dublin, Trinity College

in Candidacy for the Degree of

Doctor of Philosophy

by Keith Rochford

April 2008

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Keith Rochford

September 1, 2008

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Keith Rochford

September 1, 2008

Acknowledgments

I would like to acknowledge the encouragement, guidance and support of my supervisor Dr Brian Coghlan. His interest in the subject has been an inspiration.

My colleagues in the Computer Architecture Group, every one of whom has added to the enjoyment, experience and learning of the last few years. A special thanks to Gabriele for the collaboration in regard to the Social Grid Agents and the exciting brain storming sessions, to Stuart for his encouragement in relation to the active security experiments, and to Eamonn and Stephen for their advice and assistance. I would also like to thank John Walsh who was always ready to help. Thanks also to the other members of the Grid-Ireland operations team for their input, opinions and not least for their efforts in maintaining the production and test infrastructures that made this research possible.

I would like to thank Science Foundation Ireland and the Computer Science department in Trinity College for their support in funding this research.

My parents, my sisters and brother, and Nora for all their love and support over the last few years.

To my dearest Linda for her unwavering support, encouragement, and understanding.

KEITH ROCHFORD

University of Dublin, Trinity College
April 2008

Abstract

The centralised management of distributed computing infrastructures presents a number of considerable challenges, not least of which is the effective monitoring of physical resources and middleware components to provide an accurate operational picture for use by administrative or management staff. The detection and presentation of real-time information pertaining to the performance and availability of computing resources is a difficult yet critical activity.

This thesis presents an architecture intended to enhance the service monitoring experience of a Grid operations team. We have designed and implemented an extensible agent-based architecture capable of detecting and aggregating status information using low-level sensors, functionality tests and existing information systems. To date it has been successfully deployed across eighteen Grid-Ireland sites.

Managing the availability of the monitored services is an associated and essential task in ensuring the availability of a grid infrastructure. This project aims to take the next step in the monitoring and management of grid computing systems by developing a standards-based distributed control plane for grid resources and middleware components. While many existing projects focus on the monitoring of grid resources and infrastructures, they make little or no provision for the active or pro-active management operations required to ensure the availability of the infrastructure.

This solution employs web service technologies with the intention of producing an extensible body of work that might be applied to a range of grid projects and middlewares. With the emergence of meta-grids and the increasing importance of grid middleware interoperability, the acceptance and adoption of standard interfaces will become an important step in the development of future grid components.

Contents

Acknowledgments	iv
Abstract	v
List of Tables	viii
List of Figures	x
Chapter 1 Introduction	1
1.1 Motivations	1
1.2 Objectives	2
1.3 Contribution	3
1.4 Thesis Structure	3
Chapter 2 Context	4
2.1 Introduction to Grid Computing	4
2.2 The Evolution of Distributed Computing	6
2.3 First Generation - the birth of wide-area computing	7
2.3.1 Fafner	7
2.3.2 I-WAY	7
2.4 Second Generation - the emergence of integrated systems	9
2.4.1 Globus	9
2.4.2 Legion	9
2.4.3 UNICORE	10
2.5 Third Generation - the adoption of Web service architectures	10
2.6 Large-scale Grids	11
2.6.1 NASA Information Power Grid	11

2.6.2	European DataGrid	11
2.6.3	LCG and EGEE	12
2.6.4	CrossGrid and Int.eu.grid	12
2.6.5	Open Science Grid	13
2.6.6	The European Grid Initiative	13
2.6.7	DEISA and TeraGrid	14
2.6.8	Naregi	15
2.6.9	GridPP and UK NGS	15
2.6.10	Grid-Ireland	16
2.7	Peer-to-Peer and Community Computing	16
2.7.1	BOINC	17
2.7.2	World Community Grid	18
2.7.3	Distributed.net	18
2.8	Utility Computing and Clouds	19
2.8.1	IBM Cloud Computing	20
2.8.2	Amazon Web Services	21
2.8.3	Sun Grid	21
2.8.4	Microsoft Clouds	22
2.8.5	3Tera	22
2.9	HPC Eco-Systems	23
2.10	Evolution of Distributed Infrastructure Management	25
Chapter 3 Distributed Infrastructure Monitoring		27
3.1	Introduction	27
3.2	Information Requirements and Models	28
3.2.1	The GLUE Schema	30
3.2.2	The OGF NM-WG Network Measurement Schema	30
3.2.3	The GOCDB Schema	32
3.2.4	Common Information Model	32
3.3	Cluster and Network Monitoring Tools	34
3.3.1	Ganglia	34
3.3.2	Lemon	35
3.3.3	Nagios	35
3.4	Grid Information Systems	36
3.4.1	MDS	36
3.4.2	gLite/EGEE Information System	37

3.4.3	R-GMA	40
3.5	Grid Monitoring Systems	40
3.5.1	A Grid Monitoring Architecture	41
3.5.2	PyGMA	43
3.5.3	The Relational Grid Monitoring Architecture	44
3.5.4	GridICE	47
3.5.5	MonALISA	47
3.5.6	GSTAT	49
3.5.7	C.O.D.E.	49
3.5.8	Site Functional Test	51
3.5.9	SAM	51
3.5.10	CrossGrid Host Check	51
3.5.11	JIMS	54
3.5.12	I4C	55
3.5.13	RB-STATS	55
3.6	Real-time Grid Job Monitoring Systems	55
3.7	Summary	58
Chapter 4 Distributed Infrastructure Management		60
4.1	Introduction	60
4.2	Bodies and Standards	61
4.2.1	Global Grid Forum	61
4.2.2	Distributed Management Task Force	61
4.2.3	Internet Engineering Task Force	62
4.2.4	OASIS	62
4.2.5	Centre for Information Technology Leadership	62
4.2.6	World Wide Web Consortium	63
4.2.7	International Organization for Standardization	63
4.2.8	International Telecommunication Union	63
4.2.9	TeleManagement Forum	63
4.2.10	Infrastructure Management Standards	64
4.3	Grid Management Models	68
4.3.1	Centralised Management	68
4.3.2	Federated Management	70
4.3.3	Grid Operations Centres	70
4.4	Grid Management Groups	72

4.4.1	LCG Monitoring Working Groups	73
4.4.2	EGEE Operations Automation Team	74
4.5	Grid Management Tools	74
4.5.1	The LCG GOC Database	75
4.5.2	SMILE	75
4.5.3	ELFms	76
4.5.4	InGrid	77
4.5.5	Help Systems	78
4.6	Command and Control	79
4.6.1	Operations Other Than War	81
4.6.2	Summary	83
4.6.3	Application of Control Theory	85
4.6.4	Control Strategies	86
4.6.5	Properties of control	87
4.6.6	Feedback Loop Control	87
4.6.7	Model Predictive Control	88
4.6.8	Control in Grid Operations	89
Chapter 5 Vision		91
5.1	A Vision for Command / Control in Grid Operations	92
5.2	System Requirements	94
5.3	Human Machine Interface	100
5.4	The Case for Automated Management	104
Chapter 6 Grid Monitoring with I4C		105
6.1	Introduction	105
6.2	Motivations	105
6.3	Centralised vs. Agent-based Monitoring	106
6.4	Architecture	107
6.4.1	Monitoring Agents	107
6.4.2	Configuration Database	110
6.4.3	Central Monitoring WebService	110
6.5	Presentation and Alerting	111
6.5.1	Reports/Displays/Alerts	111
6.5.2	Navigation of Information	111
6.5.3	Republishing for use by higher-level software	114
6.6	Deployment	116

6.7	Conclusions	116
Chapter 7 Grid Command/Control with Grid4C		119
7.1	Introduction	119
7.2	Use cases	120
7.2.1	Service Outages	120
7.2.2	Automatic service recovery	120
7.2.3	Managing batch queues	120
7.2.4	Active Security	121
7.2.5	Active Authorization	121
7.2.6	Applications server monitoring	121
7.2.7	Workflow based resource management	121
7.2.8	Autonomic resource management and provision	122
7.2.9	Alerting, visualisation, and interaction	122
7.2.10	Peer resource management	122
7.2.11	Extension of existing monitoring tools	122
7.2.12	Scheduled management actions	122
7.3	Related Work	123
7.4	The potential role of WSDM	123
7.4.1	The WSDM development model	124
7.5	Architecture	126
7.5.1	Overview	126
7.5.2	The Control Element & Final Elements	128
7.5.3	The Management Server	129
7.5.4	The Control Client	130
7.6	Event Generation and Standard formats	130
7.7	Security	131
7.8	Configuration	131
7.9	Example Management Capability	132
7.10	WSDM Implementation	134
7.11	Sample Management Client	135
7.12	Summary	135
Chapter 8 Deployment and Evaluation		138
8.1	Testbeds	138
8.1.1	TestGrid	138
8.2	Deployment	139

8.3	Evaluation	139
8.3.1	Performance	139
8.3.2	Scalability	141
8.4	Use-Cases	141
8.4.1	Autonomic Policy-based Resource Management	141
8.4.2	Integration with the Nagios monitoring system	143
8.4.3	Providing reliable resource metrics for Economic Grid Markets	144
8.4.4	Integration with Active Security Infrastructure	150
8.5	Summary	152
Chapter 9 Visualisation and User Interface		153
9.1	Relevant Display Technologies	154
Chapter 10 Service Availability Models		158
10.1	Determining Site and Service availability statistics	158
10.1.1	Measures of system availability	158
10.2	Modelling Grid Service Availability as a finite space Markov Chain	161
10.2.1	Predicting future states	163
10.2.2	Evaluating the steady state	166
10.2.3	Multi-step Transition Probabilities	167
10.3	Summary	167
Chapter 11 Conclusions		168
11.1	Conclusions	168
11.2	Future Work	169
11.2.1	WSDM-Management of web services	169
11.2.2	Complex Event Processing	170
11.2.3	Social Grid Agents	170
11.2.4	Advanced Visualisation	170
11.2.5	WSRP for Grid Management Portals	171
11.2.6	Discovery and autonomic management	171
11.2.7	Developing system models	171
11.2.8	Using WS-BPEL to execute management workflows across Grid4C End-points	171
11.3	Personal Note	172
Bibliography		173

Appendices	191
Appendix A The WSDM standard capabilities	192
Appendix B The WSDM Event Format	194
B.1 WEF event components	194
B.2 WEF SituationCategory types	195
Appendix C WS-Notification Traces	200
Glossary	206

List of Tables

3.1	Selection of CE Attributes from the GLUE schema	30
3.2	Globus MDS Version Information	38
3.3	R-GMA Producer Types	47
4.1	Infrastructure Management Standards	64
5.1	Some of the dimensions of grid C ⁴ I requirements	96
6.1	HyperGraph Infrastructure Monitoring Key	114
6.2	Example Analyser event handlers	116
6.3	Grid monitoring system requirements	116
7.1	A selection of typical properties and operations exposed by Grid4C management endpoints.	133
7.2	Properties and operations of a simple GridFTP management capability. The same selection of properties and operations are provided for GRAM, GRIS, SSH, etc.	133
7.3	The PBS Management Capability	133
7.4	Example Resource Broker management capability	134
7.5	Grid monitoring system requirements	135
8.1	Baseline performance measurements for test network environment	140
8.2	Metrics for gLite Workload Management System.	148
10.1	Annual hours of downtime for availability Vs expected period of operation	159
10.2	Frequency Distribution of recorded states for Grid-Ireland Sites	160
10.3	Frequency Distribution of recorded states for Grid-Ireland site csTCDie	160
10.4	Frequency Distribution of recorded states for CE host at site csTCDie	160

10.5 Frequency Distribution of recorded states for HTTP service on GridInstall at TCD	161
--	-----

List of Figures

2.1	Timeline of Distributed Computing Projects discussed in Chapter 2	8
2.2	3Tera AppLogic user interface	24
3.1	UML illustration of GLUE schema	31
3.2	A section of the LCG Grid Operations Centre Database schema.	33
3.3	gLite information system architecture. [EGEc]	39
3.4	The Monitoring Cycle	41
3.5	Grid Monitoring Architecture components	43
3.6	A GMA compound Producer/Consumer	44
3.7	Architectural overview of the Relational Grid Monitoring Architecture	45
3.8	A Screenshot of the R-GMA browser interface	46
3.9	A Screenshot of the GridICE browser interface monitoring the Int.eu.grid production infrastructure	48
3.10	A Screenshot of the GSTAT browser interface	50
3.11	A Screenshot of the Site Functional Test browser interface	52
3.12	A Screenshot of the Service Availability Monitoring browser interface	53
3.13	Architecture of the JIMS monitoring framework.	54
3.14	A Screenshot of the RB-STATS web interface	56
3.15	A Screenshot of the GridPP real time job monitoring display	57
3.16	A Screenshot of the Grid-Ireland real time job monitoring display	58
4.1	A hierarchical operations management structure	71
4.2	The GGUS web interface	78
4.3	The Request Tracker web interface	79
4.4	The ‘MAPE’ loop representing the execution of command	81
4.5	Management by Objective applied to grid service management	84
4.6	Block diagram controller/plant	86

4.7	Block diagram of a feedback loop controller	88
4.8	Block diagram of a PID controller	88
5.1	Flow of information and control	91
5.2	Unification of monitoring and control for grid management	92
5.3	A proposed architecture for a grid management and operations support system. Areas of functionality provided by the systems developed by the author of this thesis are illustrated.	95
5.4	An example of a dedicated control facility for system operations	101
5.5	An example of a 'V-Rack' virtual instrumentation display	102
6.1	Agent-based monitoring architecture	108
6.2	Monitoring Agent start-up sequence	109
6.3	A section of the Configuration Database Schema	110
6.4	Example Nagios Display	112
6.5	Detailed service monitoring information	113
6.6	A snapshot of the Hypergraph Display of Infrastructure Status in mid-navigation	115
7.1	Architectural overview of control backplane.	127
7.2	Architecture of remote management capabilities.	130
8.1	Analysis of WS-N driven reaction times.	140
8.2	Analysis of Response time.	141
8.3	Nagios / Grid4C integration architecture.	145
8.4	SGA and G4C integration architecture.	147
8.5	Complex interaction topology between SGA and Grid4C agents.	149
8.6	Integration of Grid4C components as Active Security policy enforcement points.	152
9.1	The videowall at the AT&T global network operations centre, one of the largest command and control centres in the world. One hundred and forty-one rear-projection video screens are employed, each measuring 1.2 by 1.5 meters. . .	154
9.2	Consoles in the CERN Control Centre for monitoring and control of the LHC apparatus.	155
9.3	The Machine Control Centre (MCC) at Jefferson Labs continuous beam accelerator facility (CEBAF).	156
9.4	Prototype operator consoles engineered by the author of this thesis.	156
9.5	The videowall in the video conferencing centre used by the Grid-Ireland Operations team.	157

9.6	TFT screens displaying monitoring information at the Grid-Ireland operations centre.	157
10.1	Directed graph showing all possible transitions	163
10.2	Directed graph showing state transition probabilities for HTTP service on Gridinstall at TCD	164

'It's perfectly intelligible,' the captain said, in an offended tone,
'to anyone that understands such things.'

Lewis Carrol

Chapter 1

Introduction

This chapter introduces the subject of this dissertation; the implementation and use of a standards-based monitoring and control framework for the management of e-infrastructure resources.

The motivations for the work are outlined, along with the identification of a number of objectives and research questions that are to be addressed. Finally, the contributions made by this work are identified and an outline of the remainder of this dissertation is presented.

1.1 Motivations

The effective management of distributed computational resources is an essential task if e-infrastructures such as grids are to fulfil their potential and deliver expected levels of service. By their nature, these are complex distributed systems requiring significant management effort. There are two fundamental requirements to make such management possible; information and awareness about the managed system, its configuration and current status, and a capacity for control. It is only when these requirements are met that strategic and operational management objectives may be achieved.

A great deal of work has been carried out to date in the area of monitoring and information systems to provide the necessary awareness. So much in fact that new initiatives and working groups have been established to review and evaluate current solutions so that recommendations might be made to interested parties. Many of these development projects gave little thought to standard mechanisms for the representation and communication of data, interoperability, or future integration. Even fewer of these tools recognised the requirement to support control of, or action upon, the monitored resources. Once administrators were alerted to a problem, they were left to traditional manual processes to rectify the situation.

Given the number of existing monitoring tools, I believe that any further contributions should be based on sound accepted requirements, and endeavour to ensure support for standards, extensibility, and interoperability. Having identified the requirements for an infrastructure management tool, and the absence of any substantive grid control solution, simultaneously highlighting the magnitude of the problem and presenting an opportunity not to be missed, I believe that the research and development is a worthwhile undertaking. A standards-based system for resource management, coupled with display, control, and automation components, has the potential to significantly reduce the management effort required for the operation of grid infrastructures, and to increase availability, utilisation, and security. In order for such a system to be successful, it must meet a wide range of requirements, including several relating to security, performance, interoperability, and extensibility among others.

1.2 Objectives

The aim of this undertaking is to investigate whether it is practicable to remotely monitor and control the components of a grid infrastructure and, if so, evaluate the use of such control capabilities for both centralised and distributed control. I also wish to explore the use of this control infrastructure for both manual and automatic/policy-based control of resources. The security implications must be addressed and mechanisms put in place. The benefits of this distributed management system should be illustrated through clear and simple concepts that are validated by practical implementations and deployment.

The research objectives of this work include:

- Investigation of the monitoring and management requirements of distributed computing resources.
- Identification and evaluations of existing tools that aim to satisfy these requirements
- The investigation of existing technologies and standards that might be appropriate to this task
- The design of an architecture for command and control of grid resources, and prototype implementation with which to verify it
- Evaluation of the resulting system through its deployment and a variety of use cases.

1.3 Contribution

The thesis of this dissertation is that applying a command and control system to the operation of a grid infrastructure can reduce the management effort required and increase its availability and performance, and that there is significant benefit to the development of such systems around open standard technologies.

This is investigated by conducting research into existing and potential management activities in order to identify the system requirements and applications.

Based on the identification of a number of shortcomings in available monitoring tools, Chapter 6 presents a distributed agent-based monitoring tool, I4C, developed by the author and successfully deployed on a production infrastructure.

Chapter 7 presents a standards-based, open, and extensible architecture for the management of distributed resources, Grid4C, along with a prototype implementation. This prototype is used to demonstrate its suitability for purpose by employing it to satisfy a number of current management requirements.

1.4 Thesis Structure

I begin by discussing the evolution of distributed computing so that the reader may understand the effects that this evolution and increasing complexity has had on the requirements of system management. I then describe the background to this work: in Chapter 3 I examine related monitoring mechanisms for distributed computing infrastructures, before discussing management tools and approaches in Chapter 4. Chapter 5 presents a vision and requirements analysis for a command and control system for distributed infrastructure. In Chapter 6, I present my own contribution to the field of distributed monitoring. Chapter 7 describes an architecture and implementation of a measurement and control system that builds upon the research presented in Chapter 6 and augments it with the addition of control functionality.

Chapter 8 describes the test deployments of the control system and outlines a number of use cases and experiments. In Chapter 9, I briefly outline a number of relevant display technologies that might be used for visualisation of and interaction with the managed system. Chapter 10 presents a number of models used to describe resource availability, before introducing a novel means for the representation of grid service availability based on the results of the system described in Chapter 6. In Chapter 11 I present my conclusions based on this research, before concluding with the identification of some future work and further avenues of related research.

Chapter 2

Context

2.1 Introduction to Grid Computing

The term ‘Grid Computing’ became popular in the 1990’s based on a metaphor for the provision of compute resources that could be easily accessed in an on-demand fashion in a manner similar to the usage of electricity from a power grid. The initial concept proposed a distributed computing platform for advanced science and engineering. This would be achieved by providing a software environment that would make it possible to combine and share loosely coupled resources spanning organisations and geographical boundaries. In essence, the key objective is to co-ordinate resources belonging to, and managed by, multiple participants, and to negotiate sharing of those resources among a set of participants for the purposes of satisfying computational or communication requirements.

A rapid growth in interest ensued and considerable progress has since been made by many projects in their efforts to realise the vision. These projects have served to highlight the difficulties involved in the development and deployment of such infrastructures. Many of the concepts that have come out of research in the area of Grid computing have been successfully applied in the business arena.

Some popular definitions of Grids include:

- ‘A hardware and software infrastructure that provides dependable, consistent, and pervasive access to high end computational capabilities, despite the geographical distribution of both resources and users’[Laf].
- ‘The use of (powerful) computing resources transparently available to the user via a networked environment’ [SC92].
- ‘A hardware and software infrastructure that provides dependable, consistent, perva-

sive, and inexpensive access to high-end computational capabilities'[FK03a].

- A system that allows 'co-ordinated resource sharing and problem solving in dynamic, multi-institutional virtual organisations'[FKT01a].

Numerous architectures and classification of Grids have emerged although common to all is the objective of coordinated resource sharing, and the provision of dynamic problem solving environments involving multiple participants, often coordinated within Virtual Organisations. Some of the common classification of Grids include:

- **Computational Grids** - compute intensive operations, grand challenge problems, batch job processing
- **Data grids** - storage and sharing of large amounts of data, data mining and exploration
- **Instrument Grids** - sharing sensors and devices
- **Service Grids** - applications and multimedia, on-demand, collaborative

Key characteristics of grid systems include:

- **Scalability** - leverage growing infrastructure
- **Adaptability** - tolerate faults and transient resources
- **Flexibility**- support dynamic collections of participants and resources
- **Security** - enforce security and policy restrictions placed on component entities

The success of grid technologies to date is generally attributed to the early emergence of standard architectural models, enthusiastic early developers, adopters and evangelists, and perhaps most importantly a set of computational challenges to provide a problem for the solution and to prove its worth. They are the result of an evolution of computer architecture, made possible by advances in many areas of research.

This chapter presents a brief overview of some of the developments in computing that have contributed to the emergence of Grids, before discussing a number of significant grid projects and architectures. Currently fashionable topics such as Utility and Community computing are then discussed, before closing with a brief discussion on the evolution of management systems which has accompanied the development of distributed computing.

2.2 The Evolution of Distributed Computing

It has long been recognised that one of the most cost effective ways to satisfy our demands for increased computing power is to combine the power of multiple resources. In the late 1960s, Gene Amdahl proposed what became known as Amdahl's law [Amd00] which described the improvement in processing time that could be expected by parallelising an otherwise serially performed task. This work presented the basis and justification for multi-processor computing, although it is likely that the concept is almost as old as digital computing itself. The range of parallel computing architectures that has grown out of subsequent research is a vast topic in itself but for the predominant von Neuman architectures the results might be generally categorised in terms of the channels used for inter-processor communications, and whether they reside within or external to the computing nodes. For example, parallel architectures such as Symmetric Multiprocessors or Massively Parallel Processors typically use internal, proprietary communications busses or networks. Clusters or wide-area computing architectures usually use external commodity networks such as 1Gbps Ethernet, or near-commodity high-performance networks such as Myrinet or Infiniband. It follows that the evolution of such distributed computing architectures has been closely linked to the development, and pervasive deployment, of the communication networks on which they depend. In 1969, the ARPANET project [Rob86] succeeded in linking four different computing centres using a commodity network infrastructure. The ARPANET project is also generally accredited as being the first to develop software to take advantage of this new type of computing architecture in the form of its 'Creeper' and 'Reaper' worms. These were early examples of 'CPU scavengers', making use of idle computational power. Among the first Internet-based distributed computing projects was a factoring project undertaken by the DEC System Research Center in 1988 in response to a research challenge. This early approach relied on E-mail to distribute the data sets and applications for trivially parallel computations that would require no inter-process communication.

Although the development of an affordable, pervasive and high performance communications infrastructure was an essential precursor to the evolution of distributed computing, the linking of the hardware resources alone is not sufficient to realise a distributed computing system. Software components must also be adapted or developed to enable the unification, virtualisation and exploitation of the combined resources.

In basic terms, the creation of a distributed computing environment requires [SC92]:

- The integration of individual distributed resources into a combined networked resource.
- Middleware to provide a transparent view of the resources available.

- Optimised distributed applications to take advantage of the unified resource.

Although many of the projects developed to date differ in their objectives and aims, they all had to overcome similar problems such as communications, resource management, and security.

In addition to the technological challenges that accompany an increase in distribution and scale, the influence of social and political factors becomes increasingly pertinent. While these present the architects of distributed computing systems with a number of additional hurdles, their effect has not been entirely negative. The tendency for Governments and funding bodies to favour investment in projects closer to home has led to an increased number of distributed individual resources, and hence further justification for the development of the software components that support their distributed use and collaboration.

There follows a brief overview of a number of significant distributed computing projects.

2.3 First Generation - the birth of wide-area computing

2.3.1 Fafner

Fafner (Factoring via Network-Enabled Recursion)[FAF] was a distributed computing project developed in 1995 with the intention of using the Web to overcome some of the problems associated with factoring large numbers. It employed central web servers and CGI scripts with which users could register anonymously. Contributors downloaded worker daemons which would then use HTTP requests to GET data from and POST results back to the CGI scripts on the web servers. Multiple sites were recruited to host the necessary web servers thus forming a hierarchical network that reduced administrative and computational load. The Fafner project laid the foundations for the many web-based computing projects that would follow.

2.3.2 I-WAY

Conceived in 1995, the Information Wide Area Year (I-WAY) [FGN⁺96] [DFP⁺96] was an experimental high performance network designed to link high performance computers and advanced visualisation environments. The objective was to combine existing high-bandwidth networks using telephone systems. The testbed consisted of 17 different sites connected with networks of varying bandwidths and protocols. ATM, an emerging technology at the time, was the basis for this network.

In an effort to standardise the software interface and management, point-of-presence computers were installed at each site to act as the gateways to the I-WAY. These machines were

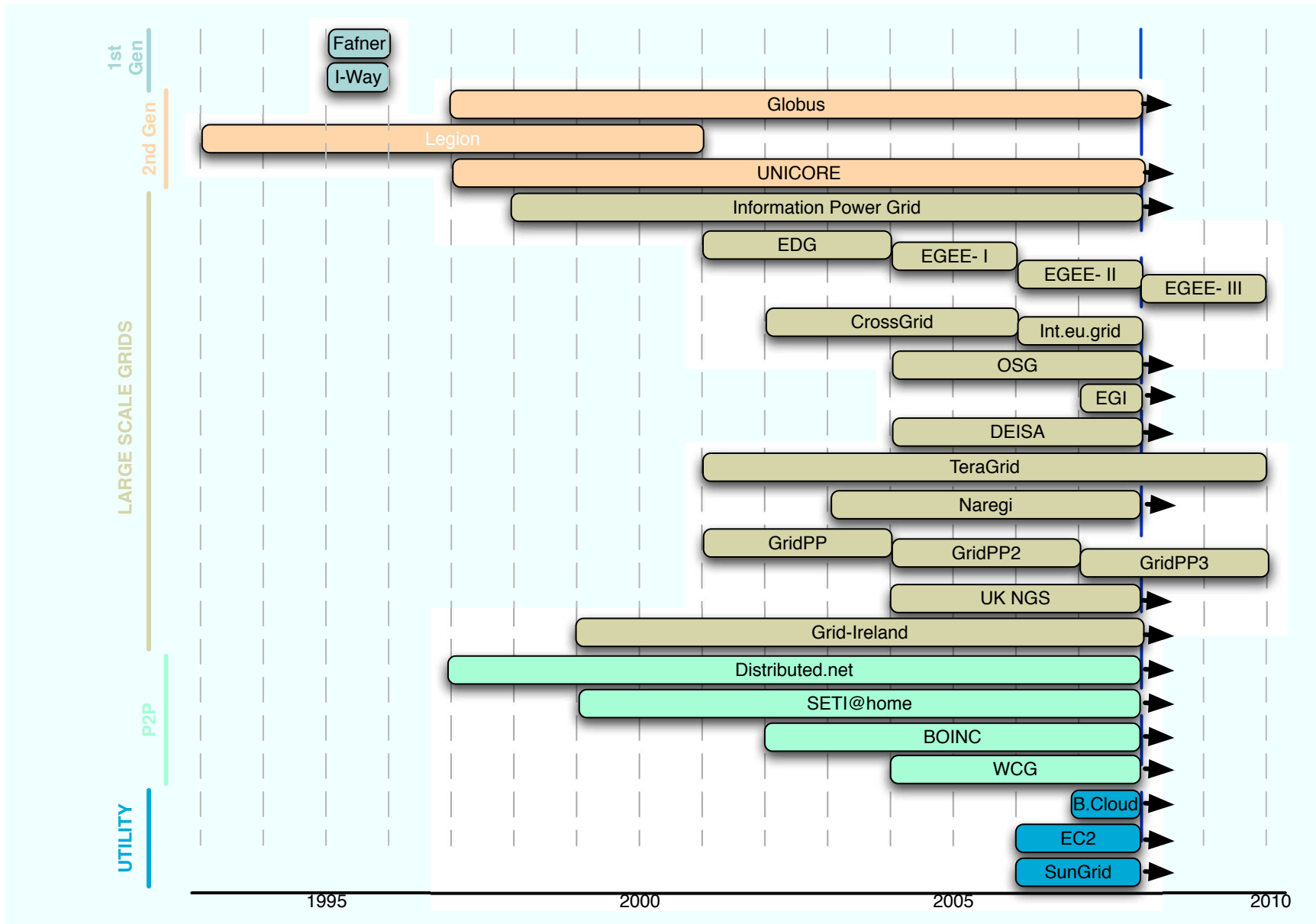


Figure 2.1: Timeline of Distributed Computing Projects discussed in Chapter 2

uniformly configured Unix workstations executing a standard software environment. This standard deployment helped overcome a number of problems including heterogeneity, scalability, security, and performance. Security was a key consideration in the I-WAY project and an effort was made to provide a common authentication environment.

I-Way was not only an operational success, which would later contribute to the Globus project, but also demonstrated how the deployment and management of wide-area distributed computing could be simplified.

2.4 Second Generation - the emergence of integrated systems

2.4.1 Globus

Globus[FK97] provides a software infrastructure that allows applications to handle distributed, heterogeneous resources as a single virtual machine. It provides a toolkit which defines the basic capabilities required to construct a computation Grid. The components of this toolkit provide fundamental services such as security, resource location, and resource management, and have well defined APIs allowing developers to tailor solutions to their individual needs.

The Globus architecture is a modular, layered one, allowing applications to employ specific features while omitting others. The services provided by the components include:

- Resource allocation and process management (GRAM)
- Authentication and security services (GSI and CAS)
- Directory service of structure and state information (MDS)
- Data Management (GridFTP and RLS, etc.)

Globus has been widely adopted and its components incorporated into a number of other projects. Its development is ongoing.

2.4.2 Legion

Developed by Andrew Grimshaw at the University of Virginia, Legion[GWT97] is an object-based meta-system providing a software infrastructure for the interaction of distributed heterogeneous computing resources. With the objective of presenting users with a single, coherent, virtual machine, the Legion system is organised into a set of classes and packages. All hardware and software resources are modelled as objects and users are free to define their own objects which can override or redefine existing ones. A number of core objects and APIs

are provided which implement the basic services required by the meta-system. Legion has been commercialised by the Avaki Corporation, now merged with Sybase [Syb].

2.4.3 UNICORE

UNICORE[Rom02] (UNiform Interface to COmputer REsources) is a technology intended to provide seamless, secure, and intuitive access to distributed computational resources such as supercomputers, cluster systems, and information stores.

Originally conceived to integrate super-computing centres across Germany, follow-up projects have sought to extend its functionality and interoperability working towards an OGF[OGF] compliant implementation. Integration of UNICORE with Globus was carried out in the EU-funded project GRIP[MW03].

Intended as an alternative to Globus, UNICORE provides its own grid toolkit and user interface portal.

2.5 Third Generation - the adoption of Web service architectures

Based on the experience gained in implementing the second generation grids, it became apparent that a primary requirement of future development would be the provision for interoperability and reuse of middleware components. It was desirable that future architectures could be engineered by assembling existing components in a flexible, extensible manner. Web services and service oriented architectures present an ideal solution to these requirements for interoperability and reuse, and their adoption is characteristic of third generation Grid projects.

The third generation also saw an increase in the importance of the users and the social groups involved. Notions such as that of the Virtual Organisation became widely accepted and greater emphasis was placed on the role played by the infrastructures in supporting science, rather than on the science of their development itself.

The Open Grid Services Architecture (OGSA)[FKNT02] announced at the Global Grid Forum in 2002 was of vital importance in promoting the adoption of service based Grid architectures. Version 1 was released in 2004 as the GGF's flagship architecture, defining a common, standard, and open architecture for Grid infrastructures. The components and services defined within the OGSA are realised using Web services in order to provide all the fundamental elements necessary for the support of e-science applications.

Due to the dynamic nature of Grid environments, the OGSA specification includes sup-

port for life-cycle management as well as support for state data associated with Grid services. The initial implementation of the OGSA architecture, the Open Grid Services Infrastructure (OGSI)[ea03] attempted to construct Grid services by extending current Web service standards to provide ‘stateful’ services. This extension to existing standards left the OGSI open to criticism and prompted a consortium of industry and research partners to design an alternative infrastructure based entirely on Web service specifications. The result was the the announcement of the WS-Resource Framework (WSRF)[Glo] in 2004. WSRF includes specifications governing resource lifetime, properties, and relationships. Also included in the specification is an event based notification system.

A number of existing projects migrated to Web service architectures over the course of their ongoing development. The Globus Toolkit versions 3(GT3) and 4 (GT4), along with UNICORE 6.0, and gLite[gLi] are examples of web service based grid middlewares.

2.6 Large-scale Grids

2.6.1 NASA Information Power Grid

Development of NASA’s Information Power Grid (IPG)[JGN99] was begun in 1998, to provide a high performance computing and data grid for use primarily by NASA scientists and engineers. It employed the Globus toolkit to provide access to heterogeneous resources distributed among several research laboratories.

The fact that the IPG was based upon the widely used Globus toolkit meant not only that NASA scientists could gain access to the computational resources from any location, but that the resources of the IPG could be used by authorised external scientists, and all via a familiar interface. In addition to data and compute services, the IPG infrastructure provides real-time access to scientific and engineering instrumentation systems.

2.6.2 European DataGrid

The European DataGrid(EDG)[Dat] project was started in January 2001, with the goal of constructing a test infrastructure to provide shared data and computing resources to the European scientific community. It involved twenty one partners from across Europe including scientific institutes and industrial partners.

Originally aimed at three core disciplines; High Energy Physics, Biology, and Earth Observation, the EDG project used Globus as it’s foundation but augmented it with the addition of several higher level services such as resource brokerage, book keeping, and data management services.

The EDG project provided European scientists with the first large-scale demonstrations of a functioning data grid and formed the basis for the LCG[lcga] infrastructure. The success of the EDG project was a critical step in the adoption of Grid computing in Europe.

2.6.3 LCG and EGEE

Designed to satisfy the computational and data-handling requirements of the experiments conducted at CERN's Large Hadron Collider, the LHC Computing Grid (LCG)[lcga] was initiated in 2003 as one of the most wide-reaching distributed computing projects to date. The project would not only integrate resources spread across Europe, America and Asia into a global virtual computing service, but also serve to further collaboration among researchers and scientists.

The LCG project developed incrementally, with the early phases operating a set of prototype services that later grew in complexity and functionality. The LCG middleware was based on that developed for the EDG project and components of the Virtual Data Toolkit[Ave].

The EGEE[egea] project started in 2004 entitled 'Enabling Grids for E-science in Europe', but was later changed to 'Enabling Grids for E-scienceE' following the addition of partners from the U.S. and Asia. Building on developments from EDG and LCG, the EGEE project initially used the second release of the LCG middleware, and later migrated to a more generic light-weight middleware developed within the project, gLite[gLi].

The EGEE-II project was commenced in April 2006 with a larger consortium of partners spanning 32 countries. Further middleware development was undertaken in EGEE-II with a focus on increasing the integration of external components and increased software testing. The EGEE project is now in phase 3 which will continue until 2010.

2.6.4 CrossGrid and Int.eu.grid

The CrossGrid[BMM⁺02] project was initiated in 2002 with the objective of developing a grid infrastructure to support a new category of applications from fields such as medicine, environmental science and physics. One of the primary aims of the CrossGrid project was to investigate and implement Grid components that would enable interactive computation, using advanced visualisation and application steering.

Grid software implemented during the course of the project included new tools for verification of source code, performance prediction, evaluation and on-line analysis. The infrastructure was also equipped with new components for monitoring of application performance, efficient distributed data access, specific resource management, as well as portals and mobile personalised user interfaces.

The work was carried out in close collaboration with the Grid Forum and the DataGrid project, in order to profit from their results and experience, and to enable interoperability. Research results were validated, tested, and demonstrated on the CrossGrid test infrastructure, where an emphasis was placed on user-friendly environments and interfaces.

Following on from the successes and expertise of the CrossGrid project, the Interactive European Grid (Int.eu.grid or I2G)[Intb] project was started on 1 May, 2006 with the objective of providing an advanced Grid-empowered production infrastructure in the European Research Area. This project is specifically oriented to support the execution of interactive demanding applications.

While interoperable with large e-Infrastructures such as EGEE thanks to common middleware services, the Interactive European Grid maintains a primary focus on powerful visualisation and interaction with demanding parallel applications that have been identified as benefiting from grid integration.

2.6.5 Open Science Grid

Built by a consortium of U.S. universities and laboratories, the Open Science Grid[Grib] is a national production-quality infrastructure for large-scale science. The basis for the Open Science Grid middleware is provided by the National Science Foundation's Middleware Initiative, incorporating Globus[FK97] and Condor[LLM88] components. The OSG middleware is packaged and supported through the Virtual Data Toolkit[Ave].

Two separate infrastructures are maintained as part of the OSG; an Integration Grid, used for testing and evaluation, and a Production Grid, which provides a stable, supported environment for sustained applications. The Operations Centre is based at Indiana University. The U.S. contribution to the LHC[CERa] experiment is being built and operated as part of the Open Science Grid.

2.6.6 The European Grid Initiative

The European Grid Initiative (EGI)[EGI] aims to protect and maximise the investment in e-infrastructures by encouraging collaboration and interoperability between national Grid infrastructures. Following considerable European investment in e-Science and the success of numerous projects, Grid technology has become recognised as a fundamental component for e-infrastructures. The EGI aims to construct a pan-European infrastructure with an emphasis on sustainability. Building on the foundation of existing National Grid Initiatives (NGI), the EGI is expected to “enable the next leap” in research infrastructures by both encouraging the linking of the existing NGIs and by supporting the development of new NGIs.

The EGI aims to coordinate middleware development and encourage standardisation by soliciting leading groups to undertake development work. It also hopes to have an advisory role in the funding of future development, based on agreed requirements and standards. The development and adoption of interface standards is obviously a high priority in facilitating the goals of the EGI. It will draw on practical experience in the areas of grid operation and middleware integration, in addition to consultation with standards organisations, in order to define its requirements in this regard.

The EGI will be expected to operate the European level of the production Grid infrastructure and to provide global services and support that complement and/or coordinate those offered by national services, e.g. VO support, user authentication, and security. Also among its objectives are the testing, integration, and packaging of existing components from leading middleware development projects and to make them widely available along with documentation and training material.

The EGI Knowledge Base encourages participants to share knowledge and experience while also being kept abreast of EGI project developments. As of September 3rd 2007, 37 National Grid Initiatives had expressed support of the EGI concept and funding for a design study has since been secured under the EU's 7th Framework Program. The EGI is expected to deliver its blueprint for future development by June 2008 and is scheduled to begin operations in early 2010.

2.6.7 DEISA and TeraGrid

DEISA (Distributed European Infrastructure for Supercomputing Applications)[DEI] is a consortium of leading national supercomputing centres that has formed a combined infrastructure to provide a persistent, production quality, distributed supercomputing environment with continental scope. The computing centres are interconnected with a high bandwidth network provided by GEANT. Research activities include middleware development in addition to supporting a wide range of scientific disciplines through the provision of enhanced HPC capabilities.

TeraGrid[Ter] is a similar venture for US supercomputing sites which aims to provide an "open scientific discovery infrastructure". Funded by the National Science Foundation, TeraGrid combines resources at 11 partner sites using high-performance network connections. Currently, Teragrid resources exceed 750 teraflops of computing capability and 30 petabytes of online and archival data storage. TeraGrid activities are coordinated by the Grid Infrastructure Group (GIG) at the University of Chicago

2.6.8 Naregi

The Japanese National Research Grid Initiative[MSA⁺05] (NAREGI) was started as a five-year project in 2003. The NAREGI project conducts research and development of the practical Grid middleware required for the operation of a robust computational research environment. In addition to the provision of the Grid middleware, the NAREGI project will conduct research on the application of high-speed network technology to the grid environment and will also conduct research and development on large-scale simulation programs for use in nano-science fields. As an example of this, the Japanese Computational Nano-science Centre at the Institute of Molecular Science is conducting research into grid-enabled nano-science and nanotechnology simulation applications.

Although a number of very interesting work packages are proposed, including programming and problem-solving environments, the research and development of the grid middleware was not started from scratch. In the initial phase, existing middleware such as UNICORE, Globus, and Condor have been used appropriately for the underlying system while research concerning the upper layers of the middleware was conducted. In the second phase, it is planned that the NAREGI middleware will evolve into the Open Grid Services Framework(OGSA) [FKNT02] .

The NAREGI Grid middleware is used as one of the software layers in the Japanese Cyber Science Infrastructure (CSI), an information technology based environment intended to boost scientific research and education activities. Other initiatives that were reorganised and incorporated into the CSI project include the university PKI and authentication system, the next-generation high-speed network project, and the academic digital contents projects. In 2006, CSI was integrated into a seven-year “Development and Application of Advanced High-performance Supercomputer” project aimed at providing Grid middleware for peta-scale computing.

2.6.9 GridPP and UK NGS

GridPP[Grid] is a UK particle physics grid constructed as a contribution to the processing requirements of the LHC projects. The project’s main objective is the construction of a large-scale e-Science infrastructure for use by the UK and international particle physics community.

GridPP is a collaboration of approximately 100 researchers from academic and research institutions including CCLRC (Council for the Central Laboratory of the Research Councils) and CERN. The aims of the project include the development of middleware and applications, and the deployment of a distributed computing infrastructure across the UK, in order to build a prototype Grid. In addition to linking resources at 17 UK sites, GridPP is connected

to other prototype Grids on an international scale. The six year project project commenced in 2001, with GridPP2, the second phase, beginning in September 2004. Phase two was intended to scale the infrastructure to the equivalent of 10,000 PCs before coming online for LHC data analysis in 2007. The project has now entered phase three, GridPP3, with funding secured until 2011.

The UK National Grid Service (NGS)[NGS] is an initiative funded by several government bodies with the goal of providing UK researchers with electronic access to the computational and data-based facilities that they require in order to carry out their research. A key function of the NGS is the training and support of users.

The resources provided by the NGS are accessed using a standard common set of services known as the Minimum Software Stack[UN]. The services provided are based on the Globus toolkit. In addition to the provision of physical resources, the NGS provides scientific software packages and libraries. The NGS currently provides services to over five hundred users and incorporates 13 sites.

2.6.10 Grid-Ireland

Grid-Ireland[CWO05] is a managed layer providing grid services to 18 sites above the Irish research network. It allows researchers to share computing and storage resources using a common interface, and facilitates international collaboration by linking Irish sites to into European grid infrastructures being developed under such EU projects as EGEE[egea], LCG[lcga], CrossGrid[BMM⁺02], and Int.EU.Grid[inta].

Grid-Ireland employs a centrally managed infrastructure, using identically configured ‘Gateways’ deployed as points of presence within each participating site. The default middleware software stack is gLite[gLi].

2.7 Peer-to-Peer and Community Computing

Peer to peer computing systems are internet applications designed to harness the resources of a large number of autonomous participants[FK03b]. The defining characteristic is generally the lack of any centralised structure and the ability of the peers to form self-organising networks, often layered above existing protocols. While both P2P networks and Grid computing are concerned with the sharing of distributed resources, the specific requirements of each has led to differing approaches, technologies, and architectures. In general, the connected resources differ in terms of architecture, scale, and connectivity.

The distributed flat architecture of many P2P systems is often regarded as a bonus leading to increased availability and fault tolerance. Many of the functionalities traditionally fulfilled

by nodes in hierarchical client-server networks are distributed among the co-operating peers in a P2P network. Described as an evolution of the traditional client-server model, peer-to-peer computing can be said to include the resources of the client into the model such that clients become servers allowing access to their own resources in return for use of the P2P service. In this case, the distinction between the clients and servers becomes less evident and what is left are peer resources both providing and consuming content or services. Applications of Peer to Peer systems include network management, content distribution, distributed computation and searching. Advantages include increased bandwidth (leveraging the accumulated bandwidth of participating resources), storage space, and computing power, in addition to robustness brought about by replication and the elimination of single points of failure.

Based on the strength and popularity of P2P networks, a number of distributed computing projects have emerged to take advantage of the architecture. Many of these community projects come under the banner of Volunteer Computing where computer owners donate their resources either for free or for community credits. One of the first such projects was the Great Internet Mersenne Prime Search[GIM] started in 1996 and was followed in 1997 by Distributed.net. Bayanihan[Sar98] , the well known JavaTM-based research project followed in 1998. In 1999, the launch of SETI@home[ACK⁺02] brought volunteer computing to the masses thanks to media coverage and popular interest. The concept has gathered pace since and there are currently a large number of active projects, some of which are described below. Several companies have been formed based in experience on volunteer computing such as Entropia and United Devices.

2.7.1 BOINC

Originally developed for the SETI@home project, the Berkeley Open Infrastructure for Network Computing (BOINC)[And04] is a non-commercial middleware system for volunteer computing. Despite its origins, BOINC is intended for use in a wide range of scientific disciplines to enable researchers to leverage the considerable computing power available from personal computers around the world.

The BOINC framework, developed at the University of California, Berkeley, is supported on a wide variety of operating systems and is available as free software under the GNU Lesser General Public License. The BOINC architecture consists of server and client components which communicate with each other in order to distribute and process the workload, and to return results.

A credit system is used to keep track of how much CPU time each volunteer has contributed. Results of calculations are validated prior to the awarding of credits in order to ensure that contributors cannot cheat by returning false results.

Some of the active projects using the BOINC framework include:

- Rosetta@home - Tests the assembly of specific proteins, using fragments of better-known proteins.
- BRaTS@Home - A gravitational lensing study.
- Climateprediction.net - Attempts to produce a forecast of the climate

A number of additional projects are currently in development.

2.7.2 World Community Grid

World Community Grid (WCG)[Gric] is an effort to create the world's largest public computing grid to perform processing for scientific research projects that benefit humanity. Launched in 2004, the WCG is funded and operated by IBM with client software currently available for a wide range of computing platforms. Many of the WCG clients currently use BOINC technology. The WCG is another example of a scavenging grid designed to utilise the idle time of connected computers from around the world. To date the project has attracted over 345,000 registered users and over 800,000 registered computers. The WCG owes its existence to the success of the Smallpox Research Grid Project, which was used to identify candidate compounds for development into smallpox treatments.

Rather than being designed specifically to work on a single project, the WCG offers participation in multiple humanitarian projects in a single package. Projects are approved for execution on the WCG by an advisory board and all users are included in all projects by default but can opt out of projects if they wish. Current research projects using the World Community Grid relate to AIDS, climate prediction, cancer research, and proteome folding

2.7.3 Distributed.net

Distributed.net[dis] is another distributed computing project that employs idle cpu time from worldwide resources for large-scale problem solving. Registered as a not-for-profit organisation and owned by Distributed Computing Technologies Inc., it was launched in 1997 and the majority of its processing has been used for encryption challenges and mathematical search algorithms. In order to contribute, users download and execute a client program called 'dnetc' which is available for a range of operating systems.

The phenomenon of Peer-to-Peer computing has had a profound and far reaching effect

on both popular and scientific computing in recent years and it likely to continue to do so for the foreseeable future. The overlap between Grid and P2P computing suggests that some degree of convergence is possible. It has been suggested[FK03b] that a combination of the two technologies, e.g. a grid services infrastructure operating above a P2P substrate, may be a solution to realising the vision of utility computing on a global scale.

2.8 Utility Computing and Clouds

Utility Computing refers to a concept wherein computing resources, either hardware or software, are accessed and utilised in an on-demand fashion in a similar way to traditionally recognised utilities such as energy supply or telecommunications. The concept presents users with a number of potential advantages including the potential for considerable cost savings and increased flexibility. Utility computing generally refers to a more dynamic system than the likes of Software as a Service (SaaS) or Hardware as a Service(HaaS).

Whereas these systems are generally statically configured as described in some contract, utility computing typically employs some form of management software to look after the physical provisioning in a more dynamic fashion. Many providers of utility services have adopted some form of virtualisation to help satisfy the requirements for reliability and scalability. The goal of the utility computing model is to maximize resource utilisation and minimize user costs

It is also possible to combine the concept of utility computing with other computing architectures, for example Peer-to-peer. In this scenario, a middle-man or broker might package and market the combined resources contributed by the peers in the network, and provide some form of remuneration based on the derived revenue.

Cloud computing is a variety of Utility computing. The analogy is based on the fact that the internet is commonly represented as a cloud where the users connected to it are relatively unaware of the scale and complexity of the system that exists beyond their browsers. Cloud computing builds on the utility model but aims to simplify deployment and management of services. In theory, a computing Cloud could encompass many different kinds of utilities,

From a business point of view, utility computing turns what was once large capital expenditure into operational costs that can be more closely compared to the revenue generated from it. Consumers can simple ‘plug-in’ to the required resources without having to invest in and manage the infrastructure. Costs of employing utility services are expected to compare favourably with the costs associated with the traditional data centre such as power, housing, cooling, and managing.

The notion of utility computing is far from new. Time-sharing of systems, which began

in the 1960s and allowed large corporations access to central mainframes for periods of time, might be considered an early precursor. It was time-sharing systems such as these that served to highlight the importance of process control and accounting that would later form key components of this business model. This was before the advent of mini-computers and PCs which would see the price of computing lowered to the point that it was affordable to the majority of businesses. Indeed it might be argued that the outsourcing of business functions such as billing to computer enabled providers in the 1960s could constitute an early example of employing computer services as a utility.

From the 1990s onwards, manufacturers of computing hardware began to show renewed interest in the field with projects from IBM, HP, and Sun among others. In more recent times, several large industry players have begun efforts to realise the opportunities offered by the utility computing model and have begun investing heavily in the technology. There follows a brief description of a number of current utility related activities

2.8.1 IBM Cloud Computing

IBM have been involved in Utility computing since the late 1990s but renewed interest in the field and the emergence of the Cloud computing concept has led to considerable recent investment of resources in the area. The IBM Blue Cloud[IBMb] was announced in late 2007 as a solution to enable and facilitate the creation of compute Clouds. Blue Cloud is a suite of components based on industry standards and open source software. It uses Red Hat's Fedora Linux, making use of its integrated Xen hypervisor in order to virtualise the hardware resources. IBM's Tivoli software is used for resource management and provision, along with DB2 and Websphere. Apache Hadoop is also included which includes Google technologies in the form of its Google File System[GGL03] and MapReduce [DG04].

In March of 2008, IBM and the Industrial Development Agency of Ireland (IDA Ireland) announced the opening of Europe's first cloud computing centre operated from its HiPODS (High Performance on Demand Solutions) lab in Dublin[IBMa]. The cloud computing centre aims to replace the traditional data centre model in which companies own and manage their individual hardware and software systems. The centre will serve as a hub for the delivery of Cloud Computing research and services to a number of facilities across Europe, the Middle East, and Africa. Experts from the IBM HiPODS (High Performance on Demand Solutions) team will work directly with customers in order to encourage and support their uptake of cloud computing solutions.

With its considerable experience in the field of Autonomic Computing, IBM is well positioned to provide cloud computing services at a successful business level by introducing increasingly self-managing behaviour to computing systems. Having been quick to recognise

the merits of cluster computing and virtualisation from the outset, Google is continuing its trend by joining forces with IBM to provide Cloud computing facilities to a number of US universities. IBM is the lead integrator for a Cloud utility that is being installed at Google's data centre in California. The hardware employed includes IBM blade and rack servers which run the Blue Cloud components. Through its own operations, Google has gained considerable experience of virtualisation and on-demand resource allocation although it has not yet begun to market this expertise or its cluster resources directly.

2.8.2 Amazon Web Services

Amazon Elastic Compute Cloud (EC2)[Amaa] is a component of the Amazon web services suite, designed to allow scalable deployment of applications by allowing users to manage the creation and termination of server instances on demand. EC2 makes extensive use of Xen virtualisation for the creation and management of the virtual servers. The service allows users to:

- Upload custom machine images installed with specific applications, libraries or data.
- Select from an existing set of pre-configured machine images.
- Control life-cycle, network configuration, security, and monitoring of the virtual servers via a web services API.

Associated with each user account are a number of 'Elastic' IP addresses which can be mapped to running instances of the virtual machines. These provide a convenient fail-over mechanism in the event of a virtual machine failure. Billing is based on the type of server instance per instance-hour and on data transfer into and out of each instance.

Amazon also offers a complimentary data storage solution, the Simple Storage Service (S3)[Amab]. S3 provides a simple standards-based (REST and SOAP) web services interface that can be used for storage and access of data on a highly scalable, reliable, fast and inexpensive data storage architecture. Users can choose whether they wish their data to be stored in the United States or in Europe and authentication mechanisms are provided to ensure data security. Billing for the service is dependent on data transfer and storage location but data transfer between Amazon EC2 and US-located S3 storage is free of charge.

2.8.3 Sun Grid

Sun Grid[Micb] is an on-demand computing service offered by Sun Microsystems. Designed to deliver enterprise computing power over the internet, Sun Grid is based upon open source

technologies including Solaris 10, Sun Grid Engine, and Java™. Users access the resources via a web portal at www.network.com or using job submission APIs. Also included is an application catalogue which provides users with business and scientific tools and libraries to be exploited by applications deployed on Sun Grid. End users and software vendors are encouraged to publish their executables in a searchable repository called the Job Catalogue.

2.8.4 Microsoft Clouds

While Microsoft believe they are in an ideal situation to satisfy utility-like customer requirements, where they would essentially rent the raw computing and data capacity, they believe that the greatest opportunities will be found through exploiting a services relationship with their existing software products. One of their most recent announcements relates to their approach to hosted database services where they will use components from their SQL Server product to build SQL Data Services[Mica].

Microsoft's services architecture consists of multiple layers:

- **Global Foundation Services** - The managed computer resources at Microsoft data centres.
- **Cloud Infrastructure Services** - The utility computing fabric supporting online services, scaling, and automatic deployment.
- **Live Platform Services** - Consumer level services such as user authentication, social networking applications, and advertising tools.

Microsoft have designed the infrastructure to support a wide range of end user devices including consumer electronics such as game consoles and media centres. Their services also offer to analyse consumer behaviour within hosted services in order to support customised content and targeted advertising.

2.8.5 3Tera

Despite being a relatively recent entrant to the field of Cloud computing, 3Tera's initial offering, AppLogic[3Te], has attracted considerable interest. AppLogic allows the definition of virtual devices that system architects can then quickly assemble into computing solutions via a simple browser interface. Figure 2.2 shows a screen shot of AppLogic being used to construct a web server cluster. Once the bundle of services is defined, it is easily deployed and started on the underlying resources. This provides an example of the simplicity that virtualisation and the utility model can bring to the deployment and management of computing resources.

Although 3Tera do offer hosting facilities, it seems their primary business intent is to provide tools and solutions to the providers of utility services so that they can easily provision and deploy scalable clustered applications.

It is reasonable to assume that some of the advantages offered by the utility model to commercial customers may also prove attractive to research and scientific users. It is interesting to consider what effect the emergence of the utility model might have on the utilisation and viability of National Grid Infrastructures in this regard. It could be argued that most of the services provided by NGIs are made available to researchers free of charge, but this is ultimately in the hands of the funding parties who could, at some point in the future, decide that it would be more cost effective to allocate the relevant funds to the consumers of computing services rather than the producers (as is already the case in Denmark). If this holds true, the directors and operators of NGIs should perhaps be quick to market their existing infrastructures as ideal platforms and delivery channels for utility computing services, or possibly cease their provision of compute and data infrastructure and focus on access gateways and central services (Operations Centres) that enable access to utility computing resources. Traditional grid jobs may have to co-exist with user-defined virtual machines submitted for execution, or at least requests for their instantiation.

All of the utility computing projects discussed here are being implemented as commercial ventures involving financial transactions and therefore, point towards an economic market for 'grid' services. Any such commercialisation or pressures to provide return on investment will highlight activities such as accounting and billing as being of crucial importance. Infrastructures with capable economic systems such as DGAS[INF], SGAS[SGE⁺04] and SGA [PKC05] [PLO⁺05a], will be to the fore.

2.9 HPC Eco-Systems

Despite finite budgets and resources, the quest for more powerful computer systems is ongoing. At a time when success in science and industry is often linked to the availability of computing power, there is growing impetus to stay at the forefront of technology and competition. In addition, those who have already invested in resources or contributed to existing projects desire and are perhaps under pressure to achieve maximum possible utilisation and efficiency.

There is currently considerable interest in efforts to increase competitiveness and further collaboration through the linking of Grid and cluster environments at the international level.

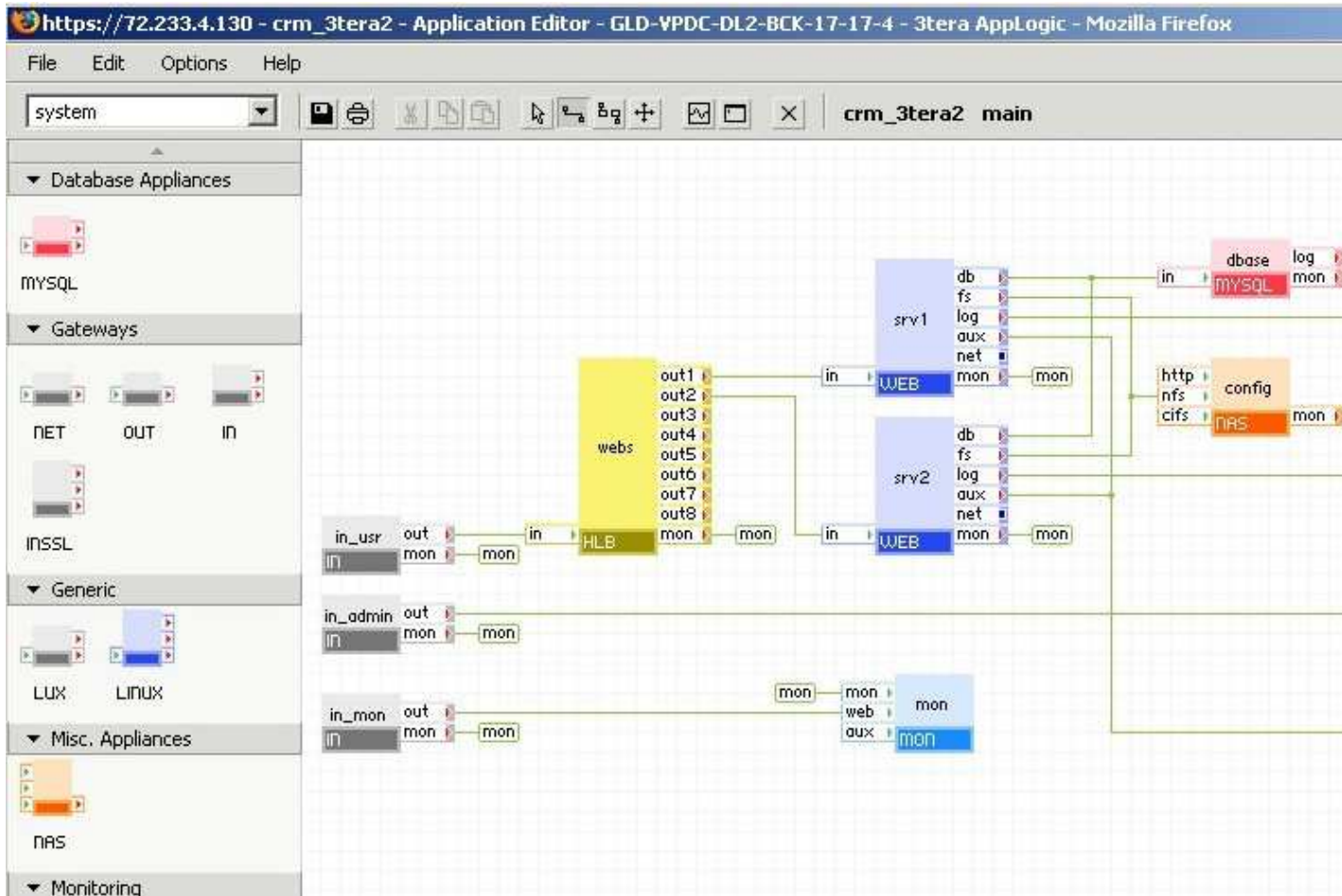


Figure 2.2: 3Tera AppLogic user interface

In Europe[Kos07], the High-Performance Computing in Europe Task force (HET)[HET] has devised a strategy for the development of a European HPC “Eco-system” (an overloading of this term) with the objective of enabling compute centres with capabilities beyond one petaflop. It is likely that despite the strategy and planning, these “Eco-systems” will be seen to evolve in a manner similar to their biological counterparts.

The future demand for these infrastructures has been highlighted by the findings of the European Strategy Forum for Research Infrastructures (ESRFI)[Cor] which has identified thirty five new experiments as requiring advanced data-management capabilities and high end computing. In response, the Partnership for Advanced Computing in Europe (PRACE, an evolution of HET)[PRAa], started on January 1st 2008, is a consortium of partners from across Europe which will provide researchers with access to world class supercomputers. The project will last for two years, and has a budget of twenty million euro. Following the development of prototypes for petaflop computing in year one, the construction of the computing centres will begin in year two. It is anticipated that integration and interoperability between PRACE and other European Grid projects such as DEISA and EGEE will be necessary.

2.10 Evolution of Distributed Infrastructure Management

As the architectures and complexities of the computing infrastructures described in the previous section have advanced, so too have the requirements for the management of their components, and of the systems as a whole. Such large scale infrastructures pose specific management problems that cannot easily be met with conventional network management tools and systems. While this topic will be further explored in chapter 4, let us briefly examine the management requirements and the developments in the area to date.

The demand for scientific computing resources has led to the accelerated growth of distributed computing systems. The increased complexity associated with the distribution of both components and users of these systems, coupled with increased user expectations with regard to performance and security, etc., further necessitates efficient and effective systems management. All of these projects share a common requirement for the efficient management of both the social and technological aspects of their composition and operation.

Users of Utility or Cloud computing will have high expectations about their levels of services and expect the infrastructures to be managed just as effectively as the many other utilities on which business, communities and individuals depend. Until autonomic systems prove to be reliable and trustworthy, there will continue to be non-trivial levels of human administration and management of the resources required.

Irrespective of the chosen tools and architectures, the basic concepts and aims of system

management do not change. These aims include:

- Software deployment, upgrade, and configuration
- Software and hardware inventory (component licensing)
- Security management and auditing
- Network fabric monitoring and management
- User administration
- System Availability maximisation
- Service Level Agreement monitoring and enforcement
- Operational cost minimisation and ROI maximisation

With continued trends towards functional and geographical distribution, along with increasing emphasis on interoperability, the development of standards-based management architectures will become increasingly important.

Chapter 3

Distributed Infrastructure Monitoring

3.1 Introduction

Grid infrastructures, like many other complex systems, are subject to change. These changes may be intentional, such as the addition of new resources, or accidental such as a hardware failure leading to a service outage. Regardless of the cause of change, there are many reasons to observe and record the structure and status of infrastructure's components. Users, operators, and middleware components themselves, all have specific requirements for information about the system. Information services are therefore critical components of distributed computing systems. Information and monitoring systems provide essential data relating to the architecture and resources of the Grid and are fundamental to the operation of many middleware components such as schedulers and brokers. In addition to static information which may be gleaned from configuration files, information systems must also perform some degree of monitoring, or rely on other components to keep them updated, in order to ensure that the information they provide is not stale.

The effective management and successful operation of such complex infrastructures in terms of job scheduling, performance analysis, optimisation, and data replication is heavily dependent on the level of awareness provided by monitoring systems [BKPV01]. Recording resource utilisation is not only a requirement for accounting but can also provide valuable insight for infrastructure optimisation or future resource planning. Resource discovery and monitoring are crucial activities in the efforts to ensure that the needs and expectation of users are met.

From an operational point of view, information is power, and efficient control cannot

be exercised without it. A combination of situational awareness and control capabilities is required if management is to take place. This chapter provides an overview of the information and monitoring requirements of distributed computing architectures and presents a number of tools and approaches that have been developed to help fulfil those requirements.

3.2 Information Requirements and Models

Many of the event types that are of interest to the operators of compute resources are common in ‘traditional’ host and network monitoring systems. When monitoring Cluster systems, these base metrics are augmented with information relating to batch systems, utilisation, and accounting. In the field of Grid monitoring, the information requirements are expanded further still to encompass site performance and availability, monitoring of site interconnects, Grid service monitoring, and operating platform or middleware versions etc. Information relating to the membership and activities of Virtual Organisations is also of interest. In centrally managed infrastructures, environmental monitoring information such as power supply and temperature might also be monitored. The field of security monitoring, a broad area unto itself can range from physical access monitoring at enclosure level to Grid-wide correlated intrusion detection.

Common terminology has been proposed for the description of the components of an information system[GGF]:

- **Event** - source of information
- **Metric / Property** - result of a measurement or event
- **Target** - the entity under measurement

Some typical examples of event types commonly monitored include:

- CPU Load
- System Uptime and load
- Disk size and utilisation
- Memory size and utilisation
- Available bandwidth
- Software or service status

- Network latency and packet loss
- Batch queue information
- Host architecture and OS
- Middleware revision
- Storage system status and availability
- Environmental Monitoring

Each instance of an event is attributed to a target, i.e. a monitored entity. These targets can be represented as a hierarchy with the most general entity at the top, and extending down to the most basic physical or logical components such as network interfaces and software processes. Examples of targets include Grids, sites, hosts, network links, services, users, jobs, batch queues, and instruments.

Many Grid communities have developed their own specific means of representing pertinent information about the status of resources and infrastructures leading to islands of information and barriers to interoperability. More recently, uniform schemas have been developed in order to allow the sharing of information across current Grid boundaries. Information must be structured based on a common or self describing schema using shared semantics. A basic requirement of such a model is the definition of common identifiers and units for the values being communicated. One approach to information modelling is to model the components and measurement methods as a hierarchy, which allows components to use the lowest level of the hierarchy that they can recognise[GGF]. Using this approach, information need not be lost due to simplification or aggregation

One of the main factors driving the adoption of common mechanisms for the representation of data has been the inclusion of schemas in grid middleware packages such as the Globus toolkit and gLite. Resources must be described in a precise and consistent manner if they are to be discovered for use. It is interesting to observe that although the relationships between abstractions often varies between schemas, in general, the information communicated about them does not. The importance of standard representations for the representation of grid information cannot be over emphasised. Information systems are fundamental to the operations of a computing infrastructure. With growing interest in security, accounting, levels of service, as Grids become more highly utilised and adapted to business environments, their importance will become even greater still. This section describes a number of schemas in use in grid infrastructures.

Table 3.1: Selection of CE Attributes from the GLUE schema

ObjectClass	Attributes
GlueCE	Service ID and name
GlueCEInfo	Batch name + version, gatekeeper host + manager, number of CPUs Application and data directories, default SE etc.
GlueCEState	Queue and jobs statistics, response times etc.
GlueCEPolicy	Limits on time and number of jobs
GlueCEAccessControlBase	Vo restrictions
GlueHostArchitecture	Platform description, CPUs in a SMP node
GlueHostApplicationSoftware	List of installed software.
GlueHostMainMemory	RAM and virtual memory size
GlueHostBenchmark	SpecInt2000 and SpecFloat2000 benchmarks.
GlueHostStorageDevice	Host storage system name, type, transfer rate.

3.2.1 The GLUE Schema

Developed as part of the DataTag[MMFM⁺05] project as a means to allow interoperability between American and European Grid resources, the Grid Laboratory Uniform Environment(GLUE)[ASV03] schema has since seen widespread use due to its adoption by the Globus project for use in MDS, see Section 3.4.1 [LST97][Fit01]. The GLUE schema provides a conceptual model of Grid resources for their description within a Grid Information Service. The model can be used for both discovery and monitoring purposes.

Information systems based on the LDAP schema generally employ LDAP, but an XML mapping of the schema was developed for Globus toolkit version 3(GT3). Within the schema, information is organised in to objectclasses, each of which can contain zero or more attributes. By way of example, some of the attributes used by the schema to represent a Compute Element are described in Table 3.1. The schema supports information for compute elements and storage resources, as well as modelling relationships between them.



An extended version of the GLUE schema was developed for use in the European DataGrid project. The extensions included additional host monitoring metrics. The UML representation of the schema illustrated in Figure 3.1 shows the EDG extensions in highlighted in red. Maintenance of the GLUE schema is now an activity in the Open Grid Forum.

3.2.2 The OGF NM-WG Network Measurement Schema

Recognising the fact that the performance of most grid applications will be dependent on the performance of the underlying networks, the Network Measurement Working Group (NM-

Title : Glue Schema 1.0 (UML diagram)
 Filename : Glue_1_0_Ext.vsd
 Author : Sergio Androozzi
 Date : 09/01/2003
 Namespace: Glue/EDT
 Status : DRAFT

Host element -
 DataTAG extensions
 for the monitoring service

-  Inheritance
-  Association
-  Aggregation
-  Equivalent to: 0..n

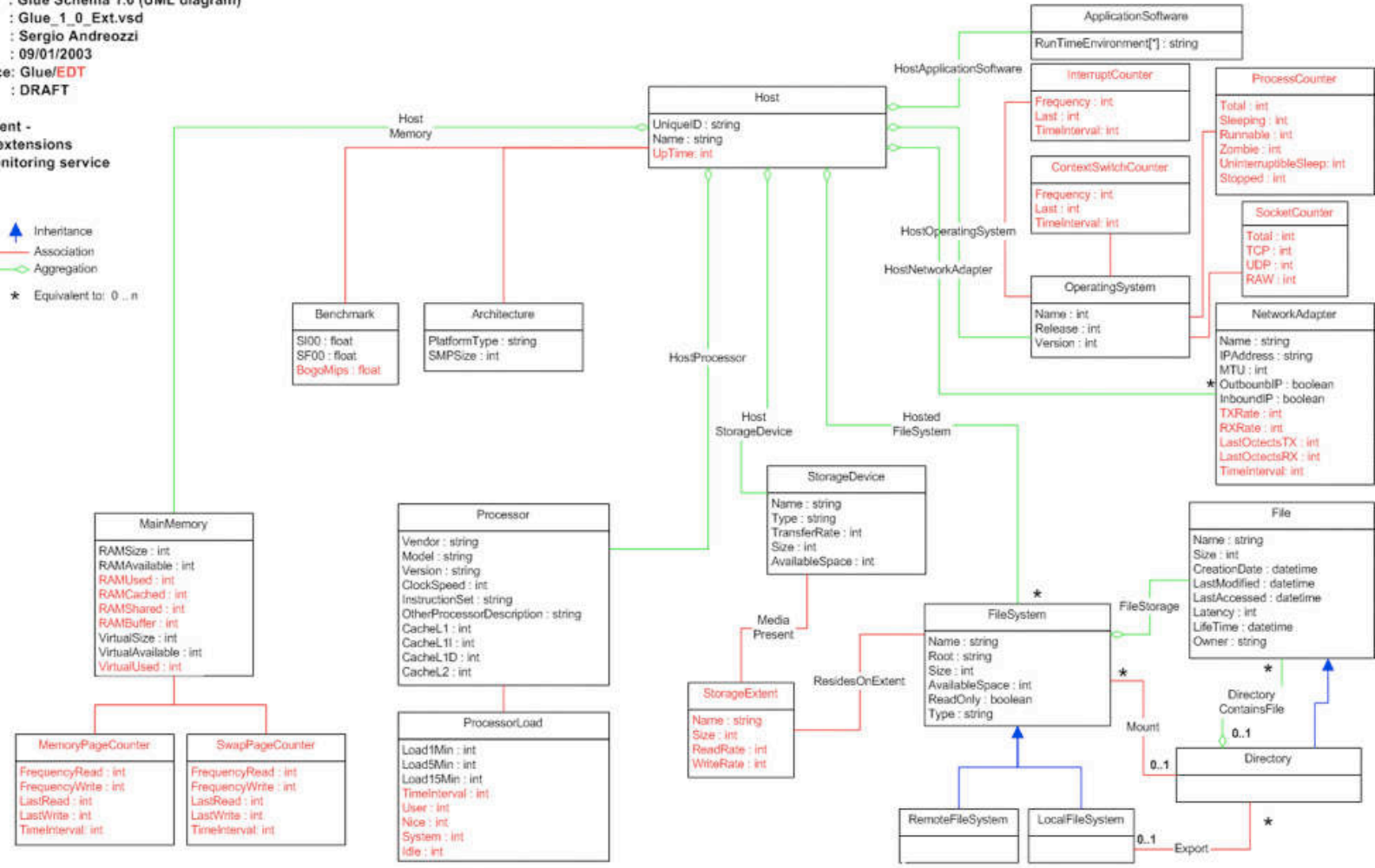


Figure 3.1: UML illustration of GLUE schema

WG)[Fora] from the Open Grid Forum is tasked with identifying network metrics that are of use to grid applications and middleware. They are also tasked with developing standard mechanisms for the the representation and communications of these metrics. Currently on version 2.0, the group's network measurement schema is an extensible XML-based encoding standard for network measurement and performance data. In addition to it's application in grid computing, the schema is expected to prove valuable in the areas of federated network management, and multidomain dynamic provisioning of network circuits.

3.2.3 The GOCDB Schema

The LCG Grid Operations Centre Database (GOCDB)[EGEd] is a central repository of general information about participating EGEE/LCG/UK-NGS sites. It is operated by the EGEE UK-Ireland Regional Operations Centre. Information stored within the database includes details of the resources at the sites, site locations and contact information, and details of any scheduled down-time when the site or resources is not expected to be available. The schema used by the GOCDB is shown in Figure 3.2.

The role of the GOCDB is discussed in greater detail in Section 4.5.1.

3.2.4 Common Information Model

CIM [Ful05] is an object oriented standard from the Distributed Management Task Force (DMTF) for the description of management system information models in a platform independent way. It's objective is to provide a common definition for the components of a computing environment including: servers, networks, applications, services, and management information systems themselves. One of the key motivations of CIM is interoperability. By defining of standard means of expression, management information system may be shared between disparate systems throughout the network. The standard is composed of a Schema, which provides the model descriptions, and a Specification, which describes it's integration with external management models.

More recent additions to the schema include support for management profiles, security, and virtualisation. The CIM-based Grid Schema Working Group is working towards a Grid schema based on CIM. At the time of writing, the most recent version of the CIM schema was released in January 2008.

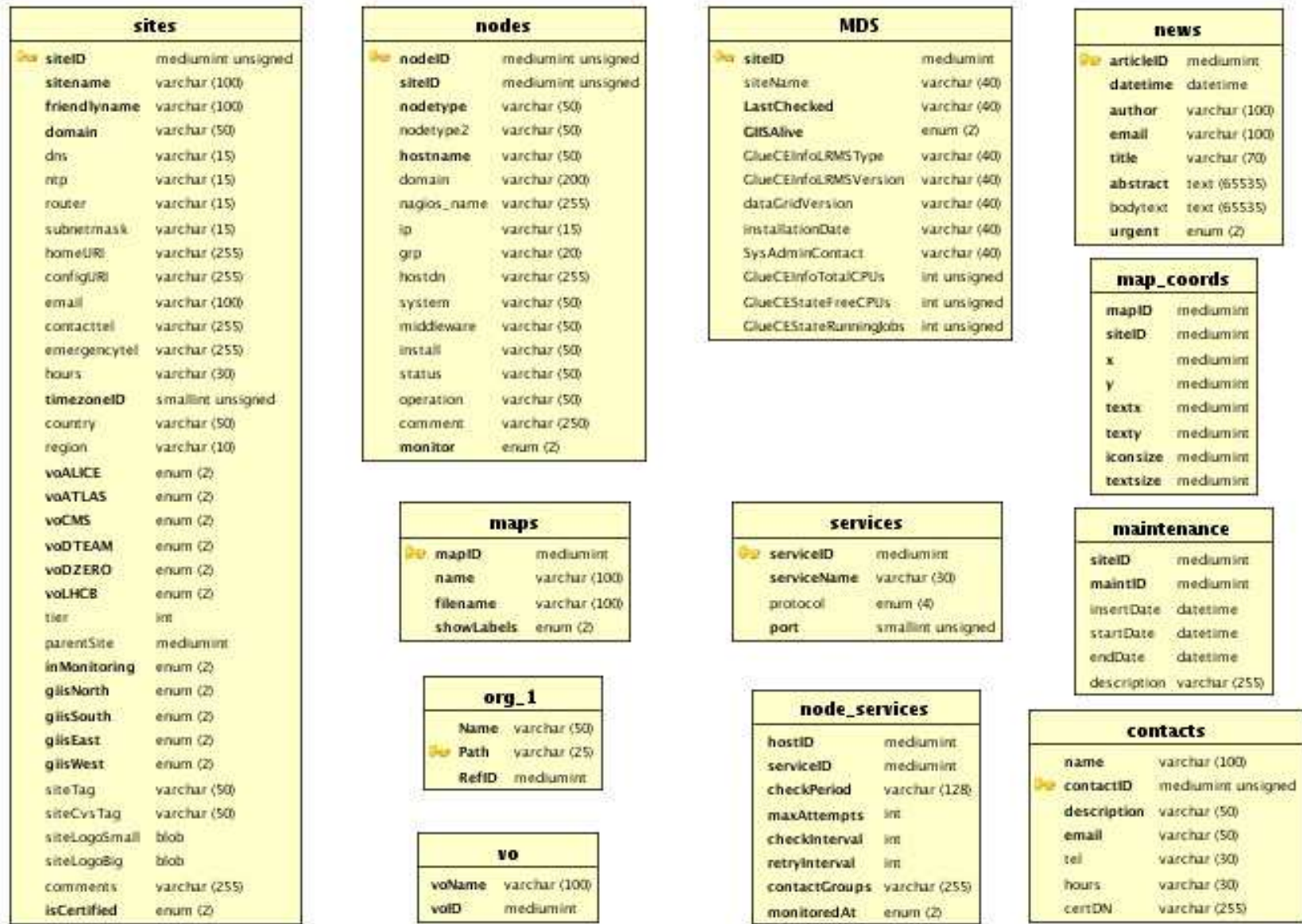


Figure 3.2: A section of the LCG Grid Operations Centre Database schema.

3.3 Cluster and Network Monitoring Tools

The monitoring requirements of grids are considerably more complex than those of clusters or local area resources. However similarities exist and many monitoring systems designed for deployment within a single administrative domain can play an active role in the overall monitoring architecture of a grid infrastructure. For this reason, this section provides an overview of monitoring tools commonly deployed within grid sites. In addition to providing monitoring services for the for the resources at the given site, these tools have the potential to act as information providers

3.3.1 Ganglia

Ganglia[Gan] is scalable distributed monitoring system based on a hierarchical design. The system is comprised of two daemons, a PHP Web front end and several utility programs. Inter-component communication relies on a multicast based listen/announce protocol and for this reason, Ganglia is generally more suited to monitoring cluster nodes than wide-area resources.

The two primary components of the architecture are the Ganglia Monitoring Daemon (gmond) and the Ganglia Meta Daemon (gmetad). XML is used for data representation and XDR for Data transport.

The multithreaded monitoring daemons are executed on each cluster node with the following responsibilities:

- Monitor changes in host state
- Multicast relevant changes
- Listen to broadcasts of other nodes
- Answer requests for XML description of node state

The meta daemons form nodes in the hierarchical tree and poll both monitoring daemons and meta daemons at regular intervals. Metrics are parsed from the collected XML and saved to a round robin database. The aggregated XML is exported over TCP.

Ganglia has become a mature and popular choice for host monitoring. Among it's high profile deployments are it's inclusion in the Rocks distribution from the Grids and Cluster Group at San Diego Supercomputer Centre and its use in the PlanetLab project[TCL⁺07] which linked over one hundred institutions around the world. The PlanetLab deployment clearly demonstrates that Ganglia is capable satisfying monitoring requirements on an international scale.

While there is no direct API support for the republishing of information in Ganglia, the use of XML for data exchange makes it amenable to exploitation by external software components.

3.3.2 Lemon

Developed at CERN as part of the LHC project, Lemon (Lhc Era MONitoring) [CERb] is a distributed monitoring system scalable to upwards of 10K nodes. The Lemon sensors collect an average of 70 metrics per host and the framework is extensible through the creation of custom sensors. The system provides persistent storage for the monitoring data and may be configured to trigger corrective actions and send notifications. A web-based GUI allows for visualisation of data from the monitored hosts which may be grouped into virtual clusters.

The architecture of the lemon system employs the following components:

- Monitoring Sensor Agents - Manages the Monitoring Sensors
- Monitoring Sensors - Collect monitoring metrics
- Monitoring Repository - Storage of monitoring information
- Lemon RRD Framework - Cache data for web graphics
- Lemon Alarm Gateway - A generic gateway for alarms allowing them to be fed into arbitrary systems.

On each lemon monitored machine, a Lemon agent (an evolution of the EDG fmon agent) gathers data from one or more sensors. The agent is responsible for the launching and configuration of sensors, scheduling measurements, and the collection of samples. Sensors can send monitoring information either synchronously, in response to a request, or asynchronously. Asynchronous measurements are intended for use where propagation of events is desirable as they occur rather than storing them for collection by a subsequently triggered sampling event. Communications between the sensors and agent uses a plain-text protocol. The architecture is extensible through the creation of custom sensors which can then be deployed and registered with the Lemon agents. An interesting capability of the Lemon framework is its ability to bind a metric ID to a complex datatype such as an array or matrix of values. Lemon is used for GridICE and can provide data to MONAlisa [NLG⁺03].

3.3.3 Nagios

Nagios is an open source host and service monitoring application designed to alert administrators to the presence of problems and outages. The Nagios monitoring daemon uses plugins

to periodically check the status of the monitored resources. Plugins, typically compiled executables or scripts (Perl or shell scripts), return the status information to Nagios which then updates its record for the resource. If a problem is detected for a given resource, the system may be configured to notify one or more groups of interested parties using a range of mechanisms including email, messaging, and SMS.

The system can be easily customised by the creation of specific plugins since there is a well defined interface between the plugins and the monitoring daemon. The current status of monitored resources along with historical logs may be viewed via the Nagios web interface. The web interface supports a variety of display formats including customisable topological displays. Some of the features of Nagios include:

- Monitoring of services (e.g. SMTP, HTTP, PING, etc)
- Monitoring of host resources (e.g. Load, disk usage, etc)
- Environmental monitoring
- Plugin sensor mechanism
- Supports network hierarchy
- User defined event handlers and notifications supporting escalation
- Uses flat files for configuration and storage
- Ability to define event handlers to be run during service or host events for proactive problem resolution

Created by Ethan Galstad, and originally known as NetSaint, Nagios has attracted a multitude of users and developers due to its open nature. A wide variety of community-developed plugins are also available. Nagios forms an integral part of the I4C[RCW06a] monitoring system and is discussed in more detail in Chapter 6.

3.4 Grid Information Systems

3.4.1 MDS

The Globus Monitoring and Discovery Service (MDS)[LST97][Fit01] is the information infrastructure provided by the Globus toolkit. The MDS provides information about the resources available on the Grid and their current status. MDS was formerly known as the Metacomputing Directory Service.

The architecture of MDS has undergone constant revision in line with the releases of the Globus toolkit but MDS2 (the pre-Web services implementation) is included alongside the later releases in each of the toolkits. An overview of the releases is presented in Table 3.2. In general, the MDS architecture consists of information providers (sensors), Information Services (producers), and Index Services (republishers)

Additional information providers may be installed to provide information such as details of accepted certificates, or to advertise software versioning information.

3.4.2 gLite/EGEE Information System

The gLite information system deployed within the EGEE infrastructure employs a 3-tier architecture based on the the Berkley Database Information Index. An overview of the information system is shown in Figure 3.3. The architecture is based on that of the MDS2 and many similarities exist. A resource-level BDII is usually co-located with the service about which it provides information. A site-level BDII then aggregates the information from all the resource-level BDII at that site. At the top level of the hierarchy, a BDII aggregates information from all site-level BDII's and therefore contains information about all grid services. It is the top-level BDII's that are queried when consumers require information. The information stored within each BDII conforms with the GLUE schema as described in Section 3.2.1. Multiple instances of the top-level database are maintained in order to eliminate any single point of failure and ensure a load balanced service.

As with MDS2, the databases are populated by executing information providers. These obtain the required information and format it as LDIF, the LDAP Data Interchange Format, so that it may be easily inserted into the BDII. In addition to acquiring information from monitored services, the information providers may query BDII's from lower levels in the hierarchy. In order to simplify the creation and deployment of information providers, a Generic Information Provider (GIP) framework exists.

In order to allow managers of Virtual Organisation to influence the usage of specific services by their VO's, an additional component called the Freedom of Choice for Resources (FCR) may be used to white list or black list sites. This information is then used by the top-level BDII's to configure ACL's for the VO on specific services from the database, thereby ensuring that results from those services are not returned when servicing queries from the given VO.

Although the BDII shows improved performance when compared to OpenLDAP (as used in MDS and frequently criticised in relation to the performance of update operations) it remains problematic and recommendations for its improvement are being investigated[AFRZ07].

Table 3.2: Globus MDS Version Information

Release	Toolkit	Architecture	Description
MDS4	GT4	WSRF	Much of the MDS functionality implemented as WSRF compliant services. An Index Service collects data from various sources and provides a query/response interface to that data. A Trigger Service can be configured to take action based on collected data. Both the Index and Trigger services are built upon the Aggregator Framework.
MDS3	GT3.x	OGSI	Implemented as OGSA compliant web services in order to facilitate interoperability among heterogeneous systems. GIIS is replaced with an Index Service which performs aggregation and indexing of service data. Hierarchies of Index Server may be constructed similarly to the GIIS in MDS2. Use of the GLUE schema is preserved but representation of information uses XML enabling XPath queries.
MDS2	GT2.x	LDAP	Employs Grid Resource Information Services (GRIS) as producers and Grid Index Information Services (GIIS) as republishers. Both are implemented as back ends to the LDAP server. Based on two core protocols; Grid Information Protocol (GRIP) - used for query/response, and search operations, and Grid Registration Protocol (GRRP) - used to maintain registrations between components. The Lightweight Directory Access Protocol is used for data representation, with entities being represented as LDAP objects defined with key-value pairs. These objects are organised into the hierarchical structure of LDAP known as the Directory Information Tree. Takes advantage of LDAP's referral capability in which queries are forwarded to authoritative providers

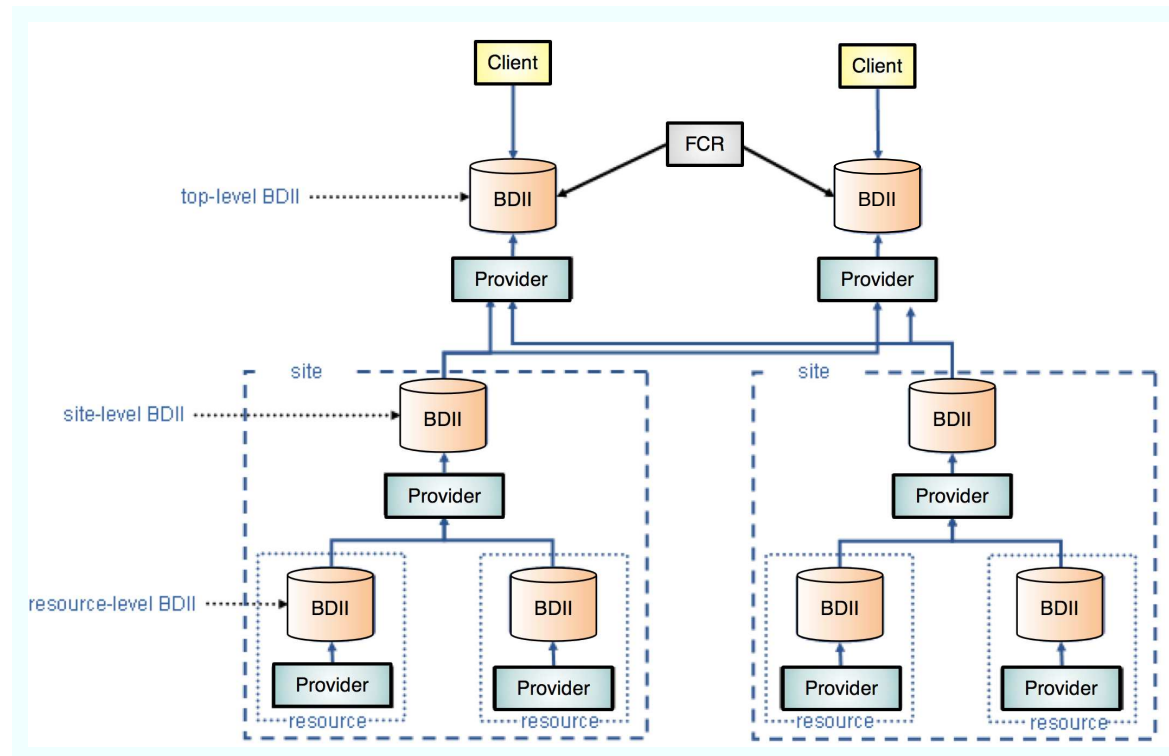


Figure 3.3: gLite information system architecture. [EGEc]

3.4.3 R-GMA

While R-GMA[BCC⁺03] is described in greater detail in Section 3.5.3 , the architecture deserves mention here due to its flexibility and the existence of two additional components which were developed to facilitate its deployment as an information system. GIN is a small tool which can be configured to invoke the MDS or BDII resource info-providers and publish the resulting information into R-GMA. GOUT is a complementary component which is capable of publishing R-GMA data into an LDAP (like MDS or BDII) database for the benefit of legacy applications

3.5 Grid Monitoring Systems

Grid monitoring may be described as the activity of measuring significant grid related resource parameters in order to analyse usage, availability, behaviour and performance[ADBF⁺03]. Its objectives include:

- Locating performance problems
- Tuning for better performance
- Fault detection and recovery
- Detection of contract or policy violations
- Input to prediction services

In general, the objective is the gathering of sufficient amounts of information such that the characteristics or status of the underlying system can be communicated and reasoned upon in an efficient and appropriate manner.

The size and nature of grid systems means that monitoring systems must be capable of measuring a wide range of metrics relating to a substantial number of entities distributed across a large geographical area. Monitoring activities present those responsible for the maintenance of the infrastructures with considerable challenges and this has become an active area of research in its own right[ZS05a]. Traditional network and host monitoring tools often lack the scalability, dynamicity, or extensibility required for effective deployment within computational or data grids.

Some of the fundamental requirements of a grid monitoring system include:

- Scalable - efficiently accommodate an increasing number of monitored entities

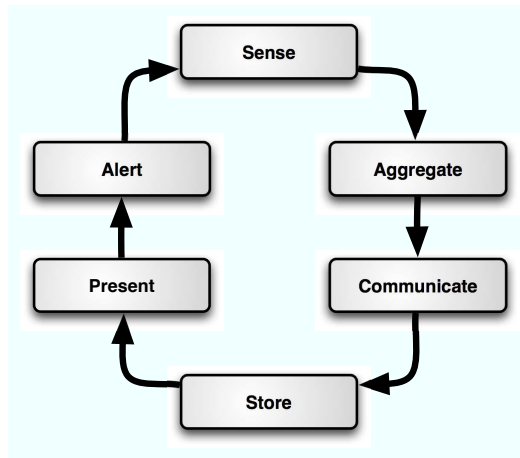


Figure 3.4: The Monitoring Cycle

- Un-intrusive - have minimal effect on the observed entities, minimise back-action
- Support time-sensitive data
- Provide efficient delivery, propagation, archival, and presentation of information
- Employ a standard schema or data format
- Ensure security of data.
- Support subscription to topic or entity related notifications.

A more detailed discussion of grid monitoring requirements is presented in section 5.2.

3.5.1 A Grid Monitoring Architecture

The GGF Grid Monitoring Architecture (GMA)[TAG⁺02] was developed by the GGF Performance Working Group with the goal of providing a minimal specification for a monitoring solution that would support required functionality and interoperability. The GMA outlines many of the fundamental concepts and requirements of a grid monitoring solution, and identifies the primary components and interactions.

Design considerations are presented which outline some of the distinguishing characteristics of the information that is handled by grid monitoring systems. The designers of the GMA assert that the lifetime of utility of monitoring information is generally relatively short suggesting that while rapid access to the information is important, the archival of such information is of secondary importance, unless it is to be used for purposes such as accounting or

historical analysis. The GMA also recognises that updates of monitoring information will be frequent, with information often being updated more frequently than it is read, and identifies this as one of the reasons why typical information-base technologies are unsuitable for such dynamic information storage.

The required characteristics of a grid monitoring system identified within the GMA include:

- **Low Latency.** Since the duration of the information's usefulness is relatively short, it must be communicated from where it is measured to where it is needed as quickly as possible.
- **High data rate.** The monitoring system must be capable of handling high data rates since information may be generated with high frequency from multiple sources.
- **Minimal measurement overhead.** The collection of measurements should be un-intrusive and have minimal impact on the measured entity.
- **Secure.** Access to monitoring information and control of the monitoring system must be restricted to authorised users. Tools must observe access restrictions or policies imposed by the owners of the information or the sensors.
- **Scalable.** Due to the very large numbers of monitored entities and sinks for monitoring information, the system must be capable of scaling to meet the requirement of large infrastructures.

The GMA also recommends that discovery mechanisms are separated from data transfer in order to ensure a precise level of control over the overhead and latency of data gathering operations. This is achieved by abstracting metadata and storing it in a separate universally accessible repository, referred to as a 'directory service'. The specification suggests that the architecture should allow the directory service to be distributed in order to increase scalability. Three models of producer/consumer interaction are supported by the GMA architecture, publish/subscribe, query/response, and notification.

The components of the GMA are illustrated in Figure 3.5 and include:

- **Producers.** Publishers of monitoring and performance information (event sources).
- **Consumers.** Receivers of information (event sinks).
- **Directory Service / Registry.** Supports discovery of producers and consumers, and publication of information. Producers and consumers publish their existence as

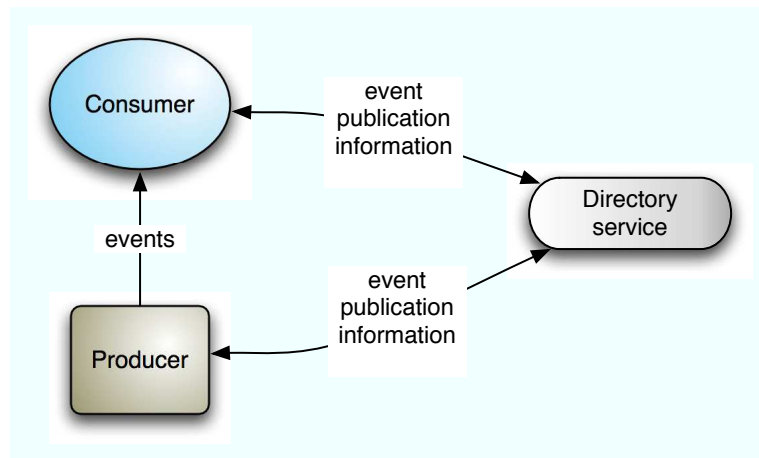


Figure 3.5: Grid Monitoring Architecture components

directory service entries, so that the directory can be used for consumers to locate producers of the required information, and for producers to find interested consumers.

The GMA architecture also allows for the combining of both producers and consumers into intermediate logical entities for the purposes of forwarding, broadcasting, filtering/analysis, and caching. These compound producer/consumers are created as single components that implement both consumer and producer interfaces, as illustrated in Figure 3.6. As an example, consider a consumer interface that might collect information from multiple producers before reasoning upon the information and publishing new derived event data.

3.5.2 PyGMA

PyGMA is an implementation of the Grid Monitoring Architecture written in the Python language. Developed at the Lawrence Berkeley National Laboratory, PyGMA provides a SOAP framework for the communications between GMA components. In order to build a monitoring system based on PyGMA, the developer must implement the monitoring components to acquire the data. The PyGMA architecture employs two types of communications endpoints; servers, endpoints responsible for servicing requests, and proxies, which provide local representations of the interfaces of the servers and handle any object marshalling that is required.

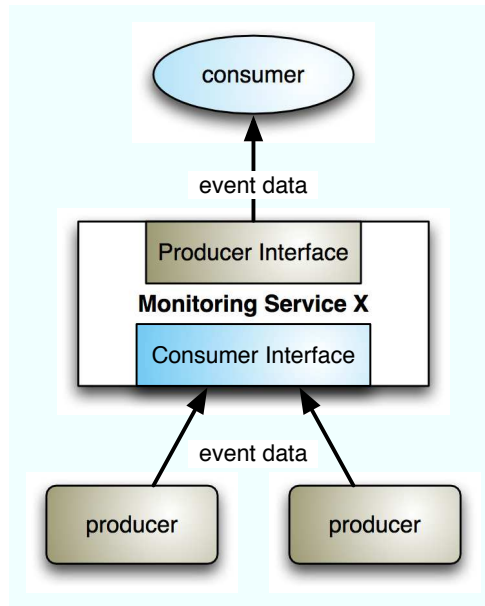


Figure 3.6: A GMA compound Producer/Consumer

3.5.3 The Relational Grid Monitoring Architecture

Developed within the European DataGrid project, the Relational Grid Monitoring Architecture (R-GMA) was designed to be an innovative grid information and monitoring system based on a powerful data model. R-GMA is based on, and compatible with the GGF GMA described above and borrows much of the terminology. One of the driving forces behind the development of R-GMA was the failure of existing LDAP based solutions (such as MDS) to satisfy the requirements of the EDG project. Some of the identified shortfalls of the LDAP solutions included their unsuitability for application monitoring and the lack of support for the streaming of data.

R-GMA applies a relational implementation to both information and monitoring. The designers of R-GMA propose that the single distinguishing characteristic of monitoring information is that it carries a time stamp and consequently all records carry a time stamp which defaults to the current time. For the purposes of inserting and retrieving information, the system creates the impression of one RDBMS per Virtual Organisation. It should be emphasised that while an RDBMS can form a component of an R-GMA system (e.g. when coupled with the relevant type of producer or the registry), the designers have not attempted to build a distributed DBMS system. Rather, they have created a way to use the relational model in a distributed environment.

Producers of information are announced to a registry which Consumers consult in order to find suitable sources of information. Once a suitable producer is found, the consumer will connect directly to it either to transfer data for a single request or to have data streamed to them. An overview of the architecture can be seen in Figure 3.7. Thin arrows represent invocation while the thicker arrows show the main flow of information. A schema is maintained separate to the registry, which describes the information available from each producer.

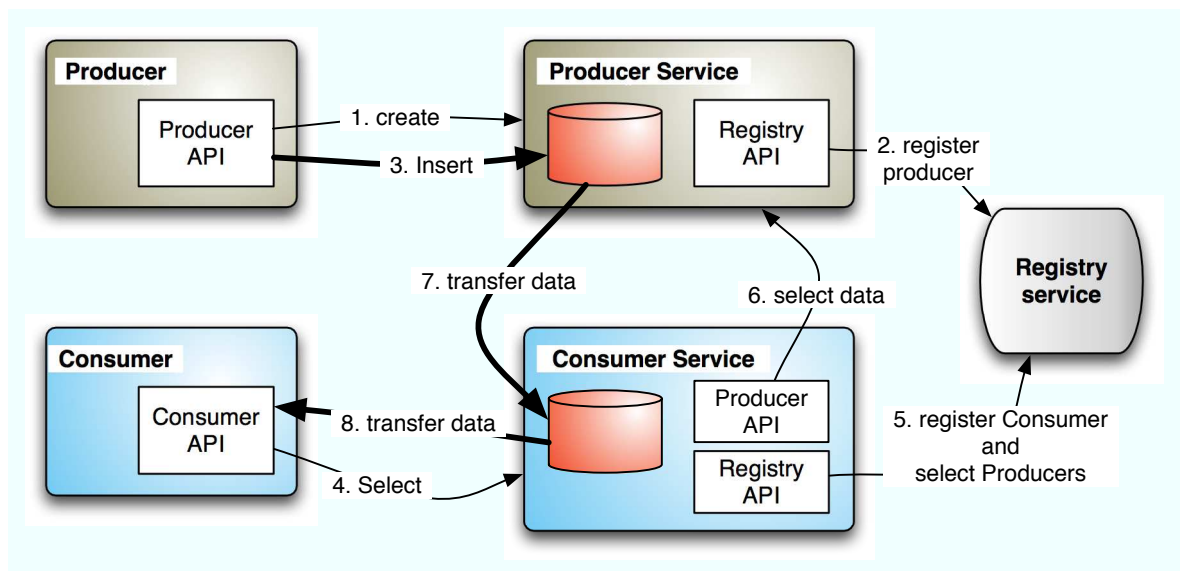


Figure 3.7: Architectural overview of the Relational Grid Monitoring Architecture

A number of producer types have been developed, as described in Table 3.3. In R-GMA, compound producer/consumers are known as Secondary producers, which can query and re-publish information. Primary and secondary producers may use either memory, files, or databases for storage of tuple information. In the case of on-demand producers, tuple storage is implemented in the user code.

While R-GMA could be applied to many situations where the distributed production and consumption of data is required, the early development included two additional tools to illustrate its use as a replacement for the MDS information system; GIN, and GOUT. R-GMA is an active project undergoing continuous development. Recent releases have featured enhanced security and performance. APIs are available for Java, C, C++, Perl, and Python. A screenshot of the R-GMA browser interface is shown in Figure 3.8. Note the SQL query used to retrieve the displayed data and the presence of the timestamp information in the form of the measurement data and measurement time fields.

R-GMA Browser Home Page

Query: **SELECT * FROM GlueServiceStatus**
Properties: Latest

GlueService_UniqueId	Status	Message	MeasurementDate	MeasurementTime
http://se.grid.rug.nl:8443/srm/managerv1	OK	No Problems	2008-04-15	07:52:12
egse.frascati.enea.it_org.glite.rgma.OnDemandProducer	OK	Service running, current memory usage at: 47%	2008-04-15	08:01:57
udo-mon01.grid.uni-dortmund.de_org.glite.rgma.Browser	OK	Service running, current memory usage at: 64%	2008-04-15	08:02:21
t2-mon-01.na.infn.it:2136	OK	No Problems	2008-04-14	23:30:02
STAR-CATANIA_fts.cr.cnaf.infn.it_org.glite.ChannelAgent	NULL	NULL	2008-04-15	07:06:58
BUDAPEST-T1_fts.cr.cnaf.infn.it_org.glite.ChannelAgent	NULL	NULL	2008-04-15	07:06:58
http://darkmass.wcss.wroc.pl:8443/srm/managerv1	OK	No Problems	2008-04-15	07:09:44
udo-mon01.grid.uni-dortmund.de_org.glite.rgma.Consumer	OK	Service running, current memory usage at: 59%	2008-04-15	08:02:19
http://atlas.phys.sinica.edu.tw:8443/srm/managerv1	OK	No Problems	2008-04-15	07:57:53
http://dpmsrm.ciemat.es:8446/srm/managerv2	OK	No Problems	2008-04-15	07:30:35
mon01.grid.info.uvt.ro_org.glite.rgma.PrimaryProducer	OK	Service running, current memory usage at: 19%	2008-04-15	08:01:47
mon01.grid.info.uvt.ro_org.glite.rgma.Browser	OK	Service running, current memory usage at: 16%	2008-04-15	08:01:50
boalice1.bo.infn.it_org.glite.rgma.SecondaryProducer	OK	Service running, current memory usage at: 30%	2008-04-14	06:56:19
se1-egee.fesb.hr_org.glite.rgma.Browser	OK	Service running, current memory usage at: 29%	2008-04-15	08:01:25
mon.grid.uni-sofia.bg_org.glite.rgma.Consumer	OK	Service running, current memory usage at: 23%	2008-04-15	08:01:52
voms.grid.sinica.edu.tw_apesci_org.glite.security.voms	OK	Service is running	2008-04-15	08:01:37
se1-egee.fesb.hr_org.glite.rgma.PrimaryProducer	OK	Service running, current memory usage at: 27%	2008-04-15	08:01:25
juno.switch.ch_org.glite.rgma.SecondaryProducer	OK	Service running, current memory usage at: 33%	2008-04-15	08:01:36
KEK1-KEK2_dg08.cc.kek.jp_org.glite.ChannelAgent	NULL	NULL	2008-04-15	07:25:41

Home
Predefined:
[GlueCE](#)
[GlueSE](#)
[GlueSubCluster](#)
[GlueService](#)
[GlueServiceStatus](#)
[GlueSite](#)
[RGMALogs](#)
Table Sets




Figure 3.8: A Screenshot of the R-GMA browser interface

Table 3.3: R-GMA Producer Types

Producer type	Description
Primary	Primary producers are sources of information and publish by means of SQL INSERT statements.
Secondary	Also referred to as archivers or republishers, secondary producers satisfy information requests by invoking queries on other producers and returning the results as inserts.
On-Demand	Formerly known as Cononical producers, the on-demand producer responds to selects (requests for information) by invoking user code.

3.5.4 GridICE

GridICE [ABF⁺04], a product of the DataTag[MMFM⁺05] project, was primarily designed to allow Grid administrators to monitor resource utilisation through a Web front end. A role-based view of information is provided at VO, site, and resource level. The display of information over a geographical map is also supported.

Sensors from the LHC Era Monitoring system (Lemon) are executed on the monitored hosts, and the information is then published via MDS using an extended GLUE schema. The extensions provide additional host-related parameters. Nagios is used at the central server for the scheduling of monitoring activities and for event notification. The central GridICE server also employs MDS for the purposes of resource discovery and life-cycle tracking. Configuration files are periodically and automatically generated in order to ensure that the Nagios scheduler is kept up to date with the current state of the infrastructure. Outputs from the Nagios checks are archived in a monitoring database for subsequent processing and rendering via the Web interface, see Figure 3.9. GridICE is a stand-alone monitoring tool with little support for republishing of information.

3.5.5 MonALISA

MonALISA (Monitoring Agents using a Large Integrated Services Architecture)[NLG⁺03] is an extensible general-purpose framework for monitoring hosts and networks in large-scale distributed systems. The system employs station servers installed at each site to orchestrate monitoring activities. Information is locally stored and made available to higher level services. Each station server manages a monitoring service which collects data from locally available sources such as SNMP, Ganglia daemons, or PBS log files. Management of services and monitoring modules is available via an administration GUI allowing authorised

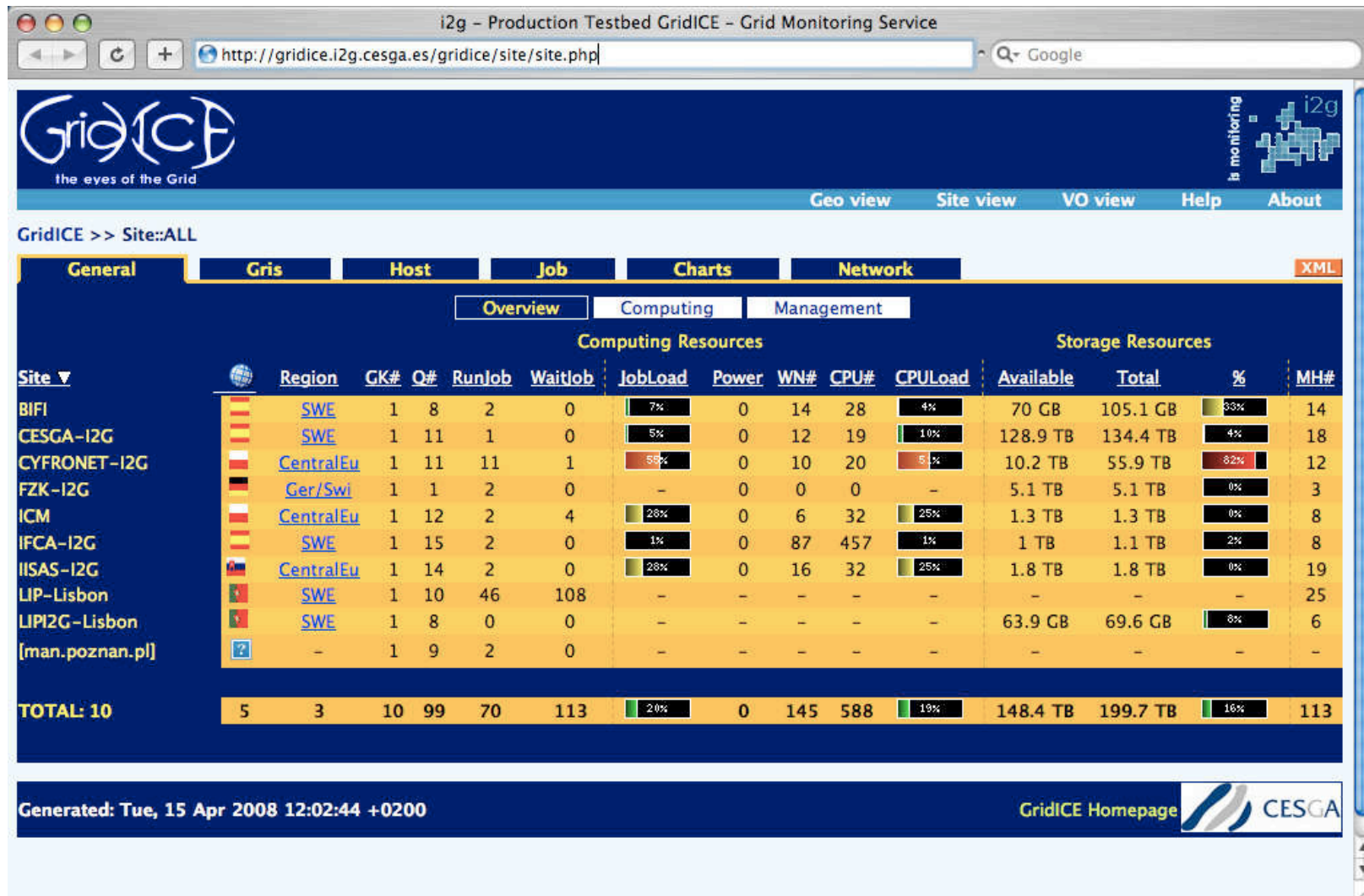


Figure 3.9: A Screenshot of the GridICE browser interface monitoring the Int.eu.grid production infrastructure

users to remotely configure the monitoring activities. A GUI front-end is also included for visualisation of monitoring information.

3.5.6 GSTAT

GSTAT[GST] is an LCG2 Grid Information System monitoring application developed by the Grid Operations Centre at Taipei. It is designed to monitor the information system by detecting faults and verifying the validity of the data. It also features a web interface for display purposes. The system allows display of information such as current job information or the number of free CPUs, from MDS-like information systems. GSTAT is composed of filters which use LDAP queries to query the information sources on the Compute Elements. The resulting data may then be analysed in order to detect faults, before the processed data is displayed via the web interface.

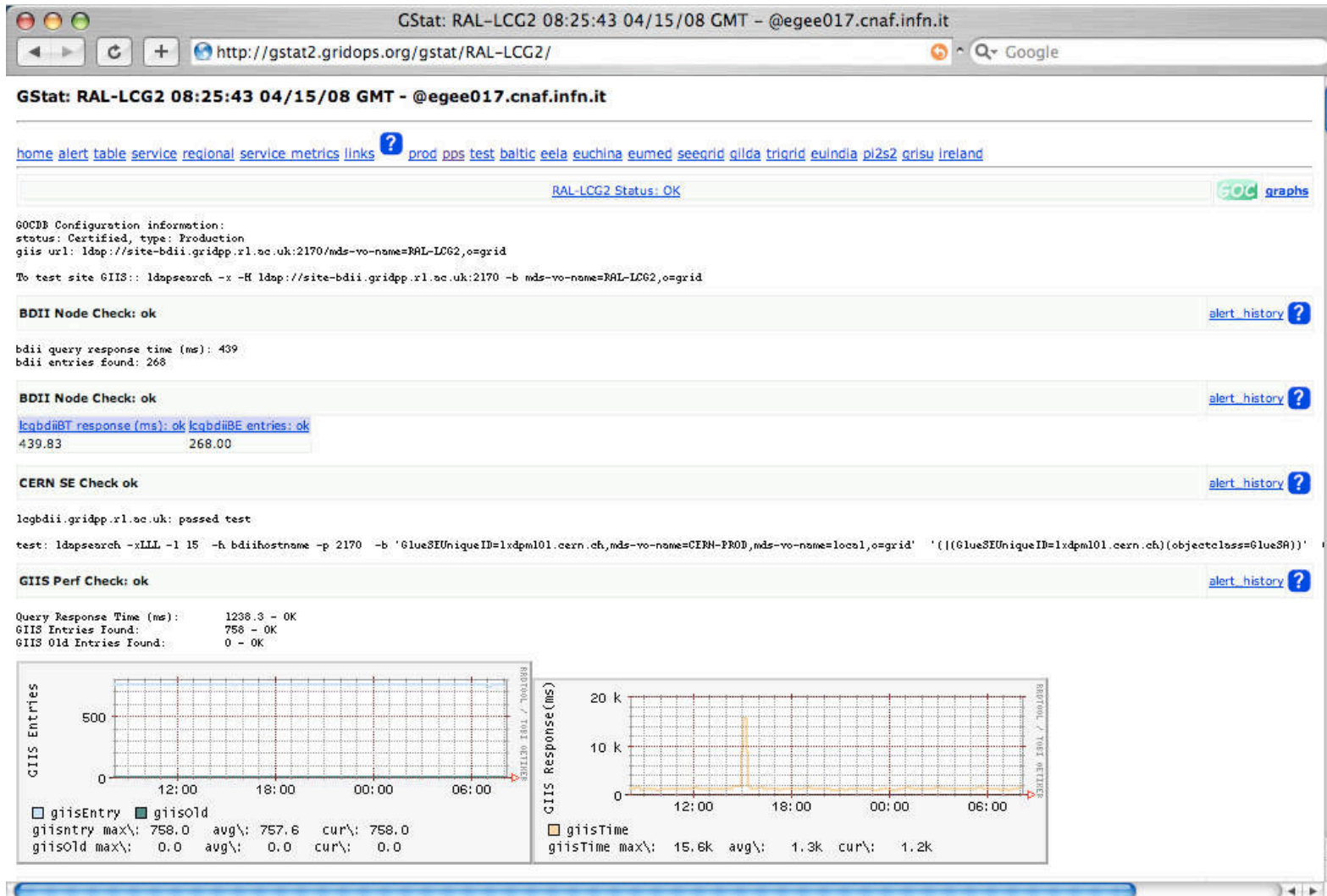
The list of sites included in the queries may be based on entries in the BDII configuration file or the GOCDB. Once a site is added to the database or information system configuration, GSTAT monitoring will begin automatically. RRD databases and R-GMA are used for data storage enabling rapid retrieval and historical analysis. A screenshot from the GSTAT installation on the EGEE production infrastructure is shown in Figure 3.10. Note that the LDAP queries used to retrieve the information are displayed alongside the test results.

3.5.7 C.O.D.E.

Developed for the NASA Information Power Grid, C.O.D.E. (Control and Observation in Distributed Environments)[Smi02] is a framework for monitoring and managing organisation-wide Globus-based grid installations. The framework is composed of observers, actors, managers, and a directory service. Management Agents, consisting of an observer, an actor, and a manager are deployed to each monitored installation, and events generated by these agents are forwarded to an event archive in the form of an XML database. A Java-based user interface queries current information from the event archive and uses it to display the current status of monitored resources.

Each observer manages a collection of sensors and provides their event data via a producer interface. Actors may be asked to perform specific actions such as restarting a daemon or sending an e-mail. Managers consume events produced by observers and, based on analysis of that event information, may instruct actors to perform specific actions.

The CODE project was among the earliest to include support for resource management in its architecture although little of the management functionality was actually implemented. Maintenance of the project finished along with that of the IPG.



50

Figure 3.10: A Screenshot of the GSTAT browser interface

3.5.8 Site Functional Test

The Site Functional Test (SFT)[sft] is a tool developed at CERN to test the operations of LCG sites from a users perspective. Scheduled tests are performed to test a range of site functionality, including job submission, data management, information systems, and software versions. Results of the monitoring operations are used to construct a web interface displaying a grid of sites and tested functionality. Colour-coding of test outputs enables the rapid detection of problems, and the display layout facilitates debugging. An additional administrative interface enables the execution of on-demand tests. Customisation of the display output is also supported. A screen shot of the SFT interface is presented in Figure 3.11.

3.5.9 SAM

The Service Availability Monitoring (SAM)[sam] system is a monitoring framework also developed at CERN. It is designed to be an evolution of, and replacement for, the Site Functional Tests. While the core framework is maintained at CERN, many of the service checking components have been developed by project partners. Monitored entities include information systems, compute elements, storage elements, resource brokers, and data management services. SAM employs a web service architecture and an Oracle database for configuration and persistence of monitoring information. A screen shot of the SAM interface is presented in Figure 3.12.

3.5.10 CrossGrid Host Check

HostCheck is a grid monitoring and diagnostic tool developed at LIP (Laboratory of Instrumentation and Experimental Particles Physics, Portugal) under the Verification and Quality Control operations of the CrossGrid project. The system supports both test and validation activities, and allows site managers to verify the correct installation, configuration and behaviour of grid resources.

HostCheck consists of an automated set of test scripts which are typically run twice a day to verify grid systems (compute elements and storage elements) located at distributed sites. The results of the monitoring operations are used to build a collection of diagnostic web pages organised by site and date. The HostCheck diagnostics may also be executed in an on-demand fashion via the web interface, a feature intended to allow site administrators to verify the operations of their resources.

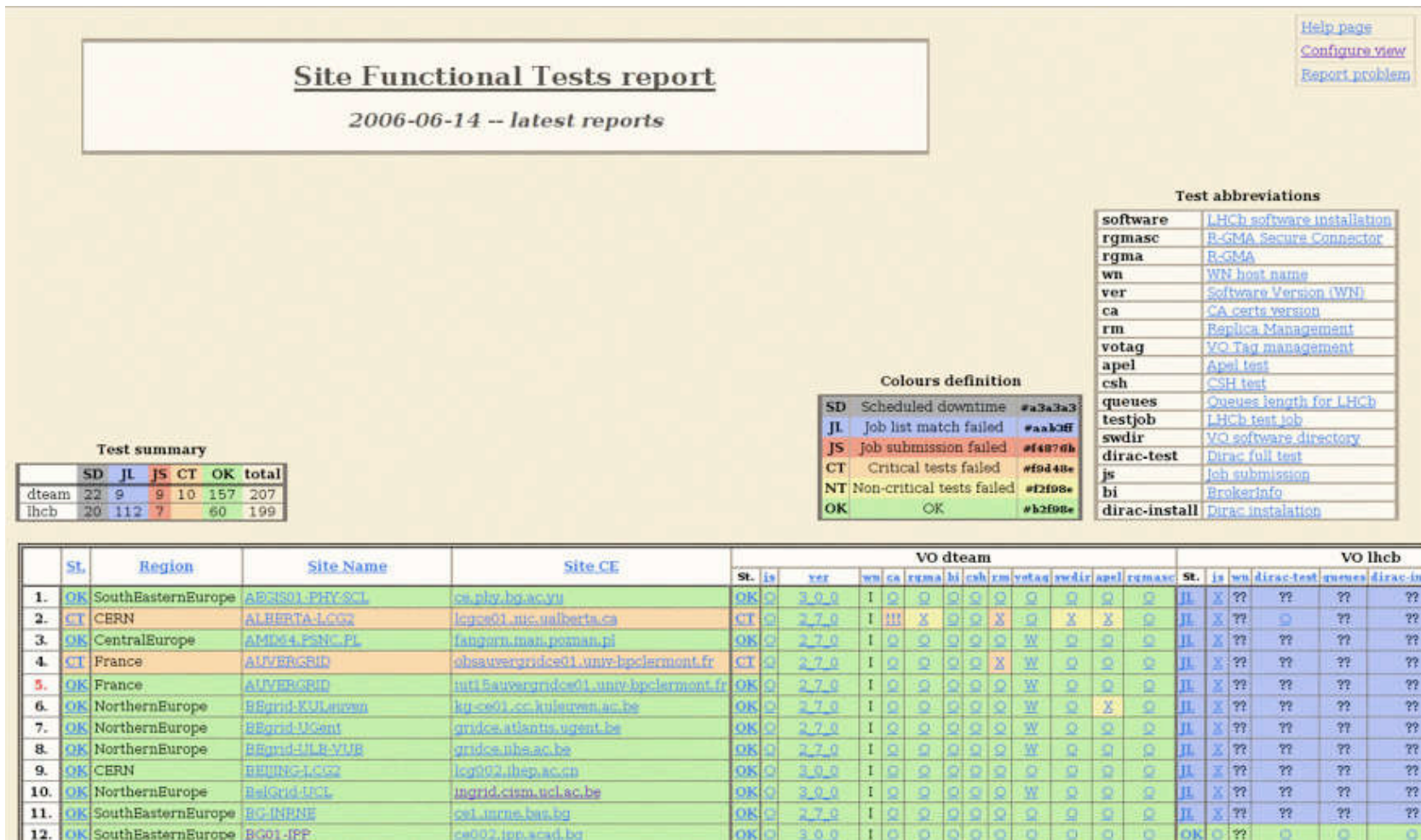


Figure 3.11: A Screenshot of the Site Functional Test browser interface

[Home Back](#)

Service Availability Monitoring - CE

2006/06/14 - 11:45:00

Tests Displayed

DTeam

- CE-sft-icg-rm-cp3
- CE-sft-rgma-sc
- CE-sft-cover
- CE-sft-icg-rm
- CE-sft-ic-rm-rep
- CE-freecpu
- CE-sft-vo-swdir

show DTeam critical tests

Sort by: SiteName

ShowSensorTests

show	stat	description	sum
<input checked="" type="checkbox"/>	NA	no status available	0
<input checked="" type="checkbox"/>	OK	normal status	142
<input checked="" type="checkbox"/>	INFO	useful information	0
<input checked="" type="checkbox"/>	NOTE	important information	0
<input checked="" type="checkbox"/>	WARN	subject may fail soon	6
<input checked="" type="checkbox"/>	ERROR	subject has failed and problem is localized	42
<input checked="" type="checkbox"/>	CRIT	subject has failed and problem is fatal	3
<input checked="" type="checkbox"/>	MAINT	subject is under maintenance	0

DTeam tests

testname	desc	crit
ca	CA certs version	CT
rm	Replica Management	CT
ver	Software Version (WN)	CT
csd	CSH test	CT
bi	BrokerInfo	CT
js	Job submission	CT

No	RegionName	SiteName	NodeName	Status	DTeam					
					ca	rm	ver	csd	bi	js
1	SouthEasternEurope	AEGIS01-PHY-SCI	ce.phy.bnl.ac.ru	OK	ok	ok	ok	ok	ok	ok
2	CERN	ALBERTA-LCG2	lgp011.nc.ku.berlin.de	CRIT	err	err	ok	ok	ok	ok
3	France	AUVERGRID	n015auvergridce01.univ-lpiclermont.fr	OK	ok	ok	ok	ok	ok	ok
4	France	AUVERGRID	obsauvergridce01.univ-lpiclermont.fr	ERROR	ok	err	ok	ok	ok	ok
5	CERN	BEHING-LCG2	lcy003.shnp.ac.cn	OK	ok	ok	ok	ok	ok	ok
6	NorthernEurope	BEgrid-KULeuven	kr-0e01.cc.kuleuven.ac.be	OK	ok	ok	ok	ok	ok	ok
7	NorthernEurope	BEgrid-LiCent	omdca.atlantis.unent.be	OK	ok	ok	ok	ok	ok	ok

Figure 3.12: A Screenshot of the Service Availability Monitoring browser interface

3.5.11 JIMS

JIMS, (the JMX-based Infrastructure Monitoring System)[aMS⁺03], is a Java-based monitoring framework developed by the Int.eu.grid project. It is based on JMX, the Java Management Extension which facilitate the management of applications and resources represented as Managed Beans. JIMS is constructed using a layered architecture, designed to allow monitoring of grid infrastructure, in addition to dynamic discovery of computational resources.

Additional modules have been developed for the the framework, including a ‘pathload’ module ,which allows JIMS to measure available bandwidth between grid sites, and an OCM-G[BBF⁺04] module which provides load information from worker nodes. Communication layers are also constructed as modules allowing a JSR-262 compliant module to be constructed which exposes the JMX management interfaces as WS-Management standard compliant web services.

The fundamental component of the architecture is the Monitoring Agent, which employs sensors to monitor resources. Above the monitoring agents, consumers of monitoring data perform management and monitoring operations. The architecture of JIMS is illustrated in Figure 3.13.

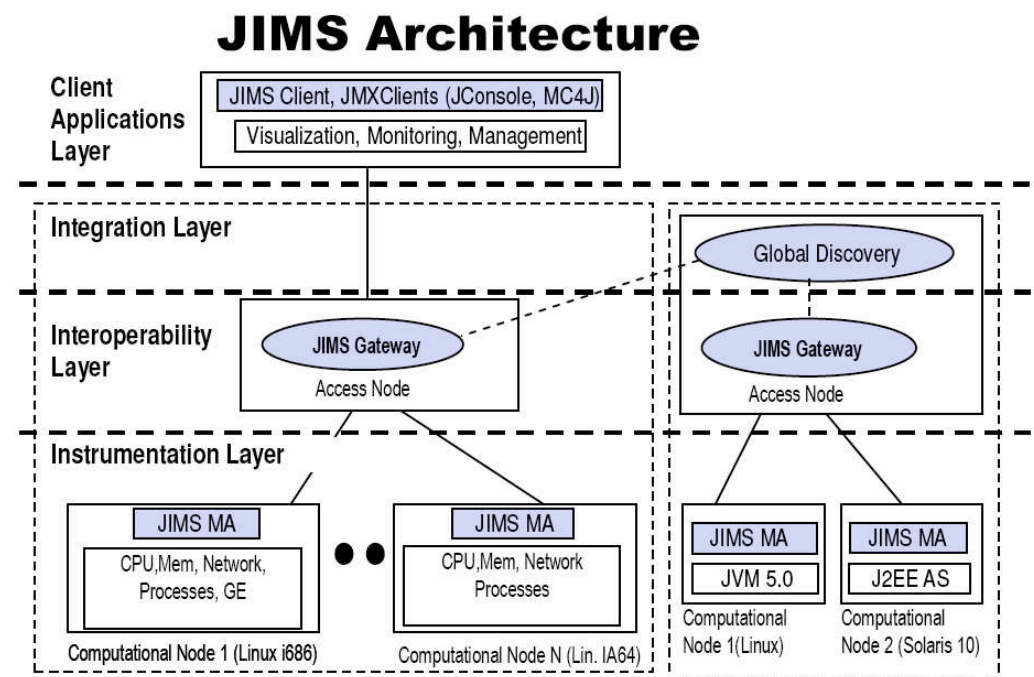


Figure 3.13: Architecture of the JIMS monitoring framework.

3.5.12 I4C

I4C[RCW06a] is an Agent-based wide-area service and resource monitoring solution developed by the author during his research at Trinity College Dublin. Its design facilitates the monitoring of network resources distributed across multiple administrative domains. In its most basic form, the system employs Nagios-style plugins and the Nagios web front end, but incorporates a web service communications layer and sensor agents to ease the deployment in remote networks. Customised displays are also possible. The architecture, development, and deployment of I4C are described in Chapter 6.

3.5.13 RB-STATS

RB-STATS (Resource Broker Statistics) is a management reporting tool developed by the author to provide reports of infrastructure usage. Reports are available on a per site or per user basis, or over a given time interval. Figure 3.14 illustrates a screen capture of the web interface showing the job submission history for a given user over the last 12 months, complete with monthly totals and colour coding based on the degree of success/completion of job execution.

3.6 Real-time Grid Job Monitoring Systems

The majority of the monitoring systems discussed thus far are concerned with monitoring the functional state of the Grid infrastructure or, in the case of the RB-STATS, the historical analysis of usage and job execution. The objective of real-time monitoring tools is to provide stakeholders with information about current usage in near real-time. This information might be as simple as the current number of active jobs in the system, or might be sufficiently detailed to show where the individual jobs are being executed and their current status. This section provides an overview of two such monitoring systems.

GridPP Real-time Monitoring

The GridPP real-time monitor[Grie], developed at Imperial College London, displays job information in near real-time organised according to job status, virtual organisation, or resource broker. Several different versions of the monitor have been developed with the first being a 2-D geographical Java applet display based on information queried from the resource brokers. More recent releases include 3-D Java displays and Google Earth overlays which use the BDII's as the information source. The components used to query the information providers for the information used in the real-time displays are also used to generate graphs

KEY:													
The information is presented in the form: number of jobs submitted / number of jobs waiting, ready, or scheduled / number of jobs failed													
No Jobs Failed			All Jobs Failed			Some Jobs Failed			No Jobs submitted but Jobs Failed				
SITE: exsted.ie													
USERS: David O'Callaghan John Walsh Brian Conlan Oliver Lattke Geoff Quigley Stephen O. Chikki Eamonn Kenny Aidan Haran Keith Rochford Grid Test User Ronan Waters Saba Moad Eamonn M. Kenny John Ryan Keith Rochford Corer Patten Stuart P. Kenny													
C>=H2O>Grid-ire.knd@exsted.ie/RA-TC/D/CN=David O'Callaghan													
	December	January	February	March	April	May	June	July	August	September	October	November	December
1	0/0/0	0/0/0	0/0/0	3/0/1	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
2	0/0/0	0/0/0	0/0/0	3/0/1	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
3	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
4	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	3/0/0	0/0/0	0/0/0
5	0/0/0	0/0/0	0/0/0	0/0/0	3/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
6	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
7	0/0/0	0/0/0	9/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
8	0/0/0	0/0/0	1/0/1	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
9	0/0/0	0/0/0	3/0/2	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
10	0/0/2	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	11/0/0	0/0/0	0/0/0	0/0/0
11	0/0/0	0/0/0	3/0/1	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
12	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	21/0/9	0/0/0
13	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	5/0/2	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
14	3/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
15	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
16	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	51/0/29	0/0/0	0/0/0	0/0/0
17	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/1	4/0/0	0/0/0	0/0/0	0/0/0
18	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	23/0/0	0/0/0	0/0/0
19	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	120/0/38	0/0/0	0/0/0	0/0/0	0/0/0
20	0/0/0	0/0/1	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	9/0/0	0/0/0	0/0/0	0/0/0
21	0/0/0	0/0/1	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	11/0/0	0/0/0	0/0/0	0/0/0
22	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	10/0/0	0/0/0	0/0/2	0/0/0
23	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	3/0/0	0/0/0
24	0/0/0	28/0/21	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
25	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
26	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	23/0/8	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
27	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	25/0/13	0/0/0	0/0/0
28	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	81/0/15	0/1/0	0/0/0	0/0/0
29	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	10/0/7	0/0/0
30	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
31	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
Totals:	3/0/2	28/0/25	16/0/6	1/0/2	1/0/0	3/0/2	0/0/0	33/0/8	120/0/39	96/0/29	129/0/18	39/1/18	0/0/0
Total Job submissions for user = 469 / 17,149 -- 68.23% Success.													

Figure 3.14: A Screenshot of the RB-STATS web interface

of ‘GridLoad’ for each CE. Archived reports are also available which provide daily activity analysis for each monitored CE. The display interface is illustrated in Figure 3.15.

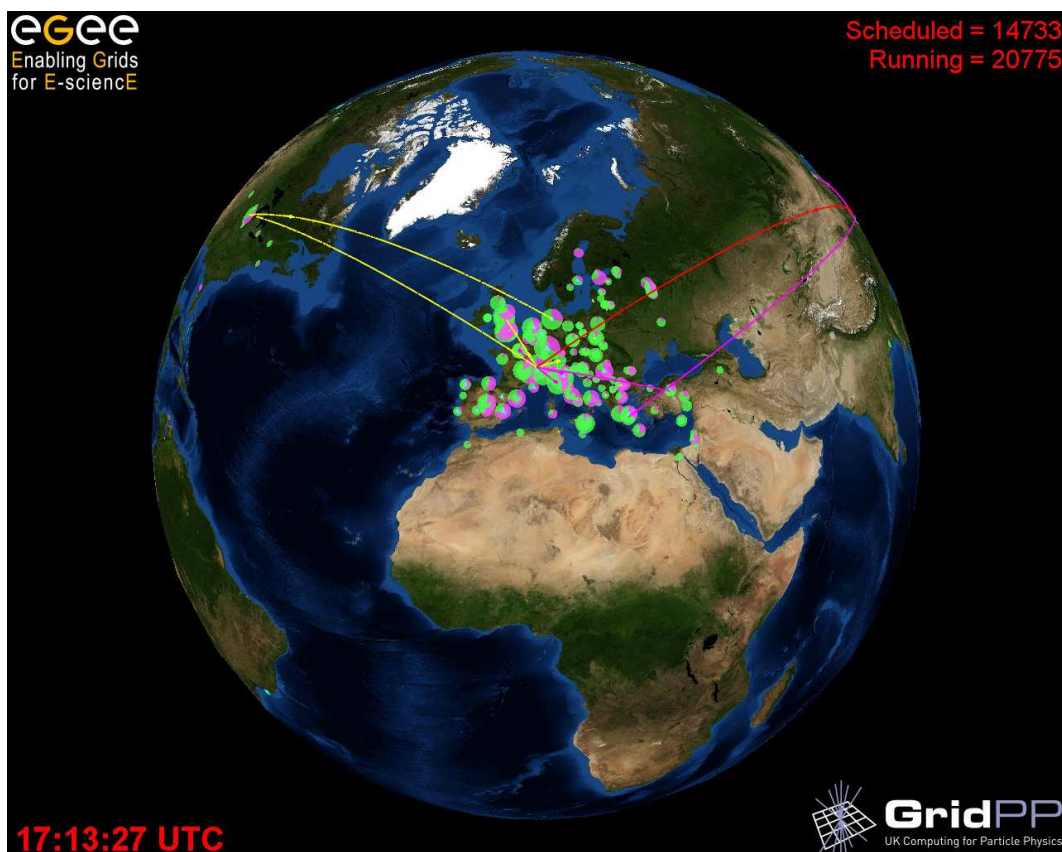


Figure 3.15: A Screenshot of the GridPP real time job monitoring display

GI Real-time Monitoring

Developed by the author for the purposes of real-time display of job activity on the Grid-Ireland infrastructure, the GI real-time monitor is a lightweight Java application producing images for display on the Web. The system uses the Java Naming and Directory Interface libraries to query the grid information system for current job activity and displays the resultant information in tables and on an overlay geographical map. Pie charts are used to represent the number of running and waiting jobs at each site, and the diameter of the chart is proportional to the percentage of the total job workload currently being handled by that site.

The monitoring of resources may be configured via a local flat file or from the site infor-

mation stored in the Grid Operations Centre database. Job information may be gathered from the site-level or top-level BDII's. The information consumers of the system are modular and variants of the system have been created to display job information for GT4 resources and for the Int.eu.grid production infrastructure. The display interface is illustrated in Figure 3.16.

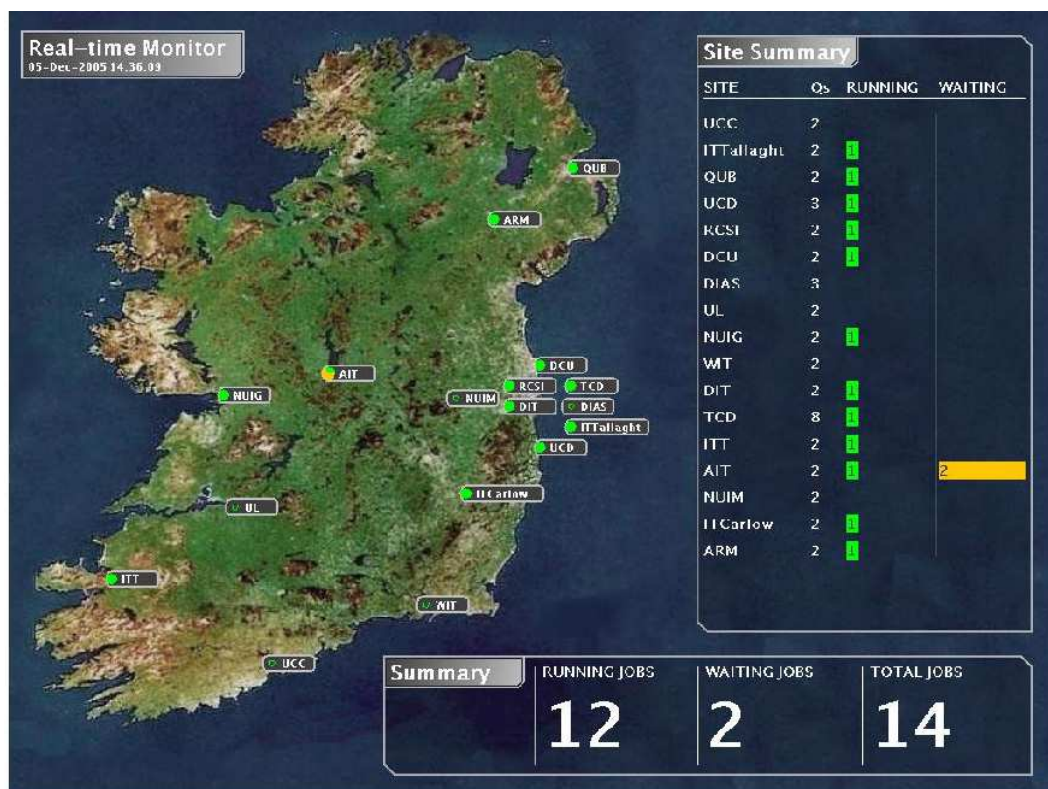


Figure 3.16: A Screenshot of the Grid-Ireland real time job monitoring display

3.7 Summary

In this chapter I have presented an overview of the field of grid monitoring. Following the description of several cluster monitoring tools, some of the additional requirements imposed by grid infrastructures were discussed before presenting a number of approaches to satisfying those requirements. The number of grid infrastructure projects, and the number of monitoring tools developed by, or available to, them serves only to highlight the importance of standard models for data representation and access.

A lack of consensus regarding monitoring requirements, data representation and information transfer has led to the development of ad-hoc solutions, often with limited scope for extension, customisation, or interoperability.

At present, it seems that there is a fashion for the development of monitoring ‘frameworks’. While this approach is generally advantageous in terms of customisation and extensibility, it does not negate the requirement for standard data models and interfaces. The closely coupled architecture of most current monitoring frameworks limits their potential for truly distributed monitoring, despite many efforts towards extensible plug-in architectures.

While the number of projects discussed in this chapter is only a subset of the available tools, it serves to illustrate that a great deal of time and effort has been expended in the research and development of monitoring systems. In addition, while some projects explicitly aim to support the information needs of the management activities of a grid operations centre, few incorporate mechanisms to exercise control over the monitored entities. The subsequent chapter addresses the requirements for, and applications of, such control in distributed infrastructures.

Chapter 4

Distributed Infrastructure Management

4.1 Introduction

A number of bodies have been established in an effort to ease the management of computer networks by defining standard approaches and architectures. Though traditionally linked to the management of local area networks and connected devices, there is a move towards architectures more suited to the management of widely distributed resources.

In the late 1980s, the importance of a standard system for network resource management was recognised by the then recently established Internet Engineering Task Force (IETF)[IETb]. Their solution, the Simple Network Management Protocol (SNMP)[IETc], was a set of standards that would enable the monitoring and configuration of network-attached devices. The SNMP standard also recognised the importance of an extensible model for resource information. This was defined in Management Information Bases, which used notation defined by ASN.1[DF01].

Further demands for enhanced desktop computer management brought about the establishment of the Desktop Management Task Force (DMTF)[DMTc] and their new management system, the Desktop Management Interface (DMI)[DMTb]. DMI is a component of the System Management BIOS and provides a system for managing and tracking components in a desktop PC.

The move towards Web service based management standards began in the 1990s, when experimentation into using HTTP for monitoring and management led to the development of architectures such as WBEM (Web-Based Enterprise Management)[DMTd], and more recently, WS-Management[DMTe] and WSDM (Web Services Distributed Management)[OASd].

DMI reached its end-of-life in 2005 and is to be replaced by another DMTF specification, DASH (Desktop and mobile Architecture for System Hardware)[DMTa]. DASH offers Web services based remote management of desktop and mobile computers using a collection of standards including WBEM, CIM[Ful05], and WS-Management.

The majority of these standards have been created in response to the demands for simplified administration of IT infrastructures within a single administrative domain. To date, few if any have considered the management requirements of distributed scientific or utility computing infrastructures.

The architects of wide-area computing infrastructures were quick to recognise the importance of defining standard interfaces and methodologies for functions such as job submission and resource description, since these are core requirements, but there has been little thought regarding the definitions of standards for resource management.

It seems probable that the future of distributed management lies in the use of web service architectures. With a move towards service oriented architectures, web-based management systems will likely provide an ideal solution for the management of both the underlying resources, and the services themselves.

4.2 Bodies and Standards

4.2.1 Global Grid Forum

The Global Grid Forum[Cat03] was established in 2000 to address issues relating to grid computing, and to define and promote standards and best practices. Having formed in the US as collection of grid users and developers, it became the Global Grid Forum following its integration with its European and Asian counterparts. The first meeting was held in March 2001, and since its inception, the GGF has produced numerous standards and specifications based on world wide collaboration. The GGF was merged with the Enterprise Grid Alliance in 2006 to form the Open Grid Forum[OGF].

4.2.2 Distributed Management Task Force

The Distributed Management Task Force (DMTF)[DMTc] is an organisation promoting the development of management standards for IT and network environments. The standards they produce allow for the development of common management infrastructure components for control and communications in a platform-independent manner. Founded in 1992, the DMTF is an open organisation whose membership now includes many of the largest technology companies. The organisation is composed of a technical committee which oversees a

collection of working groups. Standards developed by the DMTF include the Common Information Model (CIM), the Desktop Management Interface (DMI), and Web-Based Enterprise Management (WBEM). A working group for utility computing was announced in 1994 with the aim of creating interoperable and common object models for utility computing services.

4.2.3 Internet Engineering Task Force

The Internet Engineering Task Force (IETF)[IETb] is an open standards organisation of volunteer participants, with the objective of developing and promoting internet standards. Members of the IETF are organised into working groups whose lifetime is linked to that of their specific project. On completion of a project, the relevant work group is disbanded unless its charter is expanded to encompass new roles. Examples of the areas in which the working groups are active include applications, Internet routing, real-time systems, Operations and Management, and Security. The Internet Engineering Steering Group[IETa] is composed of area directors and the IETF chair and its function is to supervise the operations of the IETF.

4.2.4 OASIS

The Organisation for the Advancement of Structured Information Standards (OASIS)[OASb] was founded in 1993 to promote industry standards for e-business. Originally named SGML Open, based on their work with the Standard Generalised Markup Language[Gol97], its name was later changed to OASIS in order to reflect the growing scope of its standards. The organisation has been instrumental in developing many of the current Web-service standards in addition to standards relating to security, messaging, management, business process execution, and service discovery. OASIS currently comprises participants from over 600 organisations.

4.2.5 Centre for Information Technology Leadership

The CITL[CIT] was chartered in 2002 as a research organisation with the goal of helping healthcare providers make informed IT implementation decisions and help maximise the value of IT systems with an view towards an improvement in the quality of care. The activities of the CITL include the assessment information technologies and reporting of findings. Some of the goals of improved IT operations include the reduction of errors, improving the quality of care through alerts and reminders, and providing direct patient access to records.

Although the primary focus of the CITL is the improvement of IT operations in the healthcare sector, some of the research findings, such as those in the areas of information exchange and interoperability, might be of interest to architects and managers of distributed systems.

4.2.6 World Wide Web Consortium

Established in 1994, the World Wide Web Consortium (W3C)[W3C] aims to promote common and interoperable protocols and is the primary standards organisation for the Web. The consortium defined the first Web-services specification in 2003 and is responsible for SOAP and WSDL. It is also heavily involved in XML-related specifications. The organisation has in excess of 400 members and is currently responsible for more than 80 technical specifications, including more recent specifications in the areas of ontologies and semantic web technologies.

4.2.7 International Organization for Standardization

The ISO[ISO] is a non-government international organisation founded in 1947 to develop and promote industrial and commercial standards. Although the ISO standards are spread across a wide range of industries and disciplines, there are a number of management standards that might be considered relevant. In particular, the ISO/IEC management standards relating to IT security:

- ISO/IEC 27001:2005 (Information security management systems - Requirements)
- ISO/IEC 17799:2005 (Code of practice for information security management)
- ISO/IEC 27006:2007 (Requirements for bodies providing audit and certification of information security management systems)

4.2.8 International Telecommunication Union

ITU[ITU] is the leading United Nations agency for information and communication technologies. Founded in 1865 to standardise and regulate international radio and telecommunications, its primary tasks include standardisation, allocation of radio frequencies, and orchestrating the interconnection of national networks to enable international communications. ITU Recommendations (standards) are widely accepted and highly recognised.

The ITU might be of interest to distributed computing operators not only because many of the standards it has developed relate to the operation of underlying networks, but also because of the organisation's considerable experience in integration and interoperability.

4.2.9 TeleManagement Forum

The TM Forum[Forb] is an industry association of web, telecom and media service providers. The Forum combines the expertise of members from 65 countries to discuss and agree common issues. Areas in which the TMForum is active include technical and business innovation,

leadership and guidance, education and training, and networking and marketing. Its membership comprises system integrators, network operators, service providers, and software and equipment suppliers.

One of the forum's current activities is the the New Generation Operations Systems and Software (NGOSS) programme which is intended to provide improved business practices for communications service providers. NGOSS is based around 5 key principles:

Separation of Business Process from Component Implementation: Advocates the management of business processes as part of a centralised infrastructure in which workflows are responsible for controlling the flow of the process across applications.

Loosely Coupled Distributed System: Integrated co-operating applications of individual components

Shared Information Model: A common information model for the sharing of data between applications. The Shared Information/Data Model (SID) is proposed.

Common Communications Infrastructure: The Common Communications Infrastructure provides a uniform communications interface over which operations support systems can communicate. This means that each OSS must only implement one communications interface rather than one for each type of OSS with which it must communicate.

Contract defined interfaces: Contract defined interfaces represent the API and interface specifications required to enable the Common Communications Infrastructure.

Many of these principles are applicable to the operations of grid service providers and grid operators could benefit greatly from taking notice of existing work in this field.

4.2.10 Infrastructure Management Standards

As described in Table 7.5, a wide range of management standards have been developed to date. Many of these standards apply only to specific areas of fabric management but a cursory knowledge of them may be beneficial to architects of grid management systems, and might serve to promote discussion as to how they might be applied to the area of grid infrastructure management. Also, some of the standards may be suitable guides for the activities of grid operations centres.

Table 4.1: Infrastructure Management Standards

Name	Description
------	-------------

SNMP	<p>The Simple Network Management Protocol (SNMP) is a protocol used monitor and manage network connected devices. Management and configuration information is exposed as variables which may be queried and set by management applications. SNMP is based on a client/server model in which managers interact with SNMP agents that provide the interface to the devices being managed. Operations such as Get and Set are supported by the agents for device management. Asynchronous communication is also supported in which the agent may publish data without being queried. The variables accessible via SNMP are organised in hierarchical structures described by Management Information Bases (MIBs).</p>
CMIP	<p>The Common Management Information Protocol (CMIP) is an implementation of the CMIS specification and provides a communications protocol for use between network management applications and management agents/devices. CMIP is a product of the ISO management model and is defined by a series of ITU recommendations. CMIP might be considered a competitor of SNMP but despite being more feature rich did not achieve the levels of popularity that SNMP enjoys on the Internet. The use of CMIP has been limited mainly to telecommunications networks.</p>
CMIS	<p>The Common management information service (CMIS) is an OSI service interface specification that defines an interface exposed by network elements for the purposes of management. The interface specification includes management operations such as the creation and deletion of managed objects, notification services, and association services used to establish peer connections for the transfer of management information.</p>
WBEM	<p>Web-Based Enterprise Management (WBEM) is the result of an initiative started in 1996 with the goal of designing a unified mechanism for describing and sharing management information in a cross-platform and vendor-neutral way. WBEM is based on a set of DMTF and internet technology standards including CIM, CIM-XML, and WS-Management.</p>

CIM	<p>The Common Information Model (CIM) is a DMTF open standard for the representation of managed IT components and their relationships using a common language. The standard is composed of two components, an infrastructure specification and a schema. The infrastructure specification describes the architecture and concepts of CIM, the language in which it is described, and mechanisms for translating CIM objects to other information models. The architecture of CIM is object-oriented and based upon concepts from UML. The schema specification defines a common base set of objects and relationships that can be used to represent managed components in an IT environment.</p>
SID	<p>Shared Information/Data (SID) was developed as a component of the TMForums NGOSS programme to provide a common vocabulary and set of object/relationship definitions for use within the design of operations support systems for the telecommunications industry.</p>
WSDM	<p>WSDM (Web Services Distributed Management) is an OASIS standard describing how web service architectures and technologies can be used to manage distributed resources. The specification consists of two parts; Management Using Web Services (WSDM-MUWS), and Management Of Web Services (WSDM-MOWS). MUWS defines how an IT resource can present its manageability interface in such a way as to allow it to be remotely managed using web service technologies. MOWS is a particular case of MUWS in which the entity under management is a component of a Web services Architecture.</p>
eTOM	<p>eTOM (enhanced Telecom Operations Map) is a business process framework developed by the Tele Management Forum to guide the development and management of key processes within a telecommunications service provider. One of eTOM's objectives is to enable the end-to-end automation of business and operations processes by deploying the framework across the entire value chain, encompassing service providers, customers, and hardware/software vendors. eTOM is the NGOSS business process framework.</p>

ITIL	Created by the UK Office of Government Commerce, the Information Technology Infrastructure Library (ITIL) is a set of concepts and procedures for managing the development, deployment, and operations of IT infrastructure. The library provides descriptions of a number of important practices including checklists, tasks, and procedures. Guidance is provided in areas such as service delivery/support, incident/problem management, change management, capacity management, availability and continuity, and security management.
SMASH	The DMTF's Systems Management Architecture for Server Hardware (SMASH) initiative is an attempt to unify and simplify the management of data centres by providing a Command Line Protocol (CLP) specification, which enables simple and intuitive management of heterogeneous servers. The SMASH CLP provides a common set of commands which can be used to manage server hardware from a variety of system vendors. SMASH also includes an addressing specification, a server management CLP-to-CIM Mapping Specification, a discovery specification and server management profiles
IPMI	The Intelligent Platform Management Interface (IPMI) is an interface standard for computer system hardware and firmware. IPMI allows administrators to monitor system status and perform management actions on the system such as It is a completely out-of band solution that operates independently of the host operating system and allows management irrespective of the software state of the machine. This is achieved using an independent hardware subsystem consisting of a Baseboard Management Controller (BMC) and other satellite controllers. Communications between the BMC and the satellite controllers within the same server chassis take place over the Intelligent Platform Management Bus (IPMB). Later version support the issuing of status alerts sent out from the BMC over serial or network connections. There is broad industry support for IPMI including implementations from many of the large hardware vendors. In addition to vendor supported management software, a number of open source utilities and management tools exist

4.3 Grid Management Models

High quality management will be vital to the success of e-Infrastructure projects. The provision of a service grid infrastructure that is available to users 24 hours a day requires a formal management structure. Such management structures should be created in such a way as to take maximum advantage of the experience of partners, and should build on established best practice. Ultimately, the responsibility to provide an environment for the delivery of reliable production quality services rests with the grid management team.

Typical responsibilities of the management team include:

- Overall project management
- Reporting to stake holders
- The provision of infrastructure management systems
- Resource allocation
- Promotional and public relations activities
- Preparation of project deliverables
- Documentation and dissemination activities
- Formulation of security and usage policies
- Delivery of expected levels of service and compliance with service level agreements

These responsibilities have led to the emergence of a number of management models and approaches, two of the most salient are discussed below.

4.3.1 Centralised Management

In centrally managed infrastructures, the operations of the grid are orchestrated and managed from a single authoritative location. All core infrastructure components and activities are under central control. It should be stressed however that those components will often reside within remote networks over which the members of the grid operations team have no administrative control.

The advantages of this approach include creation of a central expert group with the potential for more efficient orchestration of grid management tasks and the provision of a central point of contact for user support. By way of example, an overview of the centralised management model employed within Grid-Ireland is presented.

The Grid-Ireland operations centre (OpsCentre) is based at Trinity College Dublin and is staffed by an operations team, with team members fulfilling the role of operator on duty according to a rota. Among the team's responsibilities is the deployment and maintenance of an homogeneous core infrastructure[CWO05] comprised of *grid gateways*[CCO⁺05b]. Installed at each site, these gateways provide a point of access both for external grid users to gain access to the site resources and local site users to gain access to remote grid resources. The gateways provide the core infrastructure functionality of a grid site, thus freeing site administrators to concentrate on the management of their own resources. While the management of the infrastructure is centralised, the deployment of middleware is mixed, with synchronous transactional deployment centrally invoked and distributed QUATTOR repositories local to each site asynchronously fulfilling the final phase of the deployment.

Currently, there are 18 such gateways in Grid-Ireland. Each of these gateways is remotely managed from the operations centre. Local cluster administrators need not concern themselves with the provision of grid services as these are an integral part of the deployed gateways.

Central infrastructure management allows for the creation of an homogeneous core infrastructure, thereby presenting a number of benefits to the users, grid operators and the site administrators. These include:

- Minimizing the proportion of software components that need to be ported to non-reference platforms.
- Maximizing the availability of the infrastructure while reducing the effort required for software deployment. The common software and hardware components facilitate centralised management and push-button transactional deployment of middleware components [CWQ⁺04], guaranteeing uniform responses to management actions.
- Decoupling of grid infrastructure and site management. The fact that the site resources and the grid infrastructure are independent of each other allows for variation in design, deployment, and management.
- The installation of heterogeneous site resources based on non-reference architectures is also supported. Resources at a site remain under the control of the site administrators who are free to manage as they see fit, without having to concern themselves with the finer details of grid integration. In order to support a wide range of connected resources, the operations team carry out porting of the necessary middleware components[KCW⁺05a].

4.3.2 Federated Management

Due to the distribution and complexity of large national and international infrastructures, their management is often divided into a hierarchy that forms a federation of co-operating entities. Infrastructures employing federated management models include EGEE, OSG, and GridPP. In the case of the EGEE operations, the hierarchy is divided, as illustrated in Figure 4.1, into the following components:

Operations Co-ordination Centre The Operations Co-ordination Centre is the top tier in the hierarchy. It performs the role of overall activity management by overseeing all operational and support activities. They develop and promote standard operating procedures and best practices, along with the co-ordination of middleware development and encourage standardisation

Core Infrastructure Centres The Core Infrastructure Centres (CIC) operate the essential grid infrastructure services. In addition, they provide second level support to the regional operations centres and act as operations centres themselves, providing monitoring and troubleshooting services. CICs are intended to function as a single distributed entity by frequently sharing operational information and experiences. While CICs share a common set of responsibilities, some may provide specialist expertise in particular areas. Specific monitoring and management duties may rotate among the CIC.

Regional Operations Centres Regional operations centres form the first line of support for grid users and resource centres. They provide sources of expertise and technical support for the establishment, operation, and integration of resource centres. Acceptance testing of new middleware releases is also performed at the regional operation centres.

Resource Centres The resource centres provide the computation and storage resources required for the execution of user executables on the grid.

The future development of information and management systems should support the distributed hierarchical deployment patterns of these infrastructures. Architectures such as these serve to highlight the importance of interoperability and the implementation of a component based loosely-coupled standards compliant solution.

4.3.3 Grid Operations Centres

Effective management of complex distributed infrastructures, such as electricity supply and traffic management systems, is essential if expected levels of service are to be achieved.

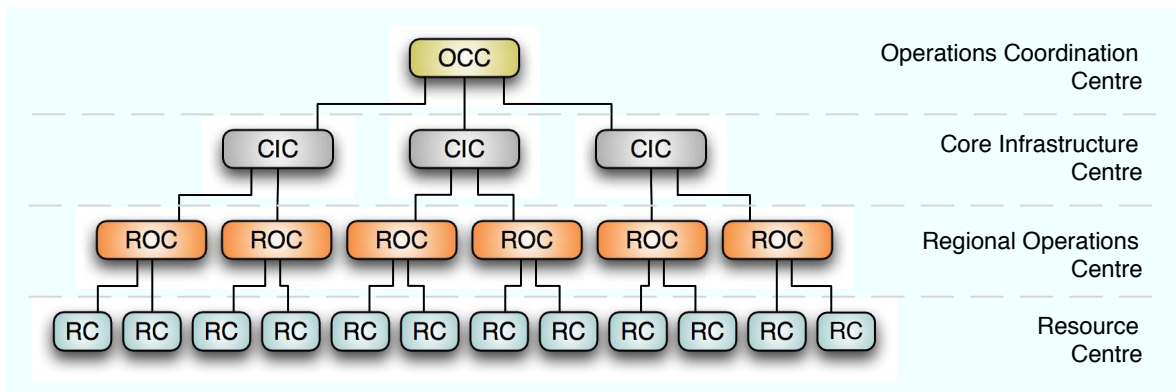


Figure 4.1: A hierarchical operations management structure

Operations Centres are frequently established to satisfy this requirement, providing a central authoritative point of operational information. They are, however, only as good as the underlying information systems on which they depend.

Utility, telecommunications, and emergency services, among others, have all recognised the value of establishing central or federated centres for operations management of complex systems. Distributed computing infrastructures such as Grids have similar monitoring requirements. For those responsible for the management and administration of the individual resources or the infrastructure as a whole it is desirable that a formal management system is put in place to ensure the efficient and continued performance of the resource. Large quantities of status information from disparate resources must be detected, aggregated and stored.

Operations centres are well established in telecommunications sector but are typically designed to provide lower levels of operational support, focusing on the monitoring and management of network elements. While these areas of monitoring are critical, for the purposes of grid operations they must be augmented by the addition of service and business/user management levels.

Functions of a grid operations centre include:

- Provide a single point of operational support to users and site administrators
- Orchestrate the deployment and maintenance of the infrastructure.
- Administration and maintenance of grid services and information repositories.
- Gather performance, availability, and usage characteristics

- Perform problem tracking and resolution
- Change Management (Infrastructure, middleware, etc.)
- Maintain configuration information
- Manage security incident response
- Operation of core infrastructure services
- Middleware deployment and resource induction
- Formulate and monitor compliance with service level agreements
- Define and enforce acceptable usage policies
- Maintain user and VO registration catalogues
- Act as Registration Authority and Certificate Authority.
- Establish and promote standard operating procedures and best practices (user registration, site operations and upgrade, etc.)
- Formulate Audit requirements and security policies
- Provision of documentation and user training
- promote collaboration and interoperability

Although not an exhaustive list of responsibilities, this list serves to illustrate the obvious demand for a great deal of up-to-the-minute operations information about the managed infrastructure and resources. In addition, given the range and complexity of the field of operation, there is significant justification for the investigation and implementation of systems for control and automation. Removing some of the burden from grid operators might result in increased levels of service and availability, increased efficiency, and reduced cost. Scalability of management can be achieved through the division of the responsibilities among a hierarchy or federations of lower-level or regional operations centres.

4.4 Grid Management Groups

A number of teams and working groups have been established in recognition of the value and complexity of the task of managing distributed computing infrastructures such as Grids. Some of the objectives of these bodies include the identification of best practice and the

increased sharing of expertise and tools. This section provides a brief overview of two such bodies established within European projects.

4.4.1 LCG Monitoring Working Groups

System Management

The Systems Management Working Group[LCGd] was established to bring together the existing expertise in the area of fabric management, and to create a central repository of tools and knowledge for the benefit of resource managers. The emphasis is on developing understanding existing tools and recommending how they might be used to remedy specific problems. It was not intended that the group should undertake any development work in the area of systems management, but rather that it would improve existing tools and provide improved documentation. An emphasis is placed on the improvement of systems management practice, and encouraging the sharing of expertise, experience, and tools. The group aims to compile a list of common infrastructure problems and document these along with recommended solutions based on past experience.

Grid Monitoring

The primary objective of the Grid Monitoring Working Group[LCGb] is to help improve the reliability of the grid infrastructure by bringing together the existing work in the area and formulating a coherent plan for its usage. The group also aims to provide interested parties with views of current and historical resource status by developing customised dashboards.

The goals of the Grid monitoring working group include:

- Evaluate the current state of monitoring archival repositories, describe the interactions between them, and provide recommendations on how this data interchange might be improved.
- Provide views of the systems tailored to the audience.
- Agree on common definitions for sensors and metrics that can be used to describe the current state of grid services.
- Define interfaces between a site and the grid monitoring fabric allowing it to both consume data available to it and to publish monitoring information.
- Promote the creation of a common sensor repository and agreed interfaces.
- Propose specific dashboard development to visualise multiple data sources.

System Analysis

The System Analysis working group[LCGc] hopes to gain improved understanding of application failures in grid environments, and to provide a view of the state of the infrastructure from an applications level. It is anticipated that this improved awareness will lead to improvements in reliability and performance. The System Analysis group also aims to improve experiment dashboards, and to enable them to provide additional information to the monitoring and management groups. An effort will be made to avoid duplication of effort between monitoring teams and working groups by ensuring that existing common underlying sensors and tools are used where possible.

4.4.2 EGEE Operations Automation Team

Part of the SA1 activity in EGEE-III, the Operations Automation Team is “tasked with coordinating monitoring tools and developments, and will have a specific goal of advising on strategic directions to take in terms of automating the operation”. One of the team’s primary objectives is the automation of what are currently manual processes so that the “overall level of operations effort can be significantly reduced in any long term infrastructure”. It is proposed that the mandate for this team will include examining the tools currently in use, and investigate their improvement and automation with a view to standardising the interactions between them. The team will also solicit and coordinate the development of new components from the project partners, and outline a workplan for the development of operational tools to ensure that any work carried out in that regard is inline with common strategy.

The areas of interest to the team will include service and availability monitoring, usage characterisation and accounting systems, and tools for Grid operators. It will also endeavour to promote the use of fabric management tools at constituent sites in order to increase availability. Provision of reporting and visualisation tools is also to be addressed.

The establishment of this team is a clear recognition of the importance of a suite of interoperable tools for grid management, and that much of the effort invested to date has been fragmented and inefficient. Their proposed mandate also serves to illustrate the importance of component based systems which can be tailored to fit the network and management structure.

4.5 Grid Management Tools

Due to the growing numbers of participants and increased awareness of the software requirements of operations centres, a number of tools have been developed in order to help staff in achieving their overall goals.

These tools should not be thought of merely as software components, that once put in place will automatically change operations management for the better. Rather they should be considered technical aids aligning with, and supporting, defined management practices.

Operational experience has shown that overall grid infrastructure reliability can be improved by putting fabric management systems in place at resource sites, minimising costly and error-prone human interventions where possible. In the absence of effective management tools, grid infrastructures may fail to meet their full potential.

4.5.1 The LCG GOC Database

The LCG Grid Operations Centre Database (GOCDB)[EGEd] is a central repository of general information about participating EGEE/LCG/UK-NGS sites. It is operated by the EGEE UK-Ireland Regional Operations Centre. Information stored within the database includes details of the resources at the sites, site locations and contact information, and details of any scheduled down-time when the site or resources is not expected to be available. The database is available via web interface to holders of certificates recognised by the LCG Certificate Authority. It is the responsibility of the site administrators to ensure that details relating to their site are kept up to date in the database.

The information stored within the database has many uses in terms of managing the operation of the infrastructure. As an example, consider the integration between the GOC database and the Information System. During the addition of a new site to the GOC DB, the URL of the site-level BDII is entered into the database. A list of all participating site-level BDII's is then easily generated and used to configure the information providers populating the top-level BDII's.

Originally deployed on a MySQL database, later versions moved to Oracle. The system is undergoing continuous improvement and bug-fixing, and the latest version 3.0.9 was released in December 2007.

4.5.2 SMILE

SMILE[SMI] is a service management interface that was proposed as part of the JRA1 activity in EGEE-III but was declined funding. The objective of SMILE was to define and create a common interface to all grid services. The proposed architecture would provide an abstraction layer between operators and services so that operators would not need to know the specific commands required to manage each service. The idea is that most services would have some commands in common, such as 'start' and 'stop', yet in order to support many different services, specific functionality may be added to the interface for a particular service.

While the SMILE proposal highlighted what is perhaps an important requirement, it did not take into account the various discovery, inspection, introspection and service description standards available or emerging in the web services arena. Some of these mechanisms may provide existing tools to achieve the goals of SMILE in a more open and standards based fashion.

4.5.3 ELFms

ELFms (Extremely Large Fabric management system)[ELF] is a project coordinated by the IT department at CERN and builds upon mechanisms developed by the European DataGrid project. ELFms is a fully modular interoperating framework for the management of large heterogeneous environments. ELFms is currently composed of three sub systems, QUATTOR, LEAF, and Lemon which is described in Section 3.3.2.

QUATTOR

QUATTOR[LLM⁺04] is a system administration toolkit enabling the automated installation, configuration, and management of computing systems running Linux and Solaris. It can be used within a small site or in a national context, e.g. Quattor is the basis for the hierarchical deployment of all middleware components across the whole of Grid-Ireland. All configuration data for managed systems is stored using hierarchical template-based structures in a Configuration Database (CDB). The information stored within the database includes hardware profiles, software package lists, and node services etc. Integration between the CDB and kickstart based installation tools, such as Quattor's Automated Installation Infrastructure (AII), allows for centrally configurable and reproducible installation of nodes.

In addition to managing the installation of nodes, Quattor includes mechanisms to manage the runtime configuration. This functionality is provided by two subsystems:

SPMA (Software Package Manager Agent) The SPMA is responsible for managing the software packages installed on the node and can handle several package formats including RPM and PKG(Solaris). Once added to a repository, new packages are deployed to client nodes by explicit configuration of the node profile in the CDB. SPMA also supports rollback.

NCM (Node Configuration Manager) The NCM uses service-specific plugins to make the configuration changes necessary to bring the node to the desired state as defined in the CDB.

LEAF

LHC Era Automated Fabric (LEAF) management toolset was developed within collaboration with GridPP as part of the LCG project. The toolset provides a collection of advanced fabric management components consisting of a State Management System, and a Hardware Management System.

State Management System (SMS) The SMS enables high level commands to be issued to sets of quattor-managed nodes enabling their operational status to be managed e.g. moved from standby into production.

Hardware Management System The HMS tracks hardware management workflows in the computer centre and supports visualisation of equipment locations. The HMS can issue formal work orders, and provide statistics for management reporting.

4.5.4 InGrid

InGrid[Pic] is described as a “a generic autonomous expert system for grid nodes real-time monitor and control”. The project is the result of a collaboration between the Spanish telecom operator Telefonica and Port d’Informaci Cientfica. The objective of inGrid is to address some of the problems associated with managing a grid resource centre, and investigate how that management can be improved using an expert system.

The system is designed to aid in the provision of a high Quality of Service by helping to overcome some of the limitations of the human element in distributed systems management. This is achieved by enabling inGrid to act as a virtual operator, capable of automatically executing actions in order to restore services to their correct state. When these automatic actions succeed, the workload on the human operators is decreased.

Once the monitoring system alerts inGrid to an incident or outage, the expert system will decide which actions are required in real-time. A graphical user interface to the expert system is included in order to eliminate some of the difficulty traditionally associated with the management of rule sets.

The inGrid workflow consists of 3 steps; Information Collection, Decision Making, and Correcting. The decision making component is at the core of the system and is based on the commonly used technique of rule-based programming. These rules consist of ‘If, Then’ statements and are used to match specific sets of patterns to their corresponding actions. The expert systems is implemented using CLIPS, a public domain development tool. inGrid is currently installed and undergoing testing at thePort d’Informaci Cientfica in Barcelona.

4.5.5 Help Systems

Grid operations teams are required to manage complex, large-scale infrastructures and support a variety of user requests. Efficient execution of these responsibilities can be a challenge in itself when multiple operators are involved in particular tasks or a large number of requests must be handled simultaneously. Hence, a number of systems and approaches have been developed.

GGUS[GGU], the Global Grid User Support system, is the centralised user support and ticketing system for the Grid developed at Forschungszentrum Karlsruhe (FZK), Germany. The system is based on Remedy[BMC] and tickets are typically created by individual user requests or automatically based on the actions of the grid operator-on-duty (GOoD). Links to monitoring information and documentation are provided for the purposes of ticket resolution. A screenshot of the GGUS web interface is illustrated in Figure 4.2.

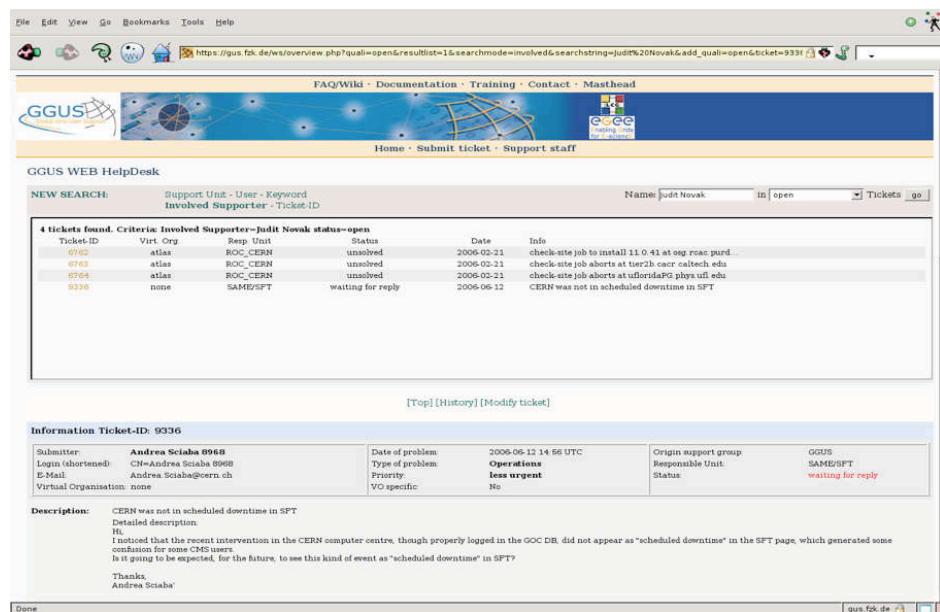


Figure 4.2: The GGUS web interface

Request Tracker (RT)[Prab] is a comprehensive and robust ticketing system which allows individuals and groups to efficiently manage tasks, support issues, and requests. Customers interact with the system via e-mail, while support staff use a web interface. Once a support ticket is created within the system, it may be given a priority and assigned to a member of staff. Its resolution or progress through the system can then be monitored, and notifications issued to management staff and relevant users/customers. A search facility is

also included, enabling operators to find out how similar requests may have been resolved in the past. RT has been applied in many fields of research and business. Common applications include help desk, project management, and managing the software development process. The Grid-Ireland operations centre uses RT for both user support and operations planning, where tasks may be inserted into the system and assigned to specific operators. A screenshot of the RT web interface is illustrated in Figure 4.3.

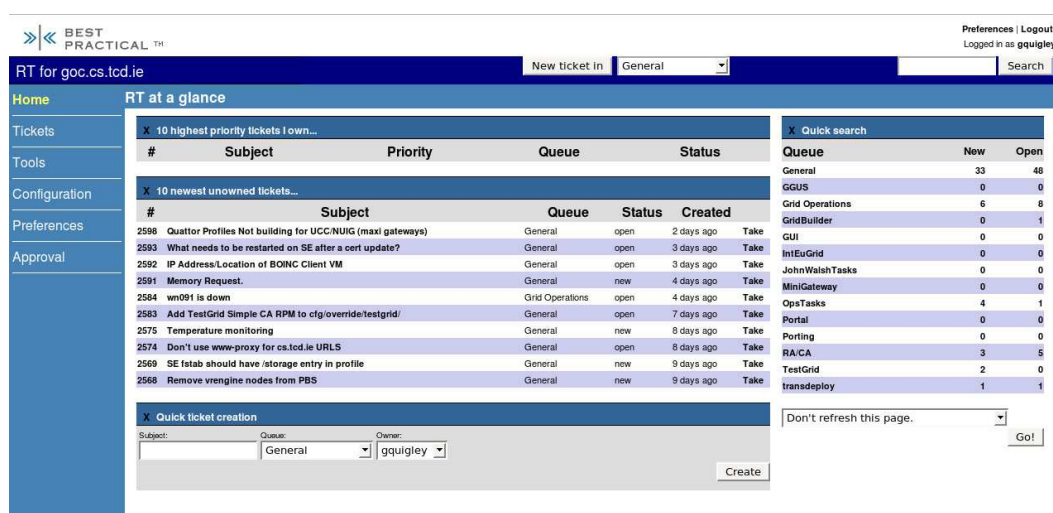


Figure 4.3: The Request Tracker web interface

These systems, and others such as Mantis[Man], are intended to orchestrate management and user support activities. They help to ensure that user / customer requests are acted upon in a timely and efficient manner, and that managers and support staff are kept informed of the progress of problem resolution. These support activities might be expanded further by incorporating other concepts from the area of customer relationship management systems such as market/demand analysis and user surveys. This might provide grid operators with a more intimate knowledge of their users' applications and requirements,

4.6 Command and Control

The concept of command and control is well recognised in its military context but may also be successfully applied in many different situations. The requirement for command arises from the size and complexity of the system in question. In the case of a system composed of distributed entities, some form of command is required in order to co-ordinate the disparate distributed resources in concert to achieve the common goal.

The act of command might be described as the exercise of authority in making use of information to co-ordinate activities and resources in such a way as to carry out a mission or to achieve an objective. The activities of command include the procurement and co-ordination of everything required for the day-to-day operation of the system, and enabling the system to fulfil its requirements through provisioning, intelligence gathering, and the planning, execution, and control of operations.

The execution of command might be divided into the following steps:

1. Gather information on ones resources and capabilities, the environment, and external influences.
2. Communicate, store, retrieve, filter, classify, distribute and display the information so that is may be reasoned upon.
3. Form an estimate of the state of the system based on the processed data.
4. Draft, transmit, and verify receipt and understanding of orders
5. Monitor execution of orders via feedback mechanisms.

These activities may also be summarised into what is known as the ‘MAPE cycle’ (monitor, analyse, plan, execute) as illustrated in Figure 4.4.

An ideal command system should be able to gather information accurately, continuously, comprehensively, and with low latency. It should provide a reliable means of differentiating between what is true and false and what is useful from what is not. The information display systems must be clear, detailed, and comprehensive. The objectives formulated by the system should be both desirable and achievable.

In military circles there is a long history of co-ordinated information gathering, command and control, initially termed *command, control, communication, and information* (C³I), now extended to C⁴I to incorporate the role of *computers* in network centric warfare. C⁴I is a strongly visual tactical tool used by war rooms to manage military field resources. Its acronym stands for:

Command The allocation and direction of resources

Control Tactical execution of programs and initiatives

Communications Transfer of state information and control

Computers Leveraging technology to achieve goals

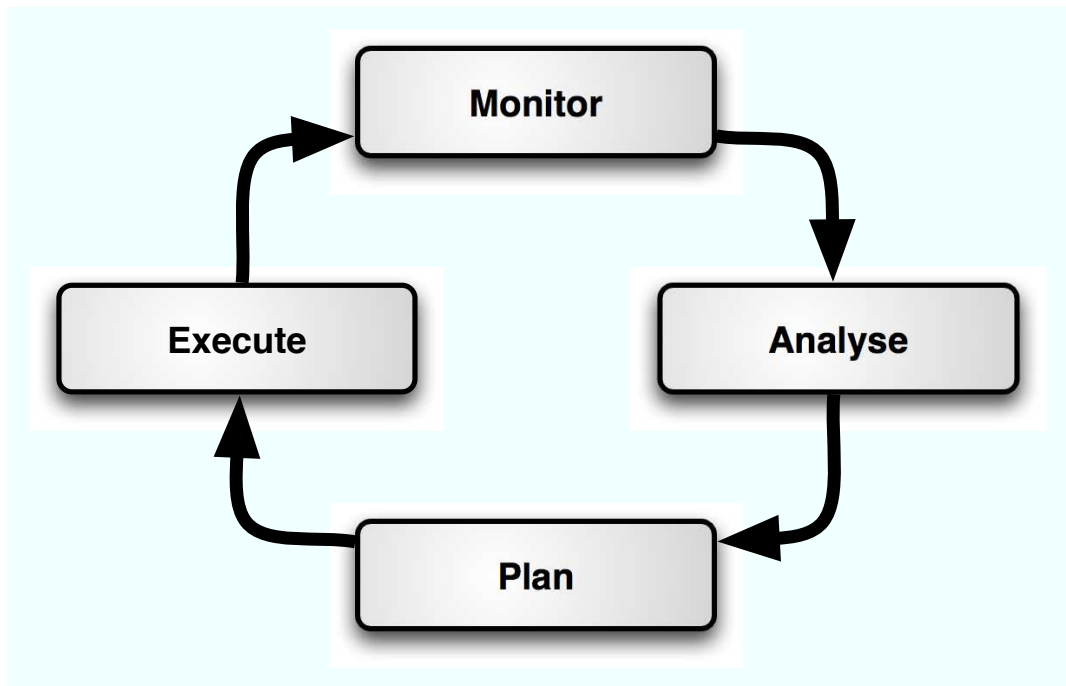


Figure 4.4: The ‘MAPE’ loop representing the execution of command

Intelligence Collection and dissemination of information

C⁴I is intended to provide an accurate and common operational picture and complete situational awareness in order to support decisive actions and reactions in theatres of war.

4.6.1 Operations Other Than War

The advantages provided by Command and Control when orchestrating complex systems has resulted in numerous applications beyond the military context. The fields of application, in so called ‘Operations Other than War’, include:

- Emergency services and disaster management
- Utility services
- Traffic management
- Network and telecommunications management
- Customer relationship management
- Complex systems with much information requiring centralised co-ordination.

Gold-Silver-Bronze

The Gold-Silver-Bronze[Les] structure was developed in the United Kingdom to provide a hierarchical framework for the command and control of major incidents. The framework is sometimes referred to as Strategic - Tactical - Operational, but the categories are equivalent. The system was designed to override the normal rank structure of emergency services personnel in the event of major incidents, as it was decided that three essential roles were more relevant to these situations than numerous ranks.

Gold For each organisation involved in the incident response, a Gold Commander is appointed to be in overall control of their organisations resources. Gold Commanders are located at the central control room where they will formulate a strategy for dealing with the incident. If the Gold Commanders for each organisation are not located within the same control facility, they will be in constant communication by radio, phone, or video-conference. This allows them to share knowledge, information feeds, and situational awareness for the formulation of a coordinated strategy.

Silver The Silver Commander is the senior member of the organisation at the incident site in charge of all of their organisations resources. The Silver Commander decides the tactics for how their resources should be deployed in order to achieve the strategic goals defined by the Gold Commander. Typically the first representative from a given organisation to arrive at the incident site assumes the role of Silver Command until he/she is relieved by a superior.

Bronze Bronze Commanders directly control and organisations resources at the scene and will be found working with those resources on site. In complex situations, multiple Bronze Commanders may be present under each Silver Commander.

Policy Primacy In general, there is a notion of policy primacy in that the ultimate charge of any incident will lie with the police. The exceptions to this are incidents involving fires or hazardous materials in which primacy is assumed by the fire service.

Incident Control System

Similar to the Gold-Silver-Bronze system, the Incident Control System (ICS)[Tea] is a standardised incident management concept developed for use within the United States. ICS is based upon a flexible scalable response organisation which provides a common framework allowing people to work together efficiently. ICS provides mechanisms for controlling personnel, facilities, equipment, and communications.

ICS was developed in the 1970s following research of emergency incidents which showed that response problems were often caused by breakdowns in communications and management deficiencies rather than shortages of resources or failure of tactics.

Some of the principles of ICS which might be applied to infrastructure management include:

Unity of Command Each individual participating in the operations reports to only one supervisor. Eliminates potential for conflicting orders and improves the flow of information.

Clear Text Promotes the use of common terminology using an associated glossary of terms.

Integrated Communications Ensured that all agents/responders can communicate with one another during the operational period.

Management by Objective All actions should contribute to achieving the goals defined in the strategic objectives as defined by the incident commander and planning section. See Figure 4.5 for a simple representation of management by objective as applied to grid services management.

Incident Action Plan Defines the measurable strategic operations to be achieved within the operational period.

Flexible/Modular Organisation The command structure efficiently scale as dictated by the incident scope or the available resources.

Comprehensive Resource Management Ensures that visibility of all resources is maintained so that they may be rapidly deployed.

4.6.2 Summary

Command and Control has been successfully applied to many systems where large amounts of information must be absorbed and reasoned upon, and can benefit from centralised coordination. This section has identified a number of concepts of command and control which are pertinent to the discussion on the application of command and control to the management of distributed computing infrastructures. Many of these concepts could provide inspiration for advanced grid management mechanisms. For example, consider the assumption of Silver Command by the first member of an organisation to reach an incident site. In terms of infrastructure monitoring or control, this might be implemented as the first agent/sensor to

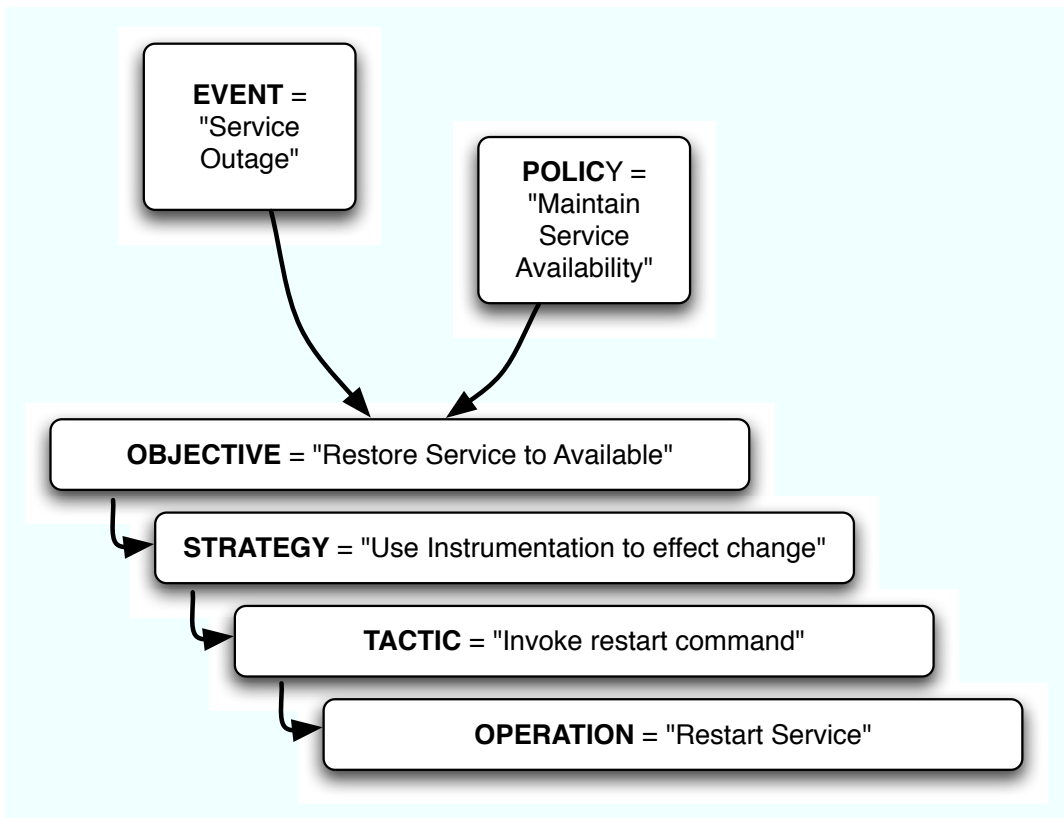


Figure 4.5: Management by Objective applied to grid service management

become aware of a problem raises an alert and assumes responsibility for management of the problem until it is relieved by a superior agent or a human operator. In grid operations, the notion of primacy might be implemented based on region or area of expertise.

4.6.3 Application of Control Theory

Introduction

Control systems are prevalent in almost every field of industry and technology. They deal with behaviour of dynamic systems and are intended to manage, direct and regulate the system in order to ensure that it operates as intended. Control theory allows the construction of high performance control devices despite large uncertainty in the individual components used to build them. Control systems are often based on the principle of feedback where some measurement of actual system output is compared to a desired reference value and the difference used to formulate corrective action.

Control system design might generally involve the following steps[DFT91]:

1. Analyse the system under control. Evaluation of sensor and actuator requirements, and their location.
2. Model of the system to be controlled
3. Simplify the model if necessary to so that it is tractable
4. Analyse the dynamics and properties of the resulting model
5. Define performance specifications
6. Select the type of controller to be used
7. Design a controller to meet the specifications
8. Simulate the controlled system
9. Repeat from step 1 if necessary
10. Choose hardware and software for the implementation of the controller
11. Perform on-line tuning of the controller if necessary.

The object under control is commonly referred to as the plant, an expression carried over from the origins of control theory in industrial process control.

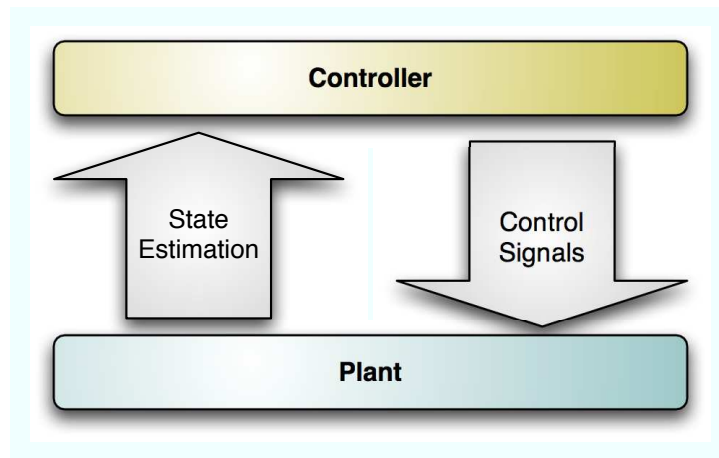


Figure 4.6: Block diagram controller/plant

4.6.4 Control Strategies

Control systems may be broadly categorised based on how the optimum control signal is determined:

Open-loop Control The fundamental assumption of open-loop control systems is that the control signal may be applied with high precision. It is assumed that the dynamics of the system are sufficiently well understood that the required control signal may be easily determined, and that during the control, there will be no unexpected disturbances causing the system to deviate from the desired outcome.

Feedback Control In feedback control systems, the control signal is adjusted based on measurements of the system output. Each measurement is compared to a reference, representing the desired state, and the feedback controller uses this variation to determine a new control signal intended to better track the reference value. Feedback controllers are valued for being adaptive and potentially independent of uncertainty within the model of system dynamics. They are also more resilient in cases where the system is susceptible to disturbances or noise.

Learning Control Learning Control uses a similar feedback system as that employed in feedback control with the difference being how the optimum control signal is evaluated. Instead of determining the control signal in real time, learning control tests a number of control signals in advance and the optimum is then selected for application to the system given the current state. Although Learning Control has a number of disadvantages, it is

often used in situations where the system states change at too high a rate for feedback control.

4.6.5 Properties of control

The choice of feedback systems used in any given scenarios is often based on evaluation of one of the following properties:

Performance This is a general description of the ability of the control system to achieve its desired objective. It may describe the ability of the system to achieve a desired value at a given time, or its ability to follow a given reference signal.

Stability The stability of a control system defines the determinism of its behaviour. Unexpected changes in state or erroneous feedback signals have the potential to induce negative effects within the control system and possibly oscillation beyond control. Where this kind of behaviour can be anticipated in advance, some form of dampening in the control systems might be appropriate.

Robustness The robustness describes the probability of the continued effective operation of the controller despite unexpected events within the system itself, or the surrounding environment.

In practise, control systems must often achieve a compromise. Engineers will choose a feedback mechanism whose properties best satisfy the given requirements.

4.6.6 Feedback Loop Control

A feedback loop controller attempts to reduce the deviation of the system output from the desired reference value by selecting and applying a corrective action that will modify the system/process accordingly. See Figure 4.7

PID Controllers

Once a mechanism is in place to monitor system outputs and reason upon them relative to the desired state, an optimum control signal must be selected and applied. A popular mechanism used to determine this control signal is the the proportional-integral-derivative controller (PID) illustrated in Figure 4.8. A PID controller generates a control value based on the weighted sum of three values; proportional (the reaction to the current error), integral (reaction based on the sum of past errors), and the derivative (reaction relative to rate of change of error). Weighting is achieved by setting the gain of each value resulting in the

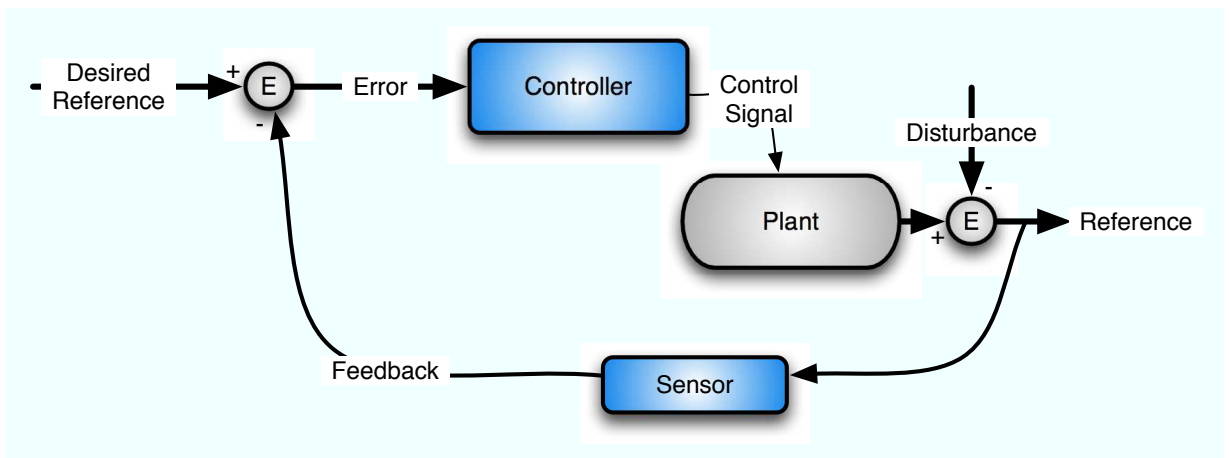


Figure 4.7: Block diagram of a feedback loop controller

tuning of the control algorithm. Tuning might, for example, result in improved response or stability by minimising the controller overshoot and the resulting oscillation of the system output.

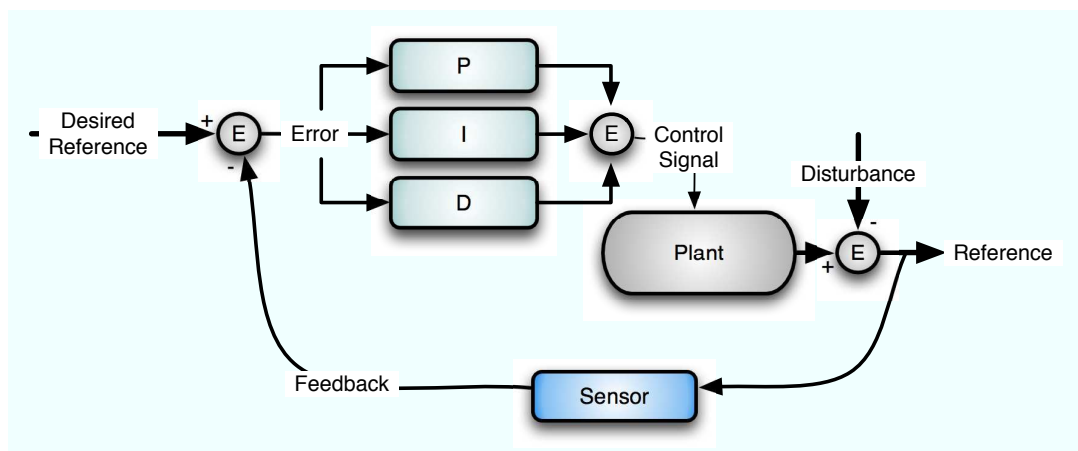


Figure 4.8: Block diagram of a PID controller

4.6.7 Model Predictive Control

Model Predictive Control is an advanced control method that relies on dynamic models of the system under control. The models are used to make predictions about the behaviour of system outputs relative to inputs. The fundamental concept of MPC is that using a

model to select a plant's control system can result in predictable behaviour over a given time period. Modelling of the controlled system is required in order to formulate a description of its dynamics. Models can be generated to represent the free evolution of the system or take into account control signals used to steer that evolution. The system model should provide reliable prediction of the input-output dynamics in such a way that it can be used to design and test a control system. Despite best efforts, no physical system can be precisely modelled as a mathematical system. Some degree of uncertainty will always be present arising from, for example, unpredictable inputs such as noise, or from unpredictable dynamics.

There are several approaches to defining system models. If the governing laws of a system are known, e.g. a mechanical system subject to the laws of Newtonian physics, then the behaviour of the system can be predicted. Another method of model construction is based on experimental analysis of how inputs to the plant affect outputs. In this scenario, the plant is considered a black box, the dynamics of which are unknown. By applying a range of input signals and measuring the corresponding outputs, an impression of the system dynamics can be obtained.

Once the dynamics of a system are understood and a desired trajectory of states is defined, an appropriate signal must be selected and applied in order to ensure that the system evolves in the desired manner. The system model facilitates this by providing an output for each input. The input which corresponds most closely with the output that most closely matches the desired state may then be chosen. When a large number of control signals are available to achieve the desired goals, the selection will often depend on one or more criteria such as minimising the time to outcome or the concord with other objectives.

4.6.8 Control in Grid Operations

In theory, control mechanisms such as those discussed in this section can be used to control any system that has a measurable output, a known ideal value for that output, and a process input that will affect the relevant output. There is considerable scope for the application of control theory to the management of distributed infrastructures, ranging from relatively simple closed-loop resource management to more advanced automated management requiring complex models of the infrastructure and resource dynamics.

An understanding of the fundamentals of control theory will facilitate the construction of precision devices / systems, that accurately perform their intended task despite disturbances from the environment. Further, the field of control theory provides mechanisms for the development of models to aid the description and understanding of complex dynamic systems.

If the future of management is in automation, those responsible for the development of the management systems would be well advised to take advantage of the wealth of research

that exists in the area of control theory. The architecture and prototype implementation of a control system presented in this thesis provides an ideal framework on which to construct advanced management tools to evaluate the application of control theory to infrastructure management.

Chapter 5

Vision

Control is exercised in many ways in society. The great enterprises of federal Europe or America, individual national democracies, public or private institutions, all respond at the highest level to external stimuli by establishing policy that informs the formulation of strategy at the next level, prompting tactical reactions at a lower level which then guide the actions at the leaf level of the enterprise, as in Figure 5.1.

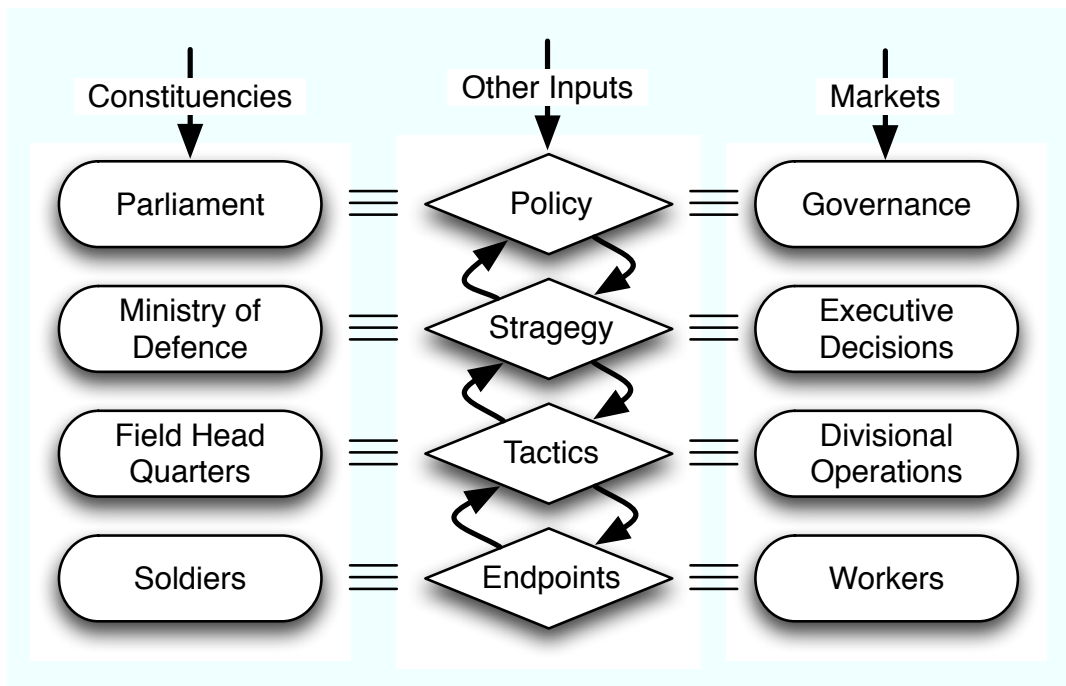


Figure 5.1: Flow of information and control

The previous discussion of distributed grid monitoring and distributed grid management does support such a model, albeit with separate information and control hierarchies, as illustrated in the left hand side of Figure 5.2. Of course, this can be collapsed into one unified hierarchy as shown in the right hand side of Figure 5.2, and it can be clearly seen that this structure adheres to the enterprise model of Figure 5.1

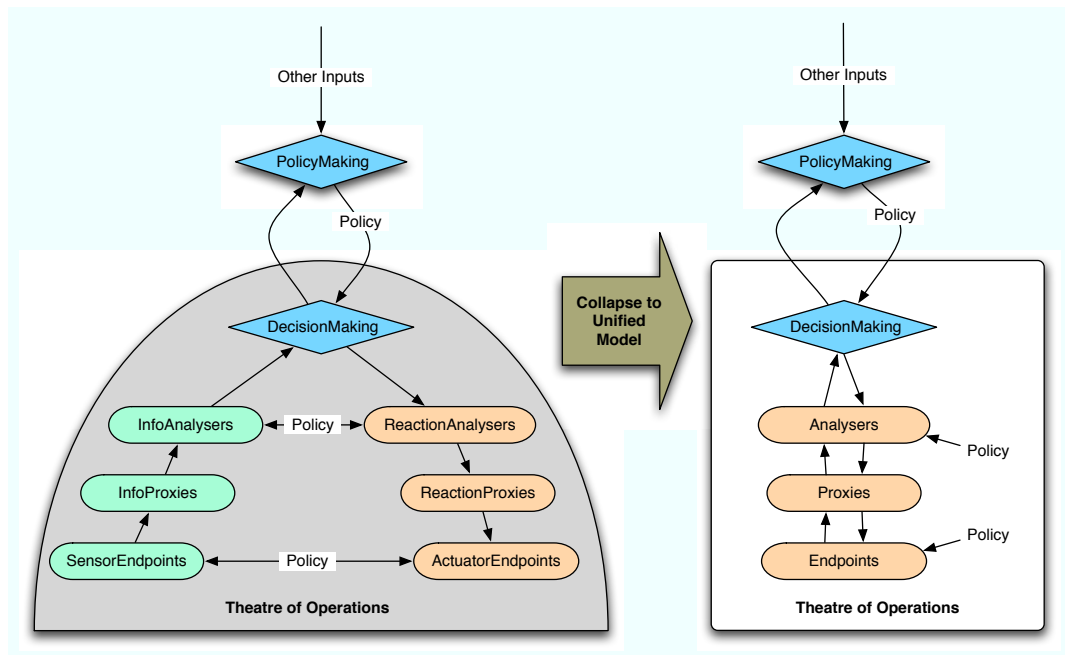


Figure 5.2: Unification of monitoring and control for grid management

5.1 A Vision for Command / Control in Grid Operations

The requirement for command of the operation of a computational resource is proportional to it's the size, complexity, and it's degree of distribution. The management of large centralised systems presents considerably fewer challenges than that of a nationally or internationally distributed computational Grid. While there are a number of advantages to constructing large resources from smaller distributed resources or services, such as those identified in Chapter 2, it introduces new challenges in relation to the management, orchestration, and and co-ordination of each resource and of the resulting infrastructure as a whole.

The responsibilities of command in Grid operations can be divided into two areas. Firstly, it is necessary to ensure the reliable day-to-day operation of the constituent resources through provision, monitoring and management. Second, it is necessary to co-ordinate and support

the combined use of these resources in order for the infrastructure to achieve its potential and goals. While the first is typically a problem of the administration of distributed hardware resources, the second introduces a wide range of additional, more specific roles relating to areas such as user and security management, middleware deployment and maintenance, and the administration and support of Virtual Organisations.

Some of the core functions that could be provided by a grid management system might include:

- Infrastructure and user management / support
Certificates, maintenance, deployment, reporting
- Identification of problems, threat capabilities, intentions
Understand ones own situation and vulnerabilities
- Filtering of information
Too much is as bad as too little. Route alerts to appropriate operators
- Triggering recovery / counter measures
Support automatic or manual resolution of problems
- Real time re-configuration of grid resources
- User Administration
- Job and data management
- Response to status and security events
- Fast, easy access to administrative and status information
- Alerting of faults / intrusions and providing means to rectify them
- Maximising impact of public relations activities and demonstration
- Early warnings to allow pre-emptive / pro-active administration

The design of a command and control system for a Grid computing infrastructure is an exciting and challenging undertaking. It must acquire information and it must exercise control, for example as in Figure 5.3. The I4C system described in Chapter 6 provides a prototype implementation of a system for data acquisition while the Grid4C system described in Chapter 7 provides for both acquisition and control.

The ultimate aim is the dramatic reduction of the management effort required to operate a distributed computing infrastructure, thus improving performance and availability. My thesis is that a grid C⁴I system is the ideal concept to effectively achieve this goal. It can provide complete situational awareness, support decision making, and enable or enact consequent reactions. I propose to construct component-level proofs-of-concept of this thesis, and use these to prove veracity via simple use cases.

The role of a grid monitoring / information system can extend far beyond the simple passive monitoring of grid resources. It can aid the operations centre in their role of system administration, user support, and problem diagnosis. It can provide an authoritative source of all infrastructure related information, allowing for its retrieval, storage, analysis, classification, and dissemination. On the other hand, if badly conceived or implemented, monitoring and information systems can serve to compound the problems faced by the operations team.

A grid control system can facilitate and demonstrate the efficient and effective management of the grid, thereby boosting confidence in the level of operations. It can serve to encourage adoption by users, contributions from resource owners, and interest from funding parties. Again, if badly conceived it may fail in these goals.

Clearly the conception is very important. The ideal concept seems not to be easily obtained, for there are no obviously successful realisations in the grid world. This is the challenge I wish to address, driven by knowledge of the system requirements, born of experience of prior attempts.

5.2 System Requirements

With such a wide variety of distributed projects employing a selection of platforms and middlewares, it would be incorrect to assume that a ‘one-size-fits-all’ approach could be adopted in engineering a management solution. It would also be naive to assume, given the number and scale, that a complete functional analysis of all requirements could be carried out. For this reason, the foundations of such a system must be open, extensible and standards based, in order to facilitate and encourage further development and deployment. The design must satisfy not only the current operational requirements, but also be able to adapt and evolve to meet future requirements.

In general, the grid C⁴I system requirements are a coupled multidimensional space where the system must simultaneously excel in as many dimensions as possible, for example, see Table 5.1.

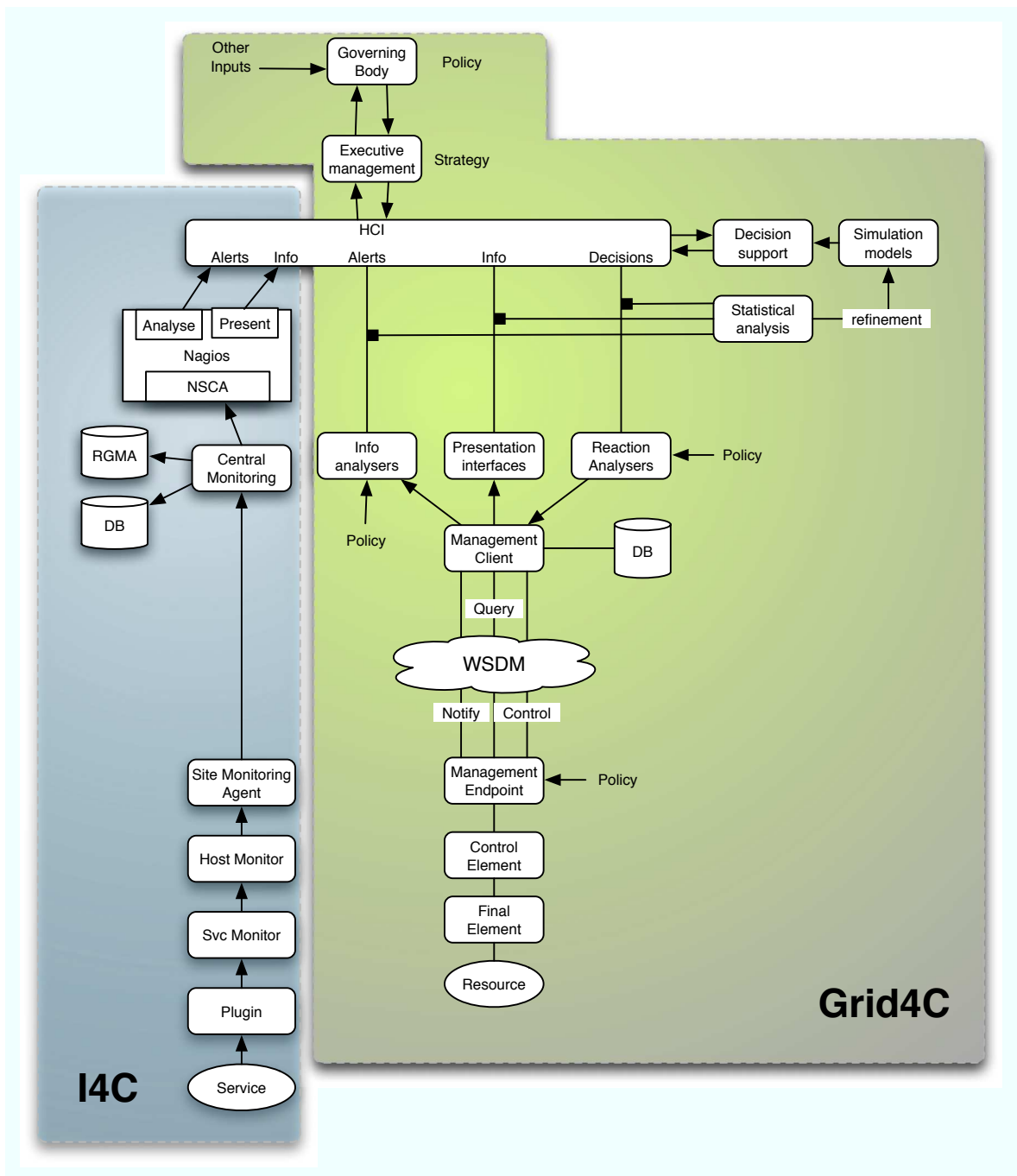


Figure 5.3: A proposed architecture for a grid management and operations support system. Areas of functionality provided by the systems developed by the author of this thesis are illustrated.

Table 5.1: Some of the dimensions of grid C⁴I requirements

Requirement	Details
Open	Interaction between software, hardware and human elements of the system should be well defined and employ a common interface/API. Interoperability, portability, the use of open standards, and community involvement should be encouraged / supported. The system should be designed and maintained by a consensus process.
Open Source	The architecture and source code should be well published / documented and be in the public domain.
Extensible	Extensibility will allow for tailoring the management components to the specific underlying architecture, e.g. in the case of the Grid-Ireland architecture, the management capabilities will be constructed to support the specific configuration of the managed site gateways by including capabilities to manage XEN-based virtual machines and software RAID storage systems. The requirement for extensibility encourages the use of a component based architecture, allowing the ‘plugging-in’ of additional components for monitoring, control, display, and communications.

Interoperable	<p>The control system should interoperate with existing monitoring and fabric management tools aggregating the information provided by them and triggering their execution via a common system interface. The area of grid computing is currently attracting a considerable amount of attention from a large number of research and development groups. This has led to many projects adopting different approaches to satisfy common requirements, often re-inventing or rehashing existing work[ZS05b]. The resulting lack of standards has resulted in a variety of tools which cannot easily be combined or extended to form more complete management solutions. It is not possible for developers or administrators to easily combine the best features of each into a system that satisfies their individual needs. One lesson that the developers and operators of grid infrastructures might learn is derived from the integration difficulties experienced within the telecommunications sector: they should differentiate their offerings based on service rather than technology. The increasingly important subject of interoperability is often hampered by each project developing their own specific means of achieving a given goal. With this in mind it seems appropriate to investigate and evaluate standard tools and mechanisms for data communications and representation when designing or planning any contribution to the field.</p> <p>The management system should also support a wide range of platforms and deployment models. Additional functionality should be supported in situations where virtualisation is employed, e.g. with XEN[BDF⁺03] or VMware[Owe07] management capabilities. It should allow integration with existing, widely-used vendor supported management tools such as HP OpenView[Mul96] or Dell OpenManage[Del].</p>
Standards-based	<p>The importance of the adoption of standards must be emphasised. Standard event formats should be used to ensure a complete and common operational picture and to facilitate the correlation of events. Every system entity capable of interaction should generate logs and events which, if not generated in the common format would be translated into the the standard representation. Standard protocols should be used, and so on. Only in this way can the considerable investment yield tools with commensurate lifespans.</p>

Secure	<p>Since the proposed system is to be capable of executing tasks normally carried out only by users with administrative privileges, security is very obviously of utmost importance. Invocation of the management web services must be restricted to authorised users or processes and all communication should take place over secure channels.</p> <p>The W3C Web Services Architecture [BHM⁺04] identifies a number of important requirements for end-to-end security. These include:</p> <p>Authentication Allows for the verification of the identity of both clients and services.</p> <p>Authorisation Used to control the access of authenticated entities to secured resources.</p> <p>Confidentiality Ensures that data in transit is protected.</p> <p>Integrity Ensures that message data cannot be modified without detection</p> <p>Non-repudiation Provides evidence of the occurrence of a transaction so that no party can later deny its involvement.</p> <p>Each of these consideration should be addressed at the relevant levels within the information and control system. Access to system information and control interfaces should be restricted to authorised users and processes, and audit trails should be maintained in order to trace access and behaviour.</p>
Flexible	<p>In so far as is possible, the underlying architecture should be free of fixed constraints. It should be flexible enough to support a variety of administrative operations. Through the use of a plugin, component-based architecture and standards-based interfaces it should be possible to extend and customise the control infrastructure to meet specific needs. Similarly, it should be possible to develop custom control interfaces provided all security concerns are satisfied.</p>
Modular	<p>The system should be composed of self-contained, standardised units exposing common defined interfaces. The alteration or replacement of components should not adversely affect its operation.</p>
Layered	<p>Modular components should be organised in to clearly defined layers with functionality implemented at the most appropriate tier</p>
Scalable	<p>The system should scale to successfully meet the requirements of very large and complex systems.</p>

Robust	The system should be sufficiently robust to ensure that it can continue to operate and meet user expectation despite difficult operating conditions or circumstances. It must be difficult for external factors to impair or corrupt its operation. This requirement must be addressed at a component level also to ensure that dependencies are properly handled.
Resilient	In the event of a component failure or a security breach of the system itself, it must be capable of conducting itself in a graceful and proper manner. It should recover if at all possible, otherwise the control necessary to resolve the situation will be absent. If recovery is not possible, efforts should be made to ensure that the outage is restricted to specific components rather than causing a chain of failures in dependent components.
Responsive	Execution of management operations should occur in near real-time and provide adequate feedback regarding their progress or output. In order to function as a control plane, it should be independent of its managed resources, fast, reliable and 'always-on'.
Easy to use	The grid service management solution should provide an intuitive and relatively transparent interface, readily accessible to the members of an operation team. The client components should be designed in such a way that they might easily be incorporated into a variety of user interfaces, including mobile, web-based, and advanced visualisation tools.
Innovative	The system should employ leading yet stable technologies and support the exploration of new approaches to meet user requirements.
Adaptive	The architecture should be capable of adapting to differing environments and requirements.
Available	Efforts should be made to ensure its availability irrespective of the status of other resources within the administrative domain.

Easily configured	In order to facilitate deployment and administration, it should be possible to define a generic set of operations to be supported by the system for each of the resource types at each of the intended target sites. For example, the properties and operations defined for a Compute Element, such as querying current jobs or starting the resource allocation manager, are likely to differ little from site to site. It should therefore be possible to deploy a largely pre-configured system with minimal reconfiguration and automatic detection of configuration variables where possible. Where possible, deployment via a fabric management system, such as Quattor[LLM ⁺ 04] should be supported. Plug-n-play configuration is also very desirable.
Lightweight	The design must aim to minimise the impact of the management system on the normal operation of the managed resources. Lightweight components will serve to minimise their impact and footprint and thereby maximise it's responsiveness.
Shared	The system should support multiple users and federations
Cross-domain	The system should be capable of operating across multiple administrative domains while observing any security policies that might be in place and ensuring its own integrity.
VO-enabled	The system should be aware of Virtual Organisation membership and support access control/filtering based upon it.

5.3 Human Machine Interface

The design of a visualisation and control interface for such a complex system as a computational grid is no mean feat. A complete management system could employ a wide selection of interactive displays ranging from dedicated control rooms sporting large format displays, as in Figure 5.4, to Web interfaces and mobile devices. Control signals might be issued from operators consoles or portable network enabled devices. Feedback could take the form of audible or visible messages or even haptic devices.

Displays might take the form of simple text based alerts, two dimensional pictographic representations or complex three dimensional models. In the case of three dimensional visualisation, the rendering of the models might be farmed off to dedicated clusters.

Displays could support multiple granularities, displaying information that is pertinent to that entity or level of abstraction. Examples of such levels might include: grid, site, machine, service user, job, etc.

The arrangement of the entities for display might be based on some abstract relationship. Simple examples might include geographical location, network topology, or the interaction of entities within a services based architecture. Where three dimensional displays are employed, interaction with the



Figure 5.4: An example of a dedicated control facility for system operations

model could reveal additional information. A simple example of this would be displaying the current state when viewed head on and displaying the history of states when rotated and viewed from the side. Heads-up displays might be included in the form of context sensitive overlays introduced to the viewing window in response to user interaction or navigation.

Rendering of streams of events in three dimensions could be included, illustrating relationships between events where they exist, and allowing the operators to manipulate or navigate within the view and perhaps take cross-sections or time slices to aid event correlation and root cause analysis.

The display system might also include a ‘screen saver’ type of functionality where, in the event of no user intervention over a given interval, the display self-navigates among the monitored entities displaying current information for each in turn. In this situation, the entities selected for display might be randomly chosen or selected based on their current level of utilisation or the number status alerts etc. Critical notifications might also be allowed to take precedence and be given display priority. These might be brought to the attention of the operators using scrolled messaging, heads-up alerting and perhaps, in the event of sever breaches, audible alarms.

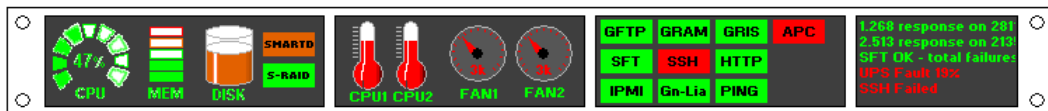


Figure 5.5: An example of a ‘V-Rack’ virtual instrumentation display

Considering the fact that many grid operators will be accustomed to hands-on systems administration and interaction with physical machines, virtual instrumentation type displays could be used to quickly communicate the relevant information for resources such as servers and network components. An example of this is the ‘V-Rack’, a mock up of which is shown in Figure 5.5. This illustrates a simple and intuitive way to convey the current state of a remote server and it’s hosted services. For example the window on the right of the display could display the outcome of status and security checks, colour-coded to reflect importance. This use of virtual instrumentation might be extended to display V-Racks contained within ‘V-Rack Enclosures’ . This graphical representation of a hosting environment could display information such as the location of the racks or servers, in addition to environmental monitoring information such as temperature, the status of the electrical supply, or the time the enclosure was last opened etc.

In the event of security alerts, the control system would allow operators to visualise the type, location, and progress of attacks. The interface would provide a fast and easy means to ‘drill down’ from initial notifications to more detailed information in order to discover the cause of, and correlation between, events. Triggers would be provided to execute pre-defined reactions and countermeasures in addition to providing the flexibility to counter intrusions dynamically, in real-time. The location, paths, and effects of an intrusion might be visualised using a number of methods taking into account geographical location, network topology, and time.

Every event within the infrastructure such as job states, machine restarts, software deployments,

and component failures, should be logged and disseminated using standard formats and notification APIs. As the all-knowing heart of the operations effort, the objective is total awareness. All management activities would also be logged using the same format, facilitating the construction of a knowledge base of operations techniques based on the actions of administrators in response to particular situations. This knowledge base could then be used for training purposes, technical support, and automation.

Despite the heavily technical nature of the majority of the functionality provided by the system, the social / human aspects to the operation of a Grid must not be ignored. User support might be enhanced through integration of the C⁴I system with help systems such as RT, providing summaries of the support activities and perhaps a means to evaluate trouble tickets.

In addition, the sensors of the management system are ideally placed to provide information to further collaboration among users. A simple example of this might be sensors deployed to the various grid User Interfaces from which information might be fed back into user portals to allow members of a Virtual Organisation and grid operators view a list of other members who are currently logged in or active in some way.

The control interface might allow grid operators to combine management operation into workflows and submit these workflows to the Command and Control system. Indeed a graphical workflow editor might be included allowing operational procedures to be assembled from a selection of predefined functions and executed on the fly or archived for later use. These workflows would allow system hierarchies and dependencies to be taken into account during management operations.

High-level decision support might also be provided, allowing Grid operators to model and simulate the effects of management operations prior to their execution of the production infrastructure.

Another component of the interface might allow drag and drop job submission, where a job could be selected from a pool of pre-defined test jobs and dropped onto a resource or broker. Job feedback would be then constantly updated (based on job-status or the information from the RB) to show the status and progression of the job displayed as a graph of job states. This graph could then be interacted with to obtain further information such as the time at which a job entered a given state, or the number of transitions between certain states before being accepted for execution. Such a display could help Grid administrators monitor jobs and diagnose problems on the fly. A simplified version of the job status graph might also be included in job submission portals to provide users with constant feedback. Interfaces might be provided to ‘replay’ problematic job submissions in a single step-through fashion in order to identify problems and test for their successful resolution.

Such detailed feedback and situational awareness could be used to evaluate and fine tune the configuration of future middleware components and systems such as autonomic managers and Social Grid Agents[PKC05]. Once automation becomes more widespread, such a window on the infrastructure could become even more important in supporting the supervisory relationship between automated agents and human operators.

All of these requirements illustrate the importance of the visual elements in ensuring the optimal effectiveness and usability of a grid C⁴I system.

5.4 The Case for Automated Management

Experience has shown that a large proportion of services outages are caused by human error. EGEE working groups have ascertained, based on their experience, that a large majority of infrastructure problems are caused by fabric management failures at the individual sites. The number of constituent resources and services within grid infrastructures creates a level of complexity that can overwhelm manual IT management procedures. Automated management needs to be investigated as a solution to some of these problems, given its potential for increased availability and reduced costs. Automatic reconfiguration of resources might provide service continuity in the event of a component failure or, more compellingly, optimum revenue/utilisation by automatic alignment of resources to meet current demand.

Chapter 6

Grid Monitoring with I4C

6.1 Introduction

This chapter presents the I4C monitoring system that was designed by the author and deployed on the Grid-Ireland infrastructure. The principles of and justifications for the monitoring system are discussed. A technical overview of the sensors, communication and presentation systems which make the aggregate real-time status information available to the members of the operations team is presented. The monitoring information is presented in such a manner as to facilitate the management of the infrastructure in response to status and security events, and provide a tool to maximise the impact of public relations activities and demonstrations. This system has been in continuous use since 2004, performing approximately 15,000 service checks per day.

6.2 Motivations

Early grid service monitoring systems, such as MapCentre[BF02] and Ganglia[Gan], offered valuable but limited functionality, required complex configuration, and were employed with limited degrees of success on our particular infrastructure[CWO05]. For example, for both tools, monitoring traffic was often denied access into the remote networks hosting the monitored grid resources, greatly reducing the usefulness of these tools. The unsuitability of these systems prompted the investigation and assessment of alternatives.

Several existing host and network monitoring tools, including some supporting distributed monitoring, were assessed but did not offer a suitable solution. Many were intended for deployment within a single administrative domain. While changes to network settings, such as firewall rules, can be quickly authorised and actioned from within such environments, this is not true from outside the domain. From the viewpoint of an operations team the majority of the monitored resources reside within remote administrative domains which are beyond their control. In cases where the resources are remotely managed the remote network still remains beyond control; this makes efficient centralised monitoring of the resources impossible. This is the case for Grid-Ireland, which has a notably inte-

grated remotely managed dedicated grid infrastructure, and in consequence a decision was taken to undertake research into this rather specialist monitoring field.

The monitoring solution described below employs a combination of distributed and centralised monitoring sensors along with the aggregation of information from existing monitoring tools and is specifically designed to speed the response to status and security events.

The immediate aims were to develop and deploy an extensible monitoring framework which would overcome the problems traditionally associated with the monitoring of distributed resources spanning multiple administrative domains to satisfy the information requirements of the operations team. The framework would take advantage of the considerable existing body of work carried out in the area of grid monitoring by aggregating existing systems and tools where provided, and allowing for the development of new sensors where none exist. A key concern was the administrative overhead involved in the deployment and management of the monitoring solution. While there were initially only six sites to monitor, the introduction of an additional twelve sites highlighted the need for an efficient solution.

6.3 Centralised vs. Agent-based Monitoring

Remote entities are typically monitored using one of two approaches; a centralised approach where one or more processes executing at a single location are responsible for the monitoring of all entities or a federated approach involving multiple distributed processes and sensors. In a centralised system, all monitoring processes are executed from a single location and attempt to determine the status of monitored entities either through the establishment of some form of connection with the entity or by performing some operation involving it. Distributed monitoring architectures employ monitoring processes on or closer to the monitored entity. These processes, often referred to as agents, perform the required tests and report the outcomes, through push or pull mechanisms, to the interested parties. The area of software agents is one of active research and many types of agents have been identified or proposed. These range from simple processes to complex intelligent autonomous mobile elements of software. It is argued that the popularity of the term has led to its misuse and it might therefore be appropriate to present an explanation of what should be inferred by the term in the context of this work. We will use the term ‘agent’ to refer to an independent software component that performs operations on behalf of a user or other piece of software and exhibits some degree of autonomic behaviour.

Both methods of monitoring have advantages and the choice of which is best suited to a given situation is dependent on a number of factors including the size and nature of the systems, the required information and the degree of control that is required. It is often claimed that one approach is superior to the other, however the merits of each must be evaluated with respect to the individual systems and monitoring requirements. While it is true that the deployment of distributed monitoring processes can incur additional effort it is not necessarily true that the configuration need be any more complex. Unfortunately, centralised monitoring typically implies a more limited depth of data gathering than is possible with remote agents. In addition the potential for the management and control of remote entities through interaction with the distributed agents makes them an attractive choice for complex

infrastructures. Also the scale of distribution can vary. The requirement for the installation of a software agent on each monitored host is dependent on the motivation for the monitoring activity and the amount of information pertaining to each host that is required. If the motivation for the adoption of an agent-based approach is merely to overcome the limitations imposed by network access policies, it may be sufficient to deploy a single agent to each network segment, domain, or site.

A combination of both approaches might be considered the best solution in terms of grid monitoring. Remote agents provide a level of access and control that would otherwise be unfeasible, yet there are a number of advantages offered by a centralised monitoring approach. It is of limited value to know the internal status information pertaining to a particular resource if we cannot determine whether or not the resource is accessible and usable through the normal operational mechanisms. Furthermore, in the interest of consistency and conceptual simplicity a central monitoring process might be implemented as a monitoring agent deployed at the Operations Centre.

6.4 Architecture

The architecture obeys the general principle of ensuring scalability by devolving work to the monitored nodes where possible. This is achieved through the combined use of monitoring agents local to each site, plus a central monitoring process, see Figure 6.1. Traditional systems like Nagios[Nag] are configured at the site for the site, but this becomes an excessive burden. Here a more efficient integrated approach is taken.

6.4.1 Monitoring Agents

A single Monitoring Agent, implemented in Java, is deployed to each of the sites with the responsibility of executing service checks for hosts and services at that site. In addition to the distributed agents, a central agent running at the Operations Centre is used to determine metrics such as site connectivity, and can optionally be configured to aggregate information from additional sources such as Site Functionality Tests[sft], Grid Intrusion Detection Systems[KC05] and existing grid information systems, so that it may be included in the analysis and presentation process.

Configuration information specific to each site, host, or service is stored in a central database and made available to the agents via a web service, thereby minimising the amount of local site-specific configuration required. As described in Figure 6.2, upon start-up a remote agent queries the central configuration server for details of the hosts and services that it should monitor at its site. It then creates local monitoring objects that determine the availability of the specific service through the use of existing or custom service checking objects. At intervals specific to each service, the monitoring objects report their status to interested subscribers via a local publisher. One such subscriber is the central monitoring service, where the results are archived and, if necessary, brought to the attention of an operator. The use of remote sensors communicating over established ports (HTTPS) not only distributes the overhead associated with the monitoring operations but also eliminates many of the connectivity problems commonly experienced with centralised network monitoring.

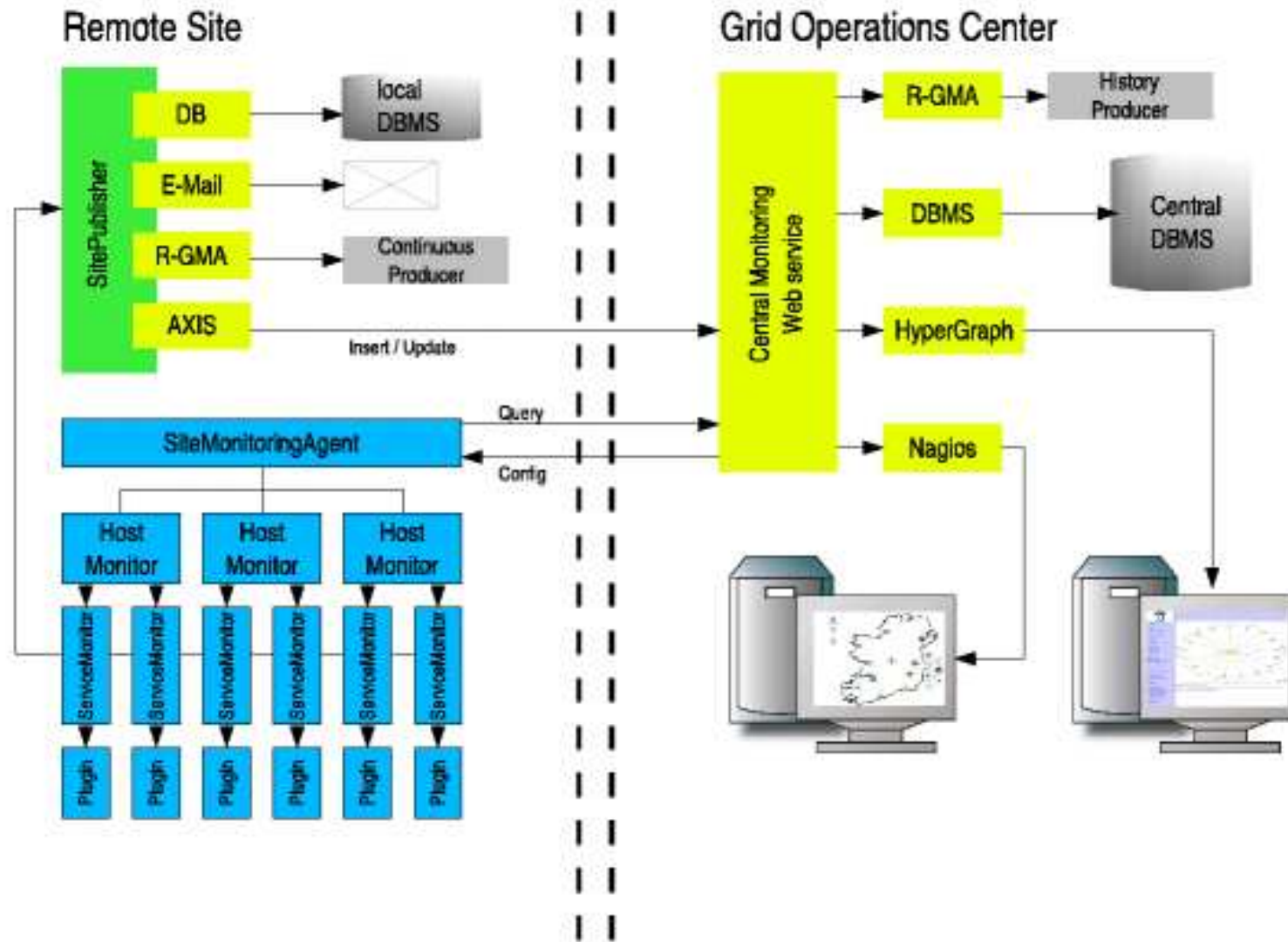


Figure 6.1: Agent-based monitoring architecture

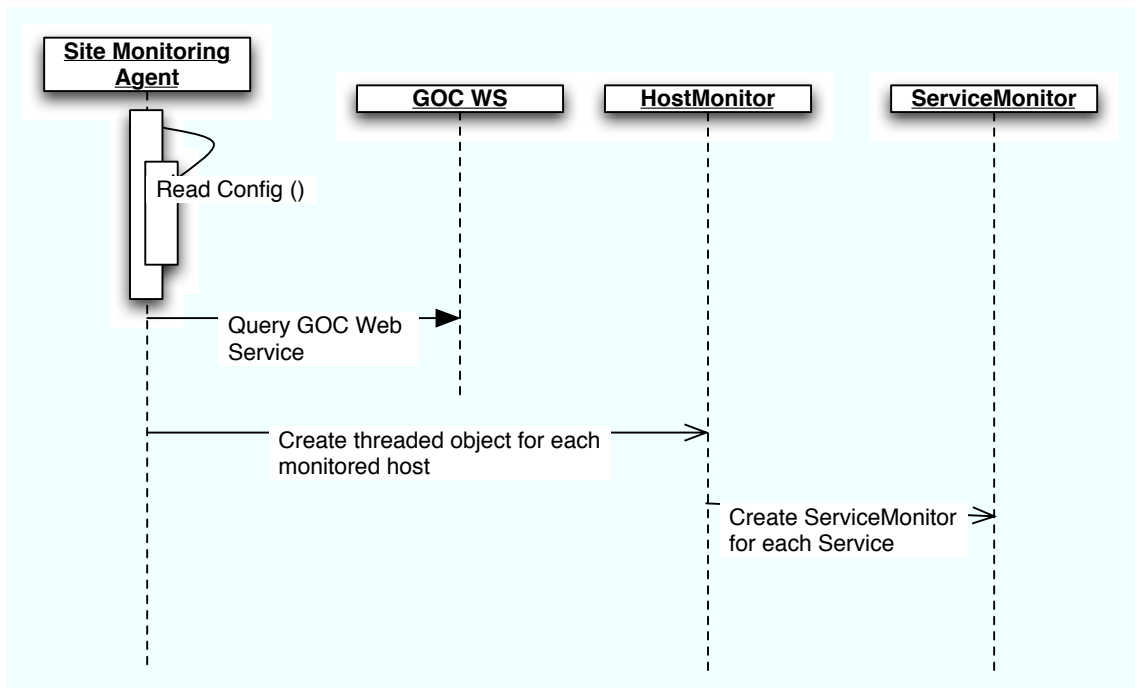


Figure 6.2: Monitoring Agent start-up sequence

In order to satisfy the requirement for extensibility, the status of each monitored entity is determined by the agent using plugin modules implemented as Java objects or as Nagios-compliant plugins written in either Perl, C, or as a shell script. This allows the agent to take advantage of a wide range of existing service checks developed for the Nagios project[Nag]. Examples of such custom plugins include log parsing mechanisms, information system consumers and mechanisms to query a Ganglia[Gan] monitoring daemon for system load information.

The collected status information includes but is by no means limited to:

- The TCP connectivity to grid service ports such as GRAM, GRIS etc.
- Network reachability of the machines comprising the grid gateway
- Queue / Job activity on the Compute Elements
- The output of the Site Functionality Tests
- Grid Portal availability
- SSH reachability of the managed hosts
- Load information obtained via the Ganglia gmon daemon

6.4.2 Configuration Database

The configuration information defining what hosts and services should be monitored at each of the sites can be represented in text files or database tables, etc. In this case, advantage is taken of the database schema for version 1 of the LCG GOCDB[GOC] maintained at Rutherford Appleton Laboratory, UK, with several extensions. Hence the configuration information is stored in a MySQL database at the Operations Centre. A sub-set of the configuration database schema is illustrated in Figure 6.3. This shows the data stored for each host, service, and the data required to relate services and hosts for monitoring purposes.

Nodes Table

Field	Type	Attributes	Null	Default
nodeID	mediumint(8)	UNSIGNED	No	
siteID	mediumint(8)	UNSIGNED	No	0
nodetype	varchar(50)		No	
nodetype2	varchar(50)		Yes	NULL
hostname	varchar(50)		No	
domain	varchar(200)		Yes	NULL
nagios_name	varchar(255)		Yes	NULL
ip	varchar(15)		Yes	NULL
grp	varchar(20)		Yes	LCG-1
hostdn	varchar(255)		Yes	NULL
system	varchar(50)		Yes	NULL
middleware	varchar(50)		Yes	NULL
install	varchar(50)		Yes	NULL
status	varchar(50)		Yes	NULL
operation	varchar(50)		Yes	NULL
comment	varchar(250)		Yes	NULL
monitor	enum('Y', 'N')		No	Y

Services Table

Field	Type	Attributes	Null	Default
serviceID	mediumint(9)		No	
serviceName	varchar(30)		No	
protocol	enum('TCP', 'UDP', 'PING', 'HTTP')		Yes	NULL
port	smallint(6)		UNSIGNED No	0

Node-Services Mapping

Field	Type	Attributes	Null	Default
hostID	mediumint(9)		No	0
serviceID	mediumint(9)		No	2
checkPeriod	varchar(128)		No	24x7
maxAttempts	int(11)		No	3
checkInterval	int(11)		No	5
retryInterval	int(11)		No	3
contactGroups	varchar(255)		No	

Figure 6.3: A section of the Configuration Database Schema

6.4.3 Central Monitoring Webservice

The central monitoring process, for entirely pragmatic reasons, needs to perform the following functions:

- provision of configuration information to each remote agent
- aggregation of information from remote agents and existing monitoring tools
- archiving and publishing of monitoring/status information

It is implemented as an AXIS[Axi] web service hosted at the Grid Operations Centre. Configuration information is made available to the monitoring agents via a web service employing the Jakarta

Database Connection Pooling mechanisms to query the MySQL database. The use of this pooling mechanism to manage the database connections is an important part of ensuring the scalability of the central Webservice. Since the Web Services are invoked frequently to transfer small amounts of information, establishing a database connection per session would result in slower response times and greater server load.

Using publication mechanisms similar to those embedded within the monitoring agents, status information reported back to the central server is made available to consumer APIs, via re-publisher mechanisms, or passed on to registered listeners. Currently implemented listeners include the Nagios NSCA client, JDBC, XML-RPC, and R-GMA[BCC⁺02] mechanisms. In addition to the central publisher mechanism, the status reports are cached in memory to provide rapid access to the latest metrics for all monitored entities. Further optimizations are being explored.

6.5 Presentation and Alerting

Just as in many traditional Operations Centres, monitoring information is brought to the attention of the operations team in real-time by means of wall-mounted TFT displays, web-based tools and email/messaging alerting systems. Currently four 18 inch LCD display panels are used at the Operations Centre, but soon a number of 40 inch displays will replace these. In the near future, monitoring information and alerts will also be made available to the members of the operations team remotely through the use of mobile devices running web browsers and thin clients.

6.5.1 Reports/Displays/Alerts

The Nagios host and network monitoring system is employed at the operations centre for presentation and alerting, solely in order to take advantage of its web based display and reporting functionality in addition to its comprehensive alerting mechanisms. The configuration is purely passive in that Nagios is not responsible for any service monitoring directly but rather the information gathered from the remote sensors is fed into Nagios by means of the Nagios Service Check Acceptor server daemon, allowing the status of defined hosts and services to be updated.

6.5.2 Navigation of Information

For many existing host and network monitoring tools, navigation of the information is rather neglected. Not only is it desirable that this be easy, it should also be fast, so that an experienced operator can exhibit a “musicians touch”, particularly in emergency situations. The most promising approach thus far assumes tree structures.

HyperGraph[Hyp] is an open source project which provides Java code to work with hyperbolic geometry and in particular hyperbolic trees. Its extensive API facilitates the visualisation of graphs and hyperbolic trees which are extremely useful when dealing with large volumes of data in a hierarchical structure.

Nagios

General

- Home
- Documentation

Monitoring

- Tactical Overview
- Service Detail
- Host Detail
- Hostgroup Overview
- Hostgroup Summary
- Hostgroup Grid
- Servicegroup Overview
- Servicegroup Summary
- Servicegroup Grid
- Status Map
- 3-D Status Map
- Service Problems
- Host Problems
- Network Outages

Show Host:

- Comments
- Downtime
- Process Info
- Performance Info
- Scheduling Queue

Reporting

- Trends
- Availability
- Alert Histogram
- Alert History
- Alert Summary
- Notifications
- Event Log

Configuration

- View Config

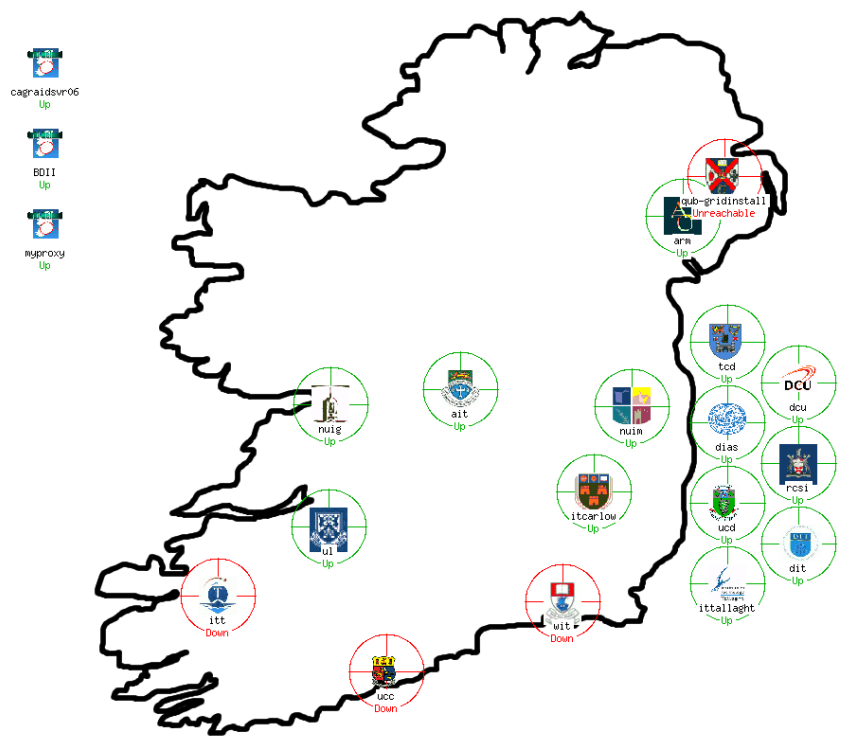


Figure 6.4: Example Nagios Display

Current Network Status
 Last Updated: Wed Jun 22 11:19:09 IST 2005
 Updated every 90 seconds
 Nagios® - www.nagios.org
 Logged in as *guest*

[View Service Status Detail For All Host Groups](#)
[View Host Status Detail For This Host Group](#)
[View Status Overview For This Host Group](#)
[View Status Summary For This Host Group](#)
[View Status Grid For This Host Group](#)

Host Status Totals

Up	Down	Unreachable	Pending
4	0	0	0
All Problems		All Types	
0		4	

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
11	0	0	0	0
All Problems		All Types		
0		11		

Service Status Details For Host Group 'tcd'

Host	Service	Status	Last Check	Duration	Attempt	Status Information
tcd-gridgate	GFTP	OK	06-08-2005 21:20:33	40d 18h 39m 26s	1/3	TCP OK - 0.459 second response time on port 2811
	GRAM	OK	06-08-2005 21:20:49	40d 18h 39m 26s	1/3	TCP OK - 0.387 second response time on port 2119
	GRIS	OK	06-08-2005 21:20:58	40d 18h 39m 26s	1/3	TCP OK - 0.345 second response time on port 2135
	SSH	OK	06-22-2005 11:15:38	16d 7h 7m 10s	1/3	SSH OK - OpenSSH_3.6.1p2-CERN20030917 (protocol 1.99)
tcd-gridinstall	APC	OK	06-22-2005 11:16:27	0d 7h 38m 9s	1/3	IN 228.9v (208.0 - 253.0) OUT 228.9v 50.0 Hz, Load 38.3 Percent, Charge 100.0 Percent, Batt 27.4v, Time left 33.0 mins, Internal Temp 31.5 C
	HTTP	OK	06-22-2005 11:17:09	49d 20h 40m 43s	1/3	HTTP ok: HTTP/1.1 200 OK - 0.003 second response time
	SSH	OK	06-22-2005 11:15:38	22d 16h 30m 58s	1/3	SSH OK - OpenSSH_3.1p1 (protocol 1.99)
tcd-gridstore	GFTP	OK	06-08-2005 21:22:57	49d 20h 41m 0s	1/3	TCP OK - 0.236 second response time on port 2811
	SSH	OK	06-22-2005 11:15:38	54d 17h 29m 15s	1/3	SSH OK - OpenSSH_3.6.1p2-CERN20030917 (protocol 1.99)
tcd-gridui	HTTPS	OK	06-22-2005 11:17:37	0d 21h 6m 42s	1/3	HTTP ok: HTTP/1.1 200 OK - 0.071 second response time
	SSH	OK	06-22-2005 11:15:38	54d 17h 31m 56s	1/3	SSH OK - OpenSSH_3.6.1p2-CERN20030917 (protocol 1.99)

Figure 6.5: Detailed service monitoring information

The HyperGraph API is used as an example of an alternative methodology for the display of status information, allowing the identification of problems ‘at a glance’ and manipulating the graph for an improved view of the necessary information. Navigation of the infrastructure tree is very fast, which is useful when attempting to find the location of an urgent problem.

In this case (see Figure 6.6), the colour of the edges between the Operations Centre node and the individual site nodes is determined by the network reachability of the gateway machine at that site. The status of the site nodes themselves is determined by the latest results of the Site Functionality Tests for that site. The colour of a host node is determined by the maximum alert value of the services monitored on that host. Nodes representing services are coloured based on the output of the plugin for that service. The graph is constructed in such a way that hovering the pointer over a node causes further information pertaining to that node to be displayed. Table 6.1 defines the colour code used in the graph. Although very useful, HyperGraph is not the panacea. Its assumption of tree structures is quite restrictive.

Colour	Connotation
Green	No problems or warnings
Amber	Warnings exist for this host or service
Red	Errors or critical warnings exist
Blue	The information for this host is considered stale
Grey	The status could not be determined

Table 6.1: HyperGraph Infrastructure Monitoring Key

6.5.3 Republishing for use by higher-level software

Of course, alerts are themselves an interesting data set, amenable to extraction of useful information regarding stability, behaviour, etc. More in-depth analysis and intelligent alerting is made possible through the publication and aggregation of the monitoring information into the R-GMA system[CGM⁺03] and the use of R-GMA consumers in the form of custom alert analysers[CK]. These analysers can trigger event handlers and notification mechanisms based on queries made on the R-GMA producers.

It is important to recognise that the power of the SQL queries within individual analysers may be extended by building trees or series of dependent analysers, allowing complex analysis of the monitoring information. Analysers may be more than mere filters of information. They may contain advanced processing logic and include historical analysis, providing alert escalation mechanisms or the investigation of correlations among service outages and higher level alerts or security events etc.

Examples of event handlers and notification mechanisms also prototyped during my development of the system are outlined in table 6.2

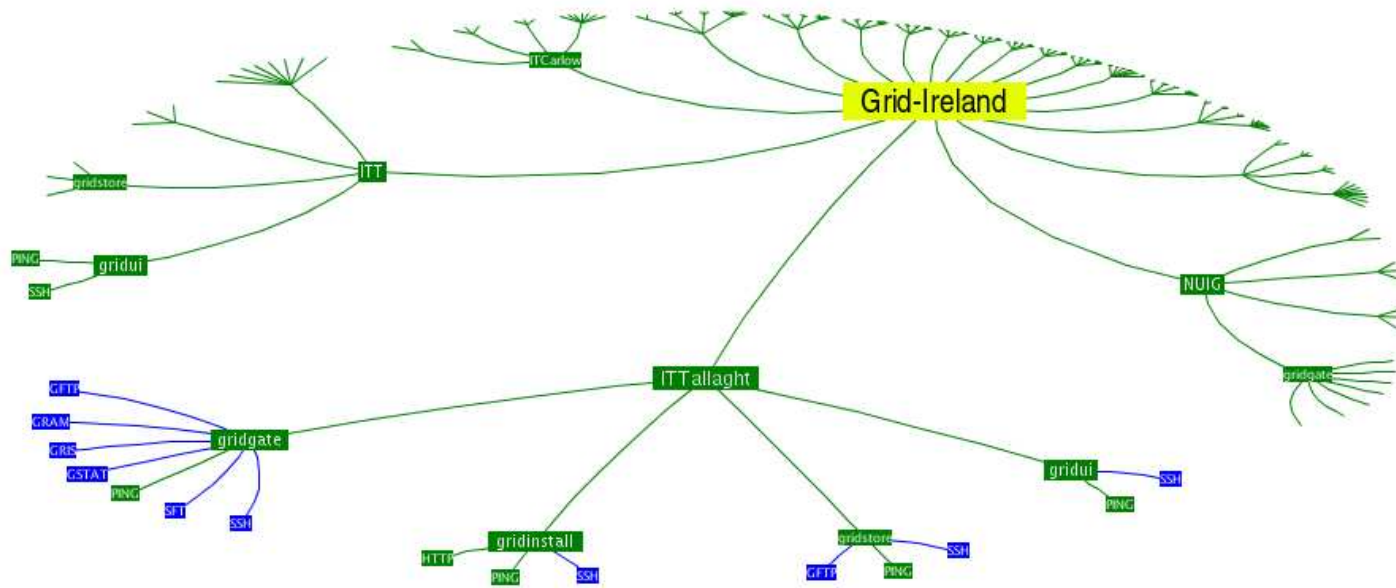


Figure 6.6: A snapshot of the Hypergraph Display of Infrastructure Status in mid-navigation

Handler Name	Description
Console Handler	Prints the alert to an open console
JDBC Handler	Invokes an insert statement on a JDBC resource
RGMA Handler	Invokes an insert statement on an RGMA producer
MAILTO Handler	Sends an email to a user/operator detailing alert
XML-RPC Handler	Invokes a method on an XML-RPC server

Table 6.2: Example Analyser event handlers

6.6 Deployment

The system described has been deployed across 18 Grid-Ireland sites with the task of monitoring almost 200 services on over 70 hosts. As indicated above, up to 15,000 service/host check results are reported to the central monitoring service each day. It has achieved its objectives and has proven to be a valuable tool for the operations team. Its extensible nature has been tested as additional checks are regularly requested.

6.7 Conclusions

This chapter has outlined some of the difficulties associated with the effective monitoring of distributed computing infrastructures and has presented an improved approach to the execution and management of grid service monitoring. When first developed in 2004 this represented a new approach. It is now an open source project [I4Cb] and has attracted interest from several parties. I4C has demonstrated the combined use of remote and centralised monitoring mechanisms along with the aggregation of existing information systems in order to satisfy the information requirements of a Grid Operations Centre. The lightweight Java agents are easily deployed at remote sites requiring minimal local configuration. Configurations are easily managed through Web interfaces to a central configuration database. The use of standard communication protocols has resulted in a reliable system, capable of operating within the constraints of tightly managed network security due to its use of standard ports (e.g. site firewall rules for these ports are not typically subject to change following security audits). Where appropriate, existing tools have been used in a flexible and extensible manner, improving the efficiency of the development effort and the overall usefulness of the system. The archiving and republishing of the monitoring information makes this a valuable component on which to base future work, and we will refer to this historical data later in this thesis.

Table 6.3: Grid monitoring system requirements

Requirement	Compliance	Details
Open	✓	The architecture and interfaces are documented and published

Open Source	✓	The system is in the public domain as a Source Forge project.
Extensible	✓	The plug-in architecture allows additional checks to be added as the need arises
Interoperable	✓	Re-publishing options and the use of standard plug-in interface allow for some degree of interoperability
Standards-based	X	Although the plug-in architecture employs the standard Nagios interface, the communications between the monitoring agents and the central server are performed using custom web services.
Secure	✓	All communications takes place over HTTPS. Configuration and monitoring information reside securely within the operations centre.
Flexible	X	The limited schema for data representation and transfer limit the flexibility of the architecture
Modular	✓	Agents and plug-ins may be added, removed, or modified provided the interfaces remain unchanged
Layered	✓	A multi-tier architecture is employed with clear distinctions between functional levels
Scalable	-	Experience has demonstrated an ability to scale well. The central management server could present a bottleneck in very large installations resulting in reduced performance.
Robust	-	Deployment has shown the system to be stable and reliable. The central monitoring service could however represent a single point of failure and should be replicated
Resilient	-	The system can tolerate changes and failures of certain components but no support for self management or repair in included.
Responsive	X	While the navigation of information via the web interfaces could be considered responsive (especially hypergraph), the remainder of the system is largely managed by timing and polling and not in response to user interaction.
Easy to use	✓	Deployment of the monitoring agents is simplified by the central configuration mechanism. Flexible re-publishing and a variety of user interfaces provide easy access to monitoring information

Innovative	✓	At the time of its conception and development the coupling of existing monitoring sensors and display mechanisms with a web service middle tier represented a novel approach to satisfying the requirements of distributed service monitoring.
Adaptive	X	Although the architecture is extensible, it is notably rigid in design.
Available	✓	The system has demonstrated a high degree of availability.
Easily configured	✓	All configuration is stored within a central database at the operations centre. A web interface to the database is provided to facilitate the addition of sites and resources.
Lightweight	✓	The site monitoring agents are relatively lightweight components being responsible only for the monitoring requirements of an individual site. The devolution of the monitoring activities to each site results in a considerably more lightweight installation at the operations centre.
Shared	✓	The monitoring information acquired by the system can easily be made available to multiple parties via the web interfaces.
Cross-domain	✓	One of the primary requirements was to satisfy the need for cross-domain monitoring and the architecture fulfils the requirement very successfully
VO-enabled	X	No VO-support is currently included although it could be added at the display level

Chapter 7

Grid Command/Control with Grid4C

7.1 Introduction

While I4C is designed to satisfy an operations centres' requirements for grid service monitoring, the development of a system to exercise control over such resources is a considerably more complex undertaking. In this chapter I describe an architecture for a grid service and resource management tool. The requirement for this system has grown out of the research into grid monitoring that led to I4C, the parallel activities relating to the R-GMA monitoring system, and the desire to ease the day-to-day administration of the infrastructure, whilst increasing its availability and speeding the response to events, especially security events.

The architecture presented in this chapter is a component and enabling technology of a larger concept called Grid4C (Grid Foresee), the grand vision of a Command and Control system for grid infrastructures. This vision, outlined in Chapter 5, describes a suite of inter-operating components that serve to facilitate the efficient manual and automatic management of computing infrastructures. The objective of the work described in this chapter is to provide the necessary sensing and control infrastructure on which to layer advanced automation, user interfaces, and visualisation in order to construct a system capable of meeting that vision.

Many of the concepts and components of the I4C system are re-used in the implementation of Grid4C, where they are extended and augmented. A capacity for control of monitored resources is added and open standards for communication and representation are used where possible. I4C provided a solid body of knowledge on which to base the development of Grid4C and while it has proved to be a stable and useful work in its own right, might best be considered a prototype implementation of Grid4C.

In addition to the components described here, the overall concept of Grid4C includes multiple management, delivery, and visualisation layers. Grid4C collapses information and control architectures into a unified model yielding a notably simple concept. In this model the leaf entity is a sensor, actuator, or both. This enables Grid4C to bring together monitoring and status information from a variety of sources. For control, Grid4C provides a distributed control plane and makes it available to human and machine operators for rich interactive or autonomous control.

As indicated in Chapter 5, the design and implementation of this system has been motivated by knowledge of the system requirements based on experience, but first let us identify some use cases.

7.2 Use cases

7.2.1 Service Outages

Upon receiving notification of a failed service on one of the grid gateways, a member of the operations team can use a Control Client to restart the service via one of the user interfaces. The Control Client will issue a service management request to the Control Element on the resource hosting the failed service. This is a generalised use case that exercises the end-to-end control functionality following manual intervention.

7.2.2 Automatic service recovery

On failure of a service, an automatic process might programmatically execute pre-defined service recovery operations in order to attempt to return the system to an operational state. The processes could then request a service check operation to verify that the service was restored. If the service was not restored, an alternative operation might be attempted or failing that, an event would be generated to escalate the alert to a member of the operations team. This use case presents an example of programmatic use of the control functionality.

7.2.3 Managing batch queues

The system could be used to exercise control over the batch processing queues connected to the Compute Element at the grid site. A very useful example of this would be the removal of 'stuck' jobs from a queue. A job may appear to be 'stuck' if the Worker Node on which it was executing hangs and the job status is not updated by the queue manager.

7.2.4 Active Security

In the event of a security incident or notification of a security flaw in the grid middleware, the operations staff can shut down or quarantine the relevant elements ‘at the touch of a button’. This use case emphasises the need for responsiveness, i.e. minimal latency from command to final action. Extensible management capabilities provide an ideal mechanism to allow security and intrusion detection systems to not only detect but actively defend. Access control policies and firewall rules could be reconfigured on the fly in response to security incidents, or user executables could be terminated, or both.

7.2.5 Active Authorization

In the event of particular security incidents, such as a breach of an acceptable usage policy (AUP), for example if a user attempted to re-nice their process beyond acceptable limits, the control mechanism might autonomically enforce security measures at the VO-level resulting in the VO members’ access being degraded (limited roles or privileges) or suspended pending review.

7.2.6 Applications server monitoring

Administrators of systems such as R-GMA running within servlet containers/application servers will be familiar with service outages caused by JavaTMmemory exceptions. Similarly, stand-alone JavaTMserver applications such as WebCom-G[PPJM03] nodes have been observed to fail due to memory problems. This architecture allows a pro-active management solution to this problem, e.g. by providing sensors to remotely monitor the state of the JavaTMvirtual machine resources and either invoke garbage collection operations or restart the service/application prior to failure.

7.2.7 Workflow based resource management

Since the management endpoints are implemented as web service with published interfaces, they are amenable to invocation from workflow languages such as WS-BPEL. This would allow operators to orchestrate management operations with specific order or dependencies and also allow for the management of multiple resources with a single action. For example, if components of overlay services such as WebCom were modelled with WSDM endpoints, the lifecycle of multiple instances of the service could be managed in tandem from a single workflow.

7.2.8 Autonomic resource management and provision

By providing endpoints to manage the configurations of batch systems and queues, computing resources can be automatically reconfigured to meet demand. This might be as simple as reallocating resources between queues, or instantiating new virtual machines of a required configuration.

7.2.9 Alerting, visualisation, and interaction

Given the flexible notification model provided by WSDM and WS-Notifications, advanced alerting systems could be constructed. The endpoints could provide considerable amounts of status information to aid system visualisation. The control functionality would allow for remote interaction with the managed resource.

7.2.10 Peer resource management

If the management endpoints are configured to implement both event publishing and subscription interfaces advanced management capabilities would become possible. An example of this would be the removal of a hierarchical system of management and devolution of the responsibility to the resources themselves. This could be achieved by creating a ‘buddy’ system in which resources or sites are assigned to peers for the purposes of monitoring and management.

7.2.11 Extension of existing monitoring tools

This architecture could be used to extend the reach or functionality of existing monitoring tools. For example, the system could be configured to publish status information into the Nagios monitoring system which could then act upon the information and use a configured event handler to rectify the situation via a management endpoint. This approach would extend the management functionality of Nagios across multiple sites and domains yet leverage it’s existing functionality as a display server and policy engine.

7.2.12 Scheduled management actions

In an individual computing / server node there are many management actions that are scheduled in a programmatic fashion, e.g. those scheduled as Cron jobs in UNIX-like systems. An equivalent facility is undoubtedly needed at the level of grid infrastructure management. Such actions would typically be examples of open loop control but clearly they could also be conditional.

For example, in the Hungarian ClusterGrid[Cen] the network router configurations are modified diurnally at specific hours to effectively allocate and de-allocate the cluster nodes. Similarly, within the Grid-Ireland Operations Centre there is a need to frequently reconfigure the firewall for the ETICS/NMI [BSM⁺06] build system to either reserve it for internal use, or provide it to the EGEE ETICS community.

Combining scheduled actions with workflows as described above would provide particularly powerful cross-resource capabilities.

7.3 Related Work

While a lot of work has been done in the area of Grid monitoring, few projects have attempted to ‘close the control loop’ by providing interfaces to rectify detected anomalies or remotely reconfigure components. One promising early work is the C.O.D.E. [Smi02] project developed for the NASA Information Power Grid. Unfortunately this project pre-dated much of the work on Web Service and management standards and the control functionality was never fully implemented.

With evolution towards grid middlewares exposing functionality via Web Services [FKNT02] [TPH⁺06], the management of these services has become an increasingly important and active research topic [TSF06] [AKS05] [CDR⁺03]. However, few of these works take into account the requirements for managing ‘legacy’ grid services, i.e. non Web Service based, or their hardware platforms via standard Web Services interfaces.

7.4 The potential role of WSDM

The increasing complexity of computer networks and infrastructures has led to many operations teams struggling to deal with the additional overhead involved in their monitoring and management. This has led resource providers to provide monitoring and management tools intended to ease the burden of the deployment and maintenance of their systems. However, in heterogeneous environments, either software or hardware, the integration of these tools can become troublesome in itself. Operations staff may be forced to use multiple systems and manually aggregate or correlate information across them in order to acquire the necessary representation of system state. The use of proprietary interfaces and persistence models without common standardised interfaces limits the possibility of managing the components of heterogeneous systems in a truly cohesive manner.

One possible solution to these integration and interoperability problems is the Web Services for Distributed Management [OASa] (WSDM) standard from the Organisation for the

Advancement of Structured Information Standards (OASIS). WSDM defines standard mechanisms for the representation of, and access to, resource management interfaces implemented as Web Services. In addition, it defines how web services themselves may be managed as resources. The standard defines two specifications; WSDM: Management Using Web Services (WSDM.MUWS) and WSDM: Management of Web Services (WSDM.MOWS)[OASc]. Mechanisms are defined for identifying, inspecting, and modifying characteristics of resources, thus ensuring the interoperability of management tools and management resources from different vendors or development groups.

Rather than being designed from the ground up, WSDM incorporates elements of numerous existing Web Service technologies, including WS-RF, WS-Notification and WS-Addressing. By employing web services, WSDM inherits essential distributed computing functionality, interoperability and implementation independence.

7.4.1 The WSDM development model

While WSDM defines the interfaces and the reference implementation provides the necessary infrastructure, the creation of the custom manageability capabilities that will be exposed via WSDM is left to the developer of the resource management solution. These capabilities, comprising data and operations, are the fundamental building blocks of resource management endpoints. Developers create WSDL files that define the web service interface, and must include one port type for the resource that includes all the resources' public operations that a client would need in order to inspect and manipulate it. In addition to those custom capabilities defined for a particular resource, the WSDM specification includes a number of standard capabilities, these include:

- Identity - The only required capability, used to differentiate among resources. This capability contains exactly one property, *ResourceID*, which is unique to the resource.
- Description - The list of captions, descriptions and version information used to provide a human readable identity for the resource.
- Metrics - Defines how to represent and access information about a specific property as well as the current time on the resource.
- State - Defines how to change the state of a resource according to a specific state model
- Operational Status - Defines three status levels for a resource (*available*, *unavailable* and *unknown*) along with status change events.

- Advertisement - A standard event to be generated when a new manageable resource is created.

Other features of the WSDM standard that are of particular interest to grid management include *Relationships* and *Notifications*.

Relationships

WSDM defines interfaces which can be used to query a manageable resource about the relationships in which it participates. This might be done by querying a resource directly to find out the relationships in which it takes part, or by querying a service implementing a WSDM interface for a relationship registry. Manageable resources therefore do not need to be aware that they are participating in a relationship. For grid management these might be used to arrange the manageable resources into groups based, for example, on their location or their resource type.

Notifications

WSDM also defines an extensible XML event format that defines a set of data elements allowing different types of management information to be transmitted, processed, correlated, and interpreted across different products and platforms using different technologies. Each WSDM capability has a corresponding WS-Notifications topic which can be used to identify and categorize capability-specific notifications such as property change events or state transition topics. The same notifications framework can be used to propagate WEF (WSDM Event Format) management events. WEF events use an extensible XML format for the exchange of event information and are discussed in more detail in Appendix B.

In addition to being the source of notifications, management capabilities can also implement NotificationConsumer interfaces, thereby allowing them to receive notification from other capabilities. More complex notification topologies then become possible with the sharing of information among entities without having to wait for it to propagate through a conventional hierarchy. Such shared awareness, coupled with distributed control, enables more advanced autonomous networks of control to be constructed. As an example of this, consider a collection of grid resources where each has subscribed directly to notifications from all others (or to a central broker, achieving the same result). If one of the resources in a group detects a network attack or violation of a user agreement policy, it might publish the details of the event for all interested parties to receive. Upon receiving the notification, each of the other resources might decide to perform some automatic reconfiguration in order to harden themselves against similar violations. The same kind of mechanism might also be used for

administrative operations such as deploying patches or configuration changes. Remembering that the inner workings of the management system might be platform or middle-ware specific, the same event might be treated differently depending on the architecture of its recipient. Local management policies might also play a part in deciding how to react to specific events.

7.5 Architecture

We are concerned not only with the management of the grid software, but also of the physical hardware resources comprising our gateways. We present an architecture which leverages the benefits of agent based design [CST⁺02][FJK04] and allows the monitoring and management of ‘legacy’ grid services via a standards-based Web Service interface. Although this architecture could, and most probably will be extended to monitor and control Grid functionalities exposed via Web Services, we consider this a separate problem domain for which there are more relevant management standards such as the Web Service Distributed Management: Management Of Web Services (MOWS)[OASa]. Based on our proposed architecture, we present a concrete, extensible implementation of a grid management tool that attempts to build on established work and contribute to future development by promoting the use of standard interfaces.

7.5.1 Overview

In this section, we describe the architecture of the management system at a high level so that the basic building blocks can be seen clearly without implementation details intruding.

The architecture of this system employs a multi-tier approach. A client/server architecture exists between the operations centre and the sites comprising the managed hosts or entities. Within the sites, a second hierarchy exists where incoming requests or operations are routed to the relevant request handlers on the managed entities.

Due the fact that the managed resources often reside within remote administrative domains, over which the grid operations team have little control, the architecture employs a single management server deployed to each site. This server acts as a gateway to the management capabilities within the site, in addition to containing middle tier logic such as the scheduling of service checks, persistence of event information and policy decision logic. This single point of presence allows de-coupling of inter-domain and intra-domain communications, so that inter-domain traffic into and out of the site can be limited to standard web service ports (HTTPS) on a single machine rather than requiring direct network access to each of the managed entities. Site administrators will appreciate this model since fewer machines need external access. The multi-tier approach also allows the software deployed to each of the

managed entities to be relatively lightweight (an important requirement) as it need only to communicate with the local management server rather than supporting an entire web services stack.

The software components deployed to each managed entity are responsible for implementing the interfaces exposed via the web services on the management server. In this respect, much of the web service logic is composed of proxy objects responsible for routing the requests to their corresponding control elements on the managed entities.

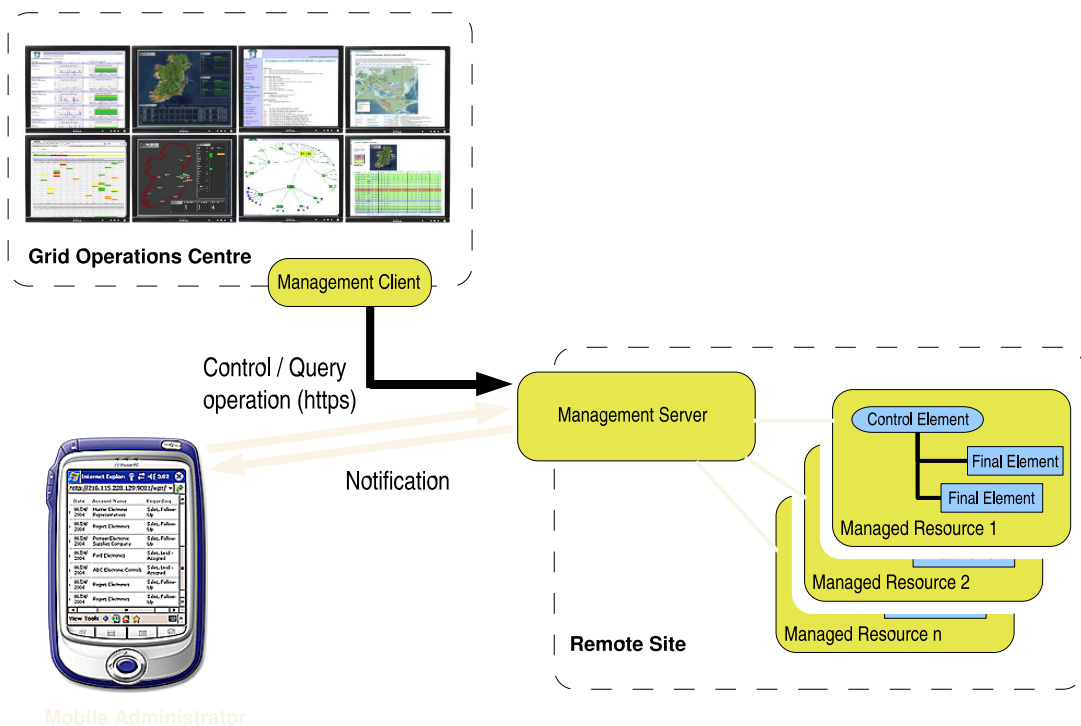


Figure 7.1: Architectural overview of control backplane.

The core components of the architecture, as illustrated in figure 7.1, include the following:

- **Management Clients** - Standalone or embedded components, capable of invoking the management capabilities exposed via Management Servers. These are typically executed within the systems of the operations centre. Mobile clients are also being investigated.
- **Management Servers** - A single management server is deployed to each site with the primary responsibility of exposing the management capabilities of the resources within

that site or domain as web service endpoints. Additional functions include the scheduling of monitoring and administrative tasks, persistence of event information and in the case of autonomous control, the role of a Policy Decision Point.

- Control Elements - These are the lightweight components deployed to each of the managed resources. They provide a control interface to the resource and implement the logic defined in the services exposed via the Management Servers. Operations requested of the Control Elements may be used to monitor or alter some property of the resource.
- Final Elements - Using terminology borrowed from the field of process control, the Final Elements are the lowest level components of the architecture. Residing on the managed resources and accessed via the Control Element, Final Elements are used to sense state and perform action operations on the resource. They are implemented as plug-in objects and can be either self-contained or rely on some proprietary local application programming interface such as the Intelligent Power Management Interface from Intel. Examples of final elements might include:
 - Service Check - used to determine the status or availability of grid services
 - Service Control - used to alter the state of a given service
 - Network control - control over network interfaces and traffic. These might for example be used to ‘quarantine’ problematic hosts
 - Hardware Management - used to expose hardware management functions via the secure web service interface, e.g. power cycling resources
 - Job Execution - execute grid jobs such as those designed to evaluate performance and availability
 - Job Management - provide remote control over local job schedulers, e.g. the removal of ‘stuck’ jobs
 - Access Control - interface to user authorisation components
 - Resource Control - more coarse grained control over the state or availability of the resource

7.5.2 The Control Element & Final Elements

Figure 7.2 shows a more detailed architecture of the control and management system. The Control Element is implemented as an XML-RPC server along with a set of request handlers. Examples of these request handlers include Final Element Managers, responsible for

maintaining a list of registered Final Elements and triggering their execution in response to requests, and Server Managers, responsible for more generic resource management operations such as start and stop, etc.

The Control Element performs a similar duty to the Management Server, albeit on a smaller scale, in that it provides a single point of access to the management functions of a resource. The advantages of this architecture include a smaller footprint on the managed resource, a single location for authentication and authorisation, and a more straightforward usage model. A Final Element may be addressed in the form *resource.requesthandler.element* although a number of XML-RPC client objects have been written to encapsulate and simplify this functionality. Objects wishing to invoke some operation on the Control Element can simply instantiate one of these client objects and use methods such as *resource.stop*, *resource.getServiceStatus(serviceName)*, or *resource.executeFinalElement(elementName)*.

The term Final Element is used to describe a component that performs the duties of a sensor, actuator or both. They are the last elements in the chain of command and constitute the bridge between the control system and the managed resource. The Final Elements are implemented as JavaTM objects implementing a specific interface allowing them to be easily 'plugged-in' to the control hierarchy. Typically the output of the operations supported by the final elements is in the form of a WEF management event. Final Elements can be self-contained JavaTM objects, include some functionality native to the resource platform, or make use of an existing application programming interface. One example of using Final Elements as wrapper objects is described in [RCW06b]; this illustrates how the extensive range of sensors developed for the Nagios project can be exploited from a Java based monitoring application.

7.5.3 The Management Server

The Management Server can be hosted on any machine capable of running a servlet container provided it has the necessary network access to the Control Elements within its domain and also to the external control clients. For this reason, the Management Server would typically be installed as part of a grid gateway. The WSDM management endpoints reside within an Axis container on this Management Server. These make the control functionality offered by the Control Elements available to the Control Clients via the published web interfaces. These interfaces make use of both standard and custom capabilities. Since the managed resources would typically not reside on the same host as the Management Server, capability proxy objects are employed to route the web service requests over XML-RPC to the Control Elements on the target resource.

7.5.4 The Control Client

Currently the Control Clients are invoked using command-line tools. More complex graphical interfaces are under development. Using the WSDL that defines the management endpoint, and the *wSDL2java* tool included in the Muse distribution, it is possible to automatically generate client code which can be used to invoke the operations defined in the WSDL. It is then relatively straightforward to create user interface code to invoke management operations without having to write code at the web services layer. The same proxy objects used within the control client are used for programmatic access to management capabilities.

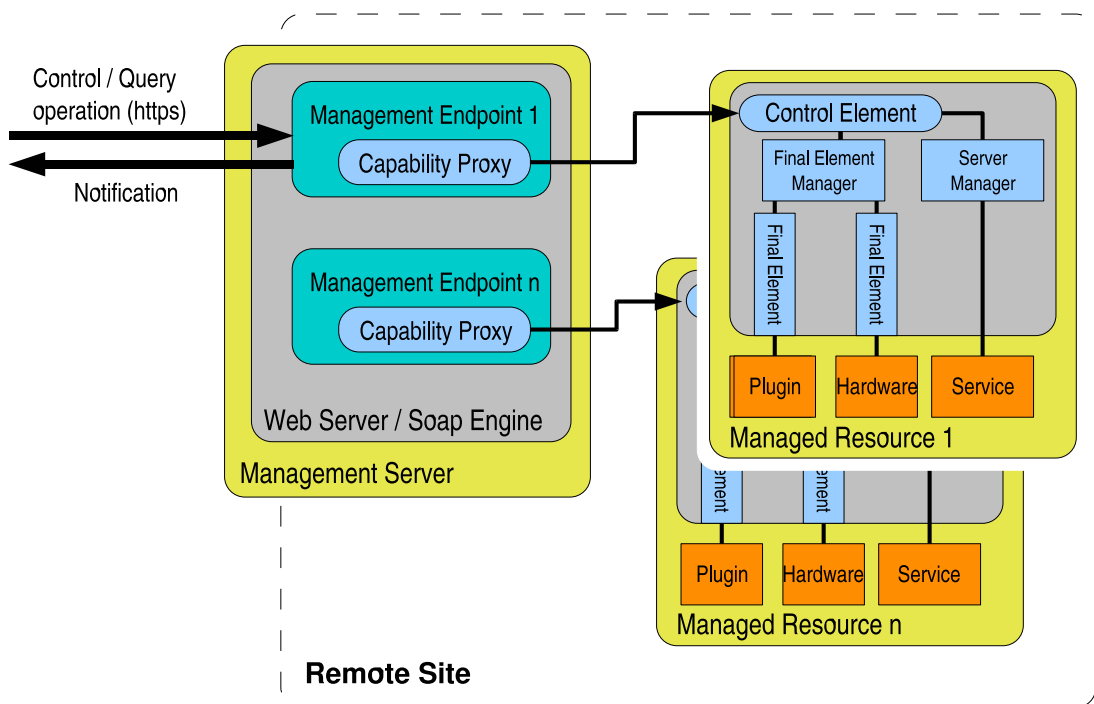


Figure 7.2: Architecture of remote management capabilities.

7.6 Event Generation and Standard formats

As stated in the requirements analysis, the importance of standard means for the representation of status and management information is paramount. Within this architecture, all component to component communication other than invocation is performed using standard

events conforming with the WSDM event format outlined in Appendix B. This format allows crucial information to be communicated while allowing extensions to accommodate additional information. Feedback from final elements and management endpoints is in the form of WEF events and all management operations performed on the endpoints are also logged and published in the same format.

7.7 Security

Since security is of critical importance in this application it is incorporated on several levels. All traffic between the Control Client and the Management Servers takes place over HTTPS connections with client and server authentication. Further security is provided at the web services layer using Apache Rampart[Apad] ensuring that all incoming SOAP requests are signed. Within a remote management domain, all traffic between the Management Server and Control Elements takes place using XML-RPC over SSL. The XML-RPC servers are also secured using access control lists which typically only allow their operations to be available to the local Management Server. In addition, in the Grid-Ireland case, because the gateways are implemented as virtual machines upon a physical host running Xen [CCO⁺05c], the physical host communications represent an out-of-band control plane. This adds greatly to the control plane's security. In the event of a total failure of the physical host, a further secure out-of-band control plane is available using the host's remote management hardware, which employs IPMI[ipm].

7.8 Configuration

Configuration of the Management Endpoints and Control Elements is performed using XML configuration files. In the case of the Managed Endpoints, the primary configuration file is the muse.xml file within the service container. The Control Element configuration file is used to specify security and port settings while the request handlers, such as the Final Element Manager and Server Manager, use independent files. The configuration files are amenable to automatic configuration by fabric management systems such as Quattor.

Control elements are typically configured for a specific type of resource, e.g. a Compute Element, and subsequently require minimal modification to the configuration when deployed to different instances of that type. A similar case applies to the configuration of the Management Endpoints which typically only need to be informed of the addresses and ports of the corresponding control elements. Indeed this configuration requirement can be negated for sites using standard naming conventions through the use of DNS and standard ports.

A degree of automatic configuration is also possible thanks to the fact that WSDM resources can, through introspection mechanisms, provide information regarding their capabilities, relationships, and the service groups to which they belong. These functions coupled with standard web service discovery mechanisms could be used to automatically configure resource management clients such as local autonomic site managers.

7.9 Example Management Capability

The Grid4C Management endpoints are composed of one or more Management Capabilities. These might be simple service capabilities (e.g. GRAM, GRIS, tomcat) or more complicated host management capabilities. Table 7.1 shows examples of some of the properties and operations provided by a selection of Grid4C capabilities. It should be noted that in addition to custom capabilities, many of the management endpoints incorporate base capabilities from the WSDM standard as detailed in Appendix A. It is intended that additional properties will be added to the capabilities in later releases. An example of this would be to add (queue and scheduler independent) job metrics to the GRAM capability.

Management endpoints for a given type of entity are created by combining existing capabilities, relevant to the managed entity, with new capabilities where necessary. For example, the management endpoint for a Compute Element incorporates the following capabilities:

Host Management Fundamental host management operations such as startService, stopService, enable, disable, etc. The properties exposed include CPU, memory and disk information in addition to current process information etc.

GRAM Operations and properties relating to the Resource Allocation Manager service

GRIS Operations and properties relating to the site BDII information provider.

SSH Operations and availability information relating to the the SSH server/damon on the host.

More than one endpoint may be deployed for a given resource and multiple endpoints can perform requests on a single ControlElement. An example of this is seen in the case of the NetworkAccessControl capability which is deployed independently of any specific entity management endpoint, e.g. the CE endpoint, yet performs operations on the same entity by requesting services from the same ControlElement. In situations where there is a large number of entities to be managed, e.g. cluster worker nodes, a different development model might be adopted. Rather than using relatively heavyweight stateful management capabilities, a

factory service might be used to generate management objects for the requested resource on the fly and map them to unique endpoint references (URIs).

Management endpoints are constructed from management capabilities. The prototype GLCE Management endpoint comprises the capabilities described in Tables 7.1, 7.2, and 7.3, among others.

Table 7.1: A selection of typical properties and operations exposed by Grid4C management endpoints.

GI Host Management Capability	
Properties	Operations
CPU Load	Enable
Uptime	Disable
Disk Size	Start Service
Disk Utilisation	Stop Service
Memory Size	Execute Final Element
Memory Utilisation	
Network Stats	
Process List	

Table 7.2: Properties and operations of a simple GridFTP management capability. The same selection of properties and operations are provided for GRAM, GRIS, SSH, etc.

GridFTP Management Capability	
Properties	Operations
Port	Start
Status	Stop
Status Information	

Table 7.3: The PBS Management Capability

PBS Management Capability	
Properties	Operations
Total Jobs	Delete Job
Running Jobs	Get Job Status
Queues	Stop Queue
Nodes	Start Queue
Server Name Size	Mark Node Down
Server Status	Mark Node Offline

A prototype management capability for a resource broker has also been developed as described in Table 7.4. This capability has been used to support research and experimentation

in the area of economic brokerage models.

Table 7.4: Example Resource Broker management capability

RB Management Capability	
Properties	Operations
	Get average Waiting Time for last n jobs
	Get average Waiting Time since T
	Get average Match Time for last n jobs
	Get average Match Time since T
	Get average Clear Time for last n jobs
	Get average Clear Time since T
	Get percent Aborted for last n jobs
	Get exit status for job j
	Get wall clock time for job j
	Get CPU count for job j

7.10 WSDM Implementation

The system has been developed in Java using Apache Muse[Apaa], a reference implementation of the WSDM specification. The Muse endpoints are deployed within an Axis2[Apac] SOAP engine on the management servers deployed to each site. Communications within each site, between the Management Server and a Control Element, are carried out using XML-RPC[Apab] with two-way authentication over SSL. Once the management server is in place and the Control Elements are installed and configured on the resources to be managed, the system makes the management functions provided by the Final Elements available to authorised users via web services interfaces compliant with the WSDM standards. The Management Server provides a single point of entry to administer the resources within its site. One of the advantages of using this reference implementation is that it includes Web Service Notification specifications which can be used to construct an event driven communication model for the management infrastructure. In this implementation, the WSDM Status capability of a managed resource is often representative of the logical value of a number of lower states. For example the status of a resource might only be set to *Available* following the successful start-up of all its services and the verification of its availability via a local or remote sensor. If this value is subsequently changed either by a local or remote process, all parties that had subscribed to the topic for that capability will be automatically notified via a property change event.

7.11 Sample Management Client

For the purposes of testing and evaluation, the management clients were limited to command line tools and API objects which facilitated the invocation of operations on the management proxies from within systems wishing to take advantage of the control and information functionality. A number of these have been successfully integrated within external software systems as described in Chapter 8. Graphical and web-based interfaces for use by operations staff are under development but are unfortunately not sufficiently complete for inclusion here.

7.12 Summary

In this chapter we have presented a number of motivations for an idealised concept of grid command and control. We described the architecture and a prototype implementation of a grid resource management tool, Grid4C, and the role of WSDM in its implementation. This is now an open source project[gr4].

The system we have presented illustrates how the use of lightweight components deployed to each of the managed resources within a site can be used to provide a single point of access to resource management. We believe that this solution is an optimal configuration allowing the use of WSDM for cross-domain fabric management without the necessity of deploying a web services container to each of the managed resources. We have also illustrated how this single point of access is particularly important on an infrastructure such as Grid-Ireland, where a large proportion of the managed resources reside within networks beyond its control.

Although there is a learning curve associated with the adoption of any new technology, the use of standard interfaces will result in software components that can be more easily integrated into existing environments. With movement towards service oriented architectures, where users will piece together solutions to meet their needs, the flexibility offered by the use of common interfaces will be critical.

This architecture is designed to facilitate management of distributed resources which is a prerequisite of the application of command to grid infrastructure management.

Table 7.5: Grid monitoring system requirements

Requirement	Compliance	Details
Open	✓	The architecture and interfaces are documented and published
Open Source	✓	The system is in the public domain as a Source Forge project.

Extensible	✓	The plug-in architecture allows addition final elements to be added as the need arises.
Interoperable	✓	All external interfaces are exposed as standards-based web services.
Standards-based	✓	The management interfaces and event formats comply with OASIS standards
Secure	✓	All communications takes place over HTTPS or XML-RPC/SSL.
Flexible	✓	The use of standard interfaces and event formats, and component-based design allows capabilities to be assembled in a flexible manner.
Modular	✓	The system employs a component based architecture.
Layered	✓	A multi-tier architecture is employed with clear distinctions between functional levels
Scalable	✓	Experience has demonstrated an ability to scale well. The distribution of management operations and the lack of dependencies on central services serves to reduce the potential for scalability problems.
Robust	✓	Deployment has shown the system to be stable and reliable.
Resilient	✓	The system can tolerate changes and failures of constituent components.
Responsive	✓	Evaluation has shown the response times to be well within acceptable ranges despite the expected overhead of the web services architecture.
Easy to use	✓	Flexible re-publishing and a variety of user interfaces provide easy access to monitoring information
Innovative	✓	There is growing recognition of the importance of standards in the field of grid monitoring and while this architecture supports this, it also paves the way for active management of resources.
Adaptive	✓	The flexible nature can be adapted to meet the demands of the supporting infrastructure.
Available	✓	The system has demonstrated a high degree of availability.
Easily configured	✓	All configuration is performed using XML configuration files on the management endpoints and is amenable to automated fabric management using mechanisms such as QUATTOR

Lightweight	✓	The architecture endeavours to minimise the footprint and communications overhead on the managed entities. The devolution of the monitoring activities to each site results in a considerably more lightweight installation at the operations centre.
Shared	✓	The monitoring information acquired by the system can easily be made available to multiple parties via flexible publish and subscription mechanisms.
Cross-domain	✓	One of the primary requirements was to satisfy the need for cross-domain monitoring and the architecture fulfils the requirement very successfully
VO-enabled	✓	The security mechanisms can be configured to restrict access based on client certificates therefore VO-enabled access to information and management is possible

Chapter 8

Deployment and Evaluation

This chapter describes a number of test applications that have been used to evaluate the Grid4C architecture. The test cases span a wide range of applications and demonstrate the suitability of the approach for the purposes of grid resource management.

8.1 Testbeds

Testbed infrastructures provide a crucial function in the testing and evaluation of middleware components and tools. Testbed infrastructures not only allow middleware developers to test new releases, but also provide a proving ground for prototypes or proof-of-concepts such as ours. Adequate testing of the Grid4C architecture would have been unfeasible without the availability of real grid services and resources on which to base measurements and test functionality.

8.1.1 TestGrid

The complexity of grid infrastructures can pose significant difficulties for the testing of user applications and middleware components. Due to the maturity of many grid projects and their migrations from investigative testbeds toward production infrastructures, an adequate means of evaluating new components in isolation from the live infrastructure is required. TestGrid[CCO⁺06] is a realistic large-scale testbed constructed at Trinity College Dublin, to accurately emulate the configuration of the production Grid-Ireland infrastructure.

Developed to allow realistic and adequate testing of middleware and applications software prior to deployment, the replica infrastructure is hosted in a self contained network allowing it to use the same system and network configurations as on the real infrastructure without conflict. Virtualisation is widely deployed within testgrid in order to reduce hardware

requirements and replicate the real infrastructure. At the time of its development this was a grid first and it continues to be a very valuable resource.

8.2 Deployment

Typically a test deployment takes place across at least three machines; a client from which control instructions are issued, a web application server hosting the WSDM Management Endpoints and associated monitoring schedulers and sensors, etc, and a machine which either hosts, or is itself the entity under management. For the purposes of evaluation, all prototype code was packaged as using tar (POSIX tape archive) files. Package formats such as RPM will be more suitable to larger-scale deployment on the production infrastructure, facilitating the use of our existing fabric management system. The software was deployed within the Grid-Ireland TestGrid described above and also within a portion of the Int.eu.grid test environment hosted at Trinity College Dublin.

8.3 Evaluation

8.3.1 Performance

While there is some overhead incurred through the use of SOAP/HTTP messages, security mechanisms, and the routing of requests from the Management Servers to the Control Elements, we feel that the benefits of the web service technologies and the reduced deployment effort justify the expense. Preliminary tests of the management operations show response times well within acceptable boundaries.

Figure 8.1 displays measurements from the autonomic resource management experiment detailed below. It shows the reaction time between a WSRP property change event being published by a management endpoint and a resulting action being invoked on the endpoint by a subscribed manager. For each event, two times are plotted; the interval between the event publication and the management operation on the WSDM service, and the interval between the event publication and the subsequent operation on the Control Element. The results show an average round-trip notify-actuation time of 65.12 ms on the management endpoint and 111.94 ms on the corresponding Control Element. These measurements represent a system where all communicating components reside within a single network segment. Baseline performance measurements for the test network environment are shown in Table 8.1.

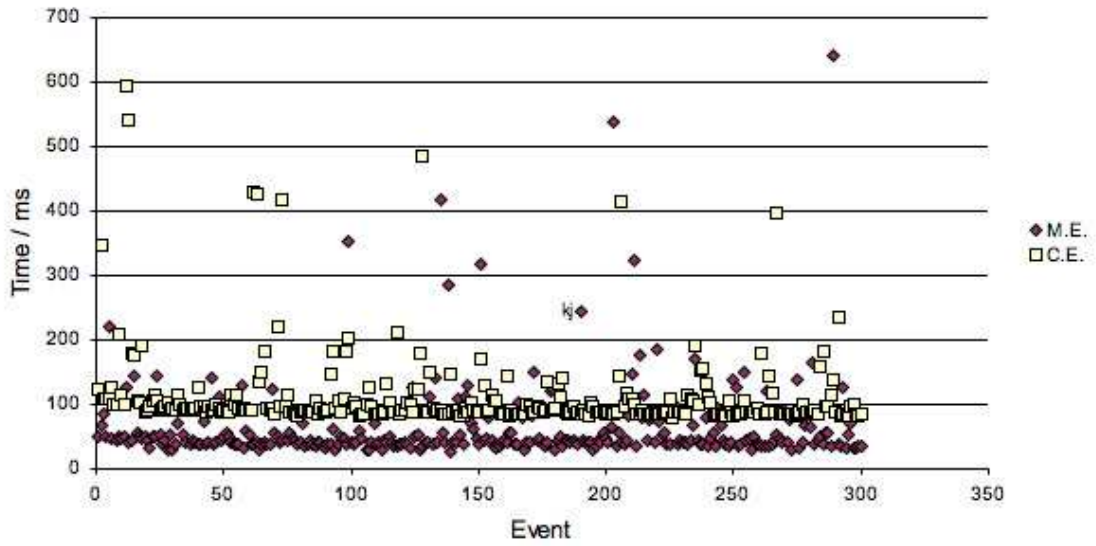


Figure 8.1: Analysis of WS-N driven reaction times.

Round trip latency ($\mu\text{sec}/\text{Trans}$)	899.970
Trans rate (Trans / sec)	1111.148
Throughput (10^6 bits / sec)	9.103

Table 8.1: Baseline performance measurements for test network environment

8.3.2 Scalability

It is expected that the architecture will scale favourably due to the devolution of much of the management functionality to the Management Servers and managed resources. The majority of the message sizes between the control clients and the managed resources will be relatively small, and multi-threaded request handlers within the Control Elements will serve to reduce execution overheads.

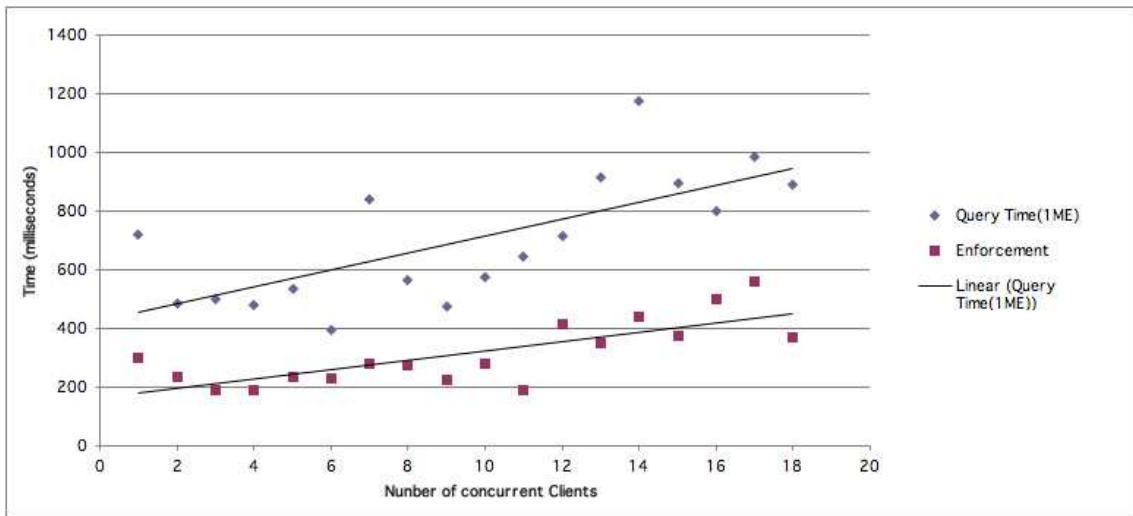


Figure 8.2: Analysis of Response time.

Figure 8.2 demonstrates the change in response time for both queries and enforcements as the number of concurrent clients increases. During this experiment, each client invoked two hundred and fifty operations on the capability at three second intervals. The objective of this experiment was to ensure that the time interval between a client invoking and management operation on a management endpoint and that operation being carried out on the managed resource is acceptable. The round-trip time taken for query operation must also be acceptable. Assuming that the number of clients is generally low, these results are very acceptable.

8.4 Use-Cases

8.4.1 Autonomic Policy-based Resource Management

For this example, an autonomic resource manager is created to manage a particular resource. The manager subscribes to property change notification events describing the operational state of the resource for which it is responsible. Upon receiving an event, the manager

consults the management policy to determine if any action is required and if so, carries out that action.

The prototype implementation consists of a manager for a CE which subscribes to the GFTP service status property. A management policy will be defined which dictates that when the status of the service is unavailable (or the response time exceeds a pre-set threshold), the management endpoint should be used to restart the service. The availability of this service can then be monitored over time and compared to the data gathered for other equivalent resources by I4C.

A more general architecture could allow these managers to be easily constructed based on runtime configuration rather than being implemented as resource specific. A configurable management object might take a configuration file specifying the resource to manage, the policy by which to manage it, and the properties to which it should subscribe. It might be possible for the management object to determine these properties on the fly from the WSDL or the property document. Either architecture is an example of automated tactical decision making at the grid site level, illustrating how the collapsed C⁴I model enables true distributed grid management. This relieves the central management of local tactical management and enables it to concentrate on strategy and global tactics. Governance policy is enforced, even at this site level.

Listing C.1 is a registration trace showing the manager subscribing to all WS-Notification topics on the managed resource.

Listings 8.1 and 8.2, excerpts from Listing C.2, show the incoming notifications to the manager on the event of a service outage. Note the gftpStatus property has changed from 0 (available) to 2 (unavailable). Also the gftpStatusInformation TCP connection test has changed from a healthy response time to ‘Connection Refused’.

```
1      <wsnt:Message>
3          <wsrf-rp:ResourcePropertyValueChangeNotification xmlns:wsrf-rp="http://docs.
              oasis-open.org/wsrf/rp-2">
4              <wsrf-rp:OldValues>
5                  <tns:gftpStatus xmlns:tns="http://www.grid.ie/Grid4C/ServiceManagement
                      /GI_GFTP_Capability">0</tns:gftpStatus>
6              </wsrf-rp:OldValues>
7              <wsrf-rp:NewValues>
8                  <tns:gftpStatus xmlns:tns="http://www.grid.ie/Grid4C/ServiceManagement
                      /GI_GFTP_Capability">2</tns:gftpStatus>
9              </wsrf-rp:NewValues>
10             </wsrf-rp:ResourcePropertyValueChangeNotification>
11     </wsnt:Message>
```

Listing 8.1: Section of incoming WSRP notification trace showing change of service state

```

1  <wsnt:Message>
      <wsrf-rp:ResourcePropertyValueChangeNotification xmlns:wsrf-rp="http://docs.
          oasis-open.org/wsrf/rp-2">
3      <wsrf-rp:OldValues>
          <tns:gftpStatusInformation xmlns:tns="http://www.grid.ie/Grid4C/
              ServiceManagement/GI_GFTP_Capability">TCP OK - 0.000 second
              response time on port 80|time=0.000186s;;;0.000000;10.000000</
5              tns:gftpStatusInformation>
          </wsrf-rp:OldValues>
          <wsrf-rp:NewValues>
7              <tns:gftpStatusInformation xmlns:tns="http://www.grid.ie/Grid4C/
                  ServiceManagement/GI_GFTP_Capability">Connection refused</
                  tns:gftpStatusInformation>
          </wsrf-rp:NewValues>
9      </wsrf-rp:ResourcePropertyValueChangeNotification>
  </wsnt:Message>

```

Listing 8.2: Section of incoming WSRP notification trace showing change of service availability

Upon receiving these events, the resource manager will solicit guidance from either its local policy or a policy server and carry out any required actions. In this case the policy states that the service should be restarted. The manager will therefore invoke a ‘start service’ operation on the management endpoint which will then get directed to the control element on the managed resource. Upon completing the requested action, the management endpoint publishes the details in a management event as illustrated in the first notification of Listing C.3. The subsequent two WSRP property change events in that listing show the properties `gftpStatus` and `gftpStatusInformation` returning to normal.

This example serves to illustrate the ease with which autonomic resource management can be established thanks to the combination of the standard notification mechanism provided by WSDM and the control capabilities provided by the Management Endpoints and Control Elements. It provides an ideal basis for the development of site-local autonomic resource managers.

8.4.2 Integration with the Nagios monitoring system

In the same way that I4C provides a mechanism to extend the monitoring functionality of the Nagios monitoring system, Grid4C can be used to extend the management functions (event handlers). The architecture, as illustrated in Figure 8.3, employs a WS-Notifications consumer to receive and parse event notifications before publishing the results into Nagios via the NSCA (Nagios Service Check Acceptor) plugin. Nagios may then use its normal configuration to decide if any action is required and if so, will trigger the defined event handler.

In this case, the event handler is a shell script that executes a Java management client to invoke an operation on the relevant management endpoint. This test serves to illustrate the integration of the Grid4C control system with existing monitoring and presentation tools.

8.4.3 Providing reliable resource metrics for Economic Grid Markets

This section is the result of a collaboration with Gabriele Pierantoni, a fellow postgraduate student in my research group at Trinity College Dublin.

Resource allocation and management in Grid Computing poses challenges of growing complexity; some of the solutions devised by the scientific community to cope with these challenges are based on economic and social paradigms. They attempt to apply the principles that form the base of economic exchange to Grid Computing in the hope that the laws of the market will yield efficiency and equilibrium in the Grid as they allegedly do in the real world.

The application of economic paradigms to Grid Computing introduces a number of complex issues regarding the process of price creation, arbitration of disputes and trust among actors. In addition, further complex challenges are presented by the need for reliable information and control systems, capable of interfacing the economic layer to the very fabric of the Grid. Grid4C Management Endpoints present an exciting solution to this problem of information provision and fabric control. The remainder of this section provides a more detailed exploration of this concept, along with a description of a prototype implementation.

In Trinity College Dublin, a prototype system that enables economic and social transactions on the Grid is being developed by Gabriele Pierantoni under the banner of Social Grid Agents (SGA)[PKC06b][PKC05]. During the development of this prototype, the need for a system to determine the price and value of resources and the need for the social agents to exercise a degree of control over their resources suggested the merging and inter-operation of Social Grid Agents with Grid4C.

The exploration described in this section is based on interoperability between Social Grid Agents and Grid4C management endpoints, allowing a two-way exchange of commands and information. Grid4C endpoints can be used by Social Grid Agents as both sensors and actuators on the Grid fabric. They provide Social Grid Agents with information describing the production parameters of the various resources such as average waiting time and success rate. The value and ultimately, the price of the resources can then be extrapolated.

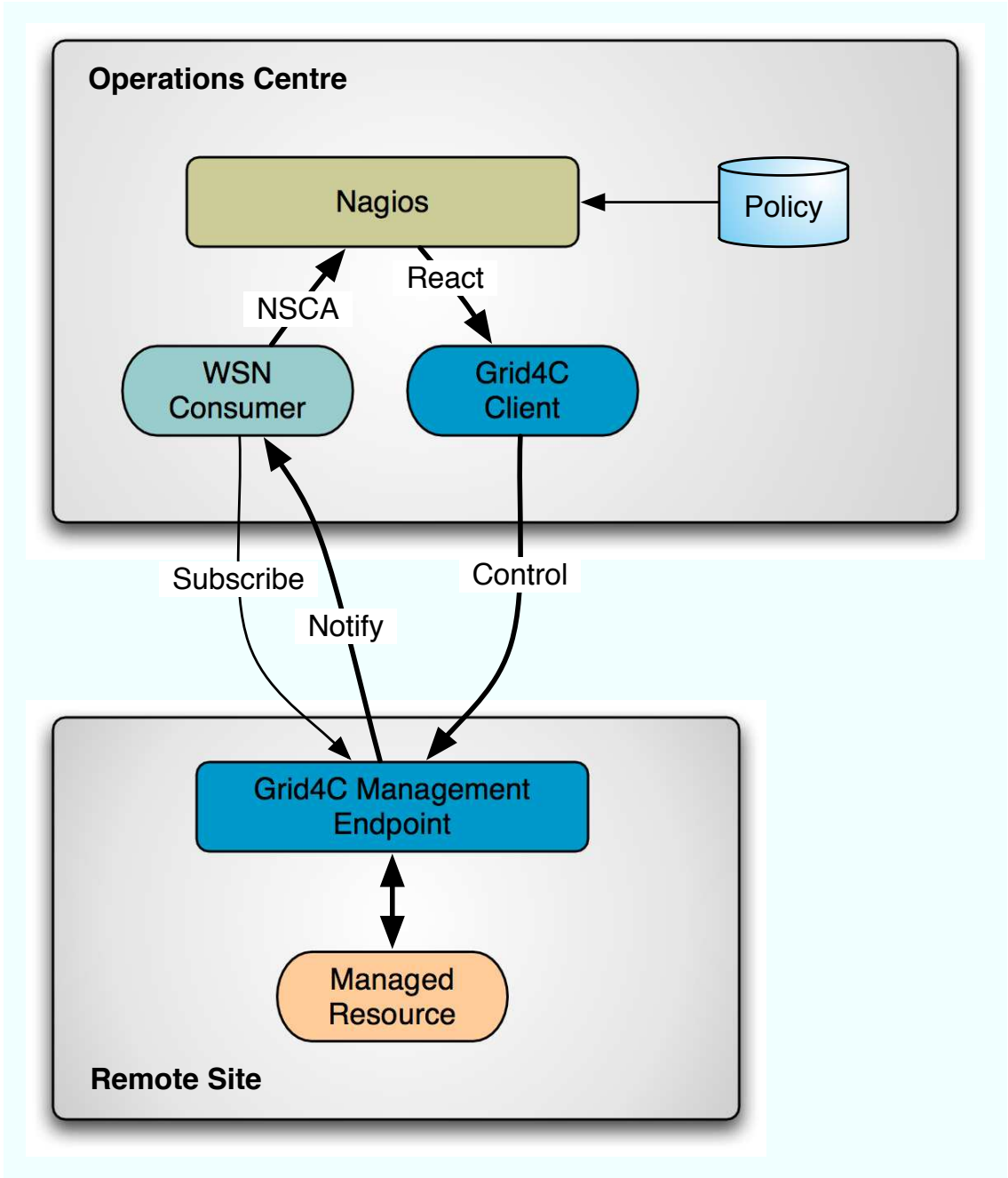


Figure 8.3: Nagios / Grid4C integration architecture.

An overview of Social Grid Agents

SGA's resource brokerage architecture follows the social and economic paradigm [FKL⁺99] [Sch02] [BAV05] [BAGS01] [GEJ⁺06] [WBPB03] and is also linked to the ongoing research on Grid interoperability [PLO⁺05a][PKC⁺06a].

The architecture is based on three layers:

- the execution layer, encompassing the existing Grid Resources (the gLite-WMS service in this particular case),
- the production layer where *Production Grid Agents* (henceforth referred to as *production agents*) compose various grid services as in a microeconomic supply chain, and
- the social layer where the *Social Grid Agents* (henceforth referred to as *social agents*) that own and control the agents in the lower layer engage in social and economic exchange.

Social agents allow different users to interact with each other in a variety of ways ranging from competitive to co-operative. Their design allows for a large variety of connection topologies among the social and production agents. Each social agent can own, control or temporarily use none, one or more production agents. A social agent can also exchange (with other social agents) the control of, or the services produced by, the production agent it controls.

Social agents make decisions in a multi-dimensional space of parameters and policies and a significant portion of such parameters and policies are based on *value* and *price*. It is important to notice that these two concepts are very different in this solution. Value is used in both the production and social level but it is not used explicitly in social transactions. Social decisions, on the other hand, are based both on value and price and social transactions can be based on price. We can explain this concept with an example. Social agent A is requested to perform service S for agent B at a price P. Agent A accepts this request based on a social policy that ties the minimum price P_{min} to the value that Social agent A associates with the required service V_S . This allows one to implement different pricing schemes based on social relations and the definition of value-based policies for bartering and non-monetary co-operative relationships.

Interaction Architecture

The architecture of the interaction between SGA agents and Grid4C capabilities is illustrated in Figure 8.4. Both agents access the underlying Grid services although through different

interfaces. SGA agents access the gLite WMS functionalities through the JAVA API interfaces while Grid4C Management Endpoints employ agents residing on the managed resources for data collection and resource administration.

The bridge between the two components is implemented as WSDM[OASd] compliant web services. This use of standards-based web service technologies not only facilitates interoperability and composition of management tools, but also provides an abstraction layer in which we can define a uniform set of metrics, properties and operations to be made available to the Social Grid Agents.

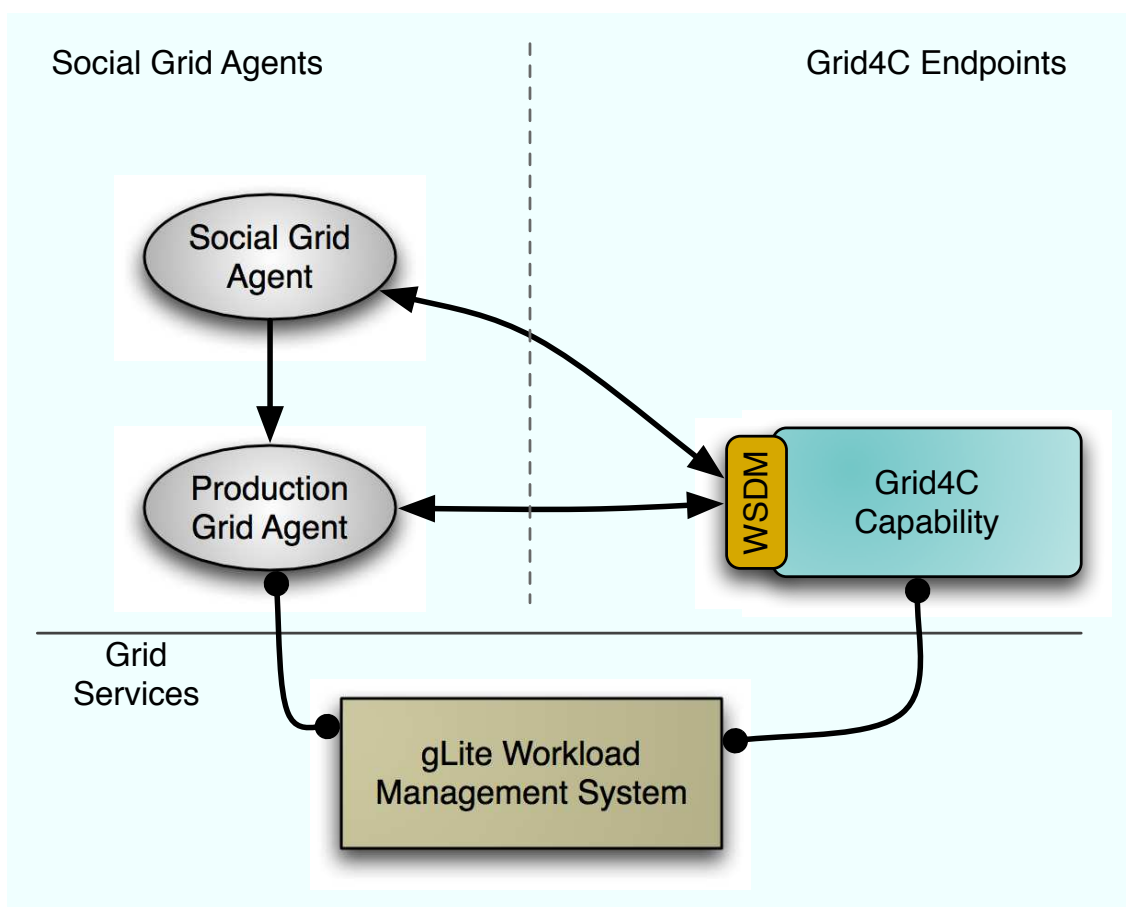


Figure 8.4: SGA and G4C integration architecture.

Interaction Topologies

This experiment investigated two main ways in which SGA and Grid4C capabilities can interact. In the first scenario described in Figure 8.4, SGA agents employ Grid4C capabilities

<i>ServiceMetrics</i>	<i>ResourceMetrics</i>
Operational Status and TCP response time	The Exit Status for a given job ID
Average Waiting Time for last n Jobs	The Wall-clock Time for a given job ID
Average Waiting Time since a given date	The CPU count for a given job ID
Average Match Time for last n Jobs	
Average Match Time since a given date	
Average Clear Time for last n Jobs	
Average Clear Time since a given date	
Percentage of the last n jobs aborted	

Table 8.2: Metrics for gLite Workload Management System.

in order to take advantage of their management functionalities and the information they are able to obtain from the Grid Services, the gLite Resource Brokers in this case. SGA agents have limited interfaces to the Grid Resources (in the gLite-WMS case the interface is currently limited to the UI API) and can make extensive use of the information provided by the Grid4C capabilities that have much richer interfaces to Grid resources. The information obtained via the Grid4C capabilities is used to assess the quality of the services and define a value both of the service provider and the specific service being performed.

The value of a service is a function of two different sets of metrics.

- Service Metrics - These metrics provide information on the abstract service.
- Resource Metrics - These metrics provide information on the resources that have been used by a specific execution of a service.

The value of a service is described as $V = f(m_s, m_r)$ where m_s represent the metrics of the service and m_r are the metrics representing the actual resource consumption. For the specific case of the gLite WMS (that is the testbed for the SGA-Grid4C interaction) the metrics are described in Table 8.2:

In the second scenario, described in Figure 8.5, Social Grid Agents and Grid4C are encompassed in a more complex topology which aim to create a virtual, implicit trust link where a direct one was missing. This interaction between SGA and Grid4C capabilities aims to resolve the issues of two social scenarios: the problem of the perception of a “fair price” and the need for arbitration that arises when the execution of a job fails. In these cases, if there is no direct trust between the client and the service provider, a third party, trusted by both, may resolve the issue. In the “fair price” problem this third actor, called the *arbitrator*, may determine a price for all the service providers that trust it. In the second case, the client may not be willing to pay for the execution of a failed job while the service provider may require

that the resources being used are to be paid for in any case. If the arbitrator is trusted by both the client and the service provider, then it can enquire into the status of the job and resolve the dispute. In this case, we say that there is an implicit trust link between the client and the service provider through the arbitrator.

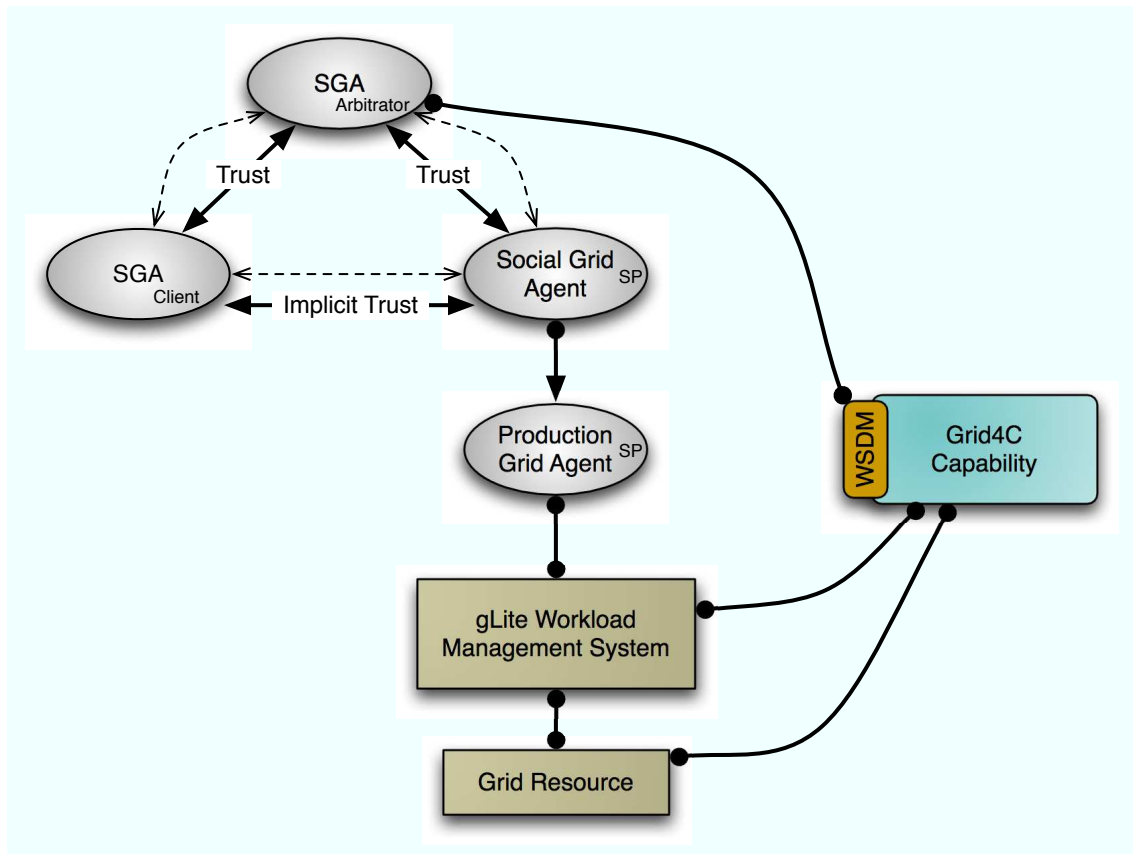


Figure 8.5: Complex interaction topology between SGA and Grid4C agents.

The Grid4C gLite Resource Broker Management Endpoint

In order to provide the information necessary for this experiment, a prototype endpoint was implemented to gather the necessary information from the gLite Resource broker. Although this test implementation was specific to the gLite RB, the exposure via a separate WSDM interface means that only the back-end logic would need to be changed for different grid middlewares, i.e. no modification of the SGA or client code would be necessary. Indeed, the use of these common interfaces should enable this economic evaluation of resources across multiple grid infrastructures. Potentially, even the use of the WMS API by the SGAs could

be replaced entirely by WSDM interfaces, enabling interoperability on an even wider scale.

In the interest of simplicity and flexibility, the processing logic for the endpoint interfaced directly to the back-end database of the Logging and Book-keeping server. This allowed for more complex and efficient queries to be constructed using SQL.

Conclusions

The flexible nature of both the SGA and Grid4C agents greatly facilitated the implementation of the interaction topologies described in this section. This allowed the implementation of pricing mechanisms that take into account resource consumption and the efficiency of the service. It also allowed the creation of a virtual, implicit trust link to solve the problems of arbitration and fair pricing of resources. Grid4C proved to be a critical enabling component of this prototype implementation for resource brokerage based on economic principles.

The experiment also served to evaluate and formalise the procedure for the integration of Grid4C management clients into external software components and provided a basis for further research in the area of programmatic control.

8.4.4 Integration with Active Security Infrastructure

This example Integrates a prototype IP-Tables Management Capability developed for integration into Grid4C management endpoints, with the Int.eu.grid Active Security Infrastructure (ASI). The Grid4C capability allows the ASI to inject rules into the firewall on the managed resource. Block rules can be injected (for all hosts not defined on an optional allowed ‘white list’) and subsequently removed if desired. It should be noted that in order to ensure that the base firewall configuration cannot be compromised, only rules injected by the system can be removed.

Overview of the Active Security Infrastructure

The Active Security project is a research activity from the Int.eu.grid project, and is being led by members of the Computer Architecture group at Trinity College Dublin. Building on concepts investigated during the CrossGrid project, Active Security is intended to go beyond existing prevention-based grid security measures by focusing on threat detection and reaction. Active Security is currently deployed in the Int.eu.grid development testbed and on Grid-Ireland. The architecture comprises three primary components:

- Security Monitoring

Security monitoring is a site-level activity in which the security status of a site is constantly monitored and any security events are reported to the operations centre where they are archived. Monitoring is performed by R-GMA enabled tools based on SNORT and Prelude-LML. The monitoring may be extended through the inclusion of additional tools such as Tripwire.

- Alert Analysis

Alert analysis occurs at the operational level. It's objective is the detection of patterns that signify an attempted attack through the filtering and analysis of events obtained from the site-level sensors. Possible outputs of the analysis phase include correlated high-priority grid alerts, and new grid policies defining actions to be taken in response to the security event. Analysis functionality may be extended through the definition of additional attack scenarios and base policies.

- Control Engine

The control engine is a site-level component powered by policies generated during the analysis phase. They constitute the policy decision point in that requests for guidance from local service agents are evaluated based on applicable policies. The results of these requests for guidance may contain actions to be carried out in order to mitigate the risk of a possible security incident. Site-level functionality may be extended through the provision of additional service agents or plug-in modules. The plug-ins are invoked when an updated policy is received.

Integration of Grid4C

The Active Security system provides an elegant solution for the analysis and detection of security events, and the definition of management policies to mitigate against detected threats. However, without a means to actively control or reconfigure components, the system remains powerless to effect the change required to reduce threat. While the developers have created a number of service agents, the true strength of the system lies in its extensible modular design allowing policy enforcement mechanisms to be easily added along with new attack scenarios and policies. This plug-in architecture makes it possible to leverage the functionality provided by Grid4C for the purposes of control configuration. The extensible nature, loose coupling, and standards-based interface of Grid4C management endpoints present an ideal and complimentary solution.

The combination of the two systems is illustrated in Figure 8.6. When evaluation of a new policy calls for action, the policy module invokes a Grid4C plugin to carry out that action via

the interfaces exposed by a management endpoint. This management endpoint may reside at the same site as the control engine or at a remote site. While the illustration shows the demonstrated reconfiguration of a network firewall, a number of additional actions are under investigation, including job and process management.

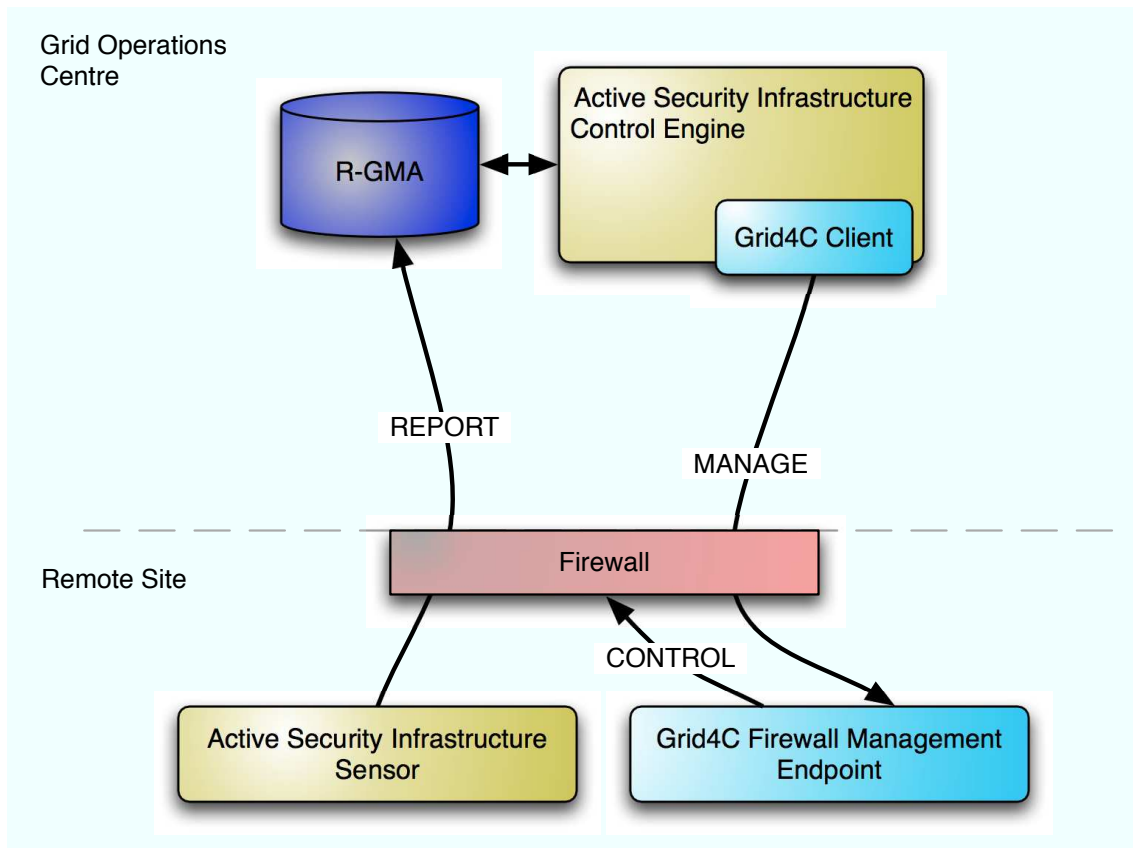


Figure 8.6: Integration of Grid4C components as Active Security policy enforcement points.

8.5 Summary

The above proofs-of-concept show that autonomic control is feasible even in the non-deterministic grid environment. The performance and scalability is very encouraging, and the simple use cases indicate the diversity of the possibilities that are attainable within a notably simple architecture.

Chapter 9

Visualisation and User Interface

Control centres are often created as central points for collecting, processing, storing, displaying, and acting upon data gathered from monitoring operations. The focal points of these facilities is often display systems that enable operators to view, navigate, and manipulate status information data. These displays provide decision makers with the degree of situational awareness required to support strategic and tactical decisions.

The display system should support a fully integrated control room environment combining access to information, monitoring, and control, and to communications technology such as voice and video etc. (for communications with site managers, remote operations teams, or federation members)

The primary objective is to provide operators with the information they need to maintain optimal levels of service by integrating and processing data from all available sources including static information such as configurations databases, asset registers, and contact information. Large format displays and communications technologies can also play an important part in supporting collaboration in management and operations.

There are many display technologies which can be applied to this scenario, some from the commodity markets, others that remain more common in industrial applications, and some emerging technologies for interaction and visualisation such as those that respond to operator movement and gesture. Interactions more commonly occur in the form of touch screens, touch tables, or operator consoles.

At the outset of this project, it was the intention to develop an entire control system for grid operations. Unfortunately, due to the magnitude of this undertaking the visualisation remains largely as future work. Rather, it was chosen to concentrate on the research and development of a open, and extensible architecture to support such visualisation and management. A bottom-up approach has been adopted based on the logic that the visualisation

and user interfaces will be of little value without the underlying infrastructure for data acquisition and control, and once in place, this infrastructure would provide an ideal platform for further development and experimentation. In this section however a number of figures illustrating the use of display technologies in the context of control centres is presented. In the context of the overall vision of Gric4C, interfaces such as these could be layered above and dependent upon the system described in Chapter 7.

9.1 Relevant Display Technologies



Figure 9.1: The videowall at the AT&T global network operations centre, one of the largest command and control centres in the world. One hundred and forty-one rear-projection video screens are employed, each measuring 1.2 by 1.5 meters.



Figure 9.2: Consoles in the CERN Control Centre for monitoring and control of the LHC apparatus.



Figure 9.3: The Machine Control Centre (MCC) at Jefferson Labs continuous beam accelerator facility (CEBAF).



(a) 1x2.

(b) 4x2.

Figure 9.4: Prototype operator consoles engineered by the author of this thesis.



Figure 9.5: The videowall in the video conferencing centre used by the Grid-Ireland Operations team.



Figure 9.6: TFT screens displaying monitoring information at the Grid-Ireland operations centre.

Chapter 10

Service Availability Models

In this chapter we explore some preliminary service state models that might be applied to grids. Some of the more commonly used expressions of availability are outlined, before a model of service behaviour based on considering service availability as a Markov chain of finite state space is described.

10.1 Determining Site and Service availability statistics

10.1.1 Measures of system availability

Availability and reliability can be described in terms of scalar quantities, although doing so can be misleading. Clear definitions of such measurements are required.

While availability is related to reliability, a clear distinction exists. Availability is the percentage of total time that the system is operational and providing the expected levels of service to its users. Reliability is an expression of the period for which the system is expected to be operational before a failure that will render it inoperable.

Availability is not a measurable attribute of a system (in the way that disk usage is for example) in that it can only be calculated historically, based on observed system behaviour.

$$Availability = \frac{OperatingTime}{ElapsedTime} \quad (10.1)$$

or

$$Availability = \frac{ElapsedTime - Sum(InoperativeTimes)}{ElapsedTime} \quad (10.2)$$

Availability is usually expressed as a percentage of a given sample period for which the resource or service was capable of fulfilling its duties and is calculated using the formula in

	99%	99.5%	99.95%	100%
24x7x365	88	44	5	0
12x5x52	32	16	2	0

Table 10.1: Annual hours of downtime for availability Vs expected period of operation

Equation 10.1. These sample periods raise an important issue relating to the expected period of operation of the service. Table 10.1 shows how the figure for the availability of a system relates the maximum hours of downtime depending on the expected period of operation.

The first line represents a system that is expected to be operational for 24 hours a day, 7 days a week, 365 days a year. The second line represents a system that is expected to be operational for 12 hours a day, 5 days a week, for 52 weeks of the year.

The table illustrates that for a given level of availability, more hours of unplanned downtime are allowed for the first system than the second. It is important to recognise however that there are 5642 hours of the year for which the second system is not expected to be available. It is for reasons such as this that figures quoted for availability should be fully explained.

The **mean time between failure** (MTBF) is a common measure of reliability and is often provided for individual system components. While the mean time between failure is a useful measurement, it does not convey any information about the expected recovery time after failure which, in the case of some components, can be considerable. The mean time between failure is calculated using the formula shown in Equation 10.3.

$$MTBF = \frac{TotalOperatingTime}{TotalNumberOfFailures} \quad (10.3)$$

The mean time between failure is calculated by summing the operating times for all units, including those for which no failure is recorded, and dividing that sum by the sum of all failures of the units. When calculating the MTBF for multiple instances of the same unit, the individual MTBF figures are divided by the number of units.

An important compliment to the figure of MTBF is the **mean time to repair** (MTTR) which is given by Equation 10.4.

$$MTTR = \frac{SumOfAllRepairTimes}{TotalNumberOfFailures} \quad (10.4)$$

Average Downtime describes the amount of time that a unit is inoperative per failure event. The ADT is calculated as per Equation 10.5

	%			
	csTCDie	cpDIASie	csULie	AITie
OK	94.47	91.69	98.18	98.59
WARNING	2.28	0.23	0.18	0.18
CRITICAL	3.23	7.38	1.62	1.21
WARNING	0.03	0.70	0.02	0.01

Table 10.2: Frequency Distribution of recorded states for Grid-Ireland Sites

	%			
	Install Server	UI	CE	SE
OK	81.78	93.40	98.80	99.90
WARNING	9.67	0.00	0.00	0.00
CRITICAL	8.54	6.60	1.01	0.09
WARNING	0.00	0.00	0.18	0.00

Table 10.3: Frequency Distribution of recorded states for Grid-Ireland site csTCDie

$$ADT = \frac{SumOfInoperativeTimes}{TotalNumberOfFailures} \quad (10.5)$$

The reliability of a unit can be represented in an easily understood value as the **Annualised Failure Rate**(AFR). The AFR takes in to account both the MTBF, MTTR, and the expected hours of operation of the unit. The formula for a unit expected to operate on a 24x7x365 basis is given in Equation 10.6

$$AFR = \frac{1}{MTBF + MTTR} * 8760 * 100\% \quad (10.6)$$

For example, a unit with an AFR of 50% will be expected to fail once every two years, while a unit with an AFR of 300% would be expected to fail three times a year.

	%			
	GRAM	GFTP	GRIS	SSH
OK	99.78	99.76	99.78	99.83
WARNING	0.00	0.00	0.00	0.00
CRITICAL	0.22	0.24	0.22	0.17
WARNING	0.00	0.00	0.00	0.00

Table 10.4: Frequency Distribution of recorded states for CE host at site csTCDie

State	Frequency	%	P
OK	21161	99.86786	0.99867857
WARNING	2	0.00943886	9.43886E-5
CRITICAL	26	0.12270518	0.0012270517
UNKNOWN	0	0.0	0.0

Table 10.5: Frequency Distribution of recorded states for HTTP service on GridInstall at TCD

10.2 Modelling Grid Service Availability as a finite space Markov Chain

While the metrics described above provide useful information regarding system availability and reliability, they provide little information about the behaviour of the system. Further analysis of the grid service status information recorded by the monitoring system described in Chapter 6 allows us to extract even more interesting information. In the following section I investigate the use of a statistical modelling technique to generate a description of system / service behaviour.

For example, if we analyse the recorded states for a particular service over a specific time interval we can construct the state frequency distribution table illustrated in 10.5. We can quickly deduce, from the mode of the distribution, that the vast majority samples showed this particular service to be available, or in an ‘OK’ state. Since the samples are taken at 15 minute intervals, we can also infer some information in relation to the total duration of downtime over the sample period. In addition, we can determine the total collection time for all samples, and consequently the Mean Time Between Failure.

Perhaps more interesting than the examination of the individual states is the examination of the transition between states over time. Consider that the system has a set of measured states $S = \{s_1, s_2, s_3, \dots, s_n\}$. The system starts in one of these states and potentially moves from one state to another. If the system is currently in state s_i , the probability of it moving to state s_j in the next step is termed the transition probability, denoted by P_{ij} . Alternatively the system might remain in the same state, denoted by P_{ii} . The starting state of the system may be described by a probability distribution on the set of states, known as a probability vector.

A Markov chain is a sequence of variables or states exhibiting the Markov property, i.e. that it is non-deterministic and that a given state does not fully determine the subsequent state. In other words, given the present state, the future state is independent of the past states. Given that the state at time t is denoted by X_t , a formal definition of a Markov chain is given by:

$$P(X_t = j | X_0 = i_0, X_1 = i_1, \dots, X_{t-1} = i_{t-1}) = P(X_t = j | X_{t-1} = i_{t-1}) \quad (10.7)$$

The set of all possible values of X defines the **state space** of the chain. For the purposes of this discussion, we that the system under measurement can be found to be in one of four possible states; available (OK), partially available (Warn), unavailable (Critical) and unknown (the state could not be determined). If we consider the status information collected for a given service to be a stochastic series of finite state space and as having the Markov property we can can analyse the state transitions over time in order to construct the stochastic matrix illustrated by:

$$P = \begin{bmatrix} 0.99972 & 0.00005 & 0.00024 & 0 \\ 0.50000 & 0.50000 & 0 & 0 \\ 0.19231 & 0 & 0.80769 & 0 \\ - & - & - & - \end{bmatrix} \quad (10.8)$$

To interpret this matrix, consider each of the columns and rows to be labelled OK, WARN, CRITICAL, and UNKNOWN respectively. The rows should be labelled as 'From' and the Columns labled 'To'. We can then determine that there is a 0.99972 probability of an OK state being followed by an OK state, or a 0.19231 probability that a Critical state will be followed by an Ok state, etc .

A blank entry in the matrix should be interpreted as meaning that no data for the transition exists within the set of sample data for this particular service rather than that the transition is not possible as might be suggested by a zero value entry. In the interest of simplicity, we could therefore reduce the row rank of this matrix to 3x3 before further calculation.

There are several classifications of Markov chains, two that are of particular relevance to this application are:

Absorbing An absorbing chain has one or more absorbing states. i.e. one the absorbing state is entered, it is impossible to exit.

Irreducible Also known as ergodic. A chain is considered irreducible if all transitions are possible, i.e. it is possible to get to any state from any state.

For example if a transition matrix representing the state of an automatically managed service was found to be absorbing, and that the absorption occurred in an adverse state, then it is likely that manual intervention would be required to remedy the situation and that the autonomic management would require tuning / review.

The same information as is illustrated in Equation 10.8 can also be illustrated using a directed graph as in 10.1 where the edges are labelled with the probability of the transition that each represents. The same data is presented without transitions for which we have no information in 10.2

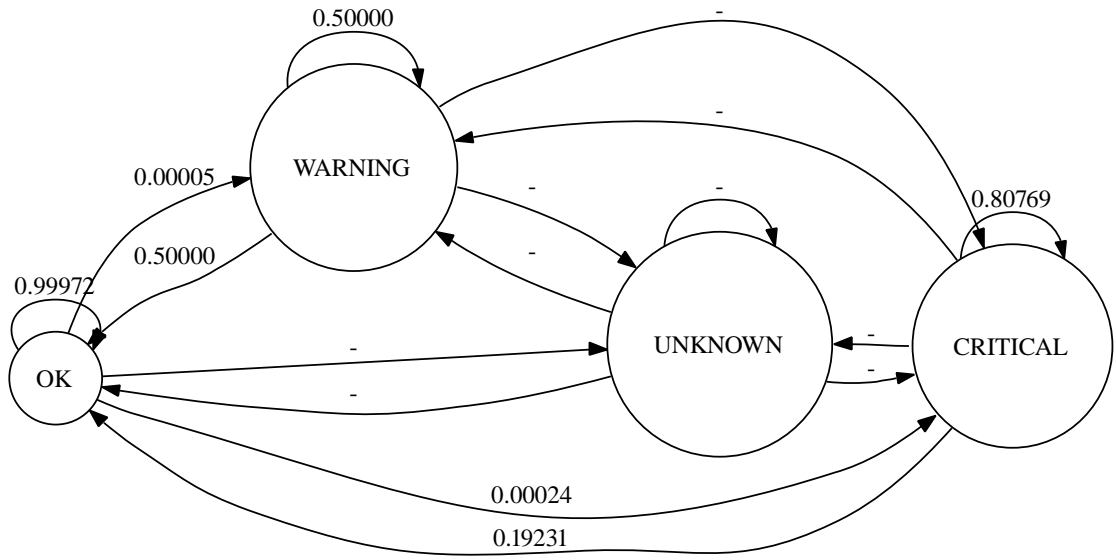


Figure 10.1: Directed graph showing all possible transitions

Modelling the service state in this way allows us to infer far more interesting information from an operations point of view. For example,

- That the 0.99972 probability of an OK state being followed by an OK state describes the reliability of the service
- The probability of a WARN or CRITICAL state being followed by an OK state tells us about the probability of recovery or the efficiency of service management. This is further discussed in 10.2.3

10.2.1 Predicting future states

The information provided by the transition matrix, such as 10.8, makes it possible to make predictions about how the state might evolve over time given an awareness of the current state.

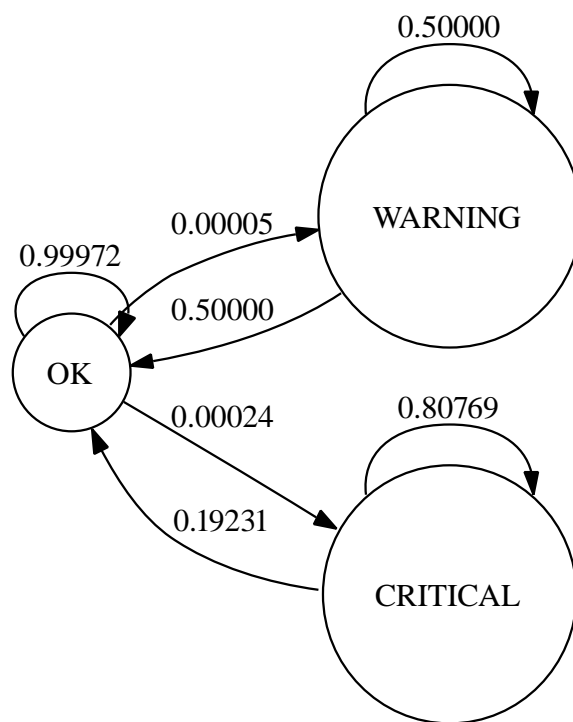


Figure 10.2: Directed graph showing state transition probabilities for HTTP service on Gridinstall at TCD

If the current state is known to be 'OK' as represented by the vector in Equation 10.9

$$x^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \quad (10.9)$$

the subsequent state can be predicted by:

$$\begin{aligned} x^{(1)} &= x^{(0)}P = \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.99972 & 0.00005 & 0.00024 & 0 \\ 0.50000 & 0.50000 & 0 & 0 \\ 0.19231 & 0 & 0.80769 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \\ &= \begin{bmatrix} 0.99972 & 0.00005 & 0.00024 & 0 \end{bmatrix} \end{aligned} \quad (10.10)$$

Thus there is a 0.99972 probability that the subsequent check will return a status of OK. We can predict the state after two time steps in a similar way:

$$\begin{aligned} x^{(2)} &= x^{(1)}P = \\ &= \begin{bmatrix} 0.99972 & 0.00005 & 0.00024 & 0 \end{bmatrix} \begin{bmatrix} 0.99972 & 0.00005 & 0.00024 & 0 \\ 0.50000 & 0.50000 & 0 & 0 \\ 0.19231 & 0 & 0.80769 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \\ &= \begin{bmatrix} 0.9995 & 0.00007 & 0.00043 & 0 \end{bmatrix} \end{aligned} \quad (10.11)$$

or

$$\begin{aligned} x^{(2)} &= x^{(0)}P^2 = \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.99972 & 0.00005 & 0.00024 & 0 \\ 0.50000 & 0.50000 & 0 & 0 \\ 0.19231 & 0 & 0.80769 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}^2 = \\ &= \begin{bmatrix} 0.9995 & 0.00007 & 0.00043 & 0 \end{bmatrix} \end{aligned} \quad (10.12)$$

In general, the probability distribution for step n is given by Equation 10.13.

$$x^{(n)} = x^{(0)}P^n \quad (10.13)$$

10.2.2 Evaluating the steady state

Using the formula above it becomes obvious that predictions of state become increasingly uncertain with every time step away from the current state. Further, the predictions can be seen to tend towards a steady state which represents the probability of each state on all samples, irrespective of the initial state. This is true for a particular type of chain known as a regular chain, where long range predictions calculated using Equation 10.13 prove independent of the starting state. i.e. the chain converges to the stationary distribution irrespective of its starting state.

The steady state vector is defined as:

$$q = \lim_{x \rightarrow \infty} x^{(n)} \quad (10.14)$$

where q is an eigenvector derived from P such that $qP = q$, therefore q has an eigenvalue of 1. The property that q is unchanged by P to calculate the steady state vector. Alternatively the steady state may be approximated by raising P to a suitably high power.

If we assume that the chain representing the state events is irreducible, i.e. that it is possible to reach any state from any state, and that it is non-absorbing, i.e. that there is no absorbing state, then it follows that it is possible to construct a regular transition matrix to represent the transitions in the chain and the steady state will therefore converge to a strictly positive vector.

The steady state can be approximated by raising P to a suitable large power, e.g. 256 as shown in Equation 10.15

$$\begin{aligned}
 P^{256} &= \begin{bmatrix} 0.99972 & 0.00005 & 0.00024 & 0 \\ 0.50000 & 0.50000 & 0 & 0 \\ 0.19231 & 0 & 0.80769 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}^{265} \\
 &= \begin{bmatrix} 0.99867 & 0.00007 & 0.00124 & 0 \end{bmatrix}
 \end{aligned} \quad (10.15)$$

We can therefore conclude that in general, for this particular service, 0.99867 percent of service checks will show the status to be ‘OK’.

10.2.3 Multi-step Transition Probabilities

The analysis of multi-step transition probabilities provide additional information of interest such as the probability of going from state i to state j in n steps

Of particular interest from an availability and service management point of view is the number of steps in the chain that is required to transition from an adverse state to an available state. For example, it might enable us to calculate, on average, how many steps (samples with intervals of known duration) it would take for a given service to return to an OK state once it had reached a Critical state. Provided the sample intervals are known and of sufficiently short duration, this would allow an estimate to be made of the duration of service outages. This estimated recovery time might allow us to base the value of a resource not only on the availability but also provides an indication of the efficiency with which the resource is managed. In addition, calculations of the mean passage time between states might provide information analogous to the mean time between failure.

10.3 Summary

In this chapter I have presented a number of typical measurements of system state. I have also outlined a proposed method for the description of service availability based on the use of Markov models. These models have the potential to provide an interesting insight into the life-cycle and availability of resources. In addition to their usefulness in the area of monitoring and operations, it is envisaged that models such as these could be usefully applied for the purposes of resource selection / valuation.

Chapter 11

Conclusions

11.1 Conclusions

This thesis has introduced an architecture and a proof-of-concept implementation for the exercise of command and control in the management of Grid infrastructures. In addition, the wider issues of grid monitoring and management have been explored.

The first contribution of this thesis is the architectural design and robust implementation of a distributed monitoring system, I4C, described in Chapter 6. This work demonstrated the integration of existing tools and mechanisms, and custom code into an architecture designed to meet the monitoring requirements of a centrally managed infrastructure. The system has been deployed on the production Grid-Ireland infrastructure for some years now and provides both a valuable tool to the operations effort, and a robust foundation and source of information on which to base further research.

The second and perhaps most important contribution is the architecture of Grid4C described in Chapter 7. This work involved the evaluation of standard web service technologies for the management of resources, and the design of an architecture that captures clear and simple concepts and allows their use in a manner that is practicable in the management of distributed computing resources. As all researchers will acknowledge, one arrives at elegance and simplicity through a difficult trajectory.

A proof-of-concept of the architecture was also implemented, showing how the management capabilities of grid resources and services can be exposed in a standard way and take advantage of existing work in the web services arena such as WSRP and WS-Notifications, a process that has been notably absent from many of the other contributions to the field of grid monitoring. The implementation also provided a secure, lightweight, and extensible architecture for intra-site host monitoring and management in the form of the Control Elements and

Final Elements.

In Chapter 10 I proposed a representation of grid service availability based on the steady state analysis of transition probability matrices constructed from availability measurements. The test data for the evaluation of the concept was acquired using the I4C monitoring system described in Chapter 6.

A number of test applications were described in Chapter 8, which served to demonstrate the suitability of the proposed control architecture for use cases in both manual and autonomic management processes. The use cases evaluated included the provision of advanced job submission metrics via a Resource Broker capability, policy based autonomic control of grid services, integration of the architecture to extend the scope of existing tools, and the provision of policy enforcement points/actuators for an active security infrastructure.

11.2 Future Work

While the prototype system described in this thesis forms the basis of a useful tool in its own right, there is considerable scope for further work.

Continued development of the components of the Grid4C system is expected in addition to further work in the areas of autonomic control, policy based management and event persistence. The use of visualisation tools as user interfaces to the control system, along with the development of mobile clients, might also be actively investigated.

A Sourceforge project has been set up [i4ca] to promote the deployment and further development of the both the I4C and Grid4C architectures. It is hoped that it will attract the interest of other users and developers, particularly those interested in tailoring the solution to their own specific infrastructure requirements.

The following sections outline a number of possibilities for future work which emerge from, and are made possible by this this research and implementation.

11.2.1 WSDM-Management of web services

In this discussion the use of WSDM has been limited to WSDM MUWS (Management using web services) for the management of resources and network services. With the migration toward grid middleware technologies based on web services, there is considerable scope for investigation into how WSDM MOWS capabilities might be incorporated into such grid services so that standards-based manageability can be natively supported.

11.2.2 Complex Event Processing

An interesting possibility is the use of complex event processing in notification handlers that subscribe to all events from a given resource or service group and reason upon the complete set. A simple example might monitor the response times of all services to detect anomalies, and once the handlers suspicions were aroused, it could invoke further operations in order to verify the situation and execute management actions if required.

11.2.3 Social Grid Agents

Demand-driven Resource Allocation with Social Grid Agents and Grid4C

Following on from the earlier work in the area of price and value determination for grid resources, described in Section 8.4.3, Social Grid Agents could take a more active role in the management and provision of grid resources by using Grid4C endpoints to provide a means for social grid agents to exercise control. This would enable several advanced scenarios. For example, an SGA representing a resource provider might detect a demand in the market and re-configure its resources in order to best align the available resources with that demand. This would allow resource providers to exercise control in order to maximise their resource utilisation or revenue. In the case where multiple sites are ‘competing’ for the work, the value metrics determined via the work described in Section 8.4.3 could be used to ascertain which site is best.

Market-based management

Another interesting concept is that when resources or management components require a complex management action to be performed, the equivalent of a request for tender might be published to invite capable agents to tender for the operations quoting their various attributes. Thus the ‘values’ offered by different managing resources could be used to choose the best agent to perform the actions given some criteria such as time-to-execution, success rate, etc.

11.2.4 Advanced Visualisation

The system properties and event streams obtainable from Grid4C could easily be leveraged to generate models for 3D rendering on systems. Users could then interact with the models and trigger actions on remote resources from within the visualisation using the Grid4C client/proxy objects. 2D overlays might be used to display relevant information during the navigation. These interfaces could greatly increase the effectiveness of activities such as problem root-cause analysis and education.

11.2.5 WSRP for Grid Management Portals

The benefits of WSDM web services could be combined with those of WSRP (Web Service for Remote Portlets) allowing not only the management functionality of resources to be exposed via standard mechanisms but also resource management user interfaces, which could be integrated into any standards-compliant portal framework.

11.2.6 Discovery and autonomic management

By creating management endpoints that implemented event consumer interfaces in addition to event producer interfaces it would be possible to create interesting management topologies. Consider for example the automatic creation of peer groups of resources or ‘buddy’ systems where one management endpoint discovers or is assigned another to monitor and manage. The managing endpoint might consult web services at the Operations Centre in order to solicit policy guidance for actions in response to status events. Replication of such managing endpoints could serve to improve availability.

11.2.7 Developing system models

Further modelling of grid service and resource behaviour is required in order enable more advanced experimentation with the application of various control mechanisms such as Model Predictive Control.

11.2.8 Using WS-BPEL to execute management workflows across Grid4C Endpoints

There are often situations in systems management when it is desirable to execute a sequence of operations involving multiple resources or to execute the same set of operations across multiple resources in parallel. A very simple example of the first might be the life-cycle management of a dependent chain of resources, such as ensuring that central services are fully operational before bringing dependent resources online. Examples of the latter include the deployment of new software revisions or a high priority reconfiguration of resource access control in response to a security alert.

Such sequences of operations may be conveniently represented as workflows. A common format for the representation of such workflows is WS-BPEL (Web Services Business Process Execution Language), an OASIS standard with broad industry support. WS-BPEL is an XML-based workflow definition language that can describe a business process based on the interactions between the process itself and its partner services. It defines how service

interactions should be co-ordinated in order to achieve a business goal. A process defined in WS-BPEL is executed in a workflow engine which is responsible for orchestrating the service invocations declared within the file.

Being web service and standards based, the architecture of the Grid4C prototype is eminently suitable to invocation from web service workflows. Typically, the required operations would be described in a workflow definition file and deployed to a workflow engine such as Apache ODE. The workflow could then be exposed and invoked as if it were a simple web service.

11.3 Personal Note

The writing of this thesis has not been an easy undertaking. The very fact that I can now say *if only I had known X at the start...*, proves just how much I have learned during the course of the research and what a valuable experience this has been. The learning and achievement of these last few years reaches far beyond the covers of this thesis and I can honestly say that personal growth along the way has contributed to making this work most rewarding. Progress can feel frustratingly slow at times, but such is the nature of the work and the rewards proportionate. Of course there are moments when ideas flourish and extra effort is required to remain focused on achievable goals, and establishing foundations for those exciting ideas. It is the exciting times such as these or conversations with fellow researchers that maintains ones drive and enthusiasm.

My interest in this area began at a undergraduate level and it has given me great satisfaction to pursue those interests to this level. I will continue to follow with interest, future developments in the field.

Bibliography

- [3Te] 3Tera. Applogic - grid operating system for web applications. <http://www.3tera.com/AppLogic/>.
- [ABF⁺04] S. Andreozzi, N. De Bortoli, S. Fantinel, G. Tortone, and M.C. Vistoli. GridICE: a Monitoring Service for the Grid. In *Proceedings of the Cracow Grid Workshop (CGW2003), Cracow, Poland, October 27-29, 2003*. Academic Computer Centre CYFRONET AGH, Feb 2004.
- [ACC⁺05] J. Astalos, R. Cecchini, B.A. Coghlan, R.D. Cowles, U. Epting, T.J. Genovese, J. Gomes, D. Groep, M. Gug, A.B. Hanushevsky, M. Helm, J.G. Jensen, C. Kanellopoulos, D.P. Kelsey, R. Marco, I. Neilson, S. Nicoud, D. O'Callaghan, D. Quesnel, I. Schaeffner, L. Shamardin, D. Skow, M. Sova, A. Wäänänen, P. Wolniewicz, and W. Xing. International grid CA interworking, peer review and policy management through the European DataGrid certification authority coordination group. In Peter M.A. Sloot, Alfons G. Hoekstra, Thierry Priol, Alexander Reinefeld, and Marian Bubak, editors, *Advances in Grid Computing — EGC 2005*, LNCS3470, pages 275–285, Amsterdam, The Netherlands, February 2005. Springer.
- [ACK⁺02] David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer. Seti@home: an experiment in public-resource computing. *Commun. ACM*, 45(11):56–61, 2002.
- [ADBF⁺03] S. Andreozzi, N. De Bortoli, S. Fantinel, A. Ghiselli, G. Tortone, and C. Vistoli. Gridice: a monitoring service for the grid. Proc. Cracow Grid Workshop, Poland, December December, 2003.
- [AFRZ07] J. Astalos, L. Flis, M. Radecki, and W. Ziajka. Performance improvements to bdi - grid information service in egee. In *CGW'07 Proceedings*, pages 398–404, ACC CYFRONET AGH, Krakow, 2007.

- [AKS05] Sahin Albayrak, Silvan Kaiser, and Jan Stender. Advanced grid management software for seamless services. *Multiagent Grid Syst.*, 1(4):263–270, 2005.
- [Amaa] Amazon. Amazon elastic compute cloud. <http://aws.amazon.com/ec2/>.
- [Amab] Amazon. Amazon simple storage service. <http://aws.amazon.com/s3/>.
- [Amd00] Gene M. Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. pages 79–81, 2000.
- [aMS⁺03] B. ?awniczeka, G Majkaa, P. S?owikowskia, K. Zieli?ska, and S?awomir Zieli?ska. Grid infrastructure monitoring service framework jiro/jmx based implementation. *Electronic Notes in Theoretical Computer Science*, pages 77–88, 2003.
- [And04] David P. Anderson. Boinc: A system for public-resource computing and storage. In *GRID '04: Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing*, pages 4–10, Washington, DC, USA, 2004. IEEE Computer Society.
- [Apa] The apache muse project. <http://ws.apache.org/muse/>.
- [Apab] Apache xml-rpc. <http://ws.apache.org/xmlrpc/>.
- [Apac] Axis2 - apache webservices project. <http://ws.apache.org/axis2/>.
- [Apad] Rampart : Ws-security module for axis2. http://ws.apache.org/axis2/modules/rampart/1_0/security-module.html.
- [ASV03] Sergio Androozzi, Massimo Sgaravatto, and Maria Cristina Vistoli. Sharing a conceptual model of grid resources and services. *CoRR*, cs.DC/0306111, 2003. informal publication.
- [Ave] Paul Avery. Virtual data and real experiments. <http://citeseer.ist.psu.edu/avery00virtual.html>.
- [Axi] Axis - apache webservices project. <http://ws.apache.org/axis/>.
- [BAGS01] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger. Economic models for management of Resources in Peer-to-Peer and Grid computing. In *Technical Track on Commercial Applications for High-Performance Computing, SPIE International Symposium on The Convergence of Information Technologies and Communications (ITCom 2001)*, Denver, Colorado, USA, August 2001.

- [BAV05] R. Buyya, D. Abramson, and S. Venogopal. The Grid Economy. In *Special Issue on Grid Computing, Proceedings of the IEEE, Manish Parashar and Craig Lee (editors), Volume 93, Issue 3, 698-714pp, IEEE Press, New York, USA, March 2005.*
- [BBF⁺04] Bartosz Balis, Marian Bubak, Wlodzimierz Funika, Tomasz Szeplieniec, and Roland Wismuller. Monitoring of interactive grid applications. pages 265–273, 2004.
- [BCC⁺02] R. Byrom, B.A. Coghlan, A. Cooke, R. Cordenonsi, L. Cornwall, A. Datta, A. Djaoui, L. Field, S. Fisher, S. Hicks, S. Kenny, J. Magowan, W. Nutt, D. O’Callaghan, M. Oever, N. Podhorszki, J. Ryan, M. Soni, P. Taylor, A. Wilson, and X. Zhu. R-gma: A relational grid information and monitoring system. Proc. Cracow Grid Workshop, Poland, December December, 2002.
- [BCC⁺03] Rob Byrom, Brian A. Coghlan, Andrew W. Cooke, Roney Cordenonsi, Linda Cornwall, Abdeslem Djaoui, Laurence Field, Steve Fisher, Steve Hicks, Stuart Kenny, Jason Leake, James Magowan, Werner Nutt, David O’Callaghan, Norbert Podhorszki, John Ryan, Manish Soni, Paul Taylor, and Antony J. Wilson. Relational grid monitoring architecture (r-gma). *CoRR*, cs.DC/0308024, 2003. informal publication.
- [BCC⁺04] R. Byrom, B. Coghlan, A. Cooke, R. Cordenonsi, L. Cornwall, A. Datta, A. Djaoui, L. Field, S. Fisher, S. Hicks, S. Kenny, J. Magowan, W. Nutt, D. O’Callaghan, M. Oever, N. Podhorszki, J. Ryan, M. Soni, P. Taylor, A. Wilson, and X. Zhu. The canonicalproducer: An instrument monitoring component of the relational grid monitoring architecture (r-gma). In *Proc. ISPDC 2004*, pages 232–237. IEEE Computer Society, July 2004.
- [BCC⁺05a] R. Byrom, B. Coghlan, A. Cooke, R. Cordenonsi, L. Cornwall, M. Craig, A. Djaoui, A. Duncan, S. Fisher, A. Gray, S. Hicks, S. Kenny, J. Leake, O. Lytleton, J. Magowan, R. Middleton, W. Nutt, D. O’Callaghan, N. Podhorszki, P. Taylor, J. Walk, and A. Wilson. Fault tolerance in the r-gma information and monitoring system. In Peter M.A. Sloot, Alfons G. Hoekstra, Thierry Priol, Alexander Reinefeld, and Marian Bubak, editors, *Advances in Grid Computing - EGC 2005*, LNCS3470, Amsterdam, The Netherlands, February 2005. Springer.
- [BCC⁺05b] R. Byrom, B. Coghlan, A. Cooke, R. Cordenonsi, L. Cornwall, A. Datta,

- A. Djaoui, L. Field, S. Fisher, S. Hicks, S. Kenny, J. Magowan, W. Nutt, D. O’Callaghan, M. Oever, N. Podhorszki, J. Ryan, M. Soni, P. Taylor, A. Wilson, and X. Zhu. The canonical producer: an instrument monitoring component of the relational grid monitoring architecture (r-gma). *Special Issue of Scientific Programming*, 13(2):151–158, October 2005.
- [BDF⁺03] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Timothy L. Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In Michael L. Scott and Larry L. Peterson, editors, *SOSP*, pages 164–177. ACM, 2003.
- [BF02] Primet P. Bonnassieux F., Harakaly R. Mapcenter: an open grid status visualization tool. In *ISCA 15th International Conference on parallel and distributed computing systems*, Louisville, Kentucky, USA, September September 19-21, 2002.
- [BHM⁺04] David Booth, Hugo Haas, Francis McCabe, Eric Newcomer, Michael Champion, Chris Ferris, and David Orchard. Web services architecture. W3C Note NOTE-*ws-arch-20040211*, World Wide Web Consortium, February 2004.
- [BKPV01] Z. Balaton, P. Kacsuk, N. Podhorszki, and F. Vajda. Use cases and the proposed grid monitoring architecture. Technical Report LDS-1/2001, Computer and Automation Research Institute of the Hungarian Academy of Sciences, 2001.
- [BKS05] Rüdiger Berlich, Marcel Kunze, and Kilian Schwarz. Grid computing in europe: from research to deployment. In *ACSW Frontiers ’05: Proceedings of the 2005 Australasian workshop on Grid computing and e-research*, pages 21–27, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.
- [BMC] BMC. Bmc remedy customer service and support. <http://www.bmc.com/remedy/>.
- [BMM⁺02] M. Bubak, J. Marco, H. Marten, N. Meyer, M. Noga, P.A.M. Sloot, and M. Tura?a. Crossgrid - development of grid environment for interactive applications. In *Pioneer Conference*, April 2002.
- [BSM⁺06] Marc-Elian Bgin, Guillermo Diez-Andino Sancho, Alberto Di Meglio, Enrico Ferro, Elisabetta Ronchieri, Matteo Selmi, and Marian Zurek. Build, configuration, integration and testing tools for large software projects: Etics. In

Nicolas Guelfi and Didier Buchs, editors, *RISE*, volume 4401 of *Lecture Notes in Computer Science*, pages 81–97. Springer, 2006.

- [Cat03] Charlie Catlett. Standards for grid computing: Global grid forum. *J. Grid Comput.*, 1(1):3–7, 2003.
- [CC06] S. Childs and B. Coghlan. How to join the virtual revolution. *CERN Courier*, 46(5), June 2006.
- [CCM06a] S. Childs, B. Coghlan, and J. McCandless. Dynamic virtual worker nodes in a production Grid. In *Frontiers of High Performance Computing and Networking – ISPA 2006 Workshops (XHPC 2006)*, volume 4331/2006, pages 417–426, Sorrento, Italy, December 2006.
- [CCM06b] S. Childs, B. Coghlan, and J. McCandless. GridBuilder: A tool for creating virtual Grid testbeds. In *2nd IEEE Conference on eScience and Grid computing*, Amsterdam, December 2006.
- [CCO⁺04] S. Childs, B.A. Coghlan, D. O’Callaghan, G. Quigley, and J. Walsh. Grid testing using virtual machines. In Marian Bubak, Michal Turala, and Kazimierz Wiatr, editors, *Proc. Cracow Grid Workshop (CGW’04)*, pages 371–378, Cracow, Poland, December 2004. Academic Computer Centre CYFRONET AGH.
- [CCO⁺05a] S. Childs, B. Coghlan, D. O’Callaghan, G. Quigley, , and J. Walsh. Centralised fabric management for a national grid infrastructure. In *Cracow Grid Workshop (CGW’05)*, Cracow, Poland, November 2005.
- [CCO⁺05b] S. Childs, B. Coghlan, D. O’Callaghan, G. Quigley, and J. Walsh. A single-computer grid gateway using virtual machines. In *Proc. AINA 2005*, pages 761–770, Taiwan, March 2005. IEEE Computer Society.
- [CCO⁺05c] S. Childs, B.A. Coghlan, D. O’Callaghan, G. Quigley, and J. Walsh. Deployment of grid gateways using virtual machines. In Peter M.A. Sloot, Alfons G. Hoekstra, Thierry Priol, Alexander Reinefeld, and Marian Bubak, editors, *Advances in Grid Computing - EGC 2005*, LNCS3470, Amsterdam, The Netherlands, February 2005. Springer.
- [CCO⁺06] S. Childs, B. Coghlan, D. O’Callaghan, G. Quigley, J. Walsh, and E. Kenny. A virtual testgrid or how to replicate a national grid. In *Proceedings of the EXPGRID workshop on Experimental Grid testbeds for the assessment of large-scale distributed applications and tools*, Paris, June 2006.

- [CDO⁺00] L. Childers, T. Disz, R. Olson, M. Papka, R. Stevens, and T. Udeshi. Access grid: Immersive group-to-group collaborative visualization, 2000.
- [CDR⁺03] K. Czajkowski, A. Dan, J. Rofrano, S. Tuecke, and M. Xu. Agreement-based grid service management (ogsi-agreement), 2003.
- [Cen] NIIF Supercomputing Center. The hungarian clustergrid infrastructure. <http://www.clustergrid.niif.hu/>.
- [CERa] CERN. The large hadron collider. <http://lh.web.cern.ch/lhc/>.
- [CERb] CERN. Lemon - lhc era monitoring. <http://lemon.web.cern.ch/>.
- [CFFK01] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman. Grid information services for distributed resource sharing, 2001.
- [CGM⁺03] A. Cooke, A. Gray, L. Ma, W. Nutt, J. Magowan, M. Oevers, P. Taylor, R. Byrom, L. Field, S. Hicks, J. Leake, M. Soni, A. Wilson, R. Cordenonsi, L. Cornwall, A. Djaoui, S. Fisher, N. Podhorszki, B. Coghlan, S. Kenny, and D. O’Callaghan. R-gma: An information integration system for grid monitoring. In *Proc.Int.Conf. Cooperative Information Systems (CoopIS’03)*, Catania, Sicily, November November, 2003.
- [CGN⁺04] A.W. Cooke, A.J.G. Gray, W. Nutt, J. Magowan, M. Oevers, P. Taylor, R. Cordenonsi, R. Byrom, L. Cornwall, A. Djaoui, L. Field, S.M. Fisher, S. Hicks, J. Leake, R. Middleton, A. Wilson, X. Zhu, N. Podhorszki, B. Coghlan, S. Kenny, D. O’Callaghan, and J. Ryan. The relational grid monitoring architecture: Mediating information about the grid. *Journal of Grid Computing*, 2:323–339, December 2004.
- [CIT] CITL. Centre for information technology leadership. <http://www.citl.org/>.
- [CJK⁺04] L.A. Cornwall, J. Jensen, D.P. Kelsey, A. Frohner, D. Kouril, F. Bonnassieux, S. Nicoud, K. Lorentey, J. Hahkala, M. Silander, R. Cecchini, V. Ciaschini, L. dell’Agnello, F. Spataro, D. O’Callaghan, O. Mulmo, G.-L. Volpato, D. Groep, M. Steenbakkens, and A. McNab. Authentication and authorization mechanisms for multi-domain grid environments. *Journal of Grid Computing*, 2:301–311, December 2004.
- [CK] B.A. Coghlan and S. Kenny. Grid-wide intrusion detection: First results after deployment. *Submitted to special issue of the Journal of Parallel and Distributed Computing on Security in Grid and Distributed Systems*.

- [CMC⁺06] Kathryn Cassidy, Jason McCandless, Stephen Childs, John Walsh, Brian Coghlan, and Declan Dagger. Combining a virtual grid testbed and grid elearning courseware. In *Proc. Cracow Grid Workshop 2006 (CGW06)*, Cracow, Poland, October 2006. Academic Computer Centre CYFRONET AGH.
- [Cor] Cordis. European strategy forum on research infrastructures. <http://cordis.europa.eu/esfri/>.
- [CST⁺02] Junwei Cao, Daniel Spooner, James D. Turner, Stephen Jarvis, Darren J. Kerbyson, Subhash Saini, and Graham Nudd. Agent-based resource management for grid computing. In *CCGRID '02: Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, page 350, Washington, DC, USA, 2002. IEEE Computer Society.
- [CWO05] B.A. Coghlan, J. Walsh, and D. O'Callaghan. Grid-ireland deployment architecture. In Peter M.A. Sloot, Alfons G. Hoekstra, Thierry Priol, Alexander Reinefeld, and Marian Bubak, editors, *Advances in Grid Computing - EGC 2005*, LNCS3470, Amsterdam, The Netherlands, February February, 2005. Springer.
- [CWQ⁺04] B.A. Coghlan, J. Walsh, G. Quigley, D. O'Callaghan, S. Childs, , and E. Kenny. Transactional grid deployment. In Marian Bubak, Michal Turala, and Kazimierz Wiatr, editors, *Proc. Crakow Grid Workshop (CGW'04)*, pages 363–370, Cracow, Poland, December 2004. Academic Computer Centre CYFRONET AGH.
- [CWQ⁺05] B.A. Coghlan, J. Walsh, G. Quigley, D. O'Callaghan, S. Childs, and E. Kenny. Principles of transactional grid deployment. In Peter M.A. Sloot, Alfons G. Hoekstra, Thierry Priol, Alexander Reinefeld, and Marian Bubak, editors, *Advances in Grid Computing - EGC 2005*, LNCS3470, pages 88–97, Amsterdam, The Netherlands, February 2005. Springer.
- [Dat] European DataGrid. Edg. <http://eu-datagrid.web.cern.ch/>.
- [DEI] DEISA. Distributed european infrastructure for supercomputing applications. <http://www.deisa.org/>.
- [Del] Dell. Openmanage systems management. <http://www.dell.com/openmanage/>.

- [DF01] Olivier Dubuisson and Philippe Fouquart. *ASN.1: communication between heterogeneous systems*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.
- [DFP⁺96] Thomas A. DeFanti, Ian Foster, Michael E. Papka, Rick Stevens, and Tim Kuhfuss. Overview of the I-WAY: Wide-area visual supercomputing. *The International Journal of Supercomputer Applications and High Performance Computing*, 10(2/3):123–131, Summer/Fall 1996.
- [DFT91] John Comstock Doyle, Bruce A. Francis, and Allen R. Tannenbaum. *Feedback Control Theory*. Prentice Hall Professional Technical Reference, 1991.
- [DG04] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. In *OSDI'04: Proceedings of the 6th conference on Symposium on Operating Systems Design & Implementation*, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.
- [dis] distributed.net. distributed.net. <http://www.distributed.net/>.
- [DMTa] DMTF. Desktop and mobile architecture for system hardware (dash) initiative. <http://www.dmtf.org/standards/mgmt/dash/>.
- [DMTb] DMTF. Desktop management interface (dmi). <http://www.dmtf.org/standards/dmi/>.
- [DMTc] DMTF. Distributed management task force. <http://www.dmtf.org/home>.
- [DMTd] DMTF. Web-based enterprise management (wbem). <http://www.dmtf.org/standards/wbem/>.
- [DMTe] DMTF. Web services for management (ws-management). <http://www.dmtf.org/standards/wsman>.
- [ea03] S. Tuecke et al. Open grid services infrastructure. Technical Report GFD.15, Global Grid Forum, 2003.
- [ea05] J. Gomes et al. Experience with the international testbed in the crossgrid project. In P.M.A. Sloot et al, editor, *Advances in Grid Computing — EGC 2005*, volume LNCS3470, Amsterdam, The Netherlands, Feb 2005. Springer.
- [egea] Enabling grids for e-science (egee). <http://public.eu-egee.org/>.

- [EGEb] Enabling Grids for E-science (EGEE). <http://www.eu-egee.org/>.
- [EGEc] EGEE. glite information system documentation. <https://twiki.cern.ch/twiki/bin/view/EGEE/InformationSystem/>.
- [EGEd] EGEE. Grid operations centre database. <http://www.ukiroc.eu/content/view/115/235/>.
- [EGI] EGI. European grid initiative. <http://web.eu-egi.eu/>.
- [ELF] ELFms. Extremely large fabric management system. <http://elfms.web.cern.ch/elfms/>.
- [FAF] FAFNER. Factoring via network-enabled recursion. <http://www.npac.syr.edu/factoring.html>.
- [FCD⁺04] S. Fisher, A. Cooke, A. Djaoui, S. Hicks, R. Middleton, J. Walk, A. Wilson, N. Podhorszki, B. Coghlan, S. Kenny, O. Lyttleton, D. O’Callaghan, A. Gray, J. Leake, W. Nutt, J. Magowan, P. Taylor, R. Cordenonsi, R. Byrom, L. Cornwall, and M. Craig. Information and monitoring services within a grid environment. In *Proc. CHEP’04*, Interlaken, Switzerland, March 2004.
- [FGK⁺] G. Fox, D. Gannon, S. Ko, S. Lee, S. Pallickara, M. Pierce, X. Qiu, X. Rao, A. Uyar, M. Wang, and W. Wu. Peer-to-peer grids.
- [FGN⁺96] I. Foster, J. Geisler, B. Nickless, W. Smith, and S. Tuecke. Software infrastructure for the i-way high-performance distributed computing experiment. In *HPDC ’96: Proceedings of the 5th IEEE International Symposium on High Performance Distributed Computing*, page 562, Washington, DC, USA, 1996. IEEE Computer Society.
- [Fit01] Steven Fitzgerald. Grid information services for distributed resource sharing. In *HPDC ’01: Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing*, page 181, Washington, DC, USA, 2001. IEEE Computer Society.
- [FJK04] I. Foster, N. Jennings, and C. Kesselman. Brain meets brawn: Why grid and agents need each other, 2004.
- [FK97] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *The International Journal of Supercomputer Applications and High Performance Computing*, 11(2):115–128, Summer 1997.

- [FK03a] Ian Foster and Carl Kesselman. *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.
- [FK03b] Ian Foster and Carl Kesselman. *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.
- [FKL⁺99] J. Foster, C. Kasselman, C. Lee, B. Lindell, K. Nahrsted, and A. Roy. A Distributed Resource Management Architecture that supports Advanced Reservations and Co-allocation. In *Seventh International Workshop on Quality of Service*, 1999.
- [FKNT02] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An open grid services architecture for distributed systems integration, 2002.
- [FKT01a] Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *Int. J. High Perform. Comput. Appl.*, 15(3):200–222, 2001.
- [FKT01b] Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, 2150, 2001.
- [Fora] Open Grid Forum. Network management working group. <http://nmwg.internet2.edu/>.
- [Forb] TM Forum. Tm forum. <http://www.tmforum.org/>.
- [Ful05] James A. Fulton. Common information model. In Laura C. Rivero, Jorge Horacio Doorn, and Viviana E. Ferragaine, editors, *Encyclopedia of Database Technologies and Applications*, pages 78–86. Idea Group, 2005.
- [Gan] Gagnlia project documentation. <http://ganglia.sourceforge.net/docs/>.
- [GC⁺05] Jorge Gomes, Brian Coghlan, et al. *Experience with the International Testbed in the CrossGrid Project*, chapter CrossGrid. LNCS. 2005.
- [GEJ⁺06] P. Gardfj??ll, E. Elmroth, L. Johnsson, O. Mulmo, and T. Sandholm. Scalable Grid-wide Capacity Allocation with the SweGrid Accounting System (sgas). 2006. Draft version. Final version to be submitted for journal publication.

- [GGF] GGF. An analysis of top n event descriptions. www-didc.lbl.gov/damed/documents/TopN_Events_final_draft.pdf.
- [GGL03] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. *SIGOPS Oper. Syst. Rev.*, 37(5):29–43, 2003.
- [GGU] GGUS. Global grid user support. <https://gus.fzk.de/>.
- [GIM] GIMPS. Great internet mersenne prime search. <http://www.mersenne.org/>.
- [gLi] gLite. Lightweight middleware for grid computing. <http://glite.web.cern.ch/>.
- [Glo] Globus. Web services resource framework. <http://www.globus.org/wsrf/>.
- [GOC] Grid operations centre database. <http://goc.grid-support.ac.uk/gridsite/gocdb/>.
- [Gol97] Charles F. Goldfarb. Sgml: The reason why and the first published hint. *Journal of the American Society for Information Science*, 48(7):656–661, 1997.
- [gria] Grid4c - Command and Control for Grids. <http://sourceforge.net/projects/grid4c>.
- [Grib] Open Science Grid. A national, distributed computing grid for data-intensive research. <http://www.opensciencegrid.org/>.
- [Gric] World Community Grid. World community grid. <http://www.worldcommunitygrid.org/>.
- [Grid] GridPP. Grid for particle physics. <http://www.gridpp.ac.uk/>.
- [Grie] GridPP. Real-time monitor. <http://gridportal.hep.ph.ic.ac.uk/rtm/>.
- [GST] GSTAT. Global grid information monitoring system. <http://grid-it.cnaf.infn.it/>.
- [GWT97] Andrew S. Grimshaw, Wm. A. Wulf, and CORPORATE The Legion Team. The legion vision of a worldwide virtual computer. *Commun. ACM*, 40(1):39–45, 1997.
- [HET] HET. The high performance computing in europe taskforce. <http://www.hpcineuropetaskforce.eu/>.

- [Hyp] Hypergraph project documentation. <http://hypergraph.sourceforge.net/docs.html>.
- [i4ca] I4c - Wide Area Network Monitoring. <http://sourceforge.net/projects/i4c>.
- [I4Cb] I4C. Wide area network monitoring. <http://i4c.sourceforge.net/>.
- [IBMa] IBM. High performance on demand solutions. http://download.boulder.ibm.com/ibmdl/pub/software/dw/wes/hipods/HiPODS_Brochure.pdf.
- [IBMb] IBM. Ready-to-use cloud computing. <http://www-03.ibm.com/press/us/en/pressrelease/22613.wss>.
- [IETa] IETF. Internet engineering steering group. <http://www.ietf.org/iesg.html/>.
- [IETb] IETF. Internet engineering task force. www.ietf.org/.
- [IETc] IETF. Simple network management protocol. <http://www.ietf.org/rfc/rfc1157.txt>.
- [INF] INFN. The distributed grid accounting system. <http://www.to.infn.it/grid/accounting/main.html>.
- [inta] Interactive european grid project. <http://www.interactive-grid.eu/>.
- [Intb] Int.eu.grid. Interactive european grid project. <http://www.interactive-grid.eu/>.
- [ipm] Intelligent platform management interface. <http://www.intel.com/design/servers/ipmi/>.
- [ISO] ISO. International organisation for standardisation. <http://www.iso.org/iso/home.htm/>.
- [ITU] ITU. International telecommunication union. <http://www.itu.int/>.
- [JGN99] William E. Johnston, Dennis Gannon, and Bill Nitzberg. Grids as production computing environments: The engineering aspects of nasa's information power grid. In *HPDC '99: Proceedings of the 8th IEEE International Symposium on High Performance Distributed Computing*, page 34, Washington, DC, USA, 1999. IEEE Computer Society.

- [KB⁺05] Stuart Kenny, Rob Byrom, et al. *SANTA-G: an instrument monitoring framework using the Relational Grid Monitoring Architecture (R-GMA)*, chapter CrossGrid. LNCS. 2005.
- [KC04] Stuart Kenny and Brian A. Coghlan. Grid-wide intrusion detection system. In Marian Bubak, Michal Turala, and Kazimierz Wiatr, editors, *Proc. Cracow Grid Workshop (CGW04)*, pages 331–337, Cracow, Poland, December 2004. Academic Computer Centre CYFRONET AGH.
- [KC05] S. Kenny and B.A. Coghlan. Towards a grid-wide intrusion detection system. In Peter M.A. Sloot, Alfons G. Hoekstra, Thierry Priol, Alexander Reinefeld, and Marian Bubak, editors, *Advances in Grid Computing - EGC 2005*, LNCS3470, Amsterdam, The Netherlands, February February, 2005. Springer.
- [KCT⁺05] E. Kenny, B. Coghlan, G. Tsouloupas, M. Dikaiakos, J. Walsh, S. Childs, D. O’Callaghan, and G. Quigley. Heterogeneous grid computing: Issues and early benchmarks. In *Proc. ICCS 2005*, volume Part III of *LNCS3516*, pages 870–874, Atlanta, USA, May 2005.
- [KCW⁺04] Eamonn Kenny, Brian A. Coghlan, John Walsh, Stephen Childs, David O’Callaghan, and Geoff Quigley. Testing heterogeneous computing nodes for grid computing. In Marian Bubak, Michal Turala, and Kazimierz Wiatr, editors, *Proc. Cracow Grid Workshop (CGW04)*, Cracow, Poland, December December, 2004. Academic Computer Centre CYFRONET AGH.
- [KCW⁺05a] E. Kenny, B. Coghlan, J. Walsh, S. Childs, D. O’Callaghan, and G. Quigley. Autobuilding multiple ports of computing nodes for grid computing. In *Cracow Grid Workshop (CGW’05)*, Cracow, Poland, November 2005.
- [KCW⁺05b] E. Kenny, B.A. Coghlan, J. Walsh, S. Childs, D. O’Callaghan, and G. Quigley. Heterogeneity of computing nodes for grid computing. In Peter M.A. Sloot, Alfons G. Hoekstra, Thierry Priol, Alexander Reinefeld, and Marian Bubak, editors, *Advances in Grid Computing - EGC 2005*, LNCS3470, pages 401–413, Amsterdam, The Netherlands, February 2005. Springer.
- [KF98] Carl Kesselman and Ian Foster. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, November 1998.
- [Kos07] Kimmo Koski. Building an hpc ecosystem in europe. In *ISPDC ’07: Proceedings of the Sixth International Symposium on Parallel and Distributed Computing*, page 3, Washington, DC, USA, 2007. IEEE Computer Society.

- [Laf] Domenico Laforenza. The grid: International efforts in grid computing. <http://citeseer.ist.psu.edu/498462.html>.
- [lcga] Lhc computing project lcg. <http://lcg.web.cern.ch/LCG/>.
- [LCGb] LCG. Grid service monitoring working group. <https://twiki.cern.ch/twiki/bin/view/LCG/GridServiceMonitoringInfo/>.
- [LCGc] LCG. System analysis working group. <https://twiki.cern.ch/twiki/bin/view/LCG/SystemAnalysisMonitoringInfo/>.
- [LCGd] LCG. System management working group. <https://twiki.cern.ch/twiki/bin/view/LCG/SystemManagement/>.
- [Les] Leslp. Gold - silver - bronze command structure. http://www.leslp.gov.uk/docs/Major_incident_procedure_manual_7th_ed.pdf.
- [LLM88] Michael Litzkow, Miron Livny, and Matthew Mutka. Condor - a hunter of idle workstations. In *Proceedings of the 8th International Conference of Distributed Computing Systems*, June 1988.
- [LLM⁺04] Rafael A. Garca Leiva, Maite Barroso Lopez, G. Cancio Meli, B. Chardi Marco, Lionel Cons, Piotr Poznanski, Andrew Washbrook, Enrico Ferro, and Alexander Holt. Quattor: Tools and techniques for the configuration, installation and management of large-scale grid computing fabrics. *J. Grid Comput.*, 2(4):313–322, 2004.
- [LST97] Laszewski, Warren Smith, and Steven Tuecke. A directory service for configuring high-performance distributed computations. In *HPDC '97: Proceedings of the 6th IEEE International Symposium on High Performance Distributed Computing*, page 365, Washington, DC, USA, 1997. IEEE Computer Society.
- [Man] Mantis. Mantis bug tracker. <http://www.mantisbt.org/>.
- [MC05] M. Manzke and B.A. Coghlan. Optimal performance state estimation of compute systems. In *Proc.IEEE MASCOTS 2005*, Atlanta, USA, September 2005.
- [MCKP06a] S. Maad, B. Coghlan, E. Kenny, and G. Pierantoni. Grid enabling e-government through a universal accessibility grid layer. In *Proc. IADIS International Conference*, July 2006.

- [MCKP06b] S. Maad, B. Coghlan, E. Kenny, and G. Pierantoni. Universal accessibility to the grid via metagrid infrastructure. In *Proc. EGEE User Forum*, March 2006.
- [MCP⁺05] S. Maad, B. Coghlan, G. Pierantoni, E. Kenny, J. Ryan, and R. Watson. Adapting the development model of the grid anatomy to meet the needs of various application domains. In *Cracow Grid Workshop (CGW'05)*, Cracow, Poland, November 2005.
- [MCQ⁺06] S. Maad, B. Coghlan, G. Quigley, J. Ryan, E. Kenny, and G. Pierantoni. A grid filesystem: A key component for managing the data centre in the enterprise. In *accepted paper for the Special Issue on Grid Computing for Enterprise, Scalable Computing: Practice and Experience Journal.*, 2006.
- [MCR⁺05] S. Maad, B. Coghlan, J. Ryan, E. Kenny, R. Watson, and G. Pierantoni. The horizon of the grid for e-government. In *Proceedings of the eGovernment Workshop*, Brunel University, United Kingdom, September 2005.
- [MCS⁺05] J.P. Morrison, B. Coghlan, A. Shearer, S. Foley, D. Power, and R. Perrot. WebCom-G: A candidate middleware for Grid-Ireland. *High Performance Computing Applications and Parallel Processing*, 2005.
- [Mica] Microsoft. Microsoft sql data services. <http://www.microsoft.com/sql/dataservices/default.aspx>.
- [Micb] SUN Microsystems. Sun utility computing. <http://www.sun.com/service/sungrid/index.jsp>.
- [MMFM⁺05] Olivier Martin, Jean-Philippe Martin-Flatin, Edoardo Martelli, Paolo Moroni, Harvey Newman, Sylvain Ravot, and Dan Nae. The datatag transatlantic testbed. *Future Generation Comp. Syst.*, 21(4):443–456, 2005.
- [MSA⁺05] S. Matsuoka, S. Shinjo, M. Aoyagi, S. Sekiguchi, H. Usami, and K. Miura. Japanese computational grid research project: NAREGI. *Proceedings of the IEEE*, 93(3):522–533, March 2005.
- [Mul96] Nathan J. Muller. The openview enterprise management framework. *Int. Journal of Network Management*, 6(5):271–283, 1996.
- [MW03] R. Menday and Ph. Wieder. Grip: The evolution of uncore towards a service oriented grid. In *Proceedings of the Cracow Grid Workshop 2003*, 2003.
- [Nag] Nagios project documentation. <http://www.nagios.org/docs/>.

- [NGS] NGS. Uk - national grid service. <http://www.ngs.ac.uk>.
- [NLG⁺03] Harvey B. Newman, I. C. Legrand, Philippe Galvez, R. Voicu, and C. Cirstoiu. Monalisa : A distributed monitoring service architecture. *CoRR*, cs.DC/0306096, 2003. informal publication.
- [OASa] Defining a web services architecture to manage distributed resources. www.oasis-open.org/committees/wsdm/.
- [OASb] OASIS. Organization for the advancement of structured information standards. <http://www.w3.org/>.
- [OASc] OASIS. Wsdm management of web services. <http://www.oasis-open.org/committees/download.php/17001/wsdm-1.0-mows-primer-cd-01.doc>.
- [OASd] OASIS. Wsdm management using web services. <http://docs.oasis-open.org/wsdm/2004/12/muws/cd-wsdm-muws-part1-1.0.pdf>.
- [OC04] David O’Callaghan and Brian Coghlan. Bridging secure WebCom and European DataGrid security for multiple VOs over multiple grids. In *Proc. ISPDC 2004*, pages 225–231, Cork, Ireland, July 2004. IEEE Computer Society.
- [OGF] OGF. Open grid forum. <http://www.ogf.org/>.
- [Owe07] Kenon Owens. Virtualization/vmware. In *LISA. USENIX*, 2007.
- [Pic] Francesc Prez Picazo. Ingrid: A generic autonomous expert system for grid nodes real time monitor and control. http://www.tid.es/html_eng/articulos_conferencias.html.
- [PKC05] G. Pierantoni, E. Kenny, and B. Coghlan. An Architecture Based on a Social-Economic Approach for Flexible Grid Resource Allocation. In *Cracow Grid Workshop (CGW05)*, Cracow, Poland, November 2005.
- [PKC⁺06a] G. Pierantoni, E. Kenny, B. Coghlan, O. Lyttleton, D. O’Callaghan, and G. Quigley. Interoperability using a Metagrid Architecture. In *HDPC 06*, Paris, France, June 2006.
- [PKC06b] Gabriele Pierantoni, Eamonn Kenny, and Brian Coghlan. An agent-based architecture for grid societies. In *PARA 06*, Umea, Sweden, June 2006.

- [PKC⁺06c] Gabriele Pierantoni, Eamonn Kenny, Brian Coghlan, Oliver Lyttleton, David O’Callaghan, and Geoff Quigley. Interoperability using a metagrid architecture. In *HDPC 06*, Paris, France, June 2006.
- [PLO⁺05a] G. Pierantoni, O. Lyttleton, D. O’Callaghan, G. Quigley, E. Kenny, and B. Coghlan. Multi-Grid and Multi-VO Job Submission based on a Unified Computational Model. In *Cracow Grid Workshop (CGW05)*, Kracow, Poland, November 2005.
- [PLO⁺05b] G. Pierantoni, O. Lyttleton, D. O’Callaghan, G. Quigley, E. Kenny, and B. Coghlan. Multi-grid and multi-vo job submission based on a unified computational model. In *Cracow Grid Workshop (CGW’05)*, Cracow, Poland, November 2005.
- [PPJM03] David A. Power, Adarsh Patil, Sunil John, and John P. Morrison. Webcomg. In Hamid R. Arabnia and Youngsong Mun, editors, *PDPTA*, pages 50–56. CSREA Press, 2003.
- [PRAa] PRACE. Partnership for advanced computing in europe. <http://www.prace-project.eu/>.
- [Prab] Best Practical. Rt: Request tracker. <http://bestpractical.com/rt/>.
- [RC04] J.P. Ryan and B.A. Coghlan. Smg: Shared memory for grids. In *Proc.PDCS’04*, pages 439–151, Boston, USA, November 2004.
- [RC05] J.P. Ryan and B.A. Coghlan. Distributed shared memory in a grid environment. In *Proc.PARCO 2005*, Malaga, September 2005.
- [RCW06a] Keith Rochford, Brian Coghlan, and John Walsh. An agent-based approach to grid service monitoring. In *ISPDC ’06: Proceedings of the Proceedings of The Fifth International Symposium on Parallel and Distributed Computing*, pages 345–351, Washington, DC, USA, 2006. IEEE Computer Society.
- [RCW06b] Keith Rochford, Brian A. Coghlan, and John Walsh. An agent-based approach to grid service monitoring. In *Proc. International Symposium on Parallel and Distributed Computing (ISPDC 2006)*, July July, 2006.
- [Rob86] Larry Roberts. The arpanet and computer networks. In *Proceedings of the ACM Conference on The history of personal workstations*, pages 51–58, New York, NY, USA, 1986. ACM.

- [Rom02] Mathilde Romberg. The unicore grid infrastructure. *Sci. Program.*, 10(2):149–157, 2002.
- [sam] Service availability monitoring. http://goc.grid.sinica.edu.tw/gocwiki/Service_Availability_Monitoring.
- [Sar98] Luis F. G. Sarmenta. Bayanihan: Web-based volunteer computing using java. In *WWCA*, pages 444–461, 1998.
- [SB94] CORPORATE Computer Science and Telecommunications Board. *Realizing the information future: the Internet and beyond*. National Academy Press, Washington, DC, USA, 1994.
- [SC92] Larry Smarr and Charles E. Catlett. Metacomputing. *Commun. ACM*, 35(6):44–52, 1992.
- [Sch02] J. M. Schopf. A General Architecture for Scheduling on the Grid. 2002.
- [sft] Site functionality tests. http://goc.grid.sinica.edu.tw/gocwiki/Site_Functional_Tests.
- [SGE⁺04] Thomas Sandholm, Peter Gardfjäll, Erik Elmroth, Lennart Johnsson, and Olle Mulmo. An ogsa-based accounting system for allocation enforcement across hpc centers. In *ICSOC '04: Proceedings of the 2nd international conference on Service oriented computing*, pages 279–288, New York, NY, USA, 2004. ACM.
- [SMI] SMILE. Service management interface. <http://egee-docs.web.cern.ch/egee-docs/list.php?dir=./smile/>.
- [Smi02] Warren Smith. A system for monitoring and management of computational grids. In *ICPP '02: Proceedings of the 2002 International Conference on Parallel Processing (ICPP'02)*, page 55, Washington, DC, USA, 2002. IEEE Computer Society.
- [Syb] Sybase. Avaki eii. <http://www.sybase.com/products/allproductsa-z/avakieii>.
- [TAG⁺02] B. TIERNEY, R. AYDT, D. GUNTER, W. SMITH, V. TAYLOR, R. WOLSKI, and M. SWANY. A grid monitoring architecture, 2002.
- [TCL⁺07] Li Tang, Yin Chen, Fei Li, Hui Zhang, and Jun Li. Empirical study on the evolution of planetlab. In *ICN*, page 64. IEEE Computer Society, 2007.

- [Tea] U.S. National Response Team. Incident command system. <http://www.nrt.org/>.
- [Ter] TeraGrid. Teragrid open scientific discovery infrastructure. <http://www.teragrid.org/>.
- [TPH⁺06] M. Theimer, S. Parastatidis, T. Hey, M. Humphrey, and G. Fox. An evolutionary approach to realizing the grid vision, 2006.
- [TSF06] Hong-Linh Truong, Robert Samborski, and Thomas Fahringer. Towards a framework for monitoring and analyzing qos metrics of grid services. In *E-SCIENCE '06: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing*, page 65, Washington, DC, USA, 2006. IEEE Computer Society.
- [UN] UK-NGS. National grid service - minimal software stack. www.grid-support.ac.uk/files/ngs2_roadmap_2007.pdf.
- [W3C] W3C. The world wide web consortium. <http://www.w3c.org/>.
- [WBPB03] R. Wolski, J. Brevik, J. S. Plank, and T. Bryan. Grid Resource Allocation and Control Using Computational Economies. 2003.
- [WMC08] Ronan Watson, Soha Maad, and Brian Coghlan. Virtual grid: Adaptive visualization of grids. In *Proceedings of the Cracow Grid Workshop 2007*, 2008.
- [ZS05a] Serafeim Zanikolas and Rizos Sakellariou. A taxonomy of grid monitoring systems. *Future Gener. Comput. Syst.*, 21(1):163–188, 2005.
- [ZS05b] Serafeim Zanikolas and Rizos Sakellariou. A taxonomy of grid monitoring systems. *Future Gener. Comput. Syst.*, 21(1):163–188, 2005.

Appendix A

The WSDM standard capabilities

The WSDM standard species a number of standard capabilities for a managed resource. This section provides a brief overview of some of the capabilities defined by WSDM MUWS 1.0

Identity The purpose of the identity capability is to distinguish between resources or determine if two capabilities manage the same resource. This is a required capability and must be provided by every manageability endpoint. The capability has exactly one property, ResourceID, which is unique to the resource.

Manageability Characteristics The manageability Characteristics capability defines properties providing information about the endpoint implementation rather than the managed resource. The capability contains one property, ManageabilityCapability, which contains a list of URIs or the manageability capabilities of the endpoint.

Correlatable Properties The Correlatable Properties capability defines the set of properties that may be used to determine if two endpoints are managing the same resource but for some reason are required to use different identities.

Description The Description capability exposes the Caption, Description, and Version properties of a manageable resource

Metrics The Metrics capability defines how collected data about the resource should be accessed and represented. Metadata and time information is included

State The State capability provides information about the current state of the resource along with details of the last state transition of the resource. The development of custom state models specific to the resource is supported.

Operational Status The Operational Status capability provides a general indication of the health/availability of the resource and enumerates to Available, PartiallyAvailable, Unavailable, or Unknown.

Configuration The Configuration Capability exposes modifiable properties of the resource that can be set in order to influence the operation / behaviour of the resource.

Advertisement The Advertisement capability defines notifications that are published upon the creation or destruction of a manageable resource.

Relationships The Relationship capability is used to expose information about the relationships in which the resource participates.

Appendix B

The WSDM Event Format

The WSDM Event Format is an extensible XML format that defines a set of basic, consistent, data elements that allow different types of management event information to be carried in a consistent manner. The standard description enables programmatic processing, correlation, and interpretation of events from different products, and management technologies.

Management event data is organised into three categories: the event reporter, the event source, and the actual situation data to be conveyed by the event. A set of standard properties is included in each category and these may be extended if necessary. WSDM also defines a standard set of priorities, severities, and situation categories, such as StartSituation, StopSituation, and CreateSituation. These facilitate common understanding of events received from a variety of resources.

WSDM supports notifications using WS-Notifications and WSDM Event Formatted messages. WS-Notification provides the publish-subscription services for Web services architectures. Various filtering methods can be employed to allow managers to subscribe based on categories of events for a resource, such as all metric changes or configuration changes rather than subscribing to each independent property change event.

B.1 WEF event components

The components of a WEF event are defined as:

EventID distinguishing identifier

SourceComponent the component from which the event originates

ReportedComponent the component reporting the event

ReportTime the time at which the event was generated

Situation container for the specifics of an event

SituationCategory (REQUIRED) categorize event into one of twelve categories

SituationTime (RECOMMENDED) the time an event actually occurred

SuccessDisposition a qualification of the SituationCategory

Priority the relative importance of an event.

Severity the relative effect this event has on the operational status of a resource.

Values are based upon the Percieved Severity as defined in the DMTF CIM Alert Indication.

Message Human readable message providing the details of the event

SubstitutableMsg contains data for which the formatting controls are catalogued elsewhere.

MsgId provides reference to a message in a catalogue

MsgIdType identifies the type of catalogue to use

Value the array of values for the message

B.2 WEF SituationCategory types

The defined SituationCategory types include:

AvailabilitySituation Situations regarding the operational state and availability of a component

CapabilitySituation This category is specified when a change in capability of a resource occurs. e.g. a printer running out of paper of a certain size etc.

ConfigurationSituation Identification of configuration changes. Any change that a component makes to its configuration should be logged using this category.

StopSituation Shutdown process for a component. e.g. "Stopped", "Stopping Service", "Stopped", "Exiting", "completed",

StartSituation Startup process for a component. e.g. "starting..", "..Started", "Initialising..", "..Initialised"

RequestSituation Identifies the completion status of a request. Typically these requests form complex management tasks or transactions. e.g. "configuration synchronisation started" or "backup procedure completed"

DestroySituation An entity or component was removed. e.g. "file deleted", "Connection Closed"

CreateSituation Occurs when a component creates an entity. e.g. document, object, file etc

DependencySituation A component cannot find a component or feature that it requires. e.g cant find version no., plugin not found, subsystem not available.

ConnectSituation Situations relating to a connection attempt from one component to another. e.g.failed, created, ended.

ReportSituation Situation that occur as a result of some setting or occurrence that causes the resource to asynchronously report various types of data. Information types that fall into this category include

Exception related an exception that is not covered by any other category

Performance related an event that does not fall under any other category and has some effect on the performance of the component.

Security related some security issue has been detected. e.g. the cabinet door to a secure rack has been opened or an attack of some sort.

Heartbeat related the resource has been configured to periodically report a heartbeat.

Status related a change in status that does not affect availability or capability. e.g. ink cartridge low

Log related log entry based on some event or interval

Debug related the resource has been enabled to turn on diagnostic information flow and will report information within this category

Trace related the resource has been enabled to turn on trace information flow and will report information within this category

OtherSituation Events that do not fall into any other category. This category is defined for syntactic completeness but any events placed in this category will not be able to be effectively correlated and its use is therefore discouraged unless absolutely necessary.

The MUSE API includes code to facilitate the generation and publishing of events in the WEF format. The code in Listing B.2 shows an example of how a WEF management event is constructed using the JavaTM API.

```

import java.net.URI;
2 import javax.xml.namespace.QName;
import org.apache.muse.ws.addressing.EndpointReference;
4 import org.apache.muse.ws.dm.muws.events.Component;
import org.apache.muse.ws.dm.muws.events.ComponentAddress;
6 import org.apache.muse.ws.dm.muws.events.ManagementEvent;
import org.apache.muse.ws.dm.muws.events.Situation;
8 import org.apache.muse.ws.dm.muws.events.WefConstants;
import org.apache.muse.ws.dm.muws.events.WefFactory;
10 import org.apache.muse.ws.dm.muws.events.impl.SimpleSubstitutableMessage;
import org.apache.muse.ws.dm.muws.events.impl.SimpleWefFactory;
12 import org.w3c.dom.Element;

```

Listing B.1: Libraries required for MUSE generation

```

ManagementEvent event = factory.createEvent();
2 //
  /// EVENT SOURCE
4 //
  //create the source component wrapper
6 Component source = factory.createComponent();
source.setName(sourceName);
8 source.setResourceID(resourceID);
  //create the EPR that will go in the wrapper
10 URI sourceURI = URI.create(uri);
EndpointReference sourceEPR = new EndpointReference(sourceURI);
12 Element eprXML = sourceEPR.toXML();
  //add the EPR to the source wrapper
14 ComponentAddress sourceAddress = factory.createComponentAddress();
sourceAddress.addExtendedElement(eprXML);
16 source.setAddress(sourceAddress);
  //add the source info to the event
18 event.setSource(source);

20
  //
22 // SITUATION
  //
24
  Situation situation = factory.createSituation();
26 situation.setCategoryType(category);
situation.setPriority(priority);
28 situation.setSeverity(severity);
situation.setSuccessDisposition(success);
30 situation.setMessage(message);

32 //
  // SUBSTITUTABLE MESSAGE
34 //

36 if (values != null && !subMessageID.equals("") && !subMessageIDtype.equals("")){

```



```

SimpleSubstitutableMessage subMessage = new SimpleSubstitutableMessage ();
38 subMessage.setMessageId ( subMessageID );
subMessage.setMessageIdType ( subMessageIDtype );
40 subMessage.setValues ( values );
situation.setSubstitutableMessage ( subMessage );
42 }

44 event.setSituation ( situation );

46

48 //
// EXTENDED ELEMENT
//
50 if ( extendedElement != null )
event.addExtendedElement ( elementName , extendedElement );
52 return event ;

```

Listing B.2: Use of the MUSE API for event generation

An example of a generated management event is shown in Listing B.3:

```

<muws1:ManagementEvent
2   xmlns:muws1=" http://docs.oasis-open.org/wsdm/muws1-2.xsd" ReportTime="2007-09-27
   T12:21:40+01:00">
   <muws1:EventId>uuid:9aa2897b-3698-57ac-ca38-40ab1e531d92</muws1:EventId>
4   <muws1:SourceComponent>
   <muws1:ComponentAddress>
6     <wsa:EndpointReference xmlns:wsa=" http://www.w3.org/2005/08/addressing">
   <wsa:Address>http://localhost/my-resources/services/event-publisher</
   wsa:Address>
8     </wsa:EndpointReference>
   </muws1:ComponentAddress>
10    <muws1:ResourceId>1234-56-7890</muws1:ResourceId>
   </muws1:SourceComponent>
12   <muws2:Situation xmlns:muws2=" http://docs.oasis-open.org/wsdm/muws2-2.xsd">
   <muws2:SituationCategory>
14     <muws2:StartSituation />
   </muws2:SituationCategory>
16   <muws2:SuccessDisposition>Successful</muws2:SuccessDisposition>
   <muws2:SituationTime>2007-09-27 T12:21:40+01:00</muws2:SituationTime>
18   <muws2:Priority>10</muws2:Priority>
   <muws2:Severity>1</muws2:Severity>
20   <muws2:Message>The application 'server1' started successfully.</muws2:Message>
   <muws2:SubstitutableMsg MsgId=" 123456" MsgIdType=" NagiosPluginOutput">
22     <muws2:Value>val1</muws2:Value>
   <muws2:Value>val2</muws2:Value>
24     <muws2:Value>val3</muws2:Value>
   </muws2:SubstitutableMsg>
26   </muws2:Situation>
   <testElementOutput
28   xmlns=" http://www.grid.ie/Grid4C/wsdm/fe-outputs.xsd"
   IP=" 134.226.53.250" host=" proteus.cs.tcd.ie" type=" TestElement">

```

```
30     <command>This is the command text value</command>
    </testElementOutput>
32 </muws1:ManagementEvent>
```

Listing B.3: The generated WEF event

Appendix C

WS-Notification Traces

This appendix provides the complete WS-Notification messages from which the excerpts in Chapter 8 are taken.

```
[CLIENT TRACE] SOAP envelope contents (outgoing):
2
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
4   <soap:Header>
      <wsa:To xmlns:wsa="http://www.w3.org/2005/08/addressing">http://134.226.53.234
          :8080/Grid4C_CE_Impl-1.2/services/GI_CE_ManageableEndpoint</wsa:To>
6      <wsa:Action xmlns:wsa="http://www.w3.org/2005/08/addressing">http://docs.oasis
          -open.org/wsn/bw-2/NotificationProducer/SubscribeRequest</wsa:Action>
      <wsa:MessageID xmlns:wsa="http://www.w3.org/2005/08/addressing">uuid:ecdbbea8
          -18ad-8c41-2532-b25187abc0e3</wsa:MessageID>
8      <wsa:From xmlns:wsa="http://www.w3.org/2005/08/addressing">
          <wsa:Address>http://www.w3.org/2005/08/addressing/role/anonymous</
              wsa:Address>
10     </wsa:From>
      </soap:Header>
12     <soap:Body>
          <wsnt:Subscribe xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2">
14             <wsnt:ConsumerReference>
                  <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
                      //134.226.53.234:8080/g4c-consumer/services/consumer</wsa:Address>
16             </wsnt:ConsumerReference>
          </wsnt:Subscribe>
18     </soap:Body>
</soap:Envelope>
20
[CLIENT TRACE] SOAP envelope contents (incoming):
22
<soapenv:Envelope
24   xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://www.w3.
      org/2005/08/addressing">
      <soapenv:Header>
26     <wsa:To>http://www.w3.org/2005/08/addressing/anonymous</wsa:To>
```

```

28     <wsa:ReplyTo>
        <wsa:Address>http://www.w3.org/2005/08/addressing/anonymous</wsa:Address>
    </wsa:ReplyTo>
30     <wsa:Action>urn:handleRequestResponse</wsa:Action>
        <wsa:RelatesTo wsa:RelationshipType="http://www.w3.org/2005/08/addressing/
            reply">uuid:ecdbbea8-18ad-8c41-2532-b25187abc0e3</wsa:RelatesTo>
32 </soapenv:Header>
    <soapenv:Body>
34     <wsnt:SubscribeResponse xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2">
        <wsnt:SubscriptionReference>
36         <wsa:Address>http://134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/
            SubscriptionManager</wsa:Address>
        <wsa:ReferenceParameters>
38         <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/
            addressing">MuseResource-1</muse-wsa:ResourceId>
        </wsa:ReferenceParameters>
40     </wsnt:SubscriptionReference>
        <wsnt:CurrentTime>2008-03-12T19:37:44+00:00</wsnt:CurrentTime>
42     </wsnt:SubscribeResponse>
    </soapenv:Body>
44 </soapenv:Envelope>

```

Listing C.1: WS-N subscription trace

```

1 <wsnt:NotificationMessage
    xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
3    xmlns:muse-wsa="http://ws.apache.org/muse/addressing"
    xmlns:tns="http://www.grid.ie/Grid4C/ServiceManagement/GI-GFTP_Capability"
5    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" xmlns:wsrfrp="http://docs.oasis-
        open.org/wsrfrp-2">
7    <wsnt:SubscriptionReference>
        <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
            //134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/SubscriptionManager</
            wsa:Address>
9        <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing">
            <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing"
                >MuseResource-1</muse-wsa:ResourceId>
11        </wsa:ReferenceParameters>
    </wsnt:SubscriptionReference>
13    <wsnt:Topic
        Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete"
        xmlns:pfx11="http://www.grid.ie/Grid4C/ServiceManagement/
            GI-GFTP_Capability">pfx11:gftpStatus</wsnt:Topic>
15    <wsnt:ProducerReference>
        <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing"/>
17        <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
            //134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/GI_CE_ManageableEndpoint
            </wsa:Address>
        </wsnt:ProducerReference>
19    </wsnt:Message>

```

```

21     <wsrf-rp:ResourcePropertyValueChangeNotification xmlns:wsrf-rp=" http://docs .
        oasis-open.org/wsrf/rp-2">
22         <wsrf-rp:OldValues>
23             <tns:gftpStatus xmlns:tns=" http://www.grid.ie/Grid4C/ServiceManagement
                /GI_GFTP_Capability">0</tns:gftpStatus>
24         </wsrf-rp:OldValues>
25         <wsrf-rp:NewValues>
26             <tns:gftpStatus xmlns:tns=" http://www.grid.ie/Grid4C/ServiceManagement
                /GI_GFTP_Capability">2</tns:gftpStatus>
27         </wsrf-rp:NewValues>
28     </wsrf-rp:ResourcePropertyValueChangeNotification>
29 </wsnt:Message>
30 </wsnt:NotificationMessage>
31 <wsnt:NotificationMessage
32     xmlns:wsnt=" http://docs.oasis-open.org/wsn/b-2"
33     xmlns:muse-wsa=" http://ws.apache.org/muse/addressing"
34     xmlns:tns=" http://www.grid.ie/Grid4C/ServiceManagement/GI_GFTP_Capability"
35     xmlns:wsa=" http://www.w3.org/2005/08/addressing"
36     xmlns:wsnt=" http://docs.oasis-open.org/wsn/b-2" xmlns:wsrf-rp=" http://docs.oasis-
        open.org/wsrf/rp-2">
37     <wsnt:SubscriptionReference>
38         <wsa:Address xmlns:wsa=" http://www.w3.org/2005/08/addressing">http:
            //134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/SubscriptionManager</
            wsa:Address>
39         <wsa:ReferenceParameters xmlns:wsa=" http://www.w3.org/2005/08/addressing">
            <muse-wsa:ResourceId xmlns:muse-wsa=" http://ws.apache.org/muse/addressing"
                >MuseResource-1</muse-wsa:ResourceId>
40         </wsa:ReferenceParameters>
41     </wsnt:SubscriptionReference>
42     <wsnt:Topic
43         Dialect=" http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete"
44         xmlns:pfx11=" http://www.grid.ie/Grid4C/ServiceManagement/
            GI_GFTP_Capability">pfx11:gftpStatusInformation</wsnt:Topic>
45     <wsnt:ProducerReference>
46         <wsa:ReferenceParameters xmlns:wsa=" http://www.w3.org/2005/08/addressing"/>
47         <wsa:Address xmlns:wsa=" http://www.w3.org/2005/08/addressing">http:
            //134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/GI_CE_ManageableEndpoint
            </wsa:Address>
48     </wsnt:ProducerReference>
49     <wsnt:Message>
50         <wsrf-rp:ResourcePropertyValueChangeNotification xmlns:wsrf-rp=" http://docs .
            oasis-open.org/wsrf/rp-2">
51             <wsrf-rp:OldValues>
52                 <tns:gftpStatusInformation xmlns:tns=" http://www.grid.ie/Grid4C/
                    ServiceManagement/GI_GFTP_Capability">TCP OK - 0.000 second
                    response time on port 80|time=0.000186s;;;0.000000;10.000000</
                    tns:gftpStatusInformation>
53             </wsrf-rp:OldValues>
54             <wsrf-rp:NewValues>
55                 <tns:gftpStatusInformation xmlns:tns=" http://www.grid.ie/Grid4C/
                    ServiceManagement/GI_GFTP_Capability">Connection refused</

```

```

    tns:gftpStatusInformation>
55     </wsrf-rp:NewValues>
        </wsrf-rp:ResourcePropertyValueChangeNotification>
57     </wsnt:Message>
</wsnt:NotificationMessage>

```

Listing C.2: Incoming WSRP notification trace showing service outage

```

<wsnt:NotificationMessage
2   xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
   xmlns:muse-wsa="http://ws.apache.org/muse/addressing"
4   xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
   xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
6   xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:wsnt="http://docs.oasis-
   open.org/wsn/b-2">
   <wsnt:SubscriptionReference>
8     <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
       //134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/SubscriptionManager</
       wsa:Address>
     <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing">
10    <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing"
       >MuseResource-1</muse-wsa:ResourceId>
     </wsa:ReferenceParameters>
12  </wsnt:SubscriptionReference>
   <wsnt:Topic
14    Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete"
     xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd">
     muws2:StartSituation</wsnt:Topic>
   <wsnt:ProducerReference>
16    <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing"/>
     <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
       //134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/GI_CE_ManageableEndpoint
       </wsa:Address>
18  </wsnt:ProducerReference>
   <wsnt:Message>
20    <muws1:ManagementEvent ReportTime="2008-03-12T19:39:54+00:00" xmlns:muws1="
       http://docs.oasis-open.org/wsdm/muws1-2.xsd">
     <muws1:EventId>uuid:e15377dc-e330-5458-4d9d-84d4542288b4</muws1:EventId>
22    <muws1:SourceComponent>
     <muws1:ComponentAddress>
24      <muws1:ComponentAddress>
     <wsa:EndpointReference xmlns:wsa="http://www.w3.org/2005/08/
       addressing">
26      <wsa:Address>http://localhost/my-resources/services/event-
        publisher</wsa:Address>
     </wsa:EndpointReference>
28      </muws1:ComponentAddress>
     </muws1:ComponentAddress>
30    </muws1:SourceComponent>
     <muws2:Situation xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
     >

```

```

32         <muws2:SituationCategory>
33             <muws2:StartSituation />
34         </muws2:SituationCategory>
35         <muws2:SuccessDisposition>Successful</muws2:SuccessDisposition>
36         <muws2:SituationTime>2008-03-12T19:39:54+00:00</muws2:SituationTime>
37         <muws2:Priority>10</muws2:Priority>
38         <muws2:Severity>1</muws2:Severity>
39         <muws2:Message>Service 'gftp' on 'testServer1.grid4c.testgrid' has
40             started Successfully...</muws2:Message>
41     </muws2:Situation>
42 </muws1:ManagementEvent>
43 </wsnt:Message>
44 </wsnt:NotificationMessage>
45 <wsnt:NotificationMessage
46     xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
47     xmlns:muse-wsa="http://ws.apache.org/muse/addressing"
48     xmlns:tns="http://www.grid.ie/Grid4C/ServiceManagement/GI_GFTP_Capability"
49     xmlns:wsa="http://www.w3.org/2005/08/addressing"
50     xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" xmlns:wsrf-rp="http://docs.oasis-
51         open.org/wsrf/rp-2">
52     <wsnt:SubscriptionReference>
53         <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
54             //134.226.53.234:8080/Grid4C-CE-Impl-1.2/services/SubscriptionManager</
55             wsa:Address>
56         <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing">
57             <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing"
58                 >MuseResource-1</muse-wsa:ResourceId>
59         </wsa:ReferenceParameters>
60     </wsnt:SubscriptionReference>
61     <wsnt:Topic
62         Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete"
63         xmlns:pfx11="http://www.grid.ie/Grid4C/ServiceManagement/
64             GI_GFTP_Capability">pfx11:gftpStatus</wsnt:Topic>
65     <wsnt:ProducerReference>
66         <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing" />
67         <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
68             //134.226.53.234:8080/Grid4C-CE-Impl-1.2/services/GI-CE-ManageableEndpoint
69             </wsa:Address>
70     </wsnt:ProducerReference>
71     <wsnt:Message>
72         <wsrf-rp:ResourcePropertyValueChangeNotification xmlns:wsrf-rp="http://docs.
73             oasis-open.org/wsrf/rp-2">
74             <wsrf-rp:OldValues>
75                 <tns:gftpStatus xmlns:tns="http://www.grid.ie/Grid4C/ServiceManagement
76                     /GI_GFTP_Capability">2</tns:gftpStatus>
77             </wsrf-rp:OldValues>
78             <wsrf-rp:NewValues>
79                 <tns:gftpStatus xmlns:tns="http://www.grid.ie/Grid4C/ServiceManagement
80                     /GI_GFTP_Capability">0</tns:gftpStatus>
81             </wsrf-rp:NewValues>

```

```

72     </wsrf-rp:ResourcePropertyValueChangeNotification>
73     </wsnt:Message>
74 </wsnt:NotificationMessage>
75 <wsnt:NotificationMessage
76     xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
77     xmlns:muse-wsa="http://ws.apache.org/muse/addressing"
78     xmlns:tns="http://www.grid.ie/Grid4C/ServiceManagement/GI_GFTP_Capability"
79     xmlns:wsa="http://www.w3.org/2005/08/addressing"
80     xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2" xmlns:wsrf-rp="http://docs.oasis-
81     open.org/wsrf/rp-2">
82     <wsnt:SubscriptionReference>
83     <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
84         //134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/SubscriptionManager</
85         wsa:Address>
86     <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing">
87     <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing"
88         >MuseResource-1</muse-wsa:ResourceId>
89     </wsa:ReferenceParameters>
90 </wsnt:SubscriptionReference>
91 <wsnt:Topic
92     Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete"
93     xmlns:pfx11="http://www.grid.ie/Grid4C/ServiceManagement/
94     GI_GFTP_Capability">pfx11:gftpStatusInformation</wsnt:Topic>
95 <wsnt:ProducerReference>
96     <wsa:ReferenceParameters xmlns:wsa="http://www.w3.org/2005/08/addressing" />
97     <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">http:
98         //134.226.53.234:8080/Grid4C_CE_Impl-1.2/services/GI_CE_ManageableEndpoint
99         </wsa:Address>
100 </wsnt:ProducerReference>
101 <wsnt:Message>
102     <wsrf-rp:ResourcePropertyValueChangeNotification xmlns:wsrf-rp="http://docs.
103     oasis-open.org/wsrf/rp-2">
104     <wsrf-rp:OldValues>
105     <tns:gftpStatusInformation xmlns:tns="http://www.grid.ie/Grid4C/
106     ServiceManagement/GI_GFTP_Capability">Connection refused</
107     tns:gftpStatusInformation>
108     </wsrf-rp:OldValues>
109     <wsrf-rp:NewValues>
110     <tns:gftpStatusInformation xmlns:tns="http://www.grid.ie/Grid4C/
111     ServiceManagement/GI_GFTP_Capability">TCP OK - 0.000 second
112     response time on port 80|time=0.000192s;;;0.000000;10.000000</
113     tns:gftpStatusInformation>
114     </wsrf-rp:NewValues>
115     </wsrf-rp:ResourcePropertyValueChangeNotification>
116 </wsnt:Message>
117 </wsnt:NotificationMessage>

```

Listing C.3: Notification trace showing service recovery management event followed by WSRP property change events showing restored service availability

Glossary

ADT	Average Down Time (also known as Mean Down Time (MDT)) - The average lengths of the periods for which a system is not operational, 159
AFR	Annualized failure rate (AFR) - The relation between the mean time between failure (MTBF) and the assumed hours that a device is run per year, expressed in percent, 160
ARPANET	Advanced Research Projects Agency Network - Developed by ARPA of the United States Department of Defense, was the world's first operational packet switching network, and the predecessor of the global Internet., 5
ASI	Active Security Infrastructure - A research activity from the Int.eu.grid project intended to go beyond existing prevention-based grid security measures by focusing on threat detection and reaction, 150
AXIS	An open-source SOAP web service engine from the Apache software foundation, 110
BDII	Berkeley Database Information Index - An OpenLDAP based implementation of an Information Index, 37
Blue Cloud	IBM's Cloud computing initiative, 20

BOINC	Berkeley Open Infrastructure for Network Computing - A non-commercial middleware system for volunteer computing., 17
C.O.D.E.	Control and Observation in Distributed Environments - A framework for monitoring and managing organisation-wide Globus-based grid installations, 49
CIM	Common Information Model - An object oriented standard for the description of management system information models, 32
CITL	Centre for Information Technology Leadership - A research organisation with the goal of helping healthcare providers make informed IT implementation decisions, 62
CrossGrid	A European grid computing project initiated in 2002 with the objective of developing a grid infrastructure to support a new category of applications from fields such as medicine, environmental science and physics, 12
DASH	Desktop and mobile Architecture for System Hardware - Web services based remote management of desktop and mobile computers using a collection of standards including WBEM, CIM, 60
DEISA	Distributed European Infrastructure for Supercomputing Applications - A consortium of leading national European supercomputing centres that has formed a combined infrastructure to provide a persistent, production quality, distributed supercomputing environment with continental scope., 14

DMI	Desktop Management Interface - A component of the System Management BIOS that provides a system for managing and tracking components in a desktop PC, 60
DMTF	Distributed Management Task Force - An organisation promoting the development of management standards for IT and network environments, 60
EC2	Amazon Elastic Compute Cloud - A component of the Amazon web services suite designed to allow scalable deployment of applications by allowing users to manage the creation and termination of server instances on demand., 21
EDG	European DataGrid - Started in January 2001, with the goal of constructing a test infrastructure to provide shared data and computing resources to the European scientific community., 11
EGEE	Enabling Grids for E-science - The largest multi-disciplinary grid infrastructure in the world designed to provide a reliable and scalable computing resource available to the European and global research community., 12
EGI	A European body created with the aims of protecting and maximising European investment in e-infrastructures by encouraging collaboration and interoperability between national Grid infrastructures., 13
ELFms	Extremely Large Fabric management system - A modular interoperating framework for the management of large heterogeneous environments, 76

ESRFI	European Strategy Forum for Research Infrastructures - A policy body for the development of research infrastructure in Europe., 25
ETICS	eInfrastructure for Testing, Integration and Configuration of Software - A software release process management tool developed at CERN, 122
Fafner	Factoring via Network-Enabled Recursion - A distributed computing project developed in 1995 with the intention of using the Web to overcome some of the problems associated with factoring large numbers., 7
Ganglia	A scalable distributed monitoring system based of hierarchical design., 34
GGUS	Global Grid User Support system - A centralised user support and ticketing system for the Grid, 78
gLite	A next generation middleware for grid computing developed by partners in the EGEE Project, 16
Global Grid Forum	Global Grid Forum (GGF) - Advisory body established in 2000 to address issues relating to grid computing, and to define and promote standards and best practices, 61
Globus	A software toolkit that allows sharing of distributed, heterogeneous resources, 9
GLUE	Grid Laboratory Uniform Environment - A standard schema for the modeling and representation of grid resources. Designed to allow interoperability between American and European Grid resources, 29

GMA	Grid Monitoring Architecture - An architecture developed by the Global Grid Forum to provide a minimal specification for monitoring solutions, 41
GOCDB	LCG Grid Operations Centre Database - A central repository of general information about participating EGEE/LCG/UK-NGS sites, 32
GRAM	Grid Resource Allocation Manager - A component of the Globus toolkit for job submission and management, 109
Grid-Ireland	The Irish National Grid Infrastructure, 16
Grid4C	A standards-based architecture for the monitoring and management of distributed computational resources developed at Trinity College Dublin, 119
GridICE	A grid monitoring tool designed to allow monitoring of resource utilisation through a Web front end, 45
GridPP	The UK particle physics grid project, 15
GRIS	Grid Resource Information Service - A component of the Globus MDS information system, 109
GSTAT	An LCG2 Grid Information System monitoring application developed by the Grid Operations Centre at Taipei, 49
HET	High-Performance Computing in Europe Task force - A policy body for the development of High Performance Computing in Europe, 23
HostCheck	A grid monitoring and diagnostic tool developed at the Laboratory of Instrumentation and Experimental Particles Physics in Portugal, 51

HyperGraph	An open source project which provides java code to work with hyperbolic geometry and trees, 111
I-WAY	Information Wide Area Year - An experimental high performance network conceived in 1995 to link high performance computers and advanced visualisation environments., 7
I4C	An Agent-based wide-area service and resource monitoring solution developed at Trinity College Dublin, 54
IETF	The Internet Engineering Task Force - An open standards organisation of volunteer participants, with the objective of developing and promoting internet standards, 60
InGrid	A generic autonomous expert system for the monitoring and control of grid resources, 77
Int.eu.grid	A continuation of the CrossGrid project initiated in 2006 with the objective of providing an advanced Grid-empowered production infrastructure in the European Research Area., 13
IPG	Information Power Grid - A high performance computing and data grid constructed in 1998 primarily for use by NASA scientists and engineers, 11
ISO	International Organization for Standardization - A non-government international organisation founded in 1947 to develop and promote industrial and commercial standards, 63

ITU	International Telecommunication Union - The leading United Nations agency for information and communication technologies. Founded in 1865 to standardise and regulate international radio and telecommunications, 63
JIMS	JMX-based Infrastructure Monitoring System - A Java-based monitoring framework developed by the Int.eu.grid project, 51
LCG	LHC Computing Grid - An international computing grid designed to meet the computational and data-handling requirements of the experiments conducted at CERN's Large Hadron Collider, 12
LDAP	Lightweight Directory Access Protocol - An application protocol for querying and modifying directory services, 30
LEAF	LHC Era Automated Fabric - A management toolset developed within collaboration between the GridPP and LCG projects, 76
Legion	An object-based meta-system providing a software infrastructure for the interaction of distributed heterogeneous computing resources., 9
Lemon	Lhc Era MONitoring - A distributed monitoring system developed at CERN as part of the LHC project, 35
MapCentre	A Grid monitoring and visualisation tool developed at CNRS, 105
MDS	Monitoring and Discovery Service - The information infrastructure provided by the Globus toolkit, 36

MonALISA	Monitoring Agents using a Large Integrated Services Architecture - An extensible general-purpose framework for monitoring hosts and networks in large-scale distributed systems, 47
MOWS	Management Of Web Services - Part of the WSDM specification from OASIS, 123
MTBF	Mean Time Between Failures - An expression for the availability of a system calculated as the average duration of periods for which system is available, 159
MTTR	Mean Time TO Repair - The average time it takes to repair a failed component, 160
MUWS	Management Using Web Services - Part of the WSDM specification from OASIS, 123
Nagios	An open source host and service monitoring application, 35
NAREGI	The Japanese National Research Grid Initiative, 14
NGS	The UK National Grid Service with the goal of providing UK researchers with electronic access to the computational and data-based facilities, 16
NSCA	Nagios Service Check Acceptor - An add-on for the Nagios monitoring system that allows passive service checks to be injected into the Nagios process., 111
OASIS	The Organisation for the Advancement of Structured Information Standards (OASIS) was founded in 1993 to develop and promote industry standards, 62

OCM-G	A monitoring tool that provides online on-line information about running parallel and distributed application for development-support and performance analysis, 54
OGSA	Open Grid Services Architecture - A service-oriented architecture for distributed computing environments., 10
OGSI	Open Grid Services Infrastructure - An infrastructure layer for the Open Grid Services Architecture (OGSA), OGSI extends Web services to accommodate grid computing resources that are both transient and stateful., 10
Open Science Grid	Built by a consortium of U.S. universities and laboratories, the Open Science Grid is a US national production-quality infrastructure for large-scale science, 13
PRACE	Partnership for Advanced Computing in Europe - An evolution of HET, 25
PyGMA	An implementation of the Grid Monitoring Architecture (GMA) written in the Python language, 43
QUATTOR	A system administration toolkit enabling the automated installation, configuration, and management of computing systems running Linux and Solaris, 76
R-GMA	An implementation of the OGF Grid Monitoring Architecture (GMA) employing a relational model, 37
RB-STATS	Resource Broker Statistics - A management reporting tool developed to provide reports of infrastructure usage, 55

RT	Request Tracker - A comprehensive and robust ticketing system which allows individuals and groups to efficiently manage tasks, support issues, and requests, 78
SAM	Service Availability Monitoring - A monitoring framework also developed at CERN, 51
SFT	Site Functional Test - A tool developed at CERN to test the operations of LCG sites from a users perspective, 49
SGA	Social Grid Agents - A prototype system developed at Trinity College Dublin that enables economic and social transactions on Grid infrastructure, 144
SNMP	Simple Network Management Protocol - A set of standards that enable the monitoring and configuration of network-attached devices, 60
Sun Grid	An on-demand computing service offered by Sun Microsystems., 21
TeraGrid	A federation of US supercomputing sites which aims to provide an open scientific discovery infrastructure., 14
TestGrid	A realistic large-scale testbed constructed at Trinity College Dublin, to accurately emulate the configuration of the production Grid-Ireland infrastructure, 138
TM Forum	TeleManagement Forum - An industry association of web, telecom and media service providers, 63
UNICORE	UNiform Interface to COmputer REsources - A technology intended to provides seamless, secure, and intuitive access to distributed computational resources, 10

W3C	World Wide Web Consortium - Aims to promote common and interoperable protocols and is the primary standards organisation for the Web, 62
WBEM	Web-Based Enterprise Management - A set of systems management technologies developed to unify the management of distributed computing environment, 60
WCG	World Community Grid - An effort to create the world's largest public computing grid to perform processing for scientific research projects that benefit humanity., 18
WS-BPEL	Web Services Business Process Execution Language - An XML-based workflow definition language from OASIS with broad industry support, 171
WSDL	Web Services Description Language - An XML-based language that provides a model for describing Web services, 129
WSDM	Web Services Distributed Management - A standard from the Organisation for the Advancement of Structured Information Standards (OASIS) that defines mechanisms for the representation of, and access to, resource management interfaces implemented as Web Services, 60
WSRF	Web Service Resource Framework - A family of OASIS-published specifications for web services encompassing topics such as statefulness and addressing., 10
XML-RPC	A remote procedure call protocol which uses XML to encode its calls and HTTP as a transport mechanism, 128