A Model for Touchpoint Simulation of Grid Services

by

Scott A. Brousseau

B.Sc., Royal Roads Military College, 1986

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Supervisory Committee

A Model for Touchpoint Simulation of Grid Services

by

Scott A. Brousseau

B.Sc., Royal Roads Military College, 1986

Supervisory Committee

Dr. Hausi A. Müller, Supervisor (Department of Computer Science)

Dr. Sudhakar Ganti, Departmental Member (Department of Computer Science)

Dr. Jens H. Weber-Jahnke, Departmental Member (Department of Computer Science)

Abstract

Supervisory Committee

Dr. Hausi A. Müller, Supervisor (Department of Computer Science)

Dr. Sudhakar Ganti, Departmental Member (Department of Computer Science)

Dr. Jens H. Weber-Jahnke, Departmental Member (Department of Computer Science)

Advances in technologies have made an unprecedented range and variety of computing resources available. A number of fields have sought to take maximum advantage of these resources, with grid computing being one of the more successful. However, the increasing complexity of these heterogeneous, distributed systems has compromised users' ability to manage them effectively. Autonomic computing, which seeks to hide the complexity of systems by making them self-managing, offers a potential solution. In order to produce autonomic managers for grid systems, realistic input is required for development and testing. This thesis proposes a model that can be used to provide simulated input, utilizing existing system logs. The simulator adheres to the standards and specifications recognized in both autonomic and grid services, and provides the detailed, accurate information that is required by developers.

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank my supervisor, Dr. Hausi Müller, for the opportunity to learn and work with the Rigi Group, and for his guidance throughout my studies at UVic.

I would like to express my gratitude to the members of the Rigi group for their assistance, companionship and friendship. In particular, I would like to thank Sweta Goyal, Qin Zhu, Sangeeta Neti, Jing Zhou, Priyanka Agrawal, Ron Desmarais, Dylan Dawson, Toni Lin, Feng Zou and Lei Lin.

I would like to thank my family and friends for their understanding and support.

Finally, I would like to thank my wife, Lisa, for her inestimable patience and steadfast support throughout this long process.

Dedication

To my wife and my family

Chapter 1 Introduction

Over the years, there has been a steady increase in both the capability and availability of computing resources. From the limited access to mainframe computers by a select few, the rise of networks and the Internet – along with the exponential increase in the power of individual machines – has given users access to an unprecedented range and variety of resources. Grid computing has sought to harness these resources and allow them to be used effectively.

1.1 Motivation – Grid Computing

One of the original definitions suggested that a grid is a system which: "…coordinates resources that are not subject to centralized control… using standard, open, general-purpose protocols and interfaces… to deliver nontrivial quality of service" [Fo02]. However, with a wide variety of systems and applications labeled as "grid computing", it remains difficult to define precisely; other definitions range from the virtualization and control of resources within an organization, to "computing power on tap", where processing power is accessed as simply as plugging in a toaster [BDG04, Ha04, Econ01]. One of the early, and perhaps best known, examples of the use of grid computing is the SETI@Home project, which harnesses the power of a multitude of PCs to search for "rational" life from outer space [ACKL02].

There is, however, agreement on the characteristics and benefits of grid computing [St07, FKNT02a, FBAK03]. The characteristics include:

- Heterogeneity – a variety of hardware and software components with differing characteristics;
- Aggregation – of individual resources into a higher-capacity virtual resource;
- Collaboration – resource sharing in a distributed fashion; and

- Virtualization – the use of interfaces to hide the complexity of the underlying resources.

The advantages and benefits of grid computing include the exploitation of under-utilised resources, the leveraging of parallel computing capacity, access to varied resources, increased reliability, improved load balancing, and better management of systems.

There are many projects worldwide that rely on grids – no less than forty are listed on one educational grid computing Website managed by CERN [Gr09]. Their uses span from very specialized scientific grids [NEES09], to national and academic research [Du09], as well as to international multi-disciplinary computing [EGEE09]. There are also many types of middleware used in these grid systems. Some are intended for very specific applications, such as dCache for data storage and retrieval [dC09], or for specific users, such as the US Department of Energy's ESnet [ES09].

This wide variety of middleware and the lack of standards in the field have hampered the adoption of grid computing by an even wider audience [St07, Fo02, FZRL08]. In order to address this, the Global Grid Forum and the Globus Alliance developed the Open Grid Services Architecture (OGSA), defining a "Grid service" which would use an open Web services interface to provision services from the grid [FKNT02b, GGF05]. In the Globus Toolkit, Version 4 (GT4), the OASIS-approved Web Services Distributed Management (WSDM) specification is used to define standard manageability interfaces that are able to manage resources using Web services technologies [OAS06a].

The Globus Toolkit is a modular product with software designed for security, data management, execution management, and information services. It is in use in a variety of existing grids, including the Open Science Grid (OSG - http://www.opensciencegrid.org/) the Enabling Grids for E-sciencE project (EGEE - http://www.eu-egee.org/), as well as the University of Victoria's GridX1 (http://www.gridx1.ca/).

There is a wide range of management and control functions available to Globus developers and users, which strive for optimum use of a grid's resources [Gl09a]. However, given the complexity and dynamism of the applications, and the need to manage the configuration, parameters, and behaviours of multiple systems, researchers have suggested the use of autonomic computing to enable self-management of grid systems [JPR09, DM07, MPWT07, LPH04].

1.1.1 Autonomic Computing

The aim of autonomic computing is to hide the complexity of modern computer systems from users. The name is borrowed from the medical field; the autonomic nervous system frees our conscious brain from the burden of having to deal with vital lower-level functions. Autonomic computing will free users and system administrators from many of today's routine management and operational tasks [KeCh03].

The management tasks to be undertaken by autonomic systems can be broken down into four aspects, also known as the self-* properties:

a) Self-configuring: Automated configuration of components and systems in accordance with high-level policies. The system will adjust to changes automatically and seamlessly;

b) Self-optimizing: Components and systems continually seek opportunities to improve their own performance and efficiency;

c) Self-healing: The system automatically detects, diagnoses, and repairs localized software and hardware problems; and

d) Self-protecting: The system automatically defends against malicious attacks or cascading failures. It uses early warning to anticipate and prevent system-wide failures.

In order to carry out these tasks, the system must be reflexive and have detailed knowledge of its components, current status, capabilities, limits, and inter-dependencies with other systems and available resources. This is accomplished in an autonomic

element through a control loop focused on a managed resource (cf. Figure 1). The control loop is often described as a monitor-analyze-plan-execute and knowledge (MAPE-K) loop. At the input side, sensors provide information to the monitoring component. This information is analyzed based on criteria drawn from the knowledge store, and a plan is formulated based on specified policies and on information gleaned from past activities. This plan is then executed through actuators or effectors interfacing with the managed resource.

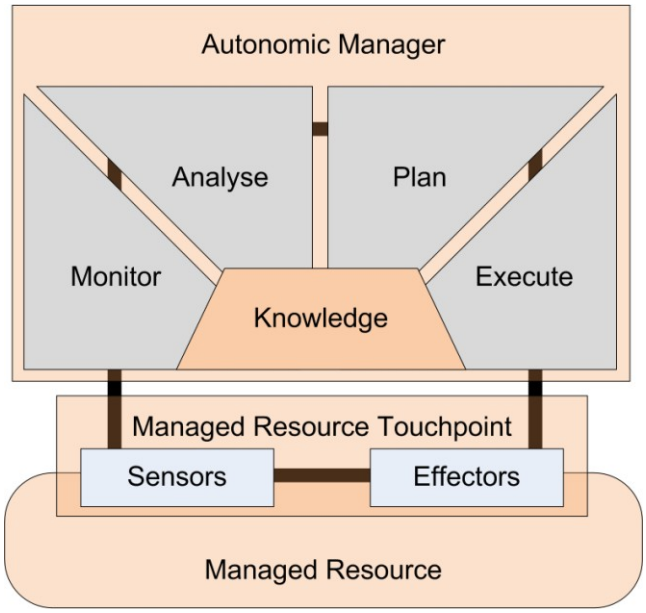

Figure 1 - The Autonomic Element

The interaction between the managed resource and autonomic manager is regulated through a manageability interface. This interface exposes the state and management operations of the resource in a consistent manner. The implementation of a manageability interface for a specific managed resource, or set of resources, is known as a manageability endpoint, or touchpoint. A key aspect of the autonomic computing architecture is the use of standard manageability interfaces, with Web Services Distributed Management (WSDM) being noted [IBM06a]. This convergence of

standards makes the use of autonomic computing with Globus grid services especially inviting.

1.1.2 The Problem

Grid computing, by its nature, can involve the control of a vast span and variety of resources. In order to properly develop and tune an autonomic controller for grid systems, realistic input is required. Options include utilising actual grid resources, using virtual resources to provide input, or providing some other input which simulates the responses of the real grid.

Using existing grid resources is seen as problematic in terms of access; grid resources are often in high demand, and the heterogeneity of grid resources only allows a small portion of the grid infrastructure to be available for a particular application [RHML07, KANK09, Na09]. Developers therefore would not necessarily have guaranteed access to the resources at the required time, or for the length of time required, in the development cycle.

1.1.2.1 Virtualization

Virtualization is a rapidly growing field, with products ranging from simple desktop servers to complete solutions that encompass whole-enterprise application and data centers. Several major companies offer no-cost applications, easily accessible to researchers and developers, including VMware, Inc (VMware Server - http://www.vmware.com/), Microsoft (MS Virtual Server - http://www.microsoft.com/windowsserversystem/virtualserver/), and Sun (VirtualBox - http://www.virtualbox.org/), which provide roughly similar capabilities. VMware Server was used by the author in the development of this thesis and will be used as the example for the category.

VMware Server provides an Infrastructure Web Access portal for management. A virtual machine (VM) is created by selecting the desired guest operating system,

memory capacity, number of processors, hard drive capacity, networking properties, and optical and USB drives. The operating system is then installed and configured to use the selected resources. Applications are installed and configured. Once a baseline configuration is established, the VM can be cloned by copying the files to a new location and having a new unique ID (and MAC address) assigned to it. In those cases where a static IP address was assigned to the baseline, it must be changed in all locations in the cloned copies.

1.1.2.2 Grid Virtualization

The feasibility of utilizing actual grid resources for the development of the autonomic manager has been discounted, but the introduction of virtualization in grids opens the possibility of exploiting this field. Resource virtualization is the dynamic and rapid configuration of virtual machines resident in grid resources to meet the needs of client applications. Jobs would no longer be constrained by the infrastructure; it could be sent to any resource, and the necessary VM would be swapped in to allow the job to be run. Research prototypes have demonstrated the feasibility of the approach, but work remains ongoing to allow its use on a wide scale [GN07, KCF06, KANK09]. The implementations of these virtualization techniques often do not adhere to existing grid standards, but research work has successfully demonstrated the ability to transport the VM configuration details along with the job and data to a virtual resource, have it configured properly and run the job [RHML07].

1.1.2.3 Cloud computing

Cloud computing has become the latest paradigm in IT. There are three main scenarios in which it is used [VMCL09]. Infrastructure as a Service uses virtualization to provide raw storage and processing capability to a user. Platform as a Service supplies a software platform on which to run systems. Software as a Service provides a Web-based

alternative to locally-run applications. The Platform as a Service is the scenario of most interest; the on-demand availability of a multiplicity of platforms on which to run manageable resources would be attractive. However, the standards on which cloud computing services are based are not consistent, and virtually all cloud services are offered for a fee, with pricing structures still being developed [KG09, FZRL08].

At the 2009 IEEE International Conference on Autonomic Computing, panelists discussed the convergence of cloud computing, grids and autonomic computing [GRR09]. It was noted that cloud computing was typically focused at the application level rather than the infrastructure level, but that autonomic computing techniques should eventually be able to be brought to bear on the Cloud environment, especially in the areas of system behaviour management, Quality of Service management and resource management.

1.1.2.1 Discussion

The use of virtualization to reproduce multiple resources offers many advantages. Because the VM contains an actual invocation of the resource, there is guaranteed fidelity in the reproduction, great control over the output, and familiarity with the implementation. The developer can set up the available virtual resource to provide what is required and be assured that the output reflects what would be received from real resources.

There are, however, two major drawbacks to using virtualization, both relating to scale. There is a significant cost in terms of effort to build and configure each of the virtual machines (VMs). It is possible to clone VMs once the initial baseline has been developed, but the modifications required in terms of specifying discrete or specific IP addresses are by no means trivial. In addition, the effort required to tailor the security environment, including certificates, proxies, and grid map files for each VM are extensive.

The second drawback is that only a limited number of instances can be run in a development or research environment. On a typical desktop computer, running more than a half dozen VMs will overwhelm the available processing power and memory. The point of saturation will depend on the CPU capacity of the computer, the installed memory, as well as the demands of the VM in terms of CPU cycles and memory, but the number of replicated resources available to the developer will be limited.

Grid virtualization offers similar advantages to local virtualization, but would not be constrained in the number of instances that could be run simultaneously. However, the level of effort to configure the grid would remain very high, and the lack of established standards would preclude its current use as a method to replicate resources, although progress is being made.

Cloud computing is a rapidly developing field and offers huge potential to allow seamless expansion of computing resources. However, similar to grid virtualization, the level of effort required to implement individual resources would be very high. There is also the additional constraint that cloud computing is a fee-for-use commodity, so its use would come with an ongoing financial cost.

Consequently, this research focuses on providing simulated responses of a grid. To ensure that valid output is provided from the simulator, log files from a grid implementation are used as input.

1.2 Objective and Approach

The objective of this research is to develop a touchpoint simulator for Globus grid services to facilitate the development of autonomic managers for grid management.

We examine autonomic managers, the workings of the autonomic touchpoint, the industry-based standards to which the touchpoint must adhere, and a framework that can be used to implement those standards. We next examine a method to transform the log output from Globus into a format useable by an automatic manager.

We then review the Globus toolkit, looking at its main components, as well as the Web services architecture and standards used. A small-scale grid is constructed utilising VMware Server 2.0 to generate a set of logs that is used in the model.

A touchpoint simulator, consisting of a simulated manageable resource and autonomic manager, is developed. We verify that it adheres to the applicable standards and is able to utilise the input from Globus logs to provide output to the simulated autonomic manager.

1.3 Outline of Thesis

This thesis is laid out as follows. This chapter discusses the motivation for the research, describing the requirement and options to provide valid input to the developers of autonomic management software, and outlines the approach used to solve the problem. Chapter 2 provides background on the technologies to be used, highlighting autonomic architecture and managers, the requisite standards, the components of grid services, and the Globus Toolkit. Chapter 3 reviews related work that influenced the present model, including the original IBM Autonomic Integrated Development Environment (AIDE), and more recent invocations of WSDM and WSRF-compliant IBM tools (Test and Performance Tools Platform and COSMO). Chapter 4 outlines the general approach including the generation of logs to be used by the simulator, the requirements of touchpoints, and the use of Apache Muse in developing the simulator. Chapter 5 details the implementation of the prototype simulator, which uses Globus logs to generate the information required by autonomic engine developers. Chapter 6 provides an evaluation of the approach to determine if it meets the objectives of the thesis. Chapter 7 provides an overview of the research conducted, highlights contributions and discusses future work.

Chapter 2 Background

2.1 Autonomic Computing Architecture and Managers

The autonomic element portrayed earlier in Figure 1 represents the simplest component in autonomic computing. A more general view of the autonomic computing reference architecture (ACRA) is depicted at Figure 2.

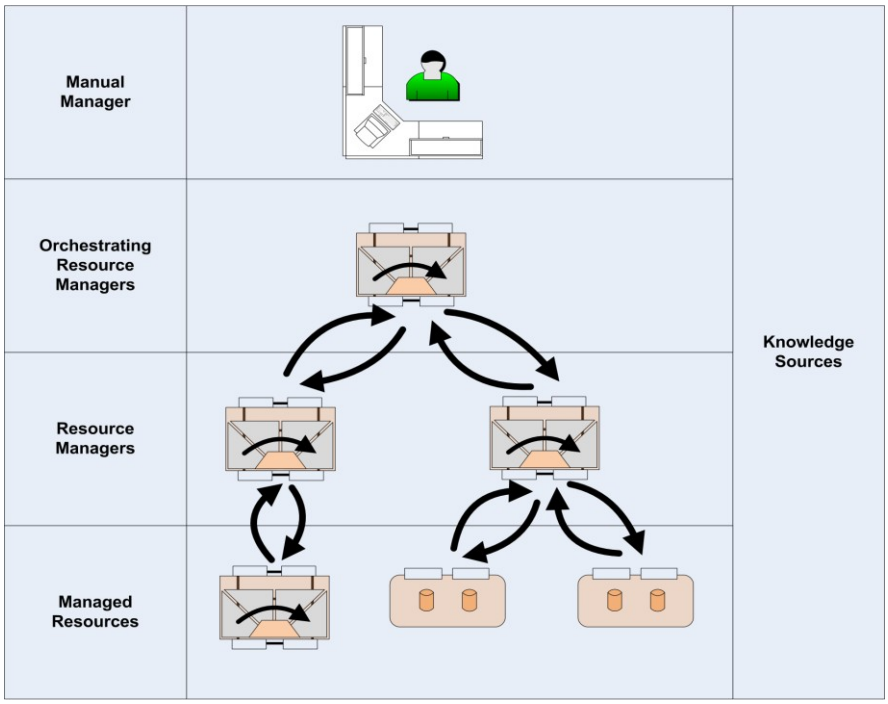

Figure 2 - IBM Autonomic Computing Reference Architecture

The bottom layer consists of the system components, or resources, to be managed. These can be either hardware or software components, may have self-managing attributes and may in fact themselves be autonomic managers. The next layer is nominally composed of autonomic resource managers, which typically will deal with a single self-* discipline. Above that are orchestrating managers which may span all four of the self-* disciplines. The autonomic managers are guided by policies to achieve the objectives of the system. At the peak is a manual manager, who can provide human guidance to the

system as required. All layers of architecture share information through access to common knowledge sources.

This nominal architecture has been the target of considerable research, especially in terms of policy. Kephart and Walsh describe three types of policies that may be used: Action policies, which are relatively simple if-then statements; Goal policies which describe a desired end-state, and Utility Function policies, the most sophisticated (and difficult to implement) which can differentiate between the desirability of various end-states [KW04].

There have been numerous implementations of autonomic managers. The IBM Autonomic Toolkit [IATK06] incorporates the Autonomic Management Engine. Melcher and Mitchell extended its functionality for autonomic network service configuration [MM04]. The Unity project demonstrated the use of a Multi-Agent System as an autonomic controller [TCWD04]. Kaiser and Pareck's implementation [KPGV03] focused on the collection of monitoring data from legacy systems. This underscores a theme common to the other implementations; the requirement to provide full and accurate input to the autonomic manager.

2.2 Web Services

The W3C defines Web services as: "… a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically, the Web Services Description Language - WSDL). Other systems interact with the Web service in a manner prescribed by its description using Simple Object Access Protocol (SOAP) messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards" [W3C04]. At its most basic, a Web-service can be described as a service requestor (client) generating a request or message. That message is passed through the WSDL stub interface, effectively translating it from the local application language to a neutral interface language. The request is then bundled into a SOAP message and

transported to the service provider. The service provider then breaks down the SOAP message, passes the request through the WSDL skeleton, where it is translated into the local application language and acted upon.

2.2.1 WSDL

The role of the Web Services Description Language (WSDL) is similar to that of an Interface Description Language (IDL) in middleware packages such as COBRA. WSDL interface specifications are often described as having an abstract part, similar to traditional IDLs, and a concert part, dealing with transport protocol binding and addressing [GSBD02].

A WSDL interface specification file (a simplified example is at Figure 3) defines the data *types* used in the Web service, which may be inline or referenced. The *messages* are listed, noting which element will be contained therein. *Operations* consist of messages which define an interaction pattern; each operation can contain an input and output message, as well as fault messages. The abstract portion of a WSDL is completed by gathering the *operations* into a *portType*.

The concrete portion of a WSDL interface includes the *binding*, *port*, and *service* constructs. The *binding* specifies the operation type, message encoding and protocol binding. For Web services, operations are normally document-style, the encoding style is literal, and the transport protocol is typically HTTP. The *port* construct, also known as the Endpoint, specifies the URI at which the implementation of the *portType* can be accessed. Finally, the *service*, or logical groupings of port types, is defined [W3C01].

```
<wsdl:definitions targetNamespace="http://ws.apache.org/muse/test/wsrf"
xmlns:tns="http://ws.apache.org/muse/test/wsrf"
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
name="WsResource">
  <wsdl:types>
    <xsd:schema elementFormDefault="qualified" targetNamespace="http://www.w3.org/2005/08/addressing">
      <xsd:include schemaLocation="WS-Addressing-2005_08.xsd" />
    </xsd:schema>
    <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsn/t-1">
      <xsd:include schemaLocation="WS-Topics-1_3.xsd" />
    </xsd:schema>
    <xsd:schema elementFormDefault="qualified" targetNamespace="http://ws.apache.org/muse/test/wsrf">
      <xsd:element name="MessageInterval" type="xsd:integer" />
      <xsd:element name="WsResourceProperties">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element ref="wsrf-rl:CurrentTime" />
            <xsd:element ref="tns:MessageInterval" />
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:schema>
  </wsdl:types>

  <wsdl:message name="Notify">
    <wsdl:part name="Notify" element="wsnt:Notify" />
  </wsdl:message>
  <wsdl:message name="GetMetadataMsg">
    <wsdl:part name="GetMetadataMsg" element="wsx:GetMetadata" />
  </wsdl:message>
  <wsdl:message name="GetMetadataResponseMsg">
    <wsdl:part name="GetMetadataResponseMsg" element="wsx:GetMetadata" />
  </wsdl:message>

  <wsdl:portType name="WsResourcePortType" wsrf-rp:ResourceProperties="tns:WsResourceProperties"
      wsrmd:Descriptor="WsResourceMetadata" wsrmd:DescriptorLocation="WsResource.rmd">
    <wsdl:operation name="Notify">
      <wsdl:input wsa:Action=http://docs.oasis-open.org/wsn/bw-2/NotificationConsumer/NotifyRequest
          name="Notify" message="tns:Notify" />
    </wsdl:operation>
    <wsdl:operation name="GetMetadata">
      <wsdl:input wsa:Action=http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata
          name="GetMetadataMsg" message="tns:GetMetadataMsg" />
      <wsdl:output wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadataResponse"
          name="GetMetadataResponseMsg" message="tns:GetMetadataResponseMsg" />
    </wsdl:operation>
  </wsdl:portType>
```

```
<wsdl:binding name="WsResourceBinding" type="tns:WsResourcePortType">
   <wsdl-soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
     <wsdl:operation name="Notify">
     <wsdl-soap:operation/>
       <wsdl:input>
         <wsdl-soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" use="literal"/>
       </wsdl:input>
     </wsdl:operation>
   <wsdl:operation name="GetMetadata">
     <wsdl-soap:operation soapAction="GetMetadata" />
       <wsdl:input>
         <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
       </wsdl:input>
       <wsdl:output>
         <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
       </wsdl:output>
   </wsdl:operation>
 </wsdl:binding>

 <wsdl:service name="WsResourceService">
   <wsdl:port name="WsResourcePort" binding="tns:WsResourceBinding">
     <wsdl-soap:address location="http://localhost:8080/ac-control01/services/WsResource" />
   </wsdl:port>
 </wsdl:service>

</wsdl:definitions>
```

Figure 3 - WSDL Example

The separation of the abstract and concrete portions of a WSDL specification makes it very easy to reuse. Several models can be aggregated to form more complete Web services. A WSDL operation can invoke another WSDL's operations, allowing the "composition" of complex Web services applications [GSBD02].

2.3 Web Services Distributed Management (WSDM)

The standard that binds autonomic computing and Globus grid services is Web Services Distributed Management (WSDM), which was approved by OASIS in August 2006. WSDM is broken down into two portions; the specification for Management Using Web Services (MUWS) defines how a resource in a network environment can provide a manageability interface to allow it to be controlled using Web services [OAS06a, OAS06b]. Management Of Web Services (MOWS) defines how to manage Web

services as resources and how to describe and access that manageability using MUWS [OAS06c].

WSDM builds upon and utilizes the other approved and emerging standards such as WS-Interoperability Basic Profile, WS-Addressing and WS-Notification. Its design goals are as follows [Kr05]:

- Resource orientation – Direct access to manageable resources by Web services;
- Architecture independent – Implementation is not reflected in the interface or how the manager interacts with it;
- Composability – Can scale by stripping down for small applications or use all features for enterprise environments;
- Design time and run-time inspection – Conducts initial and on-going discovery of environment and resources; and
- Model independent – Does not define what information resources need to provide.

MUWS defines the set of manageability capabilities that an endpoint exposes. A capability consists of properties and operations. Examples include the Identity capability, with the ResourceId property, the Description capability, with three properties: Caption, Description and Version, and the Operational Status capability with a property of the same name.

There has been ongoing research into the use of WSDM in autonomic systems; after its incorporation in an existing prototype autonomic system [MPWT07], researchers noted that the WSDM interface allows for easy expansion, provides support for effective communications and is well suited to interactions between sensors and managers. Another research team, when implementing a framework architecture for scalable adaptive Web services chose WSDM for its standardized interface, well-defined lifecycle management, notification-based messaging, and failure recovery [LMSI08]. Of particular note, it has also been used when composing multiple grid services [HPDA05].

2.4 Web Services Resource Framework (WSRF)

Web services are generally stateless - they do not retain information from one transaction to the next [Vo03]. For simple services, such as providing the temperature when given a city or postal code, this is not a limitation. However, when dealing with more sophisticated applications, especially those that may deal with multi-stage transactions, it is imperative to be able to maintain state [FFGT04].

Embedding state information within a Web service would increase complexity. WSRF mitigates this added complexity by separating the Web service and the state information; the state information is moved into an entity called a resource.

2.4.1 WSRF Specification

The Web Services Resource Framework (WSRF) is a collection of specifications which work together to allow the proper management of WS-Resources [OAS06d].

WS-ResourceProperties: A set of interfaces that allow us to access, modify and query resource properties.

WS-ResourceLifetime: Provides the basic mechanisms required to create, destroy and manage the lifecycle of resources.

WS-ServiceGroup: Specifies how to group services or WS-Resources together, to allow each group to be discovered and accessed via a single point of entry (the service group).

WS-BaseFaults: Provides a common specification for representing faults that may occur when Web services are invoked.

2.4.2 Resource Properties

Data items in a resource are called the resource properties and include the user-defined properties in a Web service; ResourceType, Name and GridProperty are

examples. Resource properties also include metadata about the resource; the TerminationTime and ResourceId fall into this category. Resource property information is initialized using a resource metadata descriptor (.rmd) document, which includes the initial, valid and static values for a property [OAS06d].

2.4.3 WS-Addressing / Endpoint References (EPR)

Typical Web services are accessed using a URI such as: http://192.168.1.212:8080/SManRes01/services/GridResource01

In order to access a particular resource for a Web service, additional information is needed. The WS-Addressing specification provides the framework for this information, combining a resource key (ResourceId) with the URI to create a WS-Resource-qualified endpoint reference.

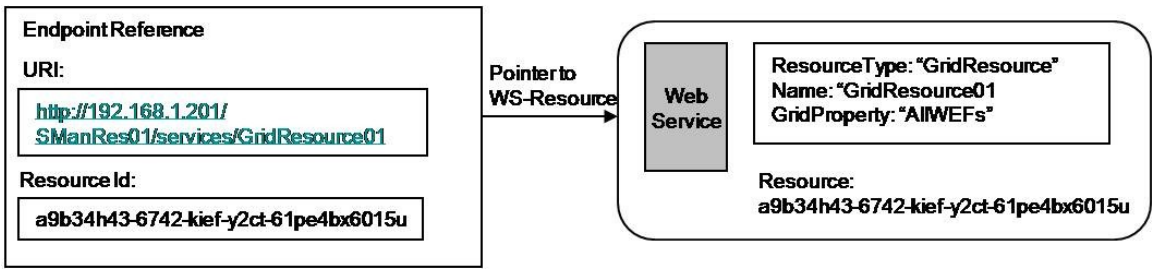

Figure 4 - Relationship of EPR to Web service and Resource

2.4.4 WS-BaseNotification

This is a related specification (http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf) as opposed to a part of WSRF. It allows the designation of a Notification Producer, a Web service that will respond to a Subscribe request from a Notification Consumer and send events of interest in Notification messages. A Subscription Manager implements the Notification message exchanges and allows querying and manipulation of the subscription. The Notifications that are exchanged may be specified through the use of WS-Topics (http://docs.oasis-open.org/wsn/wsn-

ws_topics-1.3-spec-os.pdf). WS-Topics allows the filtering and selection of Notifications, and groups them into Topic Sets, collections that supported by the Producer.

2.5 WSDM Event Format (WEF)

Tying together WSDM and WSRF is the WSDM Event Format. The MUWS specification defines a ManagementEvent type, which is used to describe all events emitted by a WSDM-compliant resource. The message consists of an EventId, SourceComponent and ReporterComponent (used only if different from the Source). The Source and ReporterComponent elements include the status of the Resources Properties described in Section 2.4.2. The individual entries are described in the next section.

2.6 Common Base Events

The IBM Common Base Event format [IBM06b] is closely related to the WSDM Event Format. It is used to describe events in a consistent, structured format, ensuring there is no ambiguity in the meaning of any facet of the event. This is critical when attempting to compare or correlate information from diverse applications and sources, such as management applications and logs.

An event is an indication of an occurrence – an external, visible manifestation of a system operation, be it the onset, evolution or conclusion of a process. Events can range from the change of state of a physical component, management signals or indications of anomalous behaviour. IBM defines an occurrence as a happening or phenomenon, a situation as the classification of an occurrence into a specific type, and an event as an indication of the situation that provides further information.

The exchange of events is critical in modern software systems, and inconsistencies in data formatting can lead to errors during these exchanges, compromising the validity of the exchange. The use of CBEs to represent the various

formats in a standardized format will allow an accurate understanding of the events [OKSC03]. A CBE consists principally of common "header" metadata, component identification, Situation information, illustrated below [IBM06b]:

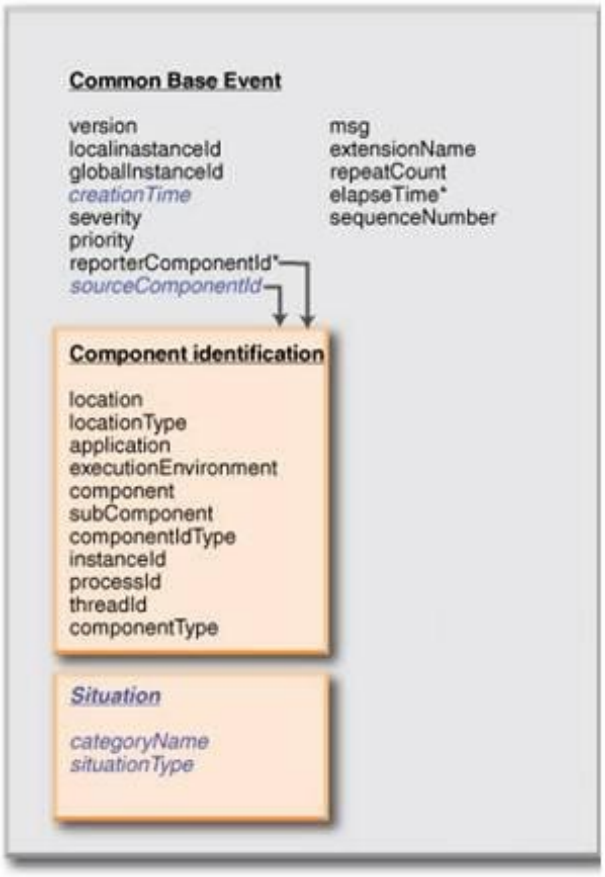

Figure 5 - Elements of the Common Base Event

The tables that follow provide amplifying information on the more important of the properties. Those in bold font are required by the specification.

Common Base Event	
Version	String identifying the CBE version – 1.0.1 is most recent version.
localInstanceId	An immutable string event identifier (need not be globally unique).
globalInstanceId	An immutable Globally Unique Id (preferably 128 or 256 bit).
creationTime	An XML dateTime value, to the greatest precision possible.
severity	The perceived severity of the event – values from 0 – 70.
priority	The importance of the event – values from 0 -100.
msg	Text accompanying an event – maximum of 1024 characters.

extensionName	The name of the event class, or XML Type, if not a CBE.
sourceComponentId	The component that was affected or impacted by the event.
reporterComponentId	The component reporting the event (required if different from the source).

Component identification	
location	Physical address of the location, e.g. 02:45:23:98:18:73, 168.192.1.201.
locationType	Type of location – IPV4, IPV6, FQ Hostname
application	The name of the application.
executionEnvironment	The immediate running environment, e.g. WebSphere Server:node:cell.
component	Logical identity (Application name), unique to environment.
subComponent	Further distinction of the logical identity.
componentIdType	Type of component (Device, System, Service, Unknown).
instanceId	Identifier or handle for specific instance (e.g. EJBHandle)
componentType	Name used to characterize instances (e.g. WebSphere or Tomcat App Server)

Table 1 - Common Base Event Amplifying Information

2.6.1 Situation

The Situation elements are used to indentify or label events in a consistent fashion. For example, the message to indicate that a server has started for one system might be simply “Server started”, and for another, it might be “Server status changed from starting to running”. Both of these identify that the server has started; however, the difference in wording would make it difficult to programmatically scan logs to determine the status of the servers. CBEs would describe this as “StartSituation – Start Completed”. The various situations include:

- StartSituation, StopSituation
- ConnectSituation, RequestSituation, ConfigureSituation
- CreateSituation, DestroySituation
- AvailableSituation, ReportSituation
- FeatureSituation, DependencySituation, OtherSituation

2.7 Generic Log Adapter

The Generic Log Adapter (GLA) is designed to render log files from their native format to the Common Base Event format, which will allow the log entries from various sources to be universally understood [GLA09]. The GLA can either employ a rule-based adapter, which uses regular expressions to match expected patterns, or a static adapter which provides more robust software-based programming. The GLA is made up of six components:

- Context – describes how a log file is to be processed, by identifying the components used in the transformation;
- Sensor – provides for the reading of content. The SingleFileSensor reads files from local storage;
- Extractor – separates the input into discrete messages, using either regular expressions, or in some cases, simple string comparisons;
- Parser – parses messages to build Common Base Events, in two phases. The first builds a series of attributes, while the second determines the value to be assigned to the attributes;
- Outputter – wraps the output in the appropriate format; destinations include standard output, a file, or a logging agent.

2.8 Apache Muse

Apache Muse [AM09] is a framework upon which developers can build Web services interfaces that are WS-ResourceFramework (WSRF) and WS-DistributedManagement (WSDM) compliant. This Java-based implementation facilitates the development of manageable resources by automating the WSDL-to-Java generation of service and client files, stubs, artifacts and proxies required to support the WS

specifications. It provides numerous utility APIs for tasks associated with WS capabilities and can be deployed in a J2EE environment.

2.9 Open Grid Services Architecture (OGSA)

As noted in the introduction, Globus utilises Web services and WSDM in its implementation. It implements the Open Grid Services Architecture (OGSA), which aims to manage resources across distributed heterogeneous platforms, deliver consistent QoS in a dynamic environment, provide a common base for autonomic management solutions, use open, standard interfaces, and to exploit industry standard integration technologies [FKNT02]. The OGSA management architecture and associated levels, depicted in Figure 6, consists of:

- OGSA Functions level: Architected services in areas such as program execution, data services and core service.
- Infrastructure level: WSDM, MUWS and WSRF provide the base manageability model for infrastructure, allowing representation and manipulation of resources.
- Resource level: Physical and logical resources – processors, network and storage.

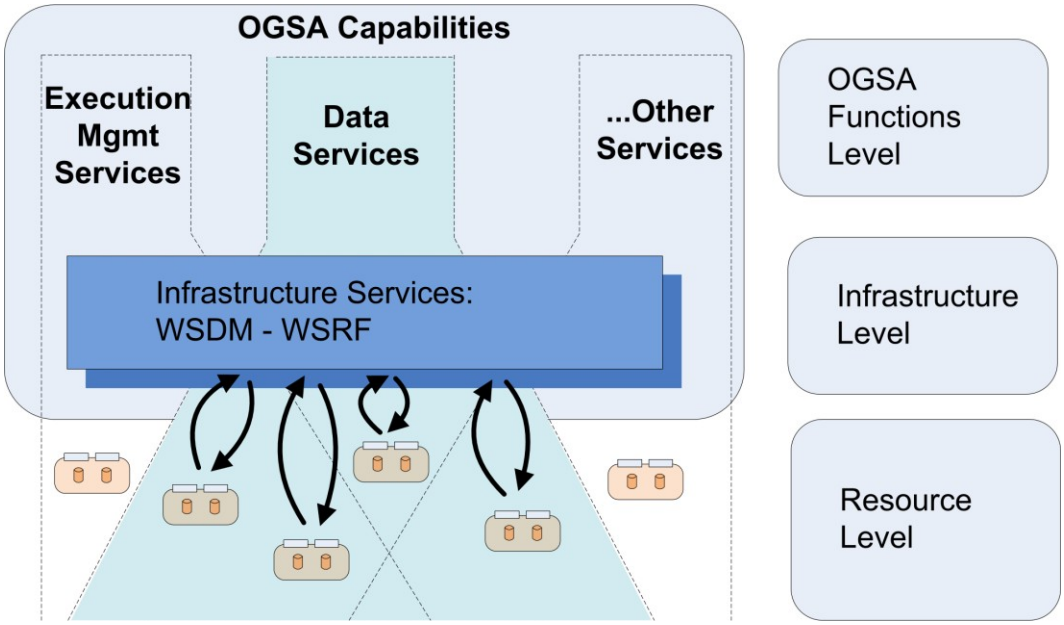

Figure 6 - OGSA Management Levels

2.10 Globus Toolkit (Overview)

The Globus Toolkit, Version 4, is an open source software toolkit used to implement grid infrastructure [Fo06]. It is comprised of five main areas: a common runtime and modules which deal with security, data management, execution management, and information services, as noted in Figure 7 below [Gl09a].

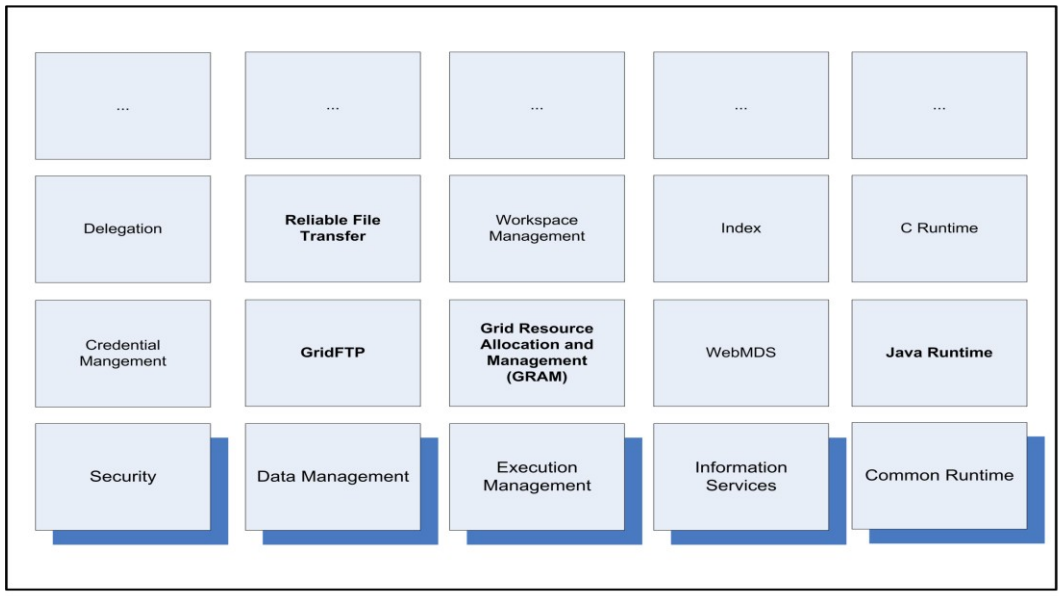

Figure 7 - Globus Toolkit Structure

GT4 provides a standalone Web services container (the shaded areas in Figure 8), within which the Java WS Core provides services including Grid Resource Allocation and Management (GRAM), Reliable File Transfer (RFT), and WS security. Client interaction involving these will utilise WS protocols. Some components, such as GridFTP, are based on C services. Direct client interaction with these services is carried out with non-WS protocols

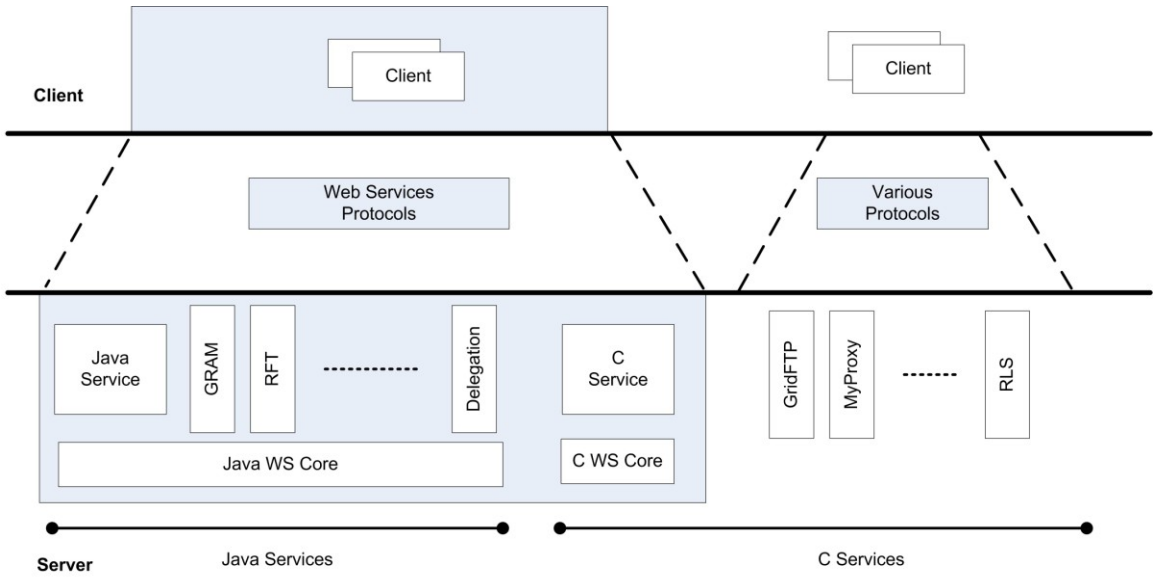

Figure 8 - GT4 Services

2.10.1 GT4 Components

Client - GT4 users can access the services through a command-line interface, or clients can be produced and packaged into a Grid Archive (GAR) file, which is deployed using the native Globus container, or another application server such as Apache Tomcat.

Security – The Grid Security Infrastructure (GSI) provides privacy, integrity, authorization and authentication, based on X.509 certificates with delegation support. Transport-level security with proxy support is currently the default (due to better performance) but message-level security based on WS-Security is also incorporated [We05].

Execution Management – GRAM provides an interface for initiating, monitoring, and managing the execution of computations on remote computers, allowing subscription to notification from resource providers [Fo06]. It allows the specification of resources desired, data staging, executables and arguments, as well as credentials and job persistence. A scheduler such as OpenPBS is still required as GRAM provides only management, not actual scheduling services.

Data Management – The base functionality for data movement in Globus is provided by GridFTP, a protocol that provides secure, robust, fast and efficient transfer of data. Layered above this is the Reliable File Transfer (RFT) system, a WSRF-compliant Web service that provides persistence and job handling for GridFTP tasks. PostgreSQL, an open source object-relational database management system is used to provide persistence.

2.11 Summary

The concepts and standards related to Web services, Web Services Distributed Management, and the Web Services Resource Framework introduced in this chapter provide an understanding of the mechanisms used in the interaction between a manageable resource and an autonomic manager. Common Base Events and the WSDM Event Format provide the mechanism to ensure that the information exchanged is unambiguous, and the Generic Log Adapter allows the extraction and categorization of information from logged sources. The WSDM-compliant Open Grid Services Architecture employed by Globus allows it to take advantage of the benefits of autonomic computing. In the next chapter, we review and discuss systems and concepts related to our work.

Chapter 3 Related Work

This chapter focuses on bodies of work related to our research, specifically, tools designed to assist in the development of Web services that support the WSDM specification.

3.1 Autonomic Integrated Development Environment (AIDE)

The IBM Autonomic Integrated Development Environment (AIDE) was developed to allow developers to take advantage of the WSDM specification. It utilized a defined architecture for connecting computing resources with management applications through standards-based interfaces [IBM05]. AIDE provided three distinct tools to facilitate the deployment of WSDM-enabled solutions.

The Manageability Endpoint builder provided a development and run-time environment to build endpoints that conform to the WSDM specification, using Eclipse or Rational Application Developer (RAD). The interfaces exposed by the endpoints allowed an autonomic manger or other WSDM-compliant tool to view or change the status of a resource.

The Manageability Resource Browser allowed for the introspection of WS-Resources and WS-ServiceGroups. Users could access and manipulate the properties exposed by a WSDM endpoint and invoke operations on the endpoint. This allowed testing to be performed for functional validation of the endpoints.

The Manageability Endpoint Simulator (MES) assisted in the development of autonomic managers by emulating a WSDM-compliant managed resource. It took advantage of the Generic Log Adapter, allowing applications whose output does not conform to Common Base Events to be utilized. For reasons noted in the discussion section, the functionality provided by AIDE was ported to another Eclipse project and AIDE itself was sunset by IBM.

3.2 Eclipse Test and Performance Tools Platform

The Test and Performance Tools Platform (TPTP) is an Eclipse project [Ec07] that permits building of tools for testing and performance evaluation applicable throughout the entire software development cycle. It includes sub-projects for monitoring, tracing and profiling, and testing. Within the Monitoring Tools portion, version 4.3.0 saw the introduction of the Build-to-Manage toolkit which provides the ability to integrate the Apache Muse runtime. Support for Apache Muse 2.2.0 was introduced in version 4.4.0, allowing developers to take advantage of the latest improvements to Muse.

TPTP 4.4.0 expands some of the functionality provided by AIDE. It allows the building of Manageability Endpoint projects, and includes a relatively simple user interface to add capabilities to the endpoint. It includes support for a range of WSDM capabilities including Description and Identity, Resource Lifetime, Resource Properties such as Get and Query, and Service Groups, however, support for Notification Producer is limited [Ec08].

Within a Manageability Endpoint project, Eclipse, TPTP builds an .mrt file that defines the capabilities of the endpoint, and a .mcap file that defines the properties of a user-generated capability. Once an endpoint is specified, a Manageability Agent Explorer allows access to and manipulation of properties, quite similar to the functionality of AIDE. Support for the TPTP Build-to-Manage toolkit was discontinued beyond version 4.4 and was ported to another Eclipse project.

3.3 COSMOS

The Eclipse support for the Build-to-Manage toolkit was moved to the Community Systems Management in Open Source (COSMOS) project [Ec09]. COSMOS is designed to provide standards-based, extensible components that will permit

management of diverse systems, to include data collection and visualization, management enablement and resource modeling. It does so using Service Modeling Language (SML) [W3C09a] and the Service Modeling Language Interchange Format (SML-IF) [W3C09b]. These two specifications detail a model which can be used to consistently describe IT resources, services and systems, as well as an implementation-neutral interchange format.

The management of data from multiple heterogeneous Management Data Repositories (MDRs) is enabled by the use of the Configuration Management Database Federation (CMDBf) [DMTF09]. It utilises a query language to access information from the various MDRs and store it, using the service modeling language noted above, in a common (and homogeneous) configuration management database.

3.4 Discussion

AIDE was IBM's first attempt at providing an IDE to assist with the development of autonomic computing. It provided a relatively simple-to-use interface that allowed developers to choose the capabilities they wanted to include in a touchpoint and have the necessary artifacts and code generated automatically. This shortened development time by hiding much of the complexity of the Web services infrastructure from the developer. It did, however, impose some limits on the customisation of the implementation as AIDE introduced a significant amount of specialized AIDE-related code in the artifacts. This made it complicated even for a developer who was familiar with Web services to modify the root Web services documents. The AIDE implementation also bundled the endpoints and capabilities into proprietary packages that could not easily be ported to another environment.

The Test and Performance Tools Platform, with the introduction of the Apache Muse package, removed the dependence on proprietary code, and ensured wider applicability of resulting projects. TPTP allows the building of manageability endpoints

in a simple Eclipse user interface and automated the generation of the required background Web services infrastructure. The resultant projects are easier to port to other environments.

There are limitations to the auto-generation capabilities, notably with Notification Producer, that restricted its usability for more complex projects, but the TPTP package is fully able to support all WSDM requirements.

The transfer of the Build-to-Manage capabilities to COSMOS was the next step in the evolution. COSMOS promises to provide a better integrated platform for the high level management of systems. However, COSMOS is under continuing development and the WSDM components are not yet able to be harnessed by developers.

The projects described above represent the initial steps in the evolution of a framework to allow development of WSDM-compliant systems. Our approach is to build on the original Touchpoint concept, utilise the capabilities of TPTP to provide the platform that will support our development, but extend it to allow the generation of all of the necessary artifacts for use by the developer. The COSMOS framework will be closely monitored as it holds the potential for use in the future once WSDM support is implemented.

3.5 Summary

There have been a number of models and toolkits to aide developers in the design and implementation of autonomic computing programs. AIDE was IBM's initial attempt, but it imposed propriety code that limited its portability. TPTP overcame this, but it suffers from limitations in the implementation of some crucial aspects of WSDM. COSMOS looks very promising, but its current lack of WSDM support precludes it from being used in the design of simulated autonomic resources. The next chapter describes the architecture that will take advantage of the foundation built by AIDE, utilising TPTP

tools that will allow the development of a touchpoint simulator that can provide realistic output from manageable resources.

Chapter 4 General Approach

The general approach of this research is to examine the requirements of grid services, the requirements of touchpoints and design a model that will provide the necessary output for developers to use when building and testing autonomic managers. We first examine Globus to determine what is output to the system logs. We then examine the requirements for touchpoints, to include the architecture, interaction patterns, and manageability capabilities. We next examine Muse, the toolkit that will allow the implementation of WSDM and WSRF standards, and describe the steps necessary to deploy a compliant resource. With this foundation, we describe the design and implementation of a touchpoint simulator that consists of a WSDM-compliant simulated manageable resource that reads a Globus log, converts the information therein to Common Base Events, transforms these to WSDM Event Format messages, and transmits them to a WSDM-compliant simulated autonomic manager.

4.1 Globus Grid Services

A three-node Globus grid was produced using VMware and the interactions of the client and grid services were examined. Examining the submission of jobs using the Grid Resource Allocation and Management (GRAM) component highlighted the functionality of GT4 and provided logged examples that will be used in the development of the touchpoint simulator.

GRAM provides execution management for GT4, providing access to data from computational tasks [Gl09b]. It is a protocol engine for communicating with local resource schedulers using a standard message format, allowing the monitoring and control of job life cycles. It is targeted for use where jobs require reliable operation, stateful monitoring, credential management, and file staging.

The interrelation of the various GRAM components is depicted at Figure 9:

① The client submits a job to GRAM services, along with appropriate delegation rights (for both GRAM and the job).

② The target file is transferred to the destination using Reliable File Transfer and GridFTP.

③ The local GRAM adapter verifies that permission has been given for the local user to execute the functions required for job management.

④ The job is scheduled by the local scheduler.

⑤ The resulting files are transferred back to the service host.

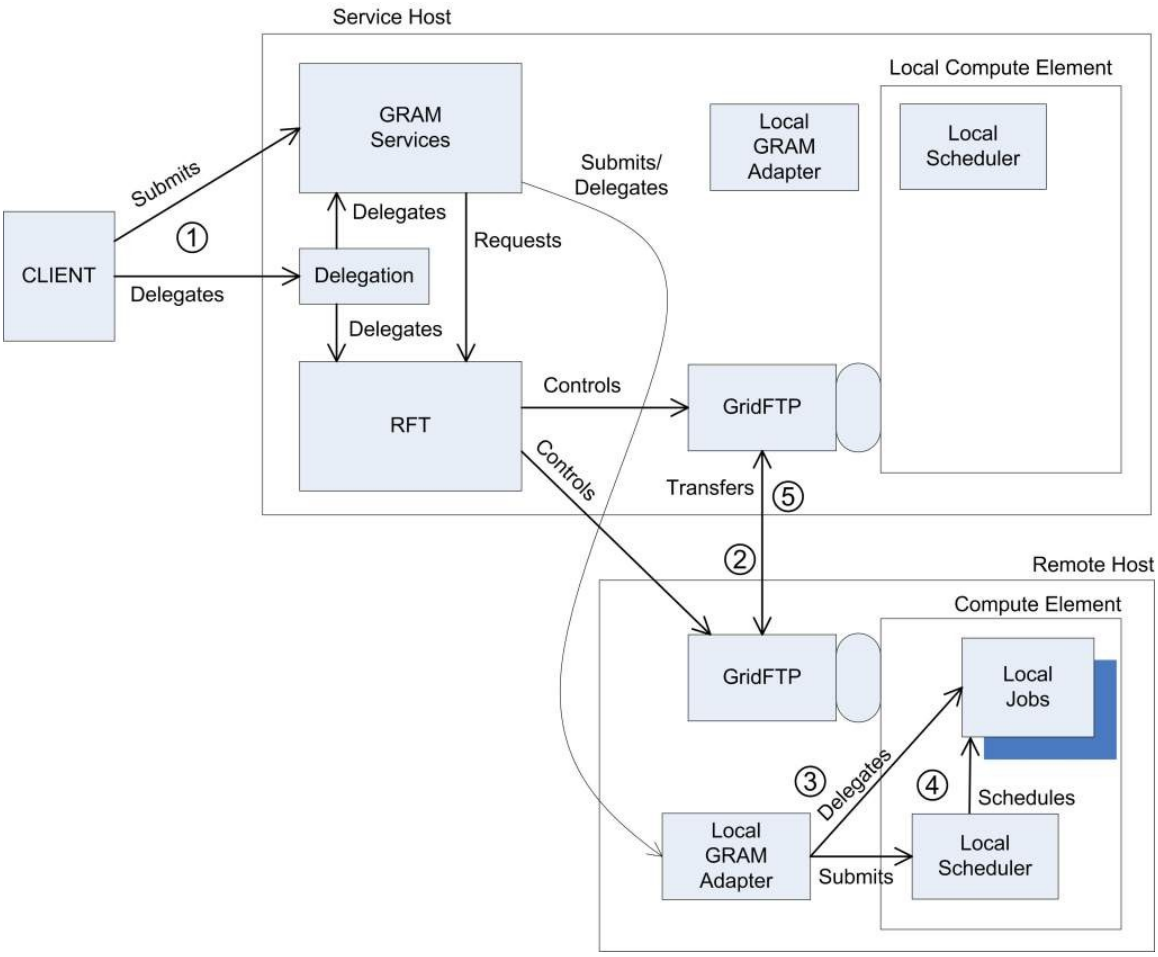

Figure 9 - GRAM Overview

4.2 Globus Client and Job

To provide a proper context for the prototype, a scenario was devised for implementation using Globus GT4. This allows us to have an understanding of the log entries generated by Globus and permits us to ensure that the information ultimately received by the autonomic manger reflects what was in the logs. A user is required to convert a very large number of audio files from .wma format to .mp3 format. The conversion is a two step process (using Linux tools); conversion from .wma to .wav using mplayer, then conversion to .mp3 using lame. The demand for the service is somewhat irregular, but the time in which to accomplish the task is very restricted, leading to sporadic peak demands which far exceed the local capability to meet that demand. A grid solution is implemented to allow the peak demand to be met by taking advantage of off-site resources.

The setup used to develop the logs for the prototype consists of three VMware virtual machines (VM) running SUSE Linux 10.3. Each has Globus Toolkit 4.2.1, including GRAM, Reliable File Transfer (RFT) and GridFTP, the postgreSQL database (for data persistence) and OpenPBS (for scheduling). Each has a globus, gt4user, and postgres user in addition to the root user. Security is provided by simpleCA, and each VM has the appropriate host and user certificates, as well as a container certificate for globus. The client machine (vm103a) will be used to submit the jobs, while resource B (vm103b) and resource C (vm103c) will be used to run the jobs, using the OpenPBS scheduler.

In order to understand the output produced in the log file properly, we will follow a representative cycle from submission to completion - the workflow is depicted at Figure 10. The task is submitted as Globus GRAM multiJob. The entire job file is included at Annex B. The endpoint reference (EPR) for the job is the ManagedJobFactoryService of vm103b at the URL https:// vm103b.hlinux.thshome: 8443/wsrf/services/ManagedJob FactoryService). The first job will see the wma file staged into vm103b, with the

mplayer executable and appropriate arguments. The wma file is converted using mplayer to wav format. The second job is the conversion of the audio file from wav format to mp3 format, transport back to vm103a and the cleanup of residual files in vm103b.

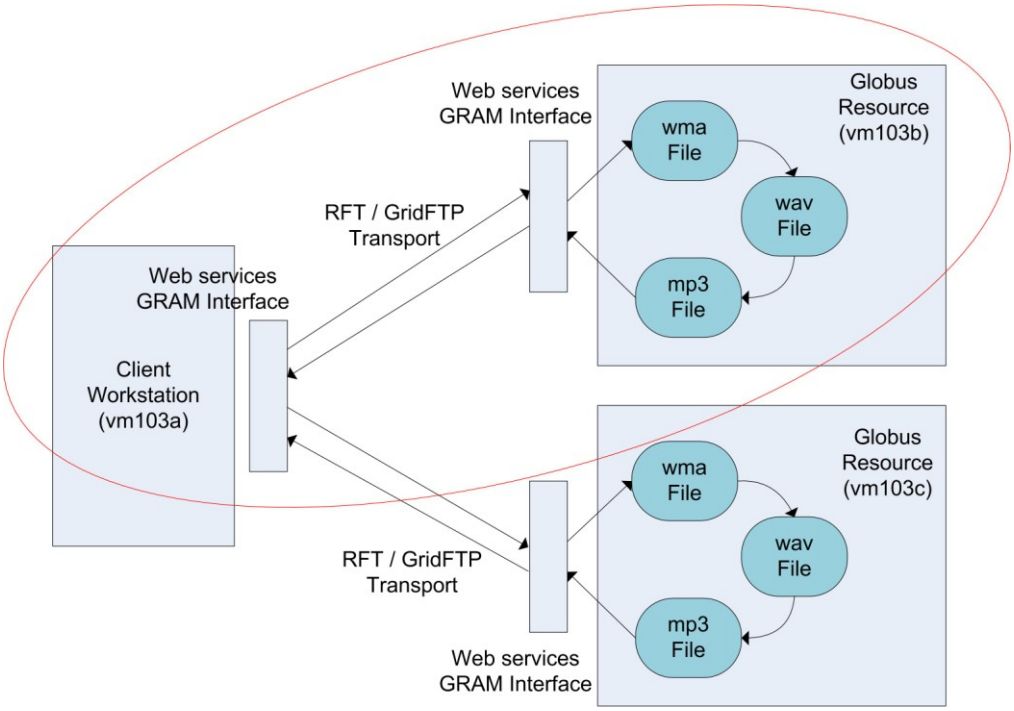

Figure 10 - Globus MultiJob Workflow

This job submission is reflected in the sequence diagram at Figure 11. Delegation of rights, then the storage and destruction of the credential are noted. The two jobs are created. Job 1 (`6a363160`) conducts the StageIn of files via RFT, and submission of mplayer conversion (.wma to .wav) component. Job 2 (`6a7bc5e0`) is the submission of the lame (.wav to .mp3) conversion. It should be noted that Job 2 is submitted to the scheduler first, so a script had to be written to delay its execution until the completion of the Job 1 conversion (from .wma to .wav). Once the two conversion jobs are completed, the resulting file is StagedOut using RFT and the intermediary files are deleted from the remote server.

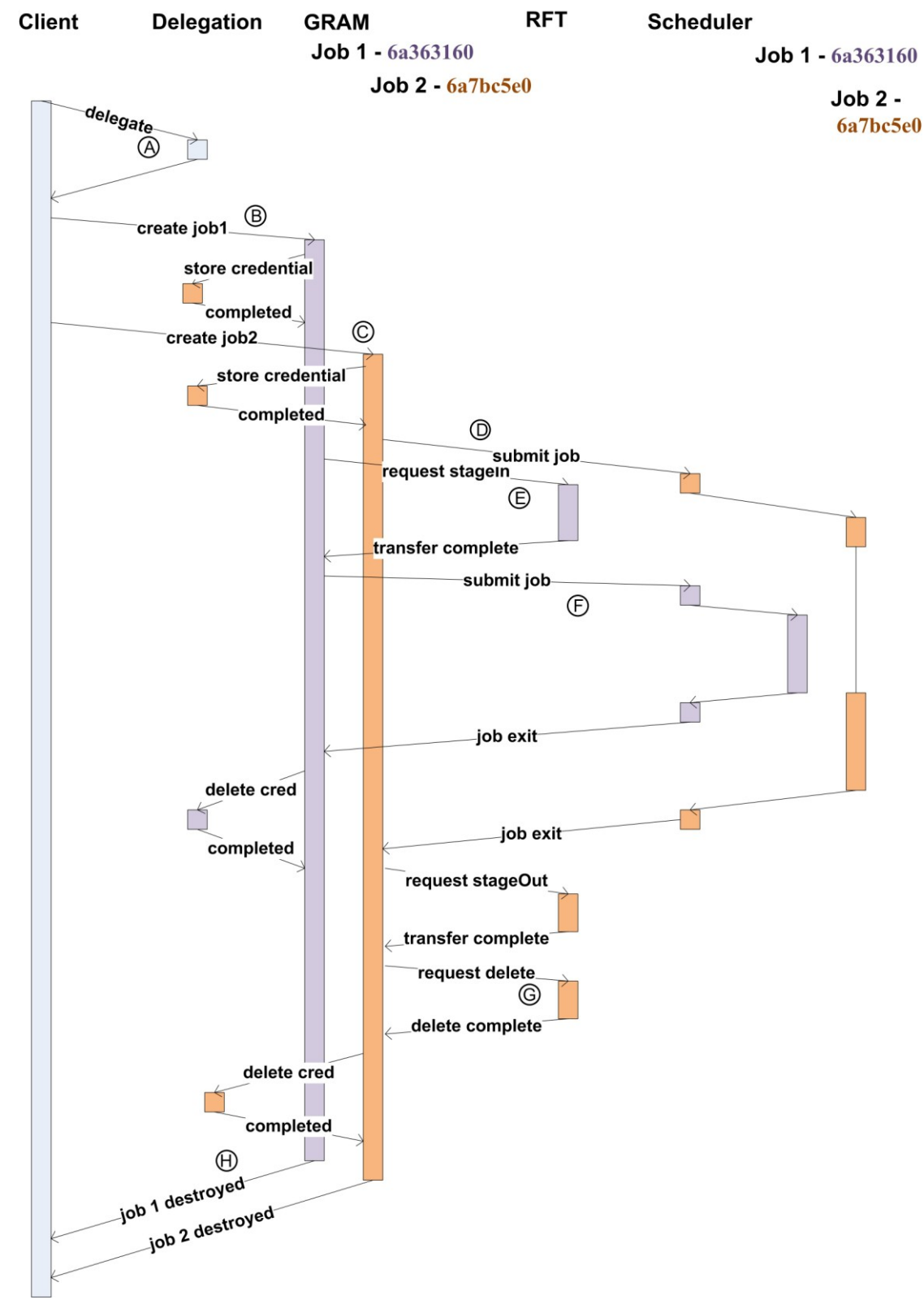

Figure 11 - RFT Sequence Diagram

These actions are reflected in the excerpt of the destination server log in Table 2 below. Only selected entries are included for readability, but they highlight the main actions taken the RFT sequence. The event letters correspond to the events in Figure 11. The resulting log entries are listed below:

Table 2 - Globus Job Log Entries

Event	Log Entry
(A)	ts=2009-11-28 12:35:12,210 level=INFO guid=68b55dc0-02ef-11df-951b-8e37a56df417 event=org.globus.authz.end service=**DelegationFactoryService** operation={http://www.globus.org/01/2008/delegationService} **requestSecurityToken**
(B)	ts=2009-11-28 12:35:14,850 level=INFO thread=ServiceThread-59 sbcomp=AdminLogger guid=6a1032d0-02ef-11df-951b-8e37a56df417 event=org.globus.execution.**job.create.end** status=0 **jobResource.id**=**6a363160**-02ef-11df-951b-8e37a56df417 service=ManagedJobFactoryService
(C)	ts=2009-11-28 12:35:15,252 level=INFO guid=6a699d70-02ef-11df-951b-8e37a56df417 event=org.globus.**execution.job.create**.end status=0 **jobResource.id=6a7bc5e0**-02ef-11df-951b-8e37a56df417 clientSubmission.id=2e581f32-edf5-11de-b33c-000c29be864e service=ManagedJobFactoryService
(D)	ts=2009-11-28 12:35:15,252 level=INFO thread=pool-1-thread-4 sbcomp=AdminLogger event=org.globus.execution.**job.submission**.start **jobResource.id**=**6a7bc5e0**-02ef-11df-951b-8e37a56df417 service=ManagedExecutableJobService
(E)	ts=2009-11-28 12:35:15,742 level=INFO event=org.globus.**execution.job.stageIn**.end status=0 **jobResource.id=6a363160**-02ef-11df-951b-8e37a56df417 transferResource.id=116 service=ManagedExecutableJobService
(F)	ts=2009-11-28 12:35:20,183 level=INFO thread=pool-1-thread-8 sbcomp=AdminLogger event=org.globus.execution.**job.submission**.start **jobResource.id**=**6a363160**-02ef-11df-951b-8e37a56df417 service=ManagedExecutableJobService
(G)	ts=2009-11-28 12:35:26,893 level=INFO thread=pool-1-thread-6 sbcomp=AdminLogger event=org.globus.execution.**job.fileCleanUp**.start **jobResource.id**=**6a7bc5e0**-02ef-11df-951b-8e37a56df417 service=ManagedExecutableJobService

(H)	ts=2009-11-28 12:35:34,152 level=INFO thread=ServiceThread-72 sbcomp=AdminLogger guid=75ae4a00-02ef-11df-951b-8e37a56df417 event=org.globus.execution.**job.terminate**.end status=0 **jobResource.id**=**6a363160**-02ef-11df-951b-8e37a56df417 requestedBy=user service=ManagedExecutableJobService

These log entries form the basis for the touchpoint simulator input and are examined in more detail in the implementation section. We next examine touchpoints in more detail to understand how they will be used.

4.3 Touchpoints

A touchpoint is the implementation of the manageability interface of a manageable resource [IBM06a]. The IBM Autonomic Computing Manageable Resource Interface Specification [IBM05] provides a reference architecture for the interface, a basic set of manageability capabilities, and a definition of interaction styles, as described below. The single, standard manageability interface provided by the touchpoint is a keystone to autonomic computing, effectively replacing numerous existing manageability interfaces required to manage various types of resources [Mi05a].

The guiding principles that determine the architecture of touchpoints are:

- All components must provide a consistent set of interfaces;
- The behaviour of manageable resources at both design and run time must be predictable and use the same description mechanism;
- The capabilities and styles of interaction of a manageable resource must be easily definable and able to be determined through self-reflection;
- Complexity should scale with functionality – a simple touchpoint should not have to deal with or define higher-level capabilities;
- Existing manageability interfaces and applications should be exploitable; and
- A touchpoint should be able to be implemented from multiple components.

Touchpoint manageability interfaces are expressed as Web services; they are exposed through a WSDL-defined interface and accessed through Web services communication protocols, as extended by the Manageability Resource Interface Specification.

Manageability interfaces are sets of properties, operations, events, and meta-information described by WSDL and Resource Properties documents. These interfaces expose manageability sensors and effectors through which autonomic managers and other autonomic applications interact with the manageable resource. An example of sensor input is an operation or event that exposes information about the operational state of a resource, and transitions of that state. An example of effector output is an operation that causes the manageable resource to update one of its properties.

Every instance of a manageable resource must have its own Endpoint Reference (EPR), as well as a unique identity, available through self-reflection. This identity must be opaque to users (i.e., it is determined by the resource itself and cannot be systematically predicted by an outside agency), must not allow characteristics of the resource to be inferred, and must become invalid once the resource is terminated.

4.3.1 Touchpoint Interaction Patterns

Four standard interaction patterns have been defined to describe the methods used by autonomic managers to access and manipulate the state of a manageable resource. When dealing with a sensor, a manager will utilize Request-Response in order to poll a resource. WS-ResourceProperties specifies which operations may be used; GetResourceProperty is used in conjunction with a resource's qualified name to retrieve the contents of a single resource property. GetMultipleResourceProperties may be used to retrieve a number of properties with a single message, and QueryResourceProperties is used with XPath to return a subset of properties. This interaction style is the only one

that is mandatory for a touchpoint, as it is used to return the Identity of a managed resource.

To send information to a manager, a manageable resource will use the Send-Notification interaction style. The manager must subscribe to events it desires, and the information is passed from the sensor as WSDM Event Format messages.

The state of a resource may be changed by a manager, through the use of an effector, by using the Perform-Operation interaction style. The WS-ResourceProperty operation SetResourceProperty can insert, update or delete the values of a property.

Finally, a resource may use the Solicit-Response interaction style to retrieve information from an autonomic manager. This is accomplished using the CalloutUser capability.

4.3.2 Touchpoint Manageability Capabilities

A touchpoint may support a range of manageability capabilities, which can be thought of as a "contract" that the touchpoint asserts it can offer to clients. The following table outlines the mandatory capabilities that all touchpoints must invoke [IBM05].

Table 3 - AC Touchpoint Required Interfaces

Capability:	**Rationale:**
wsdm:Identity *Properties:* - ResourceId	The *ResourceId* must be globally and temporally unique. Also, the same value must be returned by all endpoints that access that specific resource.
wsdm:Description *Properties* - Caption, Description, Version	Caption is a descriptive name for the resource, Description is a string describing the resource, and Version returns the version.
actp:ResourceType *Properties* - ResourceType	This is a fundamental interface that must be implemented by all manageable resources. If a resource can be considered to have multiple functional types (e.g. database server and HTTP server), it is recommended that separate manageable resources be used.

Capability:	Rationale:
wsdm:Manageability Characteristics *Properties* - ManageabilityCapability	An interface supports a capability if it supports the properties, events, metadata and operations of the capability. There should be one ManageabilityCapability property instance for each capability. This property facilitates introspection by allowing operations, events and properties to be identified by consumers.
wsrp:GetResourceProperty *Operations* - GetResourceProperty	This is a required interface for WS ResourceProperties. All MRs have resource properties so all must implement this.
wsrp:SetResourceProperty *Operations* - SetResourceProperties	If a MR has any read-write properties then this interface must be implemented.

There are a number of interfaces which are optional for touchpoints. In most cases, it is only the more complex touchpoints that will have to implement them. In simpler instances, they may be omitted, thus reducing the footprint of the touchpoint.

Table 4 - AC Touchpoint Optional Interfaces

Capability:	Rationale:
wsrp:QueryResource Properties *Operations:* - QueryResourceProperties	Manageable resources should implement this except in special circumstances, such as very small footprint requirements.
wsrp:GetMultipleResource Properties *Operations:* - GetMultipleResource Properties	Manageable resources should implement this except in special circumstances, such as very small footprint requirements.
wsdm:OperationalStatus *Properties* - OperationalStatus	This capability defines a simple representation of the availability of a resource, using the values Available, PartiallyAvailable, Unavailable and Unknown.
wsdm:Configuration *Operations* - ws-rp: SetResourceProperty	WSDM does not define any properties and expects that this will be defined for each resource domain. The SetResourceProperty operation may be used to change the configuration.
wsdm:State	Although most touchpoints will require that state be maintained, this interface is not required as part of WSDM, which defers both properties and operations to some other source for defining state.

Capability:	Rationale:
wsdm:Metrics *Properties* - CurrentTime	The property CurrentTime is read-only and indicates when a property was retrieved from a manageable resource.
wsdm:Relationships *Properties* - Relationship *Operations* - QueryRelationhipsByType	This interface is implemented by manageable resources that know about the relationships they have with other manageable resources. The same interface is used on third party store of relationship information.
actp:MetricContol *Properties* - ActiveMetric *Operations* - ControlMetricCollection	Some metrics can be costly for a resource to collect and so some manageable resource will choose to allow the collection of some metrics to be started and stopped as required. If any metrics are controllable, then the manageable resource must implement this interface, which provides the mechanism to control the metrics.
actp:CalloutUser *Properties* - RequiredCapability *Operations* - AssignProvider	This interface must be implemented if the manageable resource is able to make Callout requests, which are used in the Solicit-Response interaction style. The interface is used by the manager to assign providers to callouts used by the manageable resource and thus establish the style of management it will use.

Touchpoints provide the consistent, standards-based interface required for communication between an autonomic manager and a managed resource. By adhering to the specifications and guiding principles noted above, it is possible to ensure that touchpoints that are developed are easily deployable, and will provide a reliable link to autonomic manager. We next review Muse, the toolkit that is used to ensure that the required standards are incorporated in the implementation of the prototype.

4.4 MUSE

The Apache Muse Project is a Java-based implementation of the WSRF, WS-Notification, and WSDM specifications. It is a framework that allows users to implement Web service interfaces for manageable resources without having to manually write and implement the infrastructure required to support the WS specifications.

The programming model for Muse is very similar to typical Web services, but some concessions have been made to allow for greater portability. As can be seen in Figure 12, on the right-hand side are the capabilities expected under the WSRF model. The resource types are the interfaces defined by a WSDL. Muse allows deployment of multiple resource types, through the use of the resource manager and router; each unique resource type is specified by providing a context path that is typically the name of the WSDL. The Muse isolation layer marshals all communications between the resource router and the deployment environment. Resource types can be deployed in various environments (Apache Axis 2 and OSGi, for example) with no change to code, and only minor changes to the deployment descriptor.

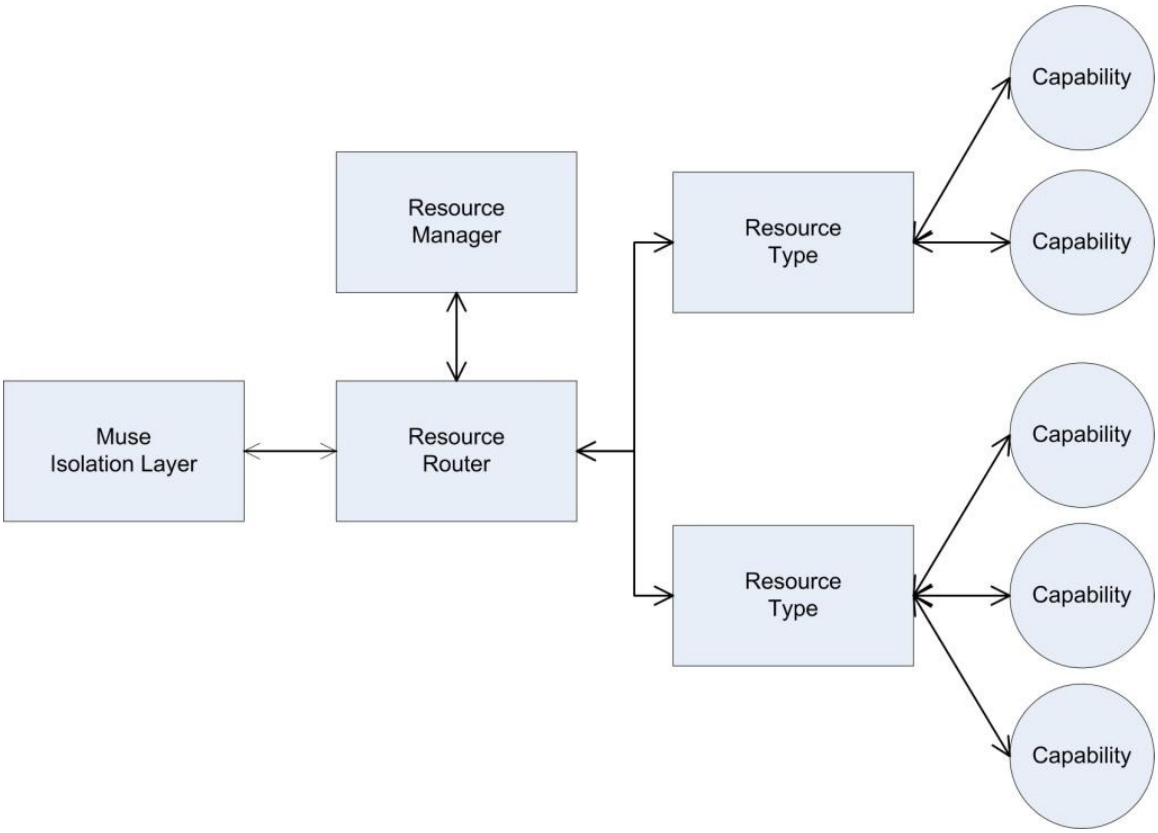

Figure 12 - Muse Programming Model

There are command line tools that facilitate the generation of the artifacts required for projects. Muse 2.2.0 includes the following features:

- Implementation of all WSRF 1.2, WSN 1.3, WSDM 1.1, and WS-MetadataExchange port types.
- Implementation of the WSDM Event Format 1.1.
- Compliance with WS-Addressing 1.0 and SOAP 1.2.
- A persistence API so that users can recover the state of a WS-resource after shutdown of the host.
- WSDL-to-Java tooling that creates service and client-side code and artifacts.

Muse includes representative WSDL file templates, written using specific conventions to ensure they are processed consistently. The templates provide the necessary details to implement all of the properties and operations defined by WSRF, WSN and WSDM.

The WSDL used in the touchpoint simulator implementation contains the definitions for the following capabilities:

- WS-ResourceProperties - *GetResourceProperty* operation
- WS-MetadataExchange - *GetMetadata* operation
- WSN NotificationProducer - *Subscribe* and *GetCurrentMessage* operations, as well as WS-Topics properties
- WSDM MUWS Identity - *ResourceId* property
- WSDM MUWS Description - *Caption*, *Description*, and *Version* properties
- WSDM MUWS OperationalStatus - *OperationalStatus* property

The following custom features are also included:

- User-Generated Properties – ResourceType, Name, GridProperty

In addition to the WSDL file, Muse allows for the use of a resource metadata descriptor document (RMD). This file indicates the name of each property, and its modifiability (read-only/read-write) and mutability (constant/mutable/appendable). It also allows for additional information such as initial and allowable values to be specified. With a valid WSDL and RMD (if required) file, invoking the wsdl2java command generates all of the files required to produce a deployable Web archive (WAR) file for a selected environment (i.e. J2EE and Axis 2) , once business logic and some initialization is incorporated in the stub files. A proxy, to allow interaction with the endpoint, is also generated.

The WAR file contains all necessary elements for deployment, including Axis 2 files, a library with Muse and Web services JARs and a number of specialized files. The first of these are consolidated WSDL files which includes all portions of the original resource WSDL file, as well as all imports. This results in a single "flattened" file that precludes the need to resolve imports and links dynamically. A router-entries directory holds endpoint persistence entries that will allow EPR-to-resource mapping to be stored and reloaded upon restart. A services.xml file collects all the operations from the WSDL file and links them with Axis 2 operations – it is this file that acts as the Muse "isolation layer" noted above.

This file will also contain entries from other services in the system, such as a factory, or subscription manager for the producer-consumer model.

The final unique file is the deployment descriptor (muse.xml) used to load, configure, and support the resource types that have been implemented. The file contains information on logging, and persistence, as well as the location of the WSDL files which describe the resource types. The capabilities which make up each resource are then described, listing their URIs, as well as their concrete implementation classes.

4.5 Summary

This chapter described the general approach of this research. The output from a Globus GRAM job which involves the transfer of files using Reliable File Transfer and GridFTP is utilised to provide the log files for the simulated manageable resource touchpoint. The touchpoint implements the specified reference architecture, and implement the necessary interaction patterns, as well as the required and some optional capabilities. The Muse project toolkit provides the necessary tools to allow the developer to define and implement the needed capabilities and properties, and deploy them in the development environment.

Now that the foundation has been established, the next chapter describes the implementation of the touchpoint simulator.

Chapter 5 Touchpoint Simulator Implementation

5.1 System Architecture Overview

In order to demonstrate the ability of the touchpoint simulator to aide in the development of autonomic managers, the system design must replicate a typical deployment environment. Given that numerous resources may be deployed at any time, a resource factory is used to generate the resources, as depicted at Figure 13. The autonomic manager interfaces with the simulated managed resource using the prescribed WSDM-compliant Web services. The mandatory capabilities (Identity, Description, ResourceType, Manageability Characteristics, Get & SetResourceProperties) as well as a number of the optional capabilities (State, Operational Status, and GetMultiple ResourceProperties) are implemented. Three of the interaction patterns are also implemented and tested. Request-Response is required for the manager to query the Identity of the managed resource, and Send-Notification is the mechanism used to transmit the simulated output from the managed resource to the manager.

The managed resource provides the logic to fulfill the requirements of a touchpoint and incorporates a Generic Log Adapter (GLA) feed. The GLA utilises a rule-based adapter to convert the GT4 logs to Common Base Events for transmission to the autonomic manager. The GLA reads the log from a physical file and performs the conversion to Common Base Events. These Common Base Events are then transformed using an XSLT transform into WSDM Event Format messages. The messages are dispatched with an appropriate topic label, to be received by the autonomic manager. The manager receives the events and parses them into their constituent sections, allowing for their use by the developer in the building and testing of the autonomic manager.

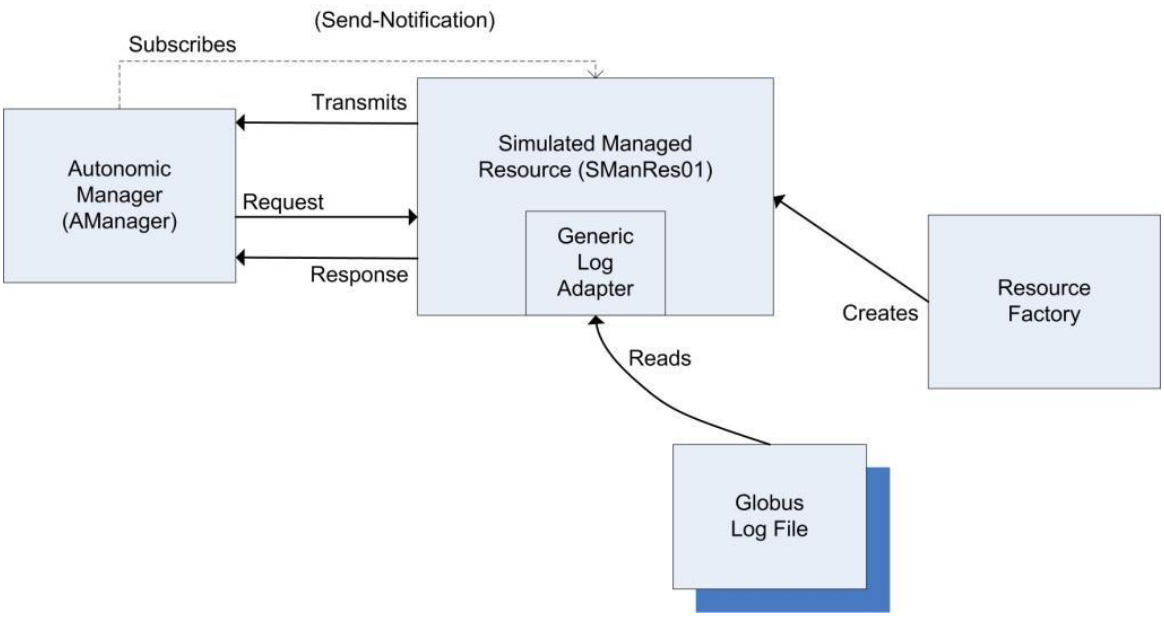

Figure 13 - Touchpoint Simulator Implementation Overview

5.2 Simulated Manageable Resource Generation

Discussion of the simulated Managed Resource must begin by examining the appropriate portions of the WSDL file used in its formation (full listing at Appendix E). Table 5 outlines the schema entry for the properties of GridResource01, beginning with the namespace declaration, then the declaration of three user-generated properties, ResourceType, Name and GridProperty. Lines 08-20 of Table 5 list the WSRF and MUWS properties that the resource contains. The two time elements are required to implement ResourceLifetime capabilities, and QueryExpressionDialect extends the GetResourceProperties capability by specifying which properties are to be returned. The next four lines (the "Topics" entries) are related to Notifications, allowing specification of which topics are to be subscribed to. The next six lines specify which Manageability Capability properties are implemented. ResourceId is the only required property, while Caption, Description and Version provide additional information about the resource. OperationalStatus provides the state of operations. The ManageabilityCapability

property is used to list those properties supported by the resource. Finally, lines 22 – 24 list the custom capabilities being implemented.

Table 5 - SManRes01.wsdl File Listing

```
(01)  <xsd:schema elementFormDefault="qualified"
        targetNamespace="http://vres.hlinux.thshome/muse/touch/impl">
(02)    <xsd:element name="ResourceType" type="xsd:string" />
(03)    <xsd:element name="Name" type="xsd:string" />
(04)    <xsd:element name="GridProperty" type="xsd:string" />
(05)      <xsd:element name="GridResource01Properties">
(06)       <xsd:complexType>
(07)        <xsd:sequence>
(08)          <xsd:element ref="wsrf-rl:CurrentTime" />
(09)          <xsd:element ref="wsrf-rl:TerminationTime" />
(10)          <xsd:element ref="wsrf-rp:QueryExpressionDialect"
                        minOccurs="0" maxOccurs="unbounded" />
(11)          <xsd:element ref="wsnt:FixedTopicSet" />
(12)          <xsd:element ref="wst:TopicSet" minOccurs="0" />
(13)          <xsd:element ref="wsnt:TopicExpression" minOccurs="0"
                        maxOccurs="unbounded" />
(14)          <xsd:element ref="wsnt:TopicExpressionDialect"
                        minOccurs="0" maxOccurs="unbounded" />
(15)          <xsd:element ref="muws1:ManageabilityCapability"
                        minOccurs="0" maxOccurs="unbounded" />
(16)          <xsd:element ref="muws1:ResourceId" />
(17)
(18)          <xsd:element ref="muws2:Caption" minOccurs="0"
                        maxOccurs="unbounded" />
(19)          <xsd:element ref="muws2:Description" minOccurs="0"
                        maxOccurs="unbounded" />
(20)          <xsd:element ref="muws2:Version" minOccurs="0" />
(21)          <xsd:element ref="muws2:OperationalStatus" />
(22)          <xsd:element ref="tns:ResourceType" />
(23)          <xsd:element ref="tns:Name" />
(24)          <xsd:element ref="tns:GridProperty" />
(25)        </xsd:sequence>
(26)       </xsd:complexType>
(27)     </xsd:element>
(28)    </xsd:schema>
(29)  </wsdl:types>
```

The GridResource01.rmd file, depicted at Table 6, shows the definition of each property in the resource. A mutability entry of “constant” indicates the property’s value must not be changed after the resource is created, while “mutable” allows removal or updating. Modifiability of “read-write” indicates whether the value can be changed via the Web services setResourceProperty function. Note that some properties, including

Caption and Description, are “read-only” yet “mutable”, signifying that the values can be changed from the server side after the resource is created. Initial values may also be specified in the .rmd file. In this case, we supply the ResourceType (http://vres.hlinux.thshome/muse/touch/impl/GridResource) for the resource being simulated, as well as Name (GridResource01).

Table 6 - SManRes01.rmd File Listing

```
(01)  <?xml version="1.0"?>
(02)  <Definitions xmlns="http://docs.oasis-open.org/wsrf/rmd-1">
(03)   <MetadataDescriptor xmlns:myns=                    "
                  http://vres.hlinux.thshome/muse/touch/impl "
(04)   xmlns: (remaining namespace declarations omitted for brevity)
(05)   name="GridResource01Metadata"
                  interface="myns:GridResource01PortType"
(06)   wsdlLocation="http://vres.hlinux.thshome/muse/touch/impl
                 GridResource01.wsdl">
(07)    <Property name="wsrl:CurrentTime" modifiability="read-only"
                  mutability="mutable" />
(08)    <Property name="wsrl:TerminationTime" modifiability="read-only"
                  mutability="mutable" />
(09)    <Property name="wsnt:FixedTopicSet" modifiability="read-only"
                  mutability="constant" />
(10)    <Property name="wst:TopicSet" modifiability="read-only"
                  mutability="mutable" />
(11)    <Property name="wsnt:TopicExpression" modifiability="read-only"
                  mutability="mutable" />
(12)    <Property name="wsnt:TopicExpressionDialect"
                  modifiability="read-only" mutability="mutable" />
(13)    <Property name="muws1:ResourceId" modifiability="read-only"
                  mutability="constant" />
(14)    <Property name="muws1:ManageabilityCapability"
                  modifiability="read-only" mutability="constant" />
(15)    <Property name="muws2:Caption" modifiability="read-only"
                  mutability="mutable" />
(16)    <Property name="muws2:Description" modifiability="read-only"
                  mutability="mutable" />
(17)    <Property name="muws2:Version" modifiability="read-only"
                  mutability="constant" />
(18)    <Property name="myns:ResourceType" modifiability="read-only"
                  mutability="mutable" />
(19)      <InitialValues>
(20)        <myns:ResourceType>
(21)          http://vres.hlinux.thshome/muse/touch/impl/GridResource
(22)        </myns:ResourceType>
(23)      </InitialValues>
```

```
(24)    <Property name="myns:Name" modifiability="read-only"
                    mutability="mutable" />
(25)      <InitialValues>
(26)        <myns:Name>
(27)          GridResource01
(28)        </myns:Name>
(29)      </InitialValues>
(30)    <Property name="myns:GridProperty" modifiability="read-write"
                    mutability="mutable" />
(31)   </MetadataDescriptor>
(32)  </Definitions>
```

To provide the factory that will produce instances of the managed resource, a WsResourceFactory WSDL file (listed at Appendix E) must be included in the resource directory. This WSDL is very simple, consisting of a single service made up of a GridResourceFactory port definition with no operation, and a binding. When it is accessed, it produces a new GridResource.

These files provide all of the information necessary for the Muse wsdl2java function (with the –axis –j2ee options) to generate the appropriate Web services and stub files for the Grid resource. A single, "flattened" WSDL file, with all referenced schema entries imported, is generated for each resource.

The resulting services.xml (Appendix F) contains entries for all components of the managed resource; GridResource01, WsResourceFactory and the Subscription Manager. The file identifies the handleRequest operation that allows messages to pass through the Muse isolation layer, and in the case of the GridResource01 resource, lists the requests that are exposed by the resource.

The muse.xml file (shown at Appendix G) identifies the router class used, provides options for logging, as well as the option for router persistence. There are separate entries for each resource type that specify the concrete java class to be used to for each capability, as well as the URI which accesses the capability.

A proxy must be generated to allow access to the capability. This is done by using the wsdl2java –proxy option. The resulting proxy is placed in the library directory under WEB-INF.

The appropriate MyCapability.java file (Annex H) which contains the business logic for the simulated resource is placed in the source directory and the SManRes01 WAR file is generated. The operation of the simulated manageable resource once deployed is discussed in subsequent sections.

5.3 Autonomic Manager Generation

The initiation of the Autonomic Manager follows a similar pattern, but does not involve a resource factory. The key element in the manager from a WSDM perspective is the ability to subscribe to notifications from the resource and the ability to receive the notifications. The WSDL file for the manager therefore contains the Notify operation:

Table 7 - Autonomic Manager WSDL Notify Operation

```
(01)  <wsdl:operation name="Notify">
(02)    <wsdl:input wsa:Action=
               "http://docs.oasis-open.org/wsn/bw-2/
               NotificationConsumer/NotifyRequest"
               name="Notify" message="tns:Notify" />
(03)  </wsdl:operation>
```

The muse.xml file describes the NotificationConsumer capability and the services.xml file has an entry for NotifyRequest to enable the notification process.

Once the wsdl2java actions are completed, and the MyCapability.java file is put in place, an AManager WAR file is generated.

5.4 Initiation of the Touchpoint Simulator

As shown in Figure 13, the three components of the simulator are the autonomic manager, the resource factory, and the manageable resource. Access to these components

is gained through a client application (Appendix I), which calls and acts on the deployed components.

When the client is started, the Autonomic Manager is deployed as a Web service accessible at:

http://localhost:8080/AManager/services/ACManager

Deployment of the Manageable Resource is a two-part process. First, the Factory is automatically generated and deployed. The second part is to "ping" the Factory (using the factoryClient.getEndpointReference() command) to activate the endpoint and initiate the creation of the Managed Resource. The Managed Resource is available at:

http://localhost:8080/SManRes01/services/GridResource01

Once the resources are deployed, WSRF properties are examined by calling getResourcePropertyDocument(). A portion of the returned value includes:

Table 8 - getResourcePropertiesDocumentResponse Message Excerpt

```
(01)  <wsrf-rp:GetResourcePropertyDocumentResponse
        xmlns:tns="http://axis2.platform.core.muse.apache.org"
        xmlns:wsrf-rp="http://docs.oasis-open.org/wsrf/rp-2">
(02)    <tns:GridResource01Properties
           xmlns:tns="http://vres.hlinux.thshome/muse/touch/impl">
            …
(03)       <tns:ResourceType>
(04)           http://vres.hlinux.thshome/muse/touch/impl/GridResource
(05)       </tns:ResourceType>
(06)       <tns:Name>GridResource01</tns:Name>
            …
(07)       <muws1:ResourceId
             xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd">
(08)           uuid:29ba1204-ba14-385b-4c98-b832f36154bc
(09)       </muws1:ResourceId>
            …
(10)       <muws2:OperationalStatus
             xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd">
(11)         Available
(12)       </muws2:OperationalStatus>
            …
(13)       <muws1:ManageabilityCapability
             xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd">
```

```
(14)          http://docs.oasis-open.org/wsdm/muws/
                                          capabilities/OperationalStatus
(15)        </muws1:ManageabilityCapability>
             …
(16)     </tns:GridResource01Properties>
(17)  </wsrf-rp:GetResourcePropertyDocumentResponse>
```

The first two lines identify that this is a Response message for GridResource01. Lines 03-06 identify the ResourceType and Name that were initialized through the .rmd file. Lines 07-09 provide the ResourceId, used to uniquely identify the resource within the WSRF framework. Lines 10-12 display the OperationalStatus – it was changed to "Available" within the capability itself. Lines 13-15 confirm that OperationalStatus is in fact included as a ManageabilityCapability.

Eligible properties – those noted as being read-write and mutable in the .rmd – are able to be updated. A value can be written to the "GridProperty" property using setResourceProperty() and the updated value is returned when a query is made.

The final step in the initiation is the subscription of the autonomic manager to the simulated manageable resource. Besides the endpoint reference address of the consumer, there are two arguments used. A termination time, which indicates when WS-ResourceLifetime is to destroy the resource, may be entered. The last argument describes the TopicFilter, used to delineate which topic the consumer (the autonomic manager) wishes to receive from the producer (the manageable resource). The resources are at this point properly deployed, and we will now examine the operation of each of the resources.

5.5 Simulated Manageable Resource Operation

In addition to the typical getter and setter methods for public variables**,** the simulated manageable resource also provides for the initialisation of desired WSRP properties (as noted above when the value of OperationalStatus was returned as "Available"). The principal function of the simulated resource is to call the Generic Log

Adapter to parse the resource logs, transform the results to the WEF format, and then have the results published through WS-Notification.

5.5.1 Generic Log Adapter

The Generic Log Adapter (GLA) is initialized with an Operating System file sensor (SingleOSFileSensor) to read the log file generated by Globus, a Regular Expression parser to capture the events listed in the log, a Common Base Event formatter to translate these events into CBEs, and finally a CBE File Outputter.

The Regular Expression parser is set up to capture the situations listed in section 2.6.1. In the scenario described earlier, these correspond to:

- Start Situation – start of Globus container, and of Reliable File Transfer;
- Stop Situation – stop of the Reliable File Transfer;
- Connect Situation – connection to SOAP server;
- Request Situation – request of security token for delegated credentials;
- Available Situation – initiation of Reliable File Transfer service;
- Create Situation – creation of job; and
- Destroy Situation – destruction of job, and delegated credentials.

The Common Base Events are then transformed into WSDM Event Format (WEF) messages which are transmitted from the simulated resource to the autonomic manager. To view the process, we will follow the thread of a log entry through the process. The first log entry on starting the Globus container reads:

```
ts=2009-12-28 09:18:32,627 level=INFO  thread=main sbcomp=ServiceContainer
event=org.globus.container.start
```

Upon being transformed by the Generic Log Adapter, the following is provided:

Table 9 - Common Base Event (CBE) Message

```
(01)  <CommonBaseEvent creationTime="2009-11-28T14:18:32.627Z"
            globalInstanceId="A1DEF43E63BCC980F2BFD730E65A5FE6"
            msg="org.globus.container.start"
            severity="1"
            version="1.0.1">

(02)      <sourceComponentId component="GT4.2.1"
            componentIdType="Product Name"
            location="192.168.1.212" locationType="IPV4"
            subComponent="ServiceContainer"
            threadId="main" componentType="Globus" />

(03)      <msgDataElement>
(04)            <msgIdType>Event</msgIdType>
(05)      </msgDataElement>

(06)      <situation categoryName="StartSituation">
(07)        <situationType
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="StartSituation"
               reasoningScope="INTERNAL"
               successDisposition="SUCCESSFUL"
               situationQualifier="START COMPLETED" />
(08)      </situation>
(09)  </CommonBaseEvent>
```

Note that the original information from the log is presented using the standard naming convention of CBEs (lines 01 and 02). ts is now creationTime, thread is threadId, sbcomp is SubComponent, and the event is reflected in the msg field (line 01). The level information, originally “INFO”, is transformed into a severity level of 1, on a scale of 0-10 (line 01).

Other data are also provided by the adapter: the CommonBaseEvent section includes globalInstanceId (line 01) which is a unique identifier for the event. The sourceComponent section (line 02) identifies the (source) component id and type, as well as the location and type.

The heart of the GLA output is the situationType section (line 07). It can be seen that the event was correctly interpreted as a StartSituation. It was noted that the reasoningScope was internal and that the Start was successfully completed.

The contextual information to determine situationType is derived primarily from the "msg" field in the CBE root element (line 01). This is populated by default from the "msg" field in the Globus log. In this example, there is no "msg" field, so the "event" field is used instead. If there is neither field, the log entry is typically a free text message (with no designator such as msg= or event=), which will be imported. The GLA uses a series of substitution rules, written as regular expressions, to interpret the information garnered from the logs. The following three regular expressions are used to test if a log event describes a StartSituation:

Table 10 - StartSituation Regular Expressions

Regular Expression:	StartSituation entry:
• \w{0,25}.){0,6}.<init>.start$	START INITIATED
• \w{0,25}.){0,6}.start.start$	START INITIATED
• \w{0,25}.){0,6}.<init>.stop$	START COMPLETED - SUCCESSFUL
• \w{0,25}.){0,6}.start.stop$	START COMPLETED - SUCCESSFUL
• \w{0,25}.){0,6}.container.start$	START COMPLETED - SUCCESSFUL

Each of the other Situations has a similar series of tests to determine whether a log entry meets the criteria for that Situation. The testing is conducted in series (Start, Feature, Configure, Stop, Connect, and so forth), and the testing is completed as soon as a match is found; thus, a log entry can only be described as a single situation. If a log entry does not meet any other test, it is described as a ReportSituation – STATUS.

5.5.2 WEF Transformation

The transformation from CBE to WEF is carried out using XSLT. The template used can be found at Appendix D. Once the WEF transformation is completed, the following message is produced:

Table 11 - WSDM Event Format (WEF) Message

```
(01)  <muws-p1:ManagementEvent xmlns:muws-p1="http://docs.oasis-
         open.org/wsdm/2004/12/muws/wsdm-muws-part1.xsd"
         ReportTime="2009-11-28T14:18:32.627Z" Severity="1">
(02)     <muws-p1:EventId>
            A1DEF43E63BCC980F2BFD730E65A5FE6
         </muws-p1:EventId>

(03)     <muws-p1:SourceComponent>
(04)        <muws-p1:ComponentAddress>
(05)           <wsa:Address xmlns:wsa=
                  "http://schemas.xmlsoap.org/ws/2004/08/addressing">
                     http://192.168.1.212:8080/
                     SManRes01/services/GridResource01
               </wsa:Address>
(06)           <wsa:ReferenceProperties xmlns:wsa=
                  "http://schemas.xmlsoap.org/ws/2004/08/addressing">
(07)              <deploy:ResourceIdentifier xmlns:deploy=
                     "http://vres.hlinux.thshome/globus/grid/vserver">
                     uuid:983g7eri-2nog-7bs2-0258-8nq3hp76v253
                  </deploy:ResourceIdentifier>
(08)           </wsa:ReferenceProperties>
(09)        </muws-p1:ComponentAddress>

(10)        <muws-p1:ResourceId>
(11)              uuid:29ba1204-ba14-385b-4c98-b832f36154bc
(12)        </muws-p1:ResourceId>
(13)     </muws-p1:SourceComponent>

(14)     <ns1:sourceComponentId xmlns:ns1=
            "http://www.ibm.com/AC/commonbaseevent1_0_1"
            component="GT4.2.1" componentIdType="Product Name"
            componentType="Globus"
            subComponent="ServiceContainer"
            location="192.168.1.212" locationType="IPV4"
            threadId="main"/>
```

```
(15)      <ns2:msgDataElement xmlns:ns2=
             "http://www.ibm.com/AC/commonbaseevent1_0_1">
             <msgDataElement>
(16)            <msg>org.globus.container.start</msg>
(17)            <msgIdType>Event</msgIdType>
(18)         </msgDataElement>
(19)      </ns2:msgDataElement>

(20)      <ns3:situation xmlns:ns3=
             "http://www.ibm.com/AC/commonbaseevent1_0_1"
             categoryName="StartSituation">
(21)         <ns3:situationType
                reasoningScope="INTERNAL"
                successDisposition="SUCCESSFUL"
                situationQualifier="START COMPLETED"/>
             </ns3:situation>
(22)   </muws-p1:ManagementEvent>
```

All of the data from the Common Base Event is retained, and a set of new information points is added. Within the source component section, the component address (line 05), consisting of the End Point Reference (EPR) of the resource, is introduced. Line 07 displays the resource identifier for the physical resource (in this case the grid vserver). There is a second resource identifier (ResourceId) at line 10 which uniquely identifies the deployed resource. Rather than the physical resource, this identifies the service being run. It is recommended [OAS06a] that this identifier be consistent across manageability endpoints (multiple endpoints which provide access to the same service should have the same ResourceId) as well as in time (power recycling should have no effect on the Id).

5.5.3 Notification Publication

When the WEF transformation is completed, the WEF events are returned to the simulated manageable resource as a collection. Each event is removed in turn and prepared to be published. Exactly what is published, and when, is determined by options set by the developer.

The first option concerns the timing of the publication. The WEFs may be published in a continuous batch, or events may be transmitted at the interval specified in the log file. For example, if there was a 2.3 second delay between two events, the simulator will pause to allow that amount of time to pass between the publication of the events.

The second option concerns the selection of events to be published. The default is to publish all events. Alternatively, the developer can set a specific subset of situationType to be published. For example, the simulator can be set to publish only Start and Stop situations. When this option is chosen, the Topic Name is modified to signify that that only selected WEFs are published.

5.5.4 Autonomic Manager Operation

When the autonomic manger is initialized, the topic name is set to receive either all WEFs or a particular subset, as described above. The manager is then set as a Message Listener, ready to accept messages from that topic.

When a message is received, it is checked against the topic. If the topic does not match, no further action is taken. If it does match, the WEF is parsed into its constituent parts, ready to be used by the developer.

5.6 Scenario Examination

We now examine the operation of the touchpoint simulator in the context of the scenario developed in Section 4.2. The tables below highlight the messages pertinent to the scenario. The table header indicates the fields being displayed.

Table 12 shows the messages initially received. The successful start of the container and SOAP server are noted, as is the availability of Reliable File Transfer and

the request of a security token. These have been captured as Start, Available, Connect, and Request situations, respectively.

Table 12 - Start Up WEFs

00	**Report Time – Severity – subComponent – msgDataElement** **Situation – disposition - qualifier**
01	2009-11-28T17:34:22.181-05:00 – 1 – ServiceContainer - org.globus.container.start **StartSituation** – START COMPLETED - SUCCESSFUL
02	2009-11-28T17:34:32.249-05:00 – 1 – ReliableFileTransferImpl - org.globus.transfer.reliable.service.ReliableFileTransferImpl.<init>.stop **AvailableSituation** – AVAILABLE
03	2009-11-28T17:34:37.613-05:00 – 1 – ServiceContainer - Starting SOAP server" address=https://192.168.1.212:8443/wsrf/services/ **ConnectSituation** – SUCCESSFUL
04	2009-11-28T17:35:12.210-05:00 – 1 – DelegationFactoryService - org.globus.delegation.requestToken.start **RequestSituation** – REQUEST INITIATED
05	2009-11-28T17:35:12.548-05:00 – 1 - DelegationFactoryService - org.globus.delegation.requestToken.end **RequestSituation** – REQUEST COMPLETED – SUCCESSFUL

Table 13 indicates that the initial job (69471670…) was created, and its job submitted. Subsequently, two jobs (**6a363160 and 6a7bc5e0**) were created; these correspond to the two portions of the GRAM multiJob described in the scenario. The job creations were properly annotated, and the submission was noted as a Report Situation.

Table 13 - Job Creation WEFs

00	**Report Time – Severity – subComponent – msgDataElement** **Situation – disposition - qualifier**
01	2009-11-28T 17:35:13.417-05:00 – 1 – AdminLogger - org.globus.execution.job.create.end - jobResource.id= **69471670**-02ef-11df-951b-8e37a56df417 **CreateSituation** - SUCCESSFUL
02	2009-11-28T 17:35:13.627-05:00 – 1 – AdminLogger - org.globus.execution.job.submission.start - jobResource.id= **69471670**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
03	2009-11-28T 17:35:14.850-05:00 – 1 – AdminLogger - org.globus.execution.job.create.end - jobResource.id= **6a363160**-02ef-11df-951b-8e37a56df417 **CreateSituation** - SUCCESSFUL
04	2009-11-28T 17:35:15.252-05:00 – 1 – AdminLogger - org.globus.execution.job.create.end - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **CreateSituation** - SUCCESSFUL

The next series simply captures a number of job actions as Report Situations. Note that the starting and stopping of Reliable File Transfer to transport the files for the StageIn operation were captured. The log event for the Stop (wsrf.destroy.end) had to be linked with the sub component (ReliableFileTransferService) to ensure it was annotated correctly.

Table 14 - Processing and Transfer WEFs – Job 6a363160

00	**Report Time – Severity – subComponent – msgDataElement** **Situation – disposition - qualifier**
01	2009-11-28T17:35:15.342-05:00 – 1 – AdminLogger - org.globus.execution.job.submission.end - subJob.ids= **6a363160**-02ef-11df-951b-8e37a56df417 --- **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
02	2009-11-28T17:35:15.575-05:00 – 1 – AdminLogger - org.globus.execution.job.processing.start - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
03	2009-11-28T 17:35:15.586-05:00 – 1 – AdminLogger - org.globus.execution.job.submission.start - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
04	2009-11-28T 17:35:15.732-05:00 – 1 – AdminLogger - org.globus.execution.job.processing.start – jobResource.id= **6a363160**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
05	2009-11-28T 17:35:15.742-05:00 – 1 – AdminLogger - org.globus.execution.job.stageIn.start – jobResource.id= **6a363160**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
06	2009-11-28T17:35:17.280-05:00 – 1 – ReliableFileTransferImpl – org.globus.transfer.reliable.service.ReliableFileTransferImpl.start.stop **StartSituation** - START COMPLETED – SUCCESSFUL
07	2009-11-28T17:35:20.165-05:00 – 1 – DestroyProvider – org.globus.wsrf.destroy.end service=ReliableFileTransferService **StopSituation** - STOP COMPLETED - SUCCESSFUL
08	2009-11-28T 17:35:20.168-05:00 – 1 – AdminLogger - org.globus.execution.job.stageIn.end – jobResource.id= **6a363160**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
09	2009-11-28T 17:35:20.183-05:00 – 1 – AdminLogger - org.globus.execution.job.submission.start – jobResource.id= **6a363160**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
10	2009-11-28T 17:35:21.430-05:00 – 1 – AdminLogger - org.globus.execution.job.submission.end – jobResource.id= **6a363160**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS

Table 15 shows the series of messages related to the second job (6a7bc5e0) of staging out the files and returning them to the originating server; again, the start and stop of the file transfer is correctly annotated.

Table 15 - Processing and Transfer WEFs – Job 6a7bc5e0

00	**Report Time – Severity – subComponent – msgDataElement** **Situation – disposition - qualifier**
01	2009-11-28T 17:35:22.813-05:00 – 1 – AdminLogger - org.globus.execution.job.stageOut.start - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
02	2009-11-28T17:35:24.325-05:00 – 1 – ReliableFileTransferImpl – org.globus.transfer.reliable.service.ReliableFileTransferImpl.start.stop **StartSituation** - START COMPLETED – SUCCESSFUL
03	2009-11-28T17:35:26.801-05:00 – 1 – DestroyProvider – org.globus.wsrf.destroy.end service=ReliableFileTransferService **StopSituation** - STOP COMPLETED - SUCCESSFUL
04	2009-11-28T 17:35:26.804-05:00 – 1 – AdminLogger - org.globus.execution.job.stageOut.end - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
05	2009-11-28T 17:35:26.893-05:00 – 1 – AdminLogger - org.globus.execution.job.fileCleanUp.start - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
06	2009-11-28T 17:35:28.224-05:00 – 1 – AdminLogger - org.globus.execution.job.fileCleanUp.start - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
07	2009-11-28T17:35:29.078-05:00 – 1 – AdminLogger - org.globus.execution.job.processing.end - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS

At the end of the processing sequence, the jobs are destroyed, as is the Delegation Service.

Table 16 - Completion Sequence WEFs

00	**Report Time – Severity – subComponent – msgDataElement** **Situation – disposition - qualifier**
01	2009-11-28T17:35:29.183-05:00 – 1 – AdminLogger - org.globus.execution.job.processing.end - jobResource.id= **69471670**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS
03	2009-11-28T17:35:34.152-05:00 – 1 – AdminLogger - org.globus.execution.job. terminate.end - jobResource.id= **6a363160**-02ef-11df-951b-8e37a56df417 **DestroySituation** – SUCCESSFUL
04	2009-11-28T17:35:34.514-05:00 – 1 – AdminLogger - org.globus.execution.job. terminate.end - jobResource.id= **6a7bc5e0**-02ef-11df-951b-8e37a56df417 **DestroySituation** – SUCCESSFUL
05	2009-11-28T17:35:33.838-05:00 – 1 – AdminLogger - org.globus.execution.job.terminate.end – jobResource.id= **69471670**-02ef-11df-951b-8e37a56df417 **DestroySituation** – SUCCESSFUL
06	2009-11-28T17:35:35.839-05:00 – 1 – AdminLogger - org.globus.wsrf.destroy.end – service=DelegationService **DestroySituation** – SUCCESSFUL

Throughout the scenario, the events were captured as expected, providing the developer with valuable cues to significant actions taking place in a routine deployment. To further explore the capabilities, the same scenario was run, but the GridFTP server on the destination server was not started, so the file transfer and subsequent processing could not occur. The significant events captured included:

Table 17 - Interrupted Transfer Sequence WEFs

00	**Report Time – Severity – subComponent – msgDataElement** **Situation – disposition - qualifier**
01	2009-11-28T 17:36:46.687-05:00 – 1 – AdminLogger - org.globus.execution.job.stageIn.start – jobResource.id= **a0e166d0**-02ef-11df-951b-8e37a56df417 **ReportSituation** – STATUS
02	2009-11-28T17:36:48.136-05:00 – 1 – ReliableFileTransferImpl – org.globus.transfer.reliable.service.ReliableFileTransferImpl.start.stop **StartSituation** - START COMPLETED – SUCCESSFUL
03	2009-11-28T17:35:48.514-05:00 – 4 – ConnectionManager - Can't create connection: java.net.ConnectException: Connection refused **ReportSituation** - STATUS
04	2009-11-28T17:35:48.517-05:00 – 4 – TransferWork - Transient transfer error **ReportSituation** - STATUS
05	2009-11-28T17:36:49.046-05:00 – 1 – DestroyProvider – org.globus.wsrf.destroy.end service=ReliableFileTransferService **StopSituation** - STOP COMPLETED - SUCCESSFUL
06	2009-11-28T17:36:49.841-05:00 – 4 – AdminLogger - Fault #1: Description: Staging error for RSL element fileStageIn. Connection creation error [Caused by: Connection refused] Cause: org.globus.exec.generated.StagingFaultType: Staging error for RSL element fileStageIn. Connection creation error [Caused by: Connection refused]- jobResource.id= **a0e166d0**-02ef-11df-951b-8e37a56df417 **ReportSituation** - STATUS

It should be noted that the Severity is reported as 4 – the "error" level – for the appropriate events; the refused connection, the transfer error and the description of the fault.

To demonstrate the ability of the touchpoint simulator to reproduce a more realistic grid environment, multiple manageable resources were deployed and initiated. The autonomic manager correctly received and displayed events from all resources.

5.7 Summary

This chapter described the design and implementation of the model for touchpoint simulation of grid services. A simulated manageable resource and a simulated autonomic controller are generated using Muse with the capabilities and properties described in the appropriate WSDL and resource property files. The touchpoint simulator resources are compliant with the WSDM and WSRF standards. The manageable resource correctly reads the Globus-generated input log, and converts the events contained therein using the generic log adapter to CBEs, then to WSDM Event Format messages. The messages are dispatched to the autonomic controller, based on the topic selected, and at the appropriate time interval. The output messages accurately reflect the events in the reference scenario and categorize the events in appropriate Situations, both when there is a single resource and when multiple resources are deployed, and are correctly annotated in fault situations.

Chapter 6 Evaluation

The aim of the touchpoint simulation model is to provide developers with a platform that can be used to aid in the development of autonomic managers for grid services. In order to assess the feasibility of our approach, we examine three areas:

- Compliance with the Web services and touchpoint specifications;
- Comparison to related work; and
- Accuracy of information provided.

6.1 Compliance with specifications

The principal specifications which apply to this work are WSDM, WSRF, Common Base Events and the manageability interface requirements of a touchpoint.

WSDM specifies how the manageability of a resource is made available to manageability consumers via Web services. WSDM MUWS Parts 1 & 2 [OAS06a, OAS06b] describe the mandatory and optional capabilities and properties, as well as the messaging format for communication between the components. The touchpoint simulator implements the Identity capability through the ResourceId property, the Description capability through the inclusion of the Caption, Description and Version properties, and the Manageability Characteristics capability, providing the Manageability Capabilities property, which lists all of the capabilities provided by the simulated manageable resource. These capabilities and properties were all demonstrated in Section 5.4, when the Resource Property Document for the resource was called and returned. A number of the capabilities, such as State, Configuration, and Metrics were not built into the implementation of the simulated managed resource, as that information is passed implicitly in the messages transmitted by the simulator.

MUWS Part 1 also describes WSDM Event Format (WEF) messages. Closely related is the Common Base Event standard [IBM06b], which expands on most of the

WEF fields. The simulator implements all of the required elements: EventId, SourceComponent, Situation, Category, as well as ReportTime. Within the Component element, the component, componentType, componentIdType, location and locationType information is provided. The simulator implements the various Situation types (Start, Stop, Create, Report, etc) to include the situation time, message, msgId, severity and success disposition. The priority field was not implemented as the Globus logs do not consistently include the information required to make a proper determination.

WSRF focuses on Resource Properties, which allow information to be stored and retrieved from instances of Web services resources. The touchpoint simulator utilises resource metadata descriptor documents (.rmd extension) to provide the definition and initialisation of properties for both deployed resources. The WSRF capabilities (GetResourceProperty, GetMultipleResourceProperties, GetResourcePropertyDocument, and SetResourceProperty) are used to set and retrieve the values of the properties. WS-ResourceLifetime is implemented (SetTerminationTime), allowing control of the lifecycle of the resources and WS-BaseFaults messages are incorporated in the resource wsdl files. The relative simple nature of the touchpoint simulator implementation did not warrant the use of the WS-ServiceGroup capabilities.

The touchpoint simulator complies with WS-BaseNotification to allow the autonomic manager to subscribe to notifications from the simulated manageable resource.

The required interfaces delineated for touchpoints [IBM05] closely mirror the specifications discussed above (Identity, Description, Manageability Characteristics, Get & SetResourceProperty) and are all implemented in the simulator, as are a number of the optional interfaces (QueryResourceProperties, GetMultipleResourceProperties, and OperationalStatus). ResourceType, the only unique touchpoint interface, is implemented.

The mandatory communication interaction pattern (Request-Response) is implemented, as are two others – PerfomOperation and Send-Notification.

6.2 Comparison to Related Work

The previous attempts at providing support for autonomic software development were discussed in Chapter 3. This section compares those attempts and technologies with the results obtained by our model of touchpoint simulation, and notes the advantages and limitations.

The original IBM AIDE framework automated the construction of manageability endpoints which conformed to the WSDM specifications and provided tools to allow the user to access and manipulate the state of a resource. However, the introduction of proprietary code and artifacts into the implementation limited its ability to be ported to other environments. The use of Apache Muse allows our model to be exported as a .war file that can be implemented in numerous environments.

The Test and Performance Tools Platform (TPTP) eclipse project integrated Apache Muse into its approach, overcoming the earlier AIDE limitation. It provided a capability to automatically generate some of the files and stubs required for WSDM-compliant manageability endpoints, but was limited in its support of Notification Producer, a critical component in the working of the simulator. Our approach allows the generation of all of the files and stubs required for the development of the manageable resources needed to implement the touchpoint simulator.

As was noted in Chapter 3, the COSMOS eclipse project shows great promise for use in the development of high-level management for systems, but is currently unable to support WSDM, and is therefore not able to be used to support Globus grid services autonomic manager development.

The use of virtualization for the simulation of manageable resources was discussed in the introduction to this thesis, and it was noted that its utility was limited by the relatively small number of resources that could be simulated in a research environment. Our model provides much more robust capabilities.

Two scenarios were explored to determine the limits of the simulation. To permit the maximal use of resources, the processing of log files by the GLA parser was conducted independently, producing .xml files containing Common Base Events that represent the events in the original logs. These files were used as input for the WEF Transform function and subsequent transmission through the simulated manageable resource.

The first scenario involved the generation of multiple versions of the same resource, where each of the simulated resources read the same CBE file. After conversion of the CBEs to WEFs, DOM is used to modify the location and component address fields, effectively simulating output from a multitude of remote machines.

The second scenario generated a number of individual simulated resources, each reading its own CBE file. This output effectively simulated a wide range of remote machines, each providing its own unique output.

Local resource utilisation is of primary concern when assessing the effectiveness of the touchpoint simulator. The scenarios above were run using CBE files which contained 200 events. The initial scenario, simulating 10 resources, had an initial peak of 60% CPU usage, during the CBE-WEF transformation and resource generation, and sustained usage of less than 35% during the transmission of notifications (cf. Figure 14).

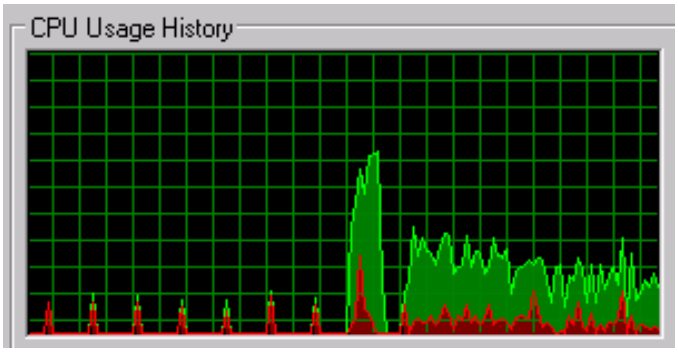

Figure 14 - CPU Usage of 10 Simulated Resources

When the number of simulated resources was raised to 500, the initial peak increased to 70% during the CBE-WEF transformation, with a plateau of 40-50% usage

during the generation of resources, and a sustained rate of approximately 35% during the transmission of notifications (cf. Figure 15).

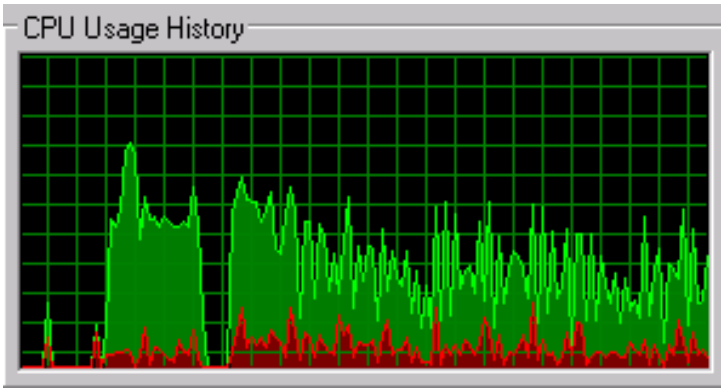

Figure 15 - CPU Usage of 500 Simulated Resources

In the second scenario, when two unique simulated resources were generated, there was an initial peak of 50% CPU usage, then sustained usage of just over 10% (cf. Figure 16).

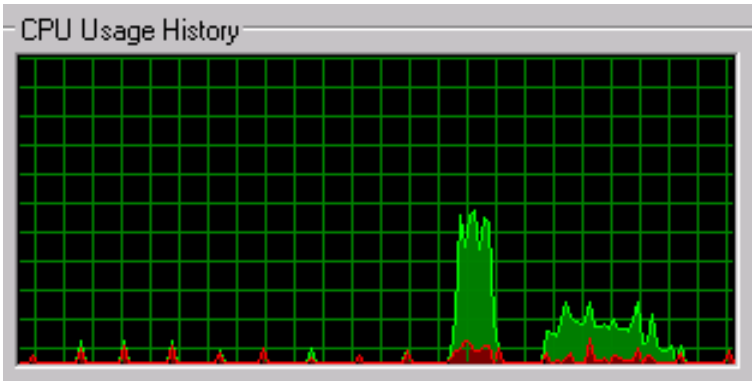

Figure 16 - CPU Usage of 2 Unique Simulated Resources

When the number of unique simulated resources was increased to 10, the CPU usage increased to an initial peak of 60% and more sustained usage of approximately 40% (cf. Figure 17).

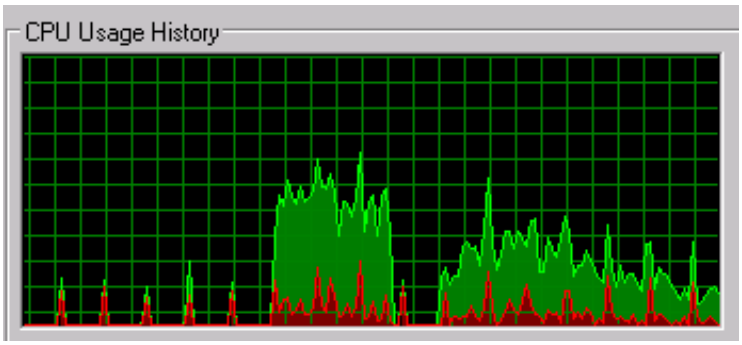

Figure 17 - CPU Usage of 10 Unique Simulated Resources

6.3 Accuracy

The examination undertaken in Chapter 5 noted the information captured in the various stages of the scenario. The touchpoint simulator successfully interpreted the various events expected in the scenario and categorized them accurately. The table below gives a summary of the results.

Table 18 - Summary of Processed Actions

Categorization Situation	Event	Reference Table / Line
Start	Start of Globus container	13 / 01
Start	Start of Reliable File Transfer	15 /06 16 / 02
Available	Initiation of Reliable File Transfer Service	13 / 02
Connect	Connection of SOAP server	13 / 03
Request	Request of Security Token	13 / 04 13 / 05
Create	Creation of job	14 / 01 14 / 03 14 / 04
Stop	Stop of Reliable File Transfer	15 / 07 16 / 03
Destroy	Destruction of Job	17 / 03 17 / 04 17 / 05
Destroy	Destruction of DelegatedCredentials	17 / 06

The events in which errors were reported were given the appropriate severity level, allowing them to be singled out. The remaining events in the scenario were categorized as Report Situations, giving them visibility, but not flagging them as noteworthy events.

The categorization of the principal events in Table 18 provides developers with invaluable cues with respect to the information being passed. In addition, the simulator also passes the original messages from the logs, providing the developer with a complete package of information with which to proceed with the development of autonomic managers, in keeping with the research observations noted earlier on the importance of the collection of full and accurate information from resources.

6.4 Summary

The touchpoint simulator provides valuable input to a developer wishing to build and implement an autonomic manager for grid services. The model complies with the WSDM and WSRF specifications, and the manageability interface requirements of a touchpoint. The messages output from the simulator meet the requirements of Common Base Events, as well as the WSDM Event Format specification.

The simulator builds on earlier work, but improves portability by not employing proprietary code, nor specialized packaging of components. It has better scalability than the simple virtualization of resources, as well as reduced complexity and level-of-effort compared to building and deploying multiple resources to be deployed either through grid virtualization or in the current incarnation of Cloud computing. The output from the simulator accurately categorizes the events transmitted and providing the raw information that may be used in the development of autonomic managers.

Chapter 7 Conclusions

7.1 Research Summary

Advances in computing have allowed users unprecedented access to computing resources. In addition to the growth in the raw power of individual machines, grid computing has brought new and innovative methods of sharing computing tasks amongst a multitude of users. These advances, however, have come at the price of a tremendous increase in complexity in the use and management of decentralized, distributed systems. Autonomic computing offers a method to tame this complexity, but requires the ability for developers to fully test autonomic managers for grid solutions.

This thesis presents a model that can be used to simulate the output of Globus grid services to allow the output to be used in the development of autonomic managers for grid management. Work included the development of a small scale grid to generate realistic log files and to provide a reference example against which the output of the simulator could be measured. The model provides a simulated managed resource and autonomic manager which comply with agreed industry standards for autonomic and grid computing, as well as for Web services. The simulator transforms the log input into WSDM Event Format messages which provide accurate and detailed reporting of events taking place in the Globus environment.

The simulator can be tuned to allow it to pass all events, or a designated subset. It ensures that the events are received at the same interval as they were original generated, while providing an option to receive them in bulk.

It is robust enough to simulate multiple resources. On a typical laptop computer, for simulations of both similar and unique resources, CPU usage was reasonable and allowed greater resource simulation than the use of local virtual resources, at a much reduced level of effort than configuring individual resources in a test grid environment.

.

7.2 Contributions

The application of autonomic computing methods to the field of grid computing is an ongoing area of research and study. The research for this thesis focused on the ability to provide the accurate input required for the development of autonomic managers for grid management. The specific contributions include:

- Determination and description of standards applicable to autonomic and grid computing;
- Determination of methods to provide grid input for the development of autonomic grid managers;
- Development of Generic Log Adapter settings for the conversion of Globus grid logs to Common Base Events;
- Design and implementation of a WSRF-compliant simulator which provides realistic Globus grid input at appropriate intervals, and allowing the selection of specific types of information to be passed.

Taken together, these contributions can act as a blueprint and framework to arm future developers with the knowledge and tools necessary to proceed with the development of autonomic mangers for grid services.

.

7.3 Future Work

The current touchpoint simulator takes the information harvested from the input logs and converts it to WSDM Event Format messages for use by the autonomic manager. The "state" of the simulated manageable resource and state transitions are passed to the manager in the text of the WEF messages. The next step in the development of the simulator is to more fully develop and utilize the WSRF capabilities and properties of the simulated resource. The simulated resource is currently WSRF-

compliant in that it will return the values of all properties when it receives a getResourcePropertyDocument call, or will update OperationalStatus when SetResourceProperty is used. The next iteration of the simulator could, for example, automatically update OperationalStatus when a "Container Start" message is parsed from the input log, and trigger the transmission of a WEF message based on this change in state.

Another area of future development is the visualization and modification of the input logs. The logs currently are read, transformed and transmitted with no intervention from the developer. The ability to tailor the logs would be advantageous in circumstances when a particular input is desired, for example, when checking boundaries and the responses to non-typical events.

A close watch need be kept on the evolution of standards and industry practices. The simulator is currently designed to conform to the OASIS-approved WSDM and WSRF specifications, but there is often fluidity in accepted standards. As a case in point, the WS-ResourceTransfer specification [W3C09c], which combines elements of WSRF and the Distributed Management Task Force (DMTF)-approved WS-Management specification was accepted as a Member Submission by the W3C in 2006 and updated to a Working Draft in September 2009. Similar to WSRF, WS-Management defined a SOAP-based protocol for the management of servers, devices, applications and more. The standards used in the simulator will obviously have to evolve as industry standards evolve.

Bibliography

[ACKL02] D.P. Anderson, J. Cobb, E. Korpola, M. Lebofsky, D. Werthimer, "SETI@home: an experiment in public-resource computing," *Communications of the ACM*, Vol. 5, Issue 11, pp. 56-61, November 2002.

[AM09] Apache MUSE 2.2.0 Website. http://ws.apache.org/muse/docs/2.2.0/tutorial/index.html

[BDG04] M. Bote-Lorenzo, Y. Dimitriadis, and E. Gomez-Sanchez, "Grid Characteristics and Uses: a Grid Definition," *Proceedings First European Across Grids Conference (ACG'03),* Springer-Verlag LNCS 2970, Santiago de Compostela, Spain, pp. 291-298, February 2004. http://ulises.tel.uva.es/uploaded_files/BoteACG03.pdf

[dC09] dCache Website. http://www.dcache.org/

[DM07] R. Desmarais, H. Müller, "A Proposal for an Autonomic Grid Management System," *Proceedings ICSE 2007 Workshops: International Workshop on Software Engineering for Adaptive and Self-Managing Systems*, (SEAMS 2007), Minneapolis, Minnesota, USA, pp. 11-17, 2007.

[DMTF09] Distributed Management Task Force (DMTF) Standard, "Configuration Management Database Federation Specification," 22 June 2009. http://www.dmtf.org/standards/published_documents/DSP0252_1.0.0.pdf

[Du09] DutchGrid – Large-scale Distributed Computing in the Netherlands Website. http://www.dutchgrid.nl/

[Ec07] Eclipse Web site, TPTP 4.4.0 Managed Agent Installation Guide, 2007. http://www.eclipse.org/tptp/monitoring/documents/tutorials/tptp_wsdm_setup_4.4.html

[Ec08] Eclipse Web site, Bugzilla entry for TPTP Build-to-Manage toolkit 4.4.0, 2008. https://bugs.eclipse.org/bugs/show_bug.cgi?id=191337

[Ec09] Eclipse COSMOS Project Web page, 2009. http://www.eclipse.org/cosmos/

[Econ01] The Economist, "Computing Power on Tap," 23 Jun 2001. http://www.economist.com/science/tq/displayStory.cfm?Story_id=662301

[EGEE09] Enabling Grids for E-sciencE Website, 2009. http://www.eu-egee.org/

[FBAK03] L. Ferreira, V. Berstis, J. Armstrong, M. Kendzierski, A. Neukoetter, M. Takagi, "Introduction to Grid Computing with Globus," IBM Redbook, September 2003. www.redbooks.ibm.com/redbooks/pdfs/sg246895.pdf

[FFGT04] I. Foster, J. Frey, S. Graham, S. Tuecke, K. Czajkowski, D. Ferguson, "Modeling Stateful Resources with Web Services," Version 1.1, 2004. http://www.ibm.com/developerworks/library/ws-resource/ws-modelingresources.pdf

[FKNT02a] I. Foster, C. Kesselman, J. Nick, S. Tueke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," The Globus Alliance, 2002. www.globus.org/alliance/publications/papers/ogsa.pdf

[FKNT02b] I. Foster, C. Kesselman, J. Nick, S. Tuecke, "Grid Services for Distributed System Integration," *IEEE Computer*, Vol. 35, Iss. 6, pp. 37-46, June 2002.

[Fo02] Foster, I, "What is the Grid? A Three Point Checklist," 20 July 2002. http://www.mcs.anl.gov/~itf/Articles/WhatIsTheGrid.pdf

[Fo06] I. Foster, "Globus Toolkit Version 4: Software for Service-Oriented Systems," *Journal of Computer Science and Technology*, Science Press, Vol. 21, No. 4, pp. 513-520, July 2006.

[FZRL08] I. Foster, Y. Zhao, I. Raicu, S. Lu, "Cloud Computing and Grid Computing 360-Degree Compared," *International Conference for High Performance Computing, Networking, Storage and Analysis (SC08) - Grid Computing Environments Workshop*, Austin, TX, 16 November 2008.

[GaCo03] A.G. Ganek and T.A. Corbi, "The Dawning of the Autonomic Computing Era," *IBM Systems Journal*, Volume 42, No 1, pp. 5-18, 2003.

[GGF05] Global Grid Forum, "The Open Grid Services Architecture, version 1.0," 29 January 2005. https://forge.gridforum.org/sf/go/doc13515?nav=1

[Gl09a] The Globus Toolkit Web site, The Globus Alliance, 2009. http://www.globus.org/

[Gl09b] Globus Execution Management: Key Concepts Web site, The Globus Alliance, 2009. http://www.globus.org/toolkit/docs/4.2/4.2.1/execution/key/#executionKey

[GLA09] IBM Corporation, Stand-alone Generic Log Adapter v4.2 User's Guide. http://eclipse.org/tptp/home/downloads/installguide/gla_42/gla_users_guide.html

[GN07] P. Garbacki, V. Naik, "Efficient Resource Virtualization and Sharing Strategies for Heterogeneous Grid Environments," *Proceedings IEEE Symposium on Integrated Network Management,* pp. 40–49, Munich, Germany, 21-25 May 2007.

[Gr09] GridCafe Website, 2009. http://www.gridcafe.org/grid-powered-project.html

[GRR09] C. Germain-Roy, O. Rana, "The Convergence of Clouds, Grids, and Autonomics," *IEEE Journal of Internet Computing,* No 9, p. 9. November/December 2009.

[GSBD02] S. Graham, S. Simeonov, T. Boubez, D. Davis, G. Daniels, Y. Nakamura, R. Neyama, "Building Web Services with Java: Making Sense of XML, SOAP, WDLS, and UDDI," Indianapolis, Indiana, Sams, 2002.

[Ha04] M. Haynos, "Perspectives on grid: Grid Computing – Next-Generation Distributed Computing," IBM developerWorks, 27 January 2004. http://ftp.utcluj.ro/pub/docs/cursuri/tarc/GRID/gr-heritage.pdf

[HPDA05] T. Heinis, C. Pautasso, O. Deak, G. Alonso, "Publishing Persistent Grid Computations as WS Resources", *IEEE Proceedings First International Conference on e-Science and Grid Computing*, e-Science 2005, Melbourne, Australia, pp 328-335, 5-8 December 2005.

[IATK06] IBM Autonomic Toolkit website, 2006. http://www.ibm.com/developerworks/autonomic/overview.html

[IBM05] IBM Corporation, "Manageable Resource Interface Specification and Touchpoint Implementation Guide," Document ACAB.CS0401E, 26 October 2005. http://www.alphaworks.ibm.com/tech/aide/

[IBM06a] IBM Corporation, "An Architectural Blueprint for Autonomic Computing," Fourth Edition, June 2006. http://www-01.ibm.com/software/tivoli/autonomic/pdfs/AC_Blueprint_White_Paper_4th.pdf

[IBM06b] IBM Corporation, "Best Practices for the Common Base Event and Common Event Infrastructure," 2006. http://download.boulder.ibm.com/ibmdl/pub/software/dw/autonomic/books/cbepractice/index.htm

[JPR09] S. Jha, M. Parashar, O. Rana, "Investigating Autonomic Behaviours in Grid-Based Computational Science Applications," *IEEE Proceedings International Conference on Autonomic Computing 2009 Conference: 6th Annual International Conference Industry Session on Grids Meets Autonomic Computing*, Barcelona, Spain, pp. 29-38, 2009.

[KANK09] O. Khalid, R. Anthony, P. Nilsson, K. Keahey, et al, "Enabling and Optimizing Pilot Jobs using Xen based Virtual Machines for the HPC Grid Applications," *IEEE Proceedings International Conference on Autonomic Computing 2009 Conference: 3rd International Workshop on Virtualization Technologies in Distributed Computing*, Barcelona, Spain, pp. 1-8, 2009.

[KCF06] K. Keahey, J. Chase, I. Foster, "Virtual Playgrounds: Managing Virtual Resources in the Grid," *IEEE Proceedings 20th International Parallel and Distributed Processing Symposium,* 2006 (IPDPS 2006), Rhodes Island, Greece, pp. 25-29 April 2006.

[KeCh03] J.O. Kephart and D.M. Chess, "The Vision of Autonomic Computing," *IEEE Computer*, Vol. 36, No. 1, pp. 41-50, January 2003.

[KG09] E.Knorr, G. Gruman, "What Cloud Computing Really Means," InfoWorld Web site, 2009. http://www.infoworld.com/d/cloud-computing/what-cloud-computing-really-means-031?page=0,0

[KPGV03] G. Kaiser, J. Parekh, P. Gross, G. Valetto, "Kinesthetics eXtreme: An External Infrastructure for Monitoring Distributed Legacy Systems," *IEEE Proceedings Autonomic Computing Workshop, Fifth Annual International Workshop on Active Middleware Services*, (AMS 2003), Washington, DC, pp. 22, 2003.

[Kr05] H. Kreger, "A Little Wisdom about WSDM," IBM developerWorks, 11 March 2005. http://www-128.ibm.com/developerworks/autonomic/library/ws-wisdom/index.html

[KW04] J. Kephart, W. Walsh, "An artificial intelligence perspective on autonomic computing policies," *IEEE Proceedings 5th International Workshop on Policies for Distributed Systems and Networks*, pp. 3–12, Yorktown Heights, NY, 7-9 June 2004.

[LMSI08] M. Litoiu, M. Mihaesca, B. Solomon, D. Ionescu, "Scalable adaptive web services," *Proceedings 2nd International Workshop on System Development in SOA Environments*, (SDSOA 08), Leipzig, Germany, pp. 47-52, 11 May 2008.

[LPH04] H. Lui, M. Parashar, S. Hariri, "A Component Based Programming Framework for Autonomic Applications," *IEEE Proceedings First International Conference on Autonomic Computing*, (ICAC 2004), pp. 10-17, New York, NY, May 2004.

[Ma05] Marsh, P, "Get on the Grid," *Computing and Engineering Journal*, Vol. 16, Iss. 1, pp. 41-45, February / March 2005.

[MKS09] H. Műller, H. Kienle, and U. Stege, "Autonomic Computing Now You See It, Now You Don't, Design and Evolution of Autonomic Software Systems," *Lecture Notes in Computer Science, International Summer Schools, ISSSE 2006-2008*, LNCS 5413, pp. 32-54, 2009.

[Mi05a] B. Miller, "The Autonomic Computing Edge: Keeping in Touch with Touchpoints," IBM developerWorks, 2 August 2005. http://www.ibm.com/developerworks/autonomic/library/ac-edge5/

[MM04] B. Melcher, B. Mitchell, "Towards an Autonomic Framework: Self-Configuring Network Services and Developing Autonomic Applications," *Intel Technical Journal,* Vol. 8, Iss. 4, pp. 279–290. http://download.intel.com/technology/itj/2004/volume08issue04/art03_autonomic/vol8_art03.pdf

[MPWT07] P. Martin, W. Powley, K. Wilson, W. Tian, T. Xu, J. Zebedee, "The WSDM of Autonomic Computing: Experiences in Implementing Autonomic Web Services," *IEEE Proceedings International Conference on Software Engineering 2007 Workshops: International Workshop on Software Engineering for Adaptive and Self-Managing Systems*, (SEAMS 2007), Minneapolis, Minnesota, USA, pp. 9-18 May 26-27 2007

[Na09] L. Navarro, "Managing democratic grids: Architecture and Lessons Learnt," *IEEE Proceedings International Conference on Autonomic Computing 2009 Conference: 6th Annual International Conference Industry Session on Grids Meets Autonomic Computing*, Barcelona, Spain, pp. 51-52, 15 June 2009.

[NEES09] Network for Earthquake Engineering Simulation Website, 2009. https://www.nees.org/it/

[OAS06a] OASIS Standard, "Web Services Distributed Management: Management Using Web Services (MUWS 1.1) Part 1," 1 August 2006. http://docs.oasis-open.org/wsdm/wsdm-muws1-1.1-spec-os-01.htm

[OAS06b] OASIS Standard, "Web Services Distributed Management: Management Using Web Services (MUWS 1.1) Part 2," 1 August 2006. http://docs.oasis-open.org/wsdm/wsdm-muws2-1.1-spec-os-01.htm

[OAS06c] OASIS Standard, "Web Services Distributed Management: Management of Web Services (WSDM-MOWS) 1.1," 1 August 2006. http://docs.oasis-open.org/wsdm/wsdm-mows-1.1-spec-os-01.pdf

[OAS06d] OASIS Standard, "Web Services Resource 1.2 (WS-Resource)," 1 April 2006. http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf

[OKSC03] D. Ogle, H. Kreger, A. Salahshour, J. Cornpropst, "Canonical Situation Data Format: The Common Base Event v1.0.1," IBM Corporation, 4 November 2003. http://www.eclipse.org/tptp/platform/documents/resources/cbe101spec/CommonBaseEvent_SituationData_V1.0.1.pdf

[RHML07] A. Ribio-Montero, E. Huedo, R. Montero, I. Llorente, "Management of Virtual Machines on Globus Grids using GridWay," *IEEE Proceedings 21st International Parallel and Distributed Processing Symposium*, (IPDPS 2007), Long Beach, CA, USA, March 2007.

[St07] Stocklinger, H, "Defining the Grid: A Snapshot on the Current View," *The Journal of Supercomputing*, Vol. 42, No. 1, pp. 3-17, October 2007.

[SWCG07] M. Steinder, I. Whalley, D. Carrera, I. Gaweda, D. Chess. "Server Virtualization in Autonomic Management of Heterogeneous Workloads," *IFIP/IEEE Proceedings 10th International Symposium on Integrated Network Management,* (IM 2007), Munich, Germany, pp. 139-148, 21-25 May 2007.

[TCWD04] G. Tesauro, D. Chess, W. Walsh, R. Das, et al, "A Multi-Agent Systems Approach to Autonomic Computing," *IEEE Proceedings Third International Joint Conference on Autonomous Agents and Multiagent Systems: International Conference on Autonomous Agents*, (AAMAS), New York, Vol. 1, pp 464-471, 19-23 July 2004.

[VMCL09] L. Vaquero, L. Rodero-Merino, J. Caceres, M. Lindner, "A Break in the Clouds: Towards a Cloud Definition," *ACM SIGCOMM Computer Communication Review*, Vol. 39, No. 1, January 2009.

[Vo03] W. Vogels, "Web Services are Not Distributed Objects," *IEEE Internet Computing*, Vol. 7, Iss. 6, pp. 59-66, November 2003.

[W3C01] W3C Note, "Web Services description Language (WSDL) 1.1," 15 March 2001. http://www.w3.org/TR/wsdl

[W3C04] W3C Working Group, "Web Services Architecture," 11 February 2004. http://www.w3.org/TR/ws-arch/

[W3C09a] W3C Recommendation, "Service Modeling Language, Version 1.1," 12 May 2009. http://www.w3.org/TR/sml/

[W3C09b] W3C Recommendation, "Service Modeling Language Interface Format, Version 1.1," 12 May 2009. http://www.w3.org/TR/sml-if/

[W3C09c] W3C Working Draft, "Web Services Resource Transfer (WS-RT)," 24 September 2009. http://www.w3.org/TR/ws-resource-transfer/

[We05] V. Welch, (Editor), "Globus Toolkit Version 4 Grid Security Infrastructure: A Standards Perspective," Version 4, 12 September 2005. http://www.globus.org/toolkit/docs/4.2/4.2.1/security/GT4-GSI-Overview.pdf

[UnHa03] J. Unger, M. Haynos, "A Visual Tour of Open Grid Services Architecture," IBM developerWorks, August 2003. http://www.ibm.com/developerworks/autonomic/library/gr-visual.pdf

Appendix A – VM Configuration and Start Up

In order to determine what output could be provided from Globus GT4 a three node grid was established. Below are the specifics of the VM names, IP addresses, a list of applications, and users in each VM.

- VM 1 – vm103a.hlinux.thshome – 192.168.1.211
  - Applications
    - Globus GT4 (GRAM, RFT, GridFTP)
    - simpleCA (Certificate Authority)
    - postgreSQL
    - openPBS
  - Users
    - gt4user – client running GT4 services
    - globus – runs GT4 container
    - postgres – runs PostgreSQL
    - root – runs GridFTP and openPBS
- VM 2 – vm103b.hlinux.thshome – 192.168.1.212
  - Applications
    - Globus GT4 (GRAM, RFT, GridFTP)
    - simpleCA (CA user)
    - postgresSQL
    - openPBS
  - Users
    - gt4user – delegated user for gridFTP
    - globus – runs GT4 container
    - postgres – runs PostgreSQL
    - root – runs GridFTP and openPBS

- VM 3 – vm103c.hlinux.thshome – 192.168.1.213
  - Applications
    - Globus GT4 (GRAM, RFT, GridFTP)
    - simpleCA (CA user)
    - postgresSQL
    - openPBS
  - Users
    - gt4user – delegated user for gridFTP
    - globus – runs GT4 container
    - postgres – runs PostgreSQL
    - root – runs GridFTP and openPBS

In order to activate all of the applications necessary to demonstrate the Grid infrastructure, the following actions must be performed:

User	*Command*
Root	pbs_server
Root	pbs_mom
Root	pbs_sched
Postgres	postgres
gt4user	grid-proxy-init
Globus	globus-start-container
Root	globus-gridftp-server -p 2811 -S

Appendix B – Globus MultiJob (Audio Conversion)

The Globus multiJob file below is submitted to initiate the jobs required in the thesis scenario. The first job is the staging in of the required file, and the conversion using lame to .wav format. The second job converts the file to mp3, stages in back to the client, and cleans up residual files.

```
(01)  <?xml version="1.0" encoding="UTF-8"?>
(02)  <multiJob xmlns:wsa="http://www.w3.org/2005/08/addressing">
(03)     <factoryEndpoint>
(04)           <wsa:Address>
(05)                 https://vm103b.hlinux.thshome:8443/
                        wsrf/services/ManagedJobFactoryService
(06)           </wsa:Address>
(07)     </factoryEndpoint>
(08)     <directory>${GLOBUS_USER_HOME}</directory>
(09)     <count>1</count>
(10)     <job>
(11)        <factoryEndpoint>
(12)           <wsa:Address>
(13)              https://vm103b.hlinux.thshome:8443/
                        wsrf/services/ManagedJobFactoryService
(14)           </wsa:Address>
(15)        </factoryEndpoint>
(16)        <executable>/usr/bin/mplayer</executable>
(17)        <argument>-vo</argument>
(18)        <argument>null</argument>
(19)        <argument>-vc</argument>
(20)        <argument>dummy</argument>
(21)        <argument>-af</argument>
(22)        <argument>resample=44100</argument>
(23)        <argument>-ao</argument>
(24)        <argument>pcm:waveheader</argument>
(25)        <argument>bjdontcry.wma</argument>
(26)        <fileStageIn>
(27)           <transfer>
(28)              <sourceUrl>
(29)                 gsiftp://vm103a.hlinux.thshome/
                           home/gt4user/AudioToGo/bjdontcry.wma
(30)              </sourceUrl>
(31)              <destinationUrl>
(32)                 gsiftp://vm103b.hlinux.thshome/
                     home/gt4user/bjdontcry.wma
(33)              </destinationUrl>
(34)           </transfer>
(35)        </fileStageIn>
(36)     </job>
(37)     <job>
(38)        <factoryEndpoint>
(39)           <wsa:Address>https://vm103b.hlinux.thshome:8443/
                  wsrf/services/ManagedJobFactoryService
(40)           </wsa:Address>
```

```
(41)         </factoryEndpoint>
(42)         <executable>/usr/bin/lame</executable>
(43)         <argument>-m</argument>
(44)         <argument>s</argument>
(45)         <argument>audiodump.wav</argument>
(46)         <argument>bjdontcry.mp3</argument>
(47)         <stdout>bjdontcry.wma.stdout</stdout>
(48)         <stderr>bjdontcry.wma.stderr</stderr>
(49)         <fileStageOut>
(50)            <transfer>
(51)               <sourceUrl>
(52)                  gsiftp://vm103b.hlinux.thshome/
                         home/gt4user/bjdontcry.mp3
(53)               </sourceUrl>
(54)               <destinationUrl>
(55)                  gsiftp://vm103a.hlinux.thshome/
                         home/gt4user/bjdontcry.mp3
(56)               </destinationUrl>
(57)            </transfer>
(58)            <transfer>
(59)               <sourceUrl>
(60)                  gsiftp://vm103b.hlinux.thshome/
                         home/gt4user/bjdontcry.wma.stdout
(61)               </sourceUrl>
(62)               <destinationUrl>
(63)                  gsiftp://vm103a.hlinux.thshome/
                         home/gt4user/AudioToGo/bjdontcry.wma.stdout
(64)               </destinationUrl>
(65)            </transfer>
(66)            <transfer>
(67)               <sourceUrl>
(68)                  gsiftp://vm103b.hlinux.thshome/
                         home/gt4user/bjdontcry.wma.stderr
(69)               </sourceUrl>
(70)               <destinationUrl>
(71)                  gsiftp://vm103a.hlinux.thshome/
                         home/gt4user/AudioToGo/bjdontcry.wma.stderr
(72)               </destinationUrl>
(73)            </transfer>
(74)         </fileStageOut>
(75)      </job>
(76)   </multiJob>
```

Appendix C – GridResource01 WSDL

The WSDL file is used to generate the simulated managed resource (GridResource01). It contains the definition of the required namespaces, schemas, messages, operations, portTypes and ports, bindings and services.

```
(01) <?xml version="1.0" encoding="UTF-8"?>
(02) <wsdl:definitions targetNamespace="http://vres.hlinux.thshome/muse/touch/impl"
(03) xmlns:tns="http://vres.hlinux.thshome/muse/touch/impl"
(04) xmlns="http://schemas.xmlsoap.org/wsdl/"
(05) xmlns:wsa="http://www.w3.org/2005/08/addressing"
(06) xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
(07) xmlns:wsdl-soap="http://schemas.xmlsoap.org/wsdl/soap/"
(08) xmlns:xsd="http://www.w3.org/2001/XMLSchema"
(09) xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
(10) xmlns:wsrf-r="http://docs.oasis-open.org/wsrf/r-2"
(11) xmlns:wsrf-rl="http://docs.oasis-open.org/wsrf/rl-2"
(12) xmlns:wsrf-bf="http://docs.oasis-open.org/wsrf/bf-2"
(13) xmlns:wsrf-rp="http://docs.oasis-open.org/wsrf/rp-2"
(14) xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2"
(15) xmlns:wsntw="http://docs.oasis-open.org/wsn/bw-2"
(16) xmlns:wst="http://docs.oasis-open.org/wsn/t-1"
(17) xmlns:wsrmd="http://docs.oasis-open.org/wsrf/rmd-1"
(18) xmlns:muws1="http://docs.oasis-open.org/wsdm/muws1-2.xsd"
(19) xmlns:muws2="http://docs.oasis-open.org/wsdm/muws2-2.xsd"
(20) name="GridResource01">
(21)    <wsdl:types>
(22)       <xsd:schema elementFormDefault="qualified" targetNamespace="http://www.w3.org/2005/08/addressing">
(23)          <xsd:include schemaLocation="WS-Addressing-2005_08.xsd" />
(24)       </xsd:schema>
(25)       <xsd:schema elementFormDefault="qualified" targetNamespace="http://schemas.xmlsoap.org/ws/2004/09/mex">
(26)          <xsd:include schemaLocation="WS-MetadataExchange-2004_09.xsd" />
(27)       </xsd:schema>
(28)       <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsrf/rl-2">
(29)          <xsd:include schemaLocation="WS-ResourceLifetime-1_2.xsd" />
(30)       </xsd:schema>
(31)       <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsrf/rp-2">
(32)          <xsd:include schemaLocation="WS-ResourceProperties-1_2.xsd" />
(33)       </xsd:schema>
(34)       <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsrf/r-2">
(35)          <xsd:include schemaLocation="WS-Resource-1_2.xsd" />
(36)       </xsd:schema>
(37)       <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsrf/rmd-1">
(38)          <xsd:include schemaLocation="WS-ResourceMetadataDescriptor-CD-01.xsd" />
(39)       </xsd:schema>
(40)       <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsdm/muws1-2.xsd">
(41)          <xsd:include schemaLocation="WSDM-MUWS-Part1-1_1.xsd" />
(42)       </xsd:schema>
(43)       <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsdm/muws2-2.xsd">
(44)          <xsd:include schemaLocation="WSDM-MUWS-Part2-1_1.xsd" />
(45)       </xsd:schema>
```

```
(46)    <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsn/b-2">
(47)      <xsd:include schemaLocation="WS-BaseNotification-1_3.xsd" />
(48)    </xsd:schema>
(49)    <xsd:schema elementFormDefault="qualified" targetNamespace="http://docs.oasis-open.org/wsn/t-1">
(50)      <xsd:include schemaLocation="WS-Topics-1_3.xsd" />
(51)    </xsd:schema>
(52)    <xsd:schema elementFormDefault="qualified" targetNamespace="http://vres.hlinux.thshome/muse/touch/impl">
(53)      <xsd:element name="ResourceType" type="xsd:string" />
(54)      <xsd:element name="Name" type="xsd:string" />
(55)      <xsd:element name="GridProperty" type="xsd:string" />
(56)      <xsd:element name="GridResource01Properties">
(57)        <xsd:complexType>
(58)          <xsd:sequence>
(59)            <xsd:element ref="wsrf-rl:CurrentTime" />
(60)            <xsd:element ref="wsrf-rl:TerminationTime" />
(61)            <xsd:element ref="wsrf-rp:QueryExpressionDialect" minOccurs="0" maxOccurs="unbounded" />
(62)            <xsd:element ref="wsnt:FixedTopicSet" />
(63)            <xsd:element ref="wst:TopicSet" minOccurs="0" />
(64)            <xsd:element ref="wsnt:TopicExpression" minOccurs="0" maxOccurs="unbounded" />
(65)            <xsd:element ref="wsnt:TopicExpressionDialect" minOccurs="0" maxOccurs="unbounded" />
(66)            <xsd:element ref="muws1:ResourceId" />
(67)            <xsd:element ref="muws1:ManageabilityCapability" minOccurs="0" maxOccurs="unbounded" />
(68)            <xsd:element ref="muws2:Caption" minOccurs="0" maxOccurs="unbounded" />
(69)            <xsd:element ref="muws2:Description" minOccurs="0" maxOccurs="unbounded" />
(70)            <xsd:element ref="muws2:Version" minOccurs="0" />
(71)            <xsd:element ref="muws2:OperationalStatus" />
(72)            <xsd:element ref="tns:ResourceType" />
(73)            <xsd:element ref="tns:Name" />
(74)            <xsd:element ref="tns:GridProperty" />
(75)          </xsd:sequence>
(76)        </xsd:complexType>
(77)      </xsd:element>
(78)    </xsd:schema>
(79)  </wsdl:types>
(80)  <wsdl:message name="GetMetadataMsg">
(81)    <wsdl:part name="GetMetadataMsg" element="wsx:GetMetadata" />
(82)  </wsdl:message>
(83)  <wsdl:message name="GetMetadataResponseMsg">
(84)    <wsdl:part name="GetMetadataResponseMsg" element="wsx:Metadata" />
```

(Message entries deleted for brevity)

```
(85)  <wsdl:message name="NoCurrentMessageOnTopicFault">
(86)    <wsdl:part name="NoCurrentMessageOnTopicFault" element="wsnt:NoCurrentMessageOnTopicFault" />
(87)  </wsdl:message>
(88)  <wsdl:portType name="GridResource01PortType" wsrf-rp:ResourceProperties="tns:GridResource01Properties"
                wsrmd:Descriptor="GridResource01Metadata" wsrmd:DescriptorLocation="GridResource01.rmd">
(89)    <wsdl:operation name="GetMetadata">
(90)      <wsdl:input wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata" name="GetMetadataMsg"
                    message="tns:GetMetadataMsg" />
(91)      <wsdl:output wsa:Action="http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadataResponse"
                    name="GetMetadataResponseMsg" message="tns:GetMetadataResponseMsg" />
(92)    </wsdl:operation>
(93)    <wsdl:operation name="Destroy">
(94)      <wsdl:input wsa:Action="http://docs.oasis-open.org/wsrf/rlw-2/ImmediateResourceTermination/DestroyRequest"
                    name="DestroyRequest" message="tns:DestroyRequest" />
```

```
(95)        <wsdl:output wsa:Action="http://docs.oasis-open.org/wsrf/rlw-2/
                          ImmediateResourceTermination/DestroyResponse"
                          name="DestroyResponse" message="tns:DestroyResponse" />
(96)        <wsdl:fault name="ResourceNotDestroyedFault" message="tns:ResourceNotDestroyedFault" />
(97)        <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(98)        <wsdl:fault name="ResourceUnavailableFault" message="tns:ResourceUnavailableFault" />
(99)      </wsdl:operation>
(100)       <wsdl:operation name="SetTerminationTime">
(101)         <wsdl:input wsa:Action="http://docs.oasis-open.org/wsrf/rlw-2/
                          ScheduledResourceTermination/SetTerminationTimeRequest"
                          name="SetTerminationTimeRequest" message="tns:SetTerminationTimeRequest" />
(102)         <wsdl:output wsa:Action="http://docs.oasis-open.org/wsrf/rlw-2/
                          ScheduledResourceTermination/SetTerminationTimeResponse"
                          name="SetTerminationTimeResponse" message="tns:SetTerminationTimeResponse" />
(103)         <wsdl:fault name="UnableToSetTerminationTimeFault" message="tns:UnableToSetTerminationTimeFault"
/>
(104)         <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(105)         <wsdl:fault name="ResourceUnavailableFault" message="tns:ResourceUnavailableFault" />
(106)         <wsdl:fault name="TerminationTimeChangeRejectedFault"
                          message="tns:TerminationTimeChangeRejectedFault" />
(107)       </wsdl:operation>
(108)       <wsdl:operation name="GetResourcePropertyDocument">
(109)         <wsdl:input wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                          GetResourcePropertyDocument/GetResourcePropertyDocumentRequest"
                          name="GetResourcePropertyDocumentRequest"
                          message="tns:GetResourcePropertyDocumentRequest" />
(110)         <wsdl:output wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/GetResourcePropertyDocument/
                          GetResourcePropertyDocumentResponse" name="GetResourcePropertyDocumentResponse"
                          message="tns:GetResourcePropertyDocumentResponse" />
(111)         <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(112)         <wsdl:fault name="ResourceUnavailableFault" message="tns:ResourceUnavailableFault" />
(113)       </wsdl:operation>
(114)       <wsdl:operation name="GetResourceProperty">
(115)         <wsdl:input wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                          GetResourceProperty/GetResourcePropertyRequest"
                          name="GetResourcePropertyRequest" message="tns:GetResourcePropertyRequest" />
(116)         <wsdl:output wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                          GetResourceProperty/GetResourcePropertyResponse"
                          name="GetResourcePropertyResponse" message="tns:GetResourcePropertyResponse" />
(117)         <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(118)         <wsdl:fault name="ResourceUnavailableFault" message="tns:ResourceUnavailableFault" />
(119)         <wsdl:fault name="InvalidResourcePropertyQNameFault"
                          message="tns:InvalidResourcePropertyQNameFault" />
(120)       </wsdl:operation>
(121)       <wsdl:operation name="GetMultipleResourceProperties">
(122)         <wsdl:input wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                          GetMultipleResourceProperties/GetMultipleResourcePropertiesRequest"
                          name="GetMultipleResourcePropertiesRequest"
                          message="tns:GetMultipleResourcePropertiesRequest" />
(123)         <wsdl:output wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                          GetMultipleResourceProperties/GetMultipleResourcePropertiesResponse"
                          name="GetMultipleResourcePropertiesResponse"
                          message="tns:GetMultipleResourcePropertiesResponse" />
(124)         <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(125)         <wsdl:fault name="ResourceUnavailableFault" message="tns:ResourceUnavailableFault" />
```

```
(126)       <wsdl:fault name="InvalidResourcePropertyQNameFault"
                    message="tns:InvalidResourcePropertyQNameFault" />
(127)     </wsdl:operation>
(128)     <wsdl:operation name="QueryResourceProperties">
(129)       <wsdl:input wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                    QueryResourceProperties/QueryResourcePropertiesRequest"
                    name="QueryResourcePropertiesRequest" message="tns:QueryResourcePropertiesRequest" />
(130)       <wsdl:output wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                    QueryResourceProperties/QueryResourcePropertiesResponse"
                    name="QueryResourcePropertiesResponse" message="tns:QueryResourcePropertiesResponse" />
(131)       <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(132)       <wsdl:fault name="ResourceUnavailableFault" message="tns:ResourceUnavailableFault" />
(133)       <wsdl:fault name="UnknownQueryExpressionDialectFault"
                     message="tns:UnknownQueryExpressionDialectFault" />
(134)       <wsdl:fault name="InvalidQueryExpressionFault" message="tns:InvalidQueryExpressionFault" />
(135)       <wsdl:fault name="QueryEvaluationErrorFault" message="tns:QueryEvaluationErrorFault" />
(136)     </wsdl:operation>
(137)     <wsdl:operation name="SetResourceProperties">
(138)       <wsdl:input wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                    SetResourceProperties/SetResourcePropertiesRequest"
                    name="SetResourcePropertiesRequest" message="tns:SetResourcePropertiesRequest" />
(139)       <wsdl:output wsa:Action="http://docs.oasis-open.org/wsrf/rpw-2/
                    SetResourceProperties/SetResourcePropertiesResponse"
                    name="SetResourcePropertiesResponse" message="tns:SetResourcePropertiesResponse" />
(140)       <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(141)       <wsdl:fault name="ResourceUnavailableFault" message="tns:ResourceUnavailableFault" />
(142)       <wsdl:fault name="InvalidModificationFault" message="tns:InvalidModificationFault" />
(143)       <wsdl:fault name="UnableToModifyResourcePropertyFault"
                    message="tns:UnableToModifyResourcePropertyFault" />
(144)       <wsdl:fault name="InvalidResourcePropertyQNameFault"
                     message="tns:InvalidResourcePropertyQNameFault" />
(145)       <wsdl:fault name="SetResourcePropertyRequestFailedFault"
                    message="tns:SetResourcePropertyRequestFailedFault" />
(146)     </wsdl:operation>
(147)     <wsdl:operation name="Subscribe">
(148)       <wsdl:input wsa:Action="http://docs.oasis-open.org/wsn/bw-2/NotificationProducer/SubscribeRequest"
                    message="tns:SubscribeRequest" />
(149)       <wsdl:output wsa:Action="http://docs.oasis-open.org/wsn/bw-2/NotificationProducer/SubscribeResponse"
                    message="tns:SubscribeResponse" />
(150)       <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(151)       <wsdl:fault name="InvalidFilterFault" message="tns:InvalidFilterFault" />
(152)       <wsdl:fault name="TopicExpressionDialectUnknownFault"
                    message="tns:TopicExpressionDialectUnknownFault" />
(153)       <wsdl:fault name="InvalidTopicExpressionFault" message="tns:InvalidTopicExpressionFault" />
(154)       <wsdl:fault name="TopicNotSupportedFault" message="tns:TopicNotSupportedFault" />
(155)       <wsdl:fault name="InvalidProducerPropertiesExpressionFault"
                    message="tns:InvalidProducerPropertiesExpressionFault" />
(156)       <wsdl:fault name="InvalidMessageContentExpressionFault"
                    message="tns:InvalidMessageContentExpressionFault" />
(157)       <wsdl:fault name="UnacceptableInitialTerminationTimeFault"
                    message="tns:UnacceptableInitialTerminationTimeFault" />
(158)       <wsdl:fault name="UnrecognizedPolicyRequestFault" message="tns:UnrecognizedPolicyRequestFault" />
(159)       <wsdl:fault name="UnsupportedPolicyRequestFault" message="tns:UnsupportedPolicyRequestFault" />
(160)       <wsdl:fault name="NotifyMessageNotSupportedFault" message="tns:NotifyMessageNotSupportedFault" />
(161)       <wsdl:fault name="SubscribeCreationFailedFault" message="tns:SubscribeCreationFailedFault" />
(162)     </wsdl:operation>
```

```
(163)         <wsdl:operation name="GetCurrentMessage">
(164)           <wsdl:input wsa:Action="http://docs.oasis-open.org/wsn/bw-2/
                        NotificationProducer/GetCurrentMessageRequest"
                        message="tns:GetCurrentMessageRequest" />
(165)           <wsdl:output wsa:Action="http://docs.oasis-open.org/wsn/bw-2/
                        NotificationProducer/GetCurrentMessageResponse"
                        message="tns:GetCurrentMessageResponse" />
(166)           <wsdl:fault name="ResourceUnknownFault" message="tns:ResourceUnknownFault" />
(167)           <wsdl:fault name="TopicExpressionDialectUnknownFault"
                        message="tns:TopicExpressionDialectUnknownFault" />
(168)           <wsdl:fault name="InvalidTopicExpressionFault" message="tns:InvalidTopicExpressionFault" />
(169)           <wsdl:fault name="TopicNotSupportedFault" message="tns:TopicNotSupportedFault" />
(170)           <wsdl:fault name="NoCurrentMessageOnTopicFault" message="tns:NoCurrentMessageOnTopicFault" />
(171)           <wsdl:fault name="MultipleTopicsSpecifiedFault" message="tns:MultipleTopicsSpecifiedFault" />
(172)         </wsdl:operation>
(173)       </wsdl:portType>
(174)       <wsdl:binding name="GridResource01Binding" type="tns:GridResource01PortType">
(175)         <wsdl-soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
(176)         <wsdl:operation name="GetMetadata">
(177)           <wsdl-soap:operation soapAction= "GetMetadata" />
(178)           <wsdl:input>
(179)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(180)           </wsdl:input>
(181)           <wsdl:output>
(182)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(183)           </wsdl:output>
(184)         </wsdl:operation>
(185)         <wsdl:operation name="Destroy">
(186)           <wsdl-soap:operation soapAction="Destroy" />
(187)           <wsdl:input name="DestroyRequest">
(188)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(189)           </wsdl:input>
(190)           <wsdl:output name="DestroyResponse">
(191)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(192)           </wsdl:output>
(193)           <wsdl:fault name="ResourceNotDestroyedFault">
(194)             <wsdl-soap:fault use="encoded" name="ResourceNotDestroyedFault" />
(195)           </wsdl:fault>
(196)           <wsdl:fault name="ResourceUnknownFault">
(197)             <wsdl-soap:fault use="encoded" name="ResourceUnknownFault" />
(198)           </wsdl:fault>
(199)           <wsdl:fault name="ResourceUnavailableFault">
(200)             <wsdl-soap:fault use="encoded" name="ResourceUnavailableFault" />
(201)           </wsdl:fault>
(202)         </wsdl:operation>
(203)         <wsdl:operation name="SetTerminationTime">
(204)           <wsdl-soap:operation soapAction="SetTerminationTime" />
(205)           <wsdl:input name="SetTerminationTimeRequest">
(206)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(207)           </wsdl:input>
(208)           <wsdl:output name="SetTerminationTimeResponse">
(209)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(210)           </wsdl:output>
        (Fault descriptions removed)
(211)         </wsdl:operation>
```

```
(212)         <wsdl:operation name="GetResourcePropertyDocument">
(213)           <wsdl-soap:operation soapAction="GetResourcePropertyDocument" />
(214)           <wsdl:input name="GetResourcePropertyDocumentRequest">
(215)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(216)           </wsdl:input>
(217)           <wsdl:output name="GetResourcePropertyDocumentResponse">
(218)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(219)           </wsdl:output>
      (Fault descriptions removed)
(220)         </wsdl:operation>
(221)         <wsdl:operation name="GetResourceProperty">
(222)           <wsdl-soap:operation soapAction="GetResourceProperty" />
(223)           <wsdl:input name="GetResourcePropertyRequest">
(224)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(225)           </wsdl:input>
(226)           <wsdl:output name="GetResourcePropertyResponse">
(227)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(228)           </wsdl:output>
      (Fault descriptions removed)
(229)         </wsdl:operation>
(230)         <wsdl:operation name="GetMultipleResourceProperties">
(231)           <wsdl-soap:operation soapAction="GetMultipleResourceProperties" />
(232)           <wsdl:input name="GetMultipleResourcePropertiesRequest">
(233)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(234)           </wsdl:input>
(235)           <wsdl:output name="GetMultipleResourcePropertiesResponse">
(236)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(237)           </wsdl:output>
       (Fault descriptions removed)
(238)         </wsdl:operation>
(239)         <wsdl:operation name="QueryResourceProperties">
(240)           <wsdl-soap:operation soapAction="QueryResourceProperties" />
(241)           <wsdl:input name="QueryResourcePropertiesRequest">
(242)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(243)           </wsdl:input>
(244)           <wsdl:output name="QueryResourcePropertiesResponse">
(245)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(246)           </wsdl:output>
       (Fault descriptions removed)
(247)         </wsdl:operation>
(248)         <wsdl:operation name="SetResourceProperties">
(249)           <wsdl-soap:operation soapAction="http://oasis.org/SetResourceProperties" />
(250)           <wsdl:input name="SetResourcePropertiesRequest">
(251)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(252)           </wsdl:input>
(253)           <wsdl:output name="SetResourcePropertiesResponse">
(254)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(255)           </wsdl:output>
     (Fault descriptions removed)
(256)         </wsdl:operation>
(257)         <wsdl:operation name="Subscribe">
(258)           <wsdl-soap:operation soapAction="http://vres.hlinux.thshome/muse/touch/impl/Subscribe" />
(259)           <wsdl:input>
(260)             <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(261)           </wsdl:input>
(262)           <wsdl:output>
```

```
(263)              <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(264)           </wsdl:output>
        (Fault descriptions removed)
(265)        </wsdl:operation>
(266)        <wsdl:operation name="GetCurrentMessage">
(267)           <wsdl-soap:operation soapAction="http://vres.hlinux.thshome/muse/touch/impl/GetCurrentMessage" />
(268)           <wsdl:input>
(269)              <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(270)           </wsdl:input>
(271)           <wsdl:output>
(272)              <wsdl-soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
(273)           </wsdl:output>
        (Fault descriptions removed)
(274)        </wsdl:operation>
(275)     </wsdl:binding>
(276)     <wsdl:service name="GridResource01Service">
(277)        <wsdl:port name="GridResource01Port" binding="tns:GridResource01Binding">
(278)           <wsdl-soap:address location="http://localhost:8080/SManRes01/services/GridResource01" />
(279)        </wsdl:port>
(280)     </wsdl:service>
(281)  </wsdl:definitions>
```

Appendix D – XSLT Transform for WSDM Event

This xslt transformation is used to convert Common Base Event descriptions to WSDM Event Format messages.

```
(01)<?xml version="1.0" encoding="UTF-8"?>
(02)<xsl:stylesheet version="1.0"
(03)xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
(04)xmlns:muws-p1="http://docs.oasis-open.org/wsdm/2004/12/muws/wsdm-muws-part1.xsd"
(05)xmlns:muws-p2="http://docs.oasis-open.org/wsdm/2004/12/muws/wsdm-muws-part2.xsd">
(06)<xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>

(07)    <xsl:template match="CommonBaseEvent">

(08)        <xsl:element name="muws-p1:ManagementEvent"
                    namespace="http://docs.oasis-open.org/wsdm/2004/12/muws/wsdm-muws-part1.xsd">
(09)            <xsl:attribute name="ReportTime"><xsl:value-of select="@creationTime"/></xsl:attribute>
(10)            <xsl:attribute name="Severity"><xsl:value-of select="@severity"/></xsl:attribute>
(11)        <xsl:element name="muws-p1:EventId" ><xsl:value-of select="@globalInstanceId"/></xsl:element>

(12)        <xsl:element name="muws-p1:SourceComponent">
(13)            <xsl:element name="muws-p1:ComponentAddress">
(14)                <xsl:element name="wsa:Address"
                                namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
                                http://192.168.1.212:8080/GLATouchpoint/services/GLATouchpoint
(15)                </xsl:element>
(16)                <xsl:element name="wsa:ReferenceProperties"
                                namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing">
(17)                    <xsl:element name="deploy:ResourceIdentifier"
                                namespace="http://vres.hlinux.thshome/globus/grid/vserver">
                                uuid:983g7eri-2nog-7bs2-0258-8nq3hp76v253
(18)                    </xsl:element>
(19)                </xsl:element>
(20)            </xsl:element>
(21)            <xsl:element name="muws-p1:ResourceId">
(22)                    uuid:29ba1204-ba14-385b-4c98-b832f36154bc
(23)            </xsl:element>

(24)        </xsl:element>
(25)
(26)        <xsl:element name="ns1:sourceComponentId"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                use-attribute-sets="sCompId">
(27)        </xsl:element>

(28)        <xsl:element name="ns2:msgDataElement"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1">
(29)            <xsl:element name="ns2:msgIdType"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1">
(30)                <xsl:value-of select="msgDataElement/msgIdType"/>
(31)            </xsl:element>
```

```
(32)            <xsl:element name="ns2:msgData"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1">
(33)                <xsl:value-of select="msgDataElement/msgIdType"/>
(34)            </xsl:element>
(35)        </xsl:element>

(36)        <xsl:if test="situation/@categoryName = 'StartSituation'">
(37)            <xsl:element name="ns3:situation" use-attribute-sets="situationAttributes"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1" >
(38)                <xsl:element name="ns3:situationType"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                use-attribute-sets="sitStartType">
(39)                </xsl:element>
(40)            </xsl:element>
(41)        </xsl:if>

(42)        <xsl:if test="situation/@categoryName = 'StopSituation'">
(43)            <xsl:element name="ns3:situation" use-attribute-sets="situationAttributes"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1" >
(44)                <xsl:element name="ns3:situationType"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                use-attribute-sets="sitStopType">
(45)                </xsl:element>
(46)            </xsl:element>
(47)        </xsl:if>

(48)        <xsl:if test="situation/@categoryName = 'ReportSituation'">
(49)            <xsl:element name="ns3:situation" use-attribute-sets="situationAttributes"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1" >
(50)                <xsl:element name="ns3:situationType"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                use-attribute-sets="sitRepType">
(51)                </xsl:element>
(52)            </xsl:element>
(53)        </xsl:if>

(54)        <xsl:if test="situation/@categoryName = 'ConfigureSituation'">
(55)            <xsl:element name="ns3:situation" use-attribute-sets="situationAttributes"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1" >
(56)                <xsl:element name="ns3:situationType"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                use-attribute-sets="sitConfigType">
(57)                </xsl:element>
(58)            </xsl:element>
(59)        </xsl:if>

(60)        <xsl:if test="situation/@categoryName = 'RequestSituation'">
(61)            <xsl:element name="ns3:situation" use-attribute-sets="situationAttributes"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1" >
(62)                <xsl:element name="ns3:situationType"
                                namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                use-attribute-sets="sitReqType">
(63)                </xsl:element>
(64)            </xsl:element>
(65)        </xsl:if>
```

```
(66)        <xsl:if test="situation/@categoryName = 'AvailableSituation'">
(67)            <xsl:element name="ns3:situation" use-attribute-sets="situationAttributes"
                                    namespace="http://www.ibm.com/AC/commonbaseevent1_0_1" >
(68)                <xsl:element name="ns3:situationType"
                                    namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                    use-attribute-sets="sitAvailType">
(69)                </xsl:element>
(70)            </xsl:element>
(71)        </xsl:if>

(72)        <xsl:if test="situation/@categoryName = 'CreateSituation'">
(73)            <xsl:element name="ns3:situation" use-attribute-sets="situationAttributes"
                                    namespace="http://www.ibm.com/AC/commonbaseevent1_0_1" >
(74)                <xsl:element name="ns3:situationType"
                                    namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                    use-attribute-sets="sitCreateType">
(75)                </xsl:element>
(76)            </xsl:element>
(77)        </xsl:if>

(78)        <xsl:if test="situation/@categoryName = 'DestroySituation'">
(79)            <xsl:element name="ns3:situation" use-attribute-sets="situationAttributes"
                                    namespace="http://www.ibm.com/AC/commonbaseevent1_0_1" >
(80)                <xsl:element name="ns3:situationType"
                                    namespace="http://www.ibm.com/AC/commonbaseevent1_0_1"
                                    use-attribute-sets="sitDestType">
(81)                </xsl:element>
(82)            </xsl:element>
(83)        </xsl:if>
(84)        </xsl:element>
(85)    </xsl:template>

(86)        <xsl:attribute-set name="sCompId">
(87)            <xsl:attribute name="component">
(88)                <xsl:value-of select="sourceComponentId/@component"/>
(89)            </xsl:attribute>
(90)            <xsl:attribute name="componentIdType">
(91)                <xsl:value-of select="sourceComponentId/@componentIdType"/>
(92)            </xsl:attribute>
(93)            <xsl:attribute name="componentType">
(94)                <xsl:value-of select="sourceComponentId/@componentType"/>
(95)            </xsl:attribute>
(96)            <xsl:attribute name="subComponent">
(97)                <xsl:value-of select="sourceComponentId/@subComponent"/>
(98)            </xsl:attribute>
(99)            <xsl:attribute name="location">
(100)                   <xsl:value-of select="sourceComponentId/@location"/>
(101)               </xsl:attribute>
(102)               <xsl:attribute name="locationType">
(103)                   <xsl:value-of select="sourceComponentId/@locationType"/>
(104)               </xsl:attribute>
(105)               <xsl:attribute name="threadId">
(106)                   <xsl:value-of select="sourceComponentId/@threadId"/>
(107)               </xsl:attribute>
            </xsl:attribute-set>
```

```
(108)   <xsl:attribute-set name="situationAttributes">
(109)       <xsl:attribute name="categoryName">
(110)           <xsl:value-of select="situation/@categoryName"/>
(111)       </xsl:attribute>
(112)   </xsl:attribute-set>

(113)   <xsl:attribute-set name="sitStartType">
(114)       <xsl:attribute name="reasoningScope">
(115)           <xsl:value-of select="situation/situationType/@reasoningScope"/>
(116)       </xsl:attribute>
(117)       <xsl:attribute name="successDisposition">
(118)           <xsl:value-of select="situation/situationType/@successDisposition"/>
(119)       </xsl:attribute>
(120)       <xsl:attribute name="situationQualifier">
(121)           <xsl:value-of select="situation/situationType/@situationQualifier"/>
(122)       </xsl:attribute>
(123)   </xsl:attribute-set>

(124)   <xsl:attribute-set name="sitStopType">
(125)       <xsl:attribute name="reasoningScope">
(126)           <xsl:value-of select="situation/situationType/@reasoningScope"/>
(127)       </xsl:attribute>
(128)       <xsl:attribute name="successDisposition">
(129)           <xsl:value-of select="situation/situationType/@successDisposition"/>
(130)       </xsl:attribute>
(131)       <xsl:attribute name="situationQualifier">
(132)           <xsl:value-of select="situation/situationType/@situationQualifier"/>
(133)       </xsl:attribute>
(134)   </xsl:attribute-set>

(135)   <xsl:attribute-set name="sitRepType">
(136)       <xsl:attribute name="reasoningScope">
(137)           <xsl:value-of select="situation/situationType/@reasoningScope"/>
(138)       </xsl:attribute>
(139)       <xsl:attribute name="reportCategory">
(140)           <xsl:value-of select="situation/situationType/@successDisposition"/>
(141)       </xsl:attribute>
(142)   </xsl:attribute-set>

(143)   <xsl:attribute-set name="sitConfigType">
(144)       <xsl:attribute name="reasoningScope">
(145)           <xsl:value-of select="situation/situationType/@reasoningScope"/>
(146)       </xsl:attribute>
(147)       <xsl:attribute name="successDisposition">
(148)           <xsl:value-of select="situation/situationType/@successDisposition"/>
(149)       </xsl:attribute>
(150)   </xsl:attribute-set>

(151)   <xsl:attribute-set name="sitReqType">
(152)       <xsl:attribute name="reasoningScope">
(153)           <xsl:value-of select="situation/situationType/@reasoningScope"/>
(154)       </xsl:attribute>
(155)       <xsl:attribute name="successDisposition">
(156)           <xsl:value-of select="situation/situationType/@successDisposition"/>
(157)       </xsl:attribute>
(158)       <xsl:attribute name="situationQualifier">
```

```
(159)                    <xsl:value-of select="situation/situationType/@situationQualifier"/>
(160)                </xsl:attribute>
(161)            </xsl:attribute-set>

(162)            <xsl:attribute-set name="sitAvailType">
(163)                <xsl:attribute name="reasoningScope">
(164)                    <xsl:value-of select="situation/situationType/@reasoningScope"/>
(165)                </xsl:attribute>
(166)                <xsl:attribute name="availabilityDisposition">
(167)                    <xsl:value-of select="situation/situationType/@successDisposition"/>
(168)                </xsl:attribute>
(169)            </xsl:attribute-set>

(170)            <xsl:attribute-set name="sitCreateType">
(171)                <xsl:attribute name="reasoningScope">
(172)                    <xsl:value-of select="situation/situationType/@reasoningScope"/>
(173)                </xsl:attribute>
(174)                <xsl:attribute name="availabilityDisposition">
(175)                    <xsl:value-of select="situation/situationType/@successDisposition"/>
(176)                </xsl:attribute>
(177)            </xsl:attribute-set>

(178)            <xsl:attribute-set name="sitDestType">
(179)                <xsl:attribute name="reasoningScope">
(180)                    <xsl:value-of select="situation/situationType/@reasoningScope"/>
(181)                </xsl:attribute>
(182)                <xsl:attribute name="availabilityDisposition">
(183)                    <xsl:value-of select="situation/situationType/@successDisposition"/>
(184)                </xsl:attribute>
(185)            </xsl:attribute-set>

(186)    </xsl:stylesheet>
```

Appendix E – WsResourceFactory WSDL

This WSDL file is used to generate the Resource Factory for the simulated managed resource (Grid Resource01). It contains the definition of the required namespaces, schemas, portTypes and ports, bindings and services.

```
(01)<?xml version="1.0" encoding="UTF-8"?>
(02)<wsdl:definitions targetNamespace="http://vres.hlinux.thshome/muse/touch/impl"
                xmlns:tns="http://vres.hlinux.thshome/muse/touch/impl"
                xmlns="http://schemas.xmlsoap.org/wsdl/"
                xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                xmlns:wsdl-soap="http://schemas.xmlsoap.org/wsdl/soap/"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                name="GridResourceFactory">
(03)      <wsdl:types>
(04)      </wsdl:types>
(05)      <wsdl:portType name="GridResourceFactoryPortType">
(06)      </wsdl:portType>
(07)      <wsdl:binding name="GridResourceFactoryBinding" type="tns:GridResourceFactoryPortType">
(08)           <wsdl-soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
(09)      </wsdl:binding>
(10)      <wsdl:service name="GridResourceFactoryService">
(11)           <wsdl:port name="GridResourceFactoryPort" binding="tns:GridResourceFactoryBinding">
(12)           <wsdl-soap:address location="http://localhost:8080/SManRes01/services/GridResourceFactory" />
(13)           </wsdl:port>
(14)      </wsdl:service>
(15)</wsdl:definitions>
```

Appendix F – GridResource01 Services.xml

This services.xml is generated by Muse when creating the simulated managed resource. The three services required by the managed resource are described, including the URLs of the actions for the services.

```
(01)<?xml version="1.0" encoding="UTF-8"?>
(02)<serviceGroup>
(03)   <service name="WsResourceFactory">
(04)      <parameter locked="false" name="ServiceClass">
(05)            org.apache.muse.core.platform.axis2.AxisIsolationLayer</parameter>
(06)      <operation name="handleRequest">
(07)         <messageReceiver class="org.apache.axis2.receivers.RawXMLINOutMessageReceiver"/>
(08)      </operation>
(09)   </service>

(10)   <service name="GridResource01">
(11)      <parameter locked="false" name="ServiceClass">
(12)            org.apache.muse.core.platform.axis2.AxisIsolationLayer</parameter>
(13)      <operation name="handleRequest">
(14)         <messageReceiver class="org.apache.axis2.receivers.RawXMLINOutMessageReceiver"/>
(15)         <actionMapping>http://docs.oasis-open.org/wsrf/rlw-2/ ImmediateResourceTermination/
(16)            DestroyRequest</actionMapping>
(17)         <actionMapping>http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata</actionMapping>
(18)         <actionMapping>http://docs.oasis-open.org/wsrf/rpw-2/
(19)            SetResourceProperties/SetResourcePropertiesRequest</actionMapping>
(20)         <actionMapping>http://docs.oasis-open.org/wsrf/rpw-2/
(21)            QueryResourceProperties/QueryResourcePropertiesRequest</actionMapping>
(22)         <actionMapping>http://docs.oasis-open.org/wsrf/rlw-2/
(23)            ScheduledResourceTermination/SetTerminationTimeRequest</actionMapping>
(24)         <actionMapping>http://docs.oasis-open.org/wsrf/rpw-2/
(25)            GetMultipleResourceProperties/GetMultipleResourcePropertiesRequest</actionMapping>
(26)         <actionMapping>http://docs.oasis-open.org/wsrf/rpw-2/
(27)            GetResourceProperty/GetResourcePropertyRequest</actionMapping>
(28)         <actionMapping>http://docs.oasis-open.org/wsrf/rpw-2/
(29)            GetResourcePropertyDocument/GetResourcePropertyDocumentRequest</actionMapping>
(30)         <actionMapping>http://docs.oasis-open.org/wsn/bw-2/
(31)            NotificationProducer/SubscribeRequest</actionMapping>
(32)         <actionMapping>http://docs.oasis-open.org/wsn/bw-2/
(33)            NotificationProducer/GetCurrentMessageRequest</actionMapping>
(34)      </operation>
(35)   </service>

(36)   <service name="SubscriptionManager">
(37)      <parameter locked="false" name="ServiceClass">
(38)            org.apache.muse.core.platform.axis2.AxisIsolationLayer</parameter>
(39)      <operation name="handleRequest">
(40)         <messageReceiver class="org.apache.axis2.receivers.RawXMLINOutMessageReceiver"/>
(41)      </operation>
(42)   </service>
(43)</serviceGroup>
```

Appendix G – GridResource01 Muse file

The Muse deployment descriptor is a generated file which contains a description of the three resources (GridResource01, the ResourceFactory and the SubscriptionManager) including the capabilities of each, their URIs, concrete implementation classes, and logging information.

```
(01)<?xml version="1.0" encoding="utf-8"?>
(02)<muse xmlns="http://ws.apache.org/muse/descriptor"
                xmlns:wsrf-sgw="http://docs.oasis-open.org/wsrf/sgw-2"
                xmlns:test="http://vres.hlinux.thshome/muse/touch/impl"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:schemaLocation="http://ws.apache.org/muse/descriptor muse-descriptor.xsd">
(03)    <router>
(04)        <java-router-class>
(05)            org.apache.muse.core.routing.SimpleResourceRouter
(06)        </java-router-class>
(07)        <logging>
(08)            <log-file>log/muse.log</log-file>
(09)            <log-level>FINEST</log-level>
(10)        </logging>
(11)        <persistence>
(12)            <java-persistence-class>
(13)                org.apache.muse.core.routing.RouterFilePersistence
(14)            </java-persistence-class>
(15)            <persistence-location>router-entries</persistence-location>
(16)        </persistence>
(17)    </router>

(18)    <resource-type use-router-persistence="true">
(19)        <context-path>WsResourceFactory</context-path>
(20)        <wsdl>
(21)            <wsdl-file>/wsdl/WsResourceFactory.wsdl</wsdl-file>
(22)            <wsdl-port-type>
(23)                test:WsResourceFactoryPortType
(24)            </wsdl-port-type>
(25)        </wsdl>
(26)        <java-id-factory-class>
(27)            org.apache.muse.core.routing.RandomResourceIdFactory
(28)        </java-id-factory-class>
(29)        <java-resource-class>
(30)            org.apache.muse.core.SimpleResource
(31)        </java-resource-class>
(32)        <capability>
(33)            <capability-uri>
(34)                http://vres.hlinux.thshome/muse/touch/impl/MyFactory
(35)            </capability-uri>
(36)            <java-capability-class>
(37)                thshome.hlinux.vres.muse.touch.impl.MyFactory
(38)            </java-capability-class>
(39)        </capability>
(40)    </resource-type>
```

```
(41)    <resource-type use-router-persistence="true">
(42)        <context-path>GridResource01</context-path>
(43)        <wsdl>
(44)            <wsdl-file>/wsdl/GridResource01.wsdl</wsdl-file>
(45)            <wsdl-port-type
                    xmlns:pfx="http://vres.hlinux.thshome/muse/touch/impl">
(46)                pfx:GridResource01PortType
(47)            </wsdl-port-type>
(48)        </wsdl>
(49)        <java-id-factory-class>
(50)            org.apache.muse.core.routing.CounterResourceIdFactory
(51)        </java-id-factory-class>
(52)        <java-resource-class>
(53)            org.apache.muse.ws.resource.impl.SimpleWsResource
(54)        </java-resource-class>
(55)        <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(56)            <desc:capability-uri>
(57)                http://docs.oasis-open.org/wsrf/rlw-2/ImmediateResourceTermination
(58)            </desc:capability-uri>
(59)            <desc:java-capability-class>
(60)                org.apache.muse.ws.resource.lifetime.impl.SimpleImmediateTermination
(61)            </desc:java-capability-class>
(62)        </desc:capability>
(63)        <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(64)            <desc:capability-uri>
(65)                http://docs.oasis-open.org/wsdm/muws/capabilities/Identity
(66)            </desc:capability-uri>
(67)            <desc:java-capability-class>
(68)                org.apache.muse.ws.dm.muws.impl.SimpleIdentity
(69)            </desc:java-capability-class>
(70)        </desc:capability>
(71)        <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(72)            <desc:capability-uri>
(73)                http://docs.oasis-open.org/wsdm/muws/capabilities/Description
(74)            </desc:capability-uri>
(75)            <desc:java-capability-class>
(76)                org.apache.muse.ws.dm.muws.impl.SimpleDescription
(77)            </desc:java-capability-class>
(78)        </desc:capability>
(79)        <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(80)            <desc:capability-uri>
(81)                http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata
(82)            </desc:capability-uri>
(83)            <desc:java-capability-class>
(84)                org.apache.muse.ws.metadata.impl.SimpleMetadataExchange
(85)            </desc:java-capability-class>
(86)        </desc:capability>
(87)        <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(88)            <desc:capability-uri>
(89)                http://docs.oasis-open.org/wsrf/rpw-2/Set
(90)            </desc:capability-uri>
(91)            <desc:java-capability-class>
(92)                org.apache.muse.ws.resource.properties.set.impl.SimpleSetCapability
(93)            </desc:java-capability-class>
(94)        </desc:capability>
```

```
(95)        <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(96)            <desc:capability-uri>http://docs.oasis-open.org/wsrf/rpw-2/Query</desc:capability-uri>
(97)            <desc:java-capability-class>
(98)                org.apache.muse.ws.resource.properties.query.impl.SimpleQueryCapability
(99)            </desc:java-capability-class>
(100)           </desc:capability>
(101)           <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(102)               <desc:capability-uri>
(103)                   http://docs.oasis-open.org/wsrf/rlw-2/ScheduledResourceTermination
(104)               </desc:capability-uri>
(105)               <desc:java-capability-class>
(106)                   org.apache.muse.ws.resource.lifetime.impl.SimpleScheduledTermination
(107)               </desc:java-capability-class>
(108)           </desc:capability>
(109)           <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(110)               <desc:capability-uri>
(111)                   http://docs.oasis-open.org/wsdm/muws/capabilities/ManageabilityCharacteristics
(112)               </desc:capability-uri>
(113)               <desc:java-capability-class>
(114)                   org.apache.muse.ws.dm.muws.impl.SimpleManageabilityCharacteristics
(115)               </desc:java-capability-class>
(116)           </desc:capability>
(117)           <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(118)               <desc:capability-uri>
(119)                   http://docs.oasis-open.org/wsrf/rpw-2/Get
(120)               </desc:capability-uri>
(121)               <desc:java-capability-class>
(122)                   org.apache.muse.ws.resource.properties.get.impl.SimpleGetCapability
(123)               </desc:java-capability-class>
(124)           </desc:capability>
(125)           <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(126)               <desc:capability-uri>
(127)                   http://docs.oasis-open.org/wsn/bw-2/NotificationProducer
(128)               </desc:capability-uri>
(129)               <desc:java-capability-class>
(130)                   org.apache.muse.ws.notification.impl.SimpleNotificationProducer
(131)               </desc:java-capability-class>
(132)           </desc:capability>
(133)           <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(134)               <desc:capability-uri>
(135)                   http://docs.oasis-open.org/wsdm/muws/capabilities/OperationalStatus
(136)               </desc:capability-uri>
(137)               <desc:java-capability-class>
(138)                   org.apache.muse.ws.dm.muws.impl.SimpleOperationalStatus
(139)               </desc:java-capability-class>
(140)           </desc:capability>
(141)           <desc:capability xmlns:desc="http://ws.apache.org/muse/descriptor">
(142)               <desc:capability-uri>
(143)                   http://vres.hlinux.thshome/muse/touch/impl/MyCapability
(144)               </desc:capability-uri>

(145)               <desc:java-capability-class>
(146)                   thshome.hlinux.vres.muse.touch.impl.MyCapability
(147)               </desc:java-capability-class>
(148)           </desc:capability>
(149)       </resource-type>
```

```
(150)       <resource-type xmlns="http://ws.apache.org/muse/descriptor"
                    xmlns:wsntw="http://docs.oasis-open.org/wsn/bw-2">
(151)           <context-path>SubscriptionManager</context-path>
(152)           <wsdl>
(153)               <wsdl-file>wsdl/SubscriptionManager.wsdl</wsdl-file>
(154)               <wsdl-port-type>wsntw:SubscriptionManager</wsdl-port-type>
(155)           </wsdl>
(156)           <java-id-factory-class>
(157)               org.apache.muse.core.routing.CounterResourceIdFactory
(158)           </java-id-factory-class>
(159)           <java-resource-class>
(160)               org.apache.muse.ws.resource.impl.SimpleWsResource
(161)           </java-resource-class>
(162)           <capability>
(163)               <capability-uri>
(164)                   http://docs.oasis-open.org/wsrf/rpw-2/Get
(165)               </capability-uri>
(166)               <java-capability-class>
(167)                   org.apache.muse.ws.resource.properties.get.impl.SimpleGetCapability
(168)               </java-capability-class>
(169)           </capability>
(170)           <capability>
(171)               <capability-uri>
(172)                   http://docs.oasis-open.org/wsrf/rlw-2/ImmediateResourceTermination
(173)               </capability-uri>
(174)               <java-capability-class>
(175)                   org.apache.muse.ws.resource.lifetime.impl.SimpleImmediateTermination
(176)               </java-capability-class>
(177)           </capability>
(178)           <capability>
(179)               <capability-uri>
(180)                   http://docs.oasis-open.org/wsrf/rlw-2/ScheduledResourceTermination
(181)               </capability-uri>
(182)               <java-capability-class>
(183)                   org.apache.muse.ws.resource.lifetime.impl.SimpleScheduledTermination
(184)               </java-capability-class>
(185)           </capability>
(186)           <capability>
(187)               <capability-uri>
(188)                   http://docs.oasis-open.org/wsn/bw-2/SubscriptionManager
(189)               </capability-uri>
(190)               <java-capability-class>
(191)                   org.apache.muse.ws.notification.impl.SimpleSubscriptionManager
(192)               </java-capability-class>
(193)               <init-param>
(194)                   <param-name>trace-notifications</param-name>
(195)                   <param-value>true</param-value>
(196)               </init-param>
(197)           </capability>
(198)           <init-param>
(199)               <param-name>validate-wsrp-schema</param-name>
(200)               <param-value>false</param-value>
(201)           </init-param>
(202)       </resource-type>
(203)   </muse>
```

Appendix H – MyCapability Code

This java file contains the programming used by the managed resource. When initiated, it starts a new Thread, runs the GLAParser to convert the Globus log to Common Base events, runs the WEFTransform, to transform the CBEs to WEF messages, and ouputs the messages based on options chosen.

```
(01)  package thshome.hlinux.vres.muse.touch.impl;
(02)
(03)  import javax.xml.namespace.QName;
(04)
(05)  import org.w3c.dom.Element;
(06)  import org.w3c.dom.Node;
(07)  import org.w3c.dom.NamedNodeMap;
(08)
(09)  import org.apache.muse.ws.addressing.soap.SoapFault;
(10)  import org.apache.muse.ws.notification.NotificationProducer;
(11)  import org.apache.muse.ws.notification.WsnConstants;
(12)  import org.apache.muse.ws.resource.impl.AbstractWsResourceCapability;
(13)  import org.apache.muse.ws.dm.muws.OperationalStatus;
(14)  import org.apache.muse.ws.dm.muws.MuwsConstants;
(15)  import org.apache.muse.ws.resource.WsResource;
(16)
(17)  import java.util.Date;
(18)  import java.text.DateFormat;
(19)  import java.text.SimpleDateFormat;
(20)  import java.text.ParseException;
(21)
(22)
(23)  public class MyCapability extends AbstractWsResourceCapability implements
                                                                  IMyCapability
(24)  {
(25)
(26)      private static final QName[] _PROPERTIES = new QName[]
(27)      {
(28)          new QName(NAMESPACE_URI, "ResourceType", PREFIX),
(29)          new QName(NAMESPACE_URI, "Name", PREFIX),
(30)          new QName(NAMESPACE_URI, "GridProperty", PREFIX)
(31)      };
(32)
(33)     private QName _TOPIC_NAME = new QName(NAMESPACE_URI, "AllWEFs", PREFIX);


(34)  // public static final QName AVAILABILITY_SITUATION_QNAME;
(35)
(36)      public QName[] getPropertyNames()
(37)      {
(38)          return _PROPERTIES;
(39)      }
```

```
(40)      private String _ResourceType = null;
(41)      private String _Name = null;
(42)      private String _GridProperty = null;
(43)      private Boolean DelayTx = false;
(44)      private Boolean allWEFs = true;
(45)      private String validWEF1 = "StartSituation";
(46)      private String validWEF2 = "StopSituation";
(47)      private Boolean validWEF = false;
(48)      private Date repDate = null;
(49)
(50)     public String getGridProperty()
(51)      {
(52)          return _GridProperty;
(53)      }
(54)
(55)      public void setGridProperty(String param0)
(56)      {
(57)          _GridProperty = param0;
(58)      }
(59)
(60)      public String getName()
(61)      {
(62)          return _Name;
(63)      }
(64)
(65)      public void setName(String param0)
(66)      {
(67)          _Name = param0;
(68)      }
(69)
(70)      public String getResourceType()
(71)      {
(72)          return _ResourceType;
(73)      }
(74)
(75)      public void setResourceType(String param0)
(76)      {
(77)          _ResourceType = param0;
(78)      }
(79)
(80)      public void initializeCompleted()
(81)          throws SoapFault
(82)      {
(83)          //initialise super class
(84)        //
(85)        super.initializeCompleted();
(86)
(87)        //Set Operational Status to Available
(88)        //
(89)          WsResource res = this.getWsResource();
(90)        OperationalStatus stat =
              (OperationalStatus)res.getCapability(MuwsConstants.OP_STATUS_URI);
(91)        stat.setOperationalStatus(OperationalStatus.AVAILABLE);
(92)        this.getLog().info(stat.getOperationalStatus());
(93)
(94)
```

```
(95)          // Access Manageable Resource's WSN capability and
(96)          //add topic based on GridProperty
(97)          //
(98)          if (_GridProperty == "selectedWEFs") {
(99)           _TOPIC_NAME = new QName(NAMESPACE_URI, "selectedWEFs", PREFIX);
(100)          allWEFs = false;
(101)         }
(102)       final NotificationProducer wsn = (NotificationProducer)getResource().
                                    getCapability(WsnConstants.PRODUCER_URI);
(103)         wsn.addTopic(_TOPIC_NAME);
(104)
(105)
(106)         Thread producer = new Thread() {
(107)          Node returnedWEFs = null;
(108)
(109)             public void run()
(110)             {
(111)
(112)                Boolean glaFinished = false;
(113)
(114)                //Run WEFTransform to return WEFs
(115)                //
(116)                try {
(117)                   //Run GLA Parser, checking periodically for completion
(118)                   //
(119)                   GLAParser glaParser = new GLAParser();
(120)                   glaParser.runGLAParser();
(121)                   glaFinished = glaParser.glaCompleted;
(122)                   getLog().info("Entering Awaiting glaFinished");
(123)                   while (!glaFinished){
(124)                      this.sleep(5000);
(125)                      glaFinished = glaParser.glaCompleted;
(126)                      getLog().info("Awaiting glaFinished");
(127)                   }//EndWhile
(128)
(129)                   //Run GlaBound to insert file <CBE> open and close
(130)                   //
(131)                   try {
(132)                      GlaBound boundFile = new GlaBound();
(133)                      boundFile.encase();
(134)                      this.sleep(10000);
(135)
(136)                   }catch (Exception e){
(137)                         System.err.println("Error: " + e.getMessage());
(138)                   }//end catch
(139)
(140)                   //Call WEFTransform
(141)                   //
(142)                   returnedWEFs = WEFTransform.getMyElement();
(143)                   this.sleep(20000);
(144)
(145)                }catch (Exception e){
(146)                System.err.println("Error: " + e.getMessage());
(147)                }//end catch
```

```
(148)                //Fetch WEFs from root element
(149)                //
(150)                Node returnedEvents = returnedWEFs.getFirstChild();
(151)
(152)                Element txElement = null;
(153)
(154)                try
(155)                {
(156)                     Boolean firstEvent = true;
(157)                     Long currentTime = null;
(158)                     Long previousTime = null;
(159)                     Long msgDelay = null;
(160)                     Long maxDelay = (long)6000;
(161)
(162)
(163)                   //Iterate through all child nodes (WEFs),
(164)                   //sending selected WEF
(165)                   //
(166)                   for(Node childNode = returnedEvents.getFirstChild();
(167)
(168)                         childNode!=null;){
(169)
(170)                      //Store nextChild and fetch current child
(171)                      //(Management Event) as txElement
(172)                      //
(173)                      Node nextChild = childNode.getNextSibling();
(174)
(175)                      txElement = (Element)childNode;
(176)
(177)                      //LastChild of txElement (Management Event) is
(178)                      //situation element
(179)                      //
(180)                      Node lastChild = txElement.getLastChild();
(181)
(182)                      //LastChild of situation element is Situation Type
(183)                      //
(184)                      NamedNodeMap sitAttrList = lastChild.getAttributes();
(185)                      Node sitAttr = sitAttrList.item(0);
(186)
(187)                      //Test if current element is a valid WEF for publication
(188)                      //
(189)                      if (sitAttr.getNodeValue().equalsIgnoreCase(validWEF1))
                                                          validWEF = true;
(190)                      if (sitAttr.getNodeValue().equalsIgnoreCase(validWEF2))
                                                          validWEF = true;
(191)
(192)                      if (allWEFs || validWEF){
(193)
(194)                         //Get time stamp (1st attribute) of management event
(195)                         //
(196)                         NamedNodeMap manEventAttrList =
                                          txElement.getAttributes();
(197)                         Node manEventAttr = manEventAttrList.item(0);
(198)
(199)                         //Establish delay time between previous message
(200)                         //and this one
```

```
(201)                          //
(202)                          if (DelayTx){
(203)
(204)                               try{
(205)                               repDate = IsoDateParser.
                                                parse(manEventAttr.getNodeValue());
(206)
(207)                               //Get message timestamp and compare to previous
(208)                               //
(209)                               currentTime = repDate.getTime();
(210)                               if (firstEvent){
(211)                                     previousTime = currentTime;
(212)                                     firstEvent = false;
(213)                               }
(214)                               //If delay between messages is
(215)                               //
(216)                               msgDelay = currentTime - previousTime;
(217)                               if (msgDelay >= 5000) {msgDelay = maxDelay;}
(218)                               previousTime = currentTime;
(219)
(220)                               }catch (ParseException e) {
(221)                               System.out.println(e);
(222)                              }//endCatch
(223)                               Thread.currentThread().sleep(msgDelay);
(224)
(225)                          }//EndIf - DelayTx
(226)
(227)                             getLog().info("Sending message to consumers...");
(228)                             wsn.publish(_TOPIC_NAME, txElement);
(229)                             validWEF = false;
(230)
(231)                        }//EndIf
(232)                          childNode = nextChild;
(233)                     }//endFor
(234)
(235)                  }//endTry
(236)                   catch (Throwable error){
(237)                       error.printStackTrace();
(238)                   }//endCatch
(239)
(240)              }//end run
(241)
(242)          };//end Thread
(243)          producer.start();
(244)      }//end initComp
(245)
(246) }
```

Appendix I – ACManagerClient.java File

The ACManagerClient creates and activates the ResourceFactory, triggering the creation of a managed resource and subscribes the ACManager to selected topics of the managed resource.

```
(01)     package thshome.hlinux.vres.muse.touch.impl;
(02)
(03)     import java.net.InetAddress;
(04)     import java.net.URI;
(05)     import java.util.Date;
(06)     import java.net.UnknownHostException;
(07)
(08)     import javax.xml.namespace.QName;
(09)
(10)     import org.apache.muse.util.xml.XPathUtils;
(11)     import org.apache.muse.ws.addressing.EndpointReference;
(12)     import org.apache.muse.ws.notification.remote.NotificationProducerClient;
(13)     import org.apache.muse.ws.resource.lifetime.WsrlConstants;
(14)     import org.apache.muse.ws.resource.remote.WsResourceClient;
(15)     import org.apache.muse.ws.notification.impl.TopicFilter;
(16)
(17)     public class ACManagerClient
(18)     {
(19)
(20)         public static String PREFIX = "tns";
(21)
(22)         public static String NAMESPACE_URI =
                                "http://vres.hlinux.thshome/muse/touch/impl";
(23)
(24)        private static final QName _TOPIC_NAME =
                                 new QName(NAMESPACE_URI, "AllWEFs", PREFIX);
(25)
(26)        public static URI getLocalAddress(String contextPath, int port)
(27)             throws UnknownHostException
(28)         {
(29)             String ip = InetAddress.getLocalHost().getHostAddress();
(30)
(31)             StringBuffer address = new StringBuffer();
(32)             address.append("http://");
(33)             address.append(ip);
(34)             address.append(':');
(35)             address.append(port);
(36)
(37)             if (contextPath.charAt(0) != '/')
(38)                 address.append('/');
(39)
(40)             address.append(contextPath);
(41)             return URI.create(address.toString());
(42)         }
(43)
(44)         public static void getResourceProperty
                            (NotificationProducerClient client) throws Exception
(45)         {
```

```
(46)             client.getPropertyAsObject(WsrlConstants.CURRENT_TIME_QNAME,
                                                                    Date.class);
(47)         }
(48)
(49)         public static void getResourcePropertyDocument(
                                              NotificationProducerClient client)
(50)         throws Exception
(51)     {
(52)         client.getResourcePropertyDocument();
(53)     }
(54)
(55)         public static void setResourceProperties(WsResourceClient client)
(56)         throws Exception
(57)     {
(58)         QName name = new QName("http://vres.hlinux.thshome/muse/touch/impl",
                                           "GridProperty", "tns");
(59)
(60)         Object[] values = new Object[]{ "AllWEFs" };
(61)         client.updateResourceProperty(name, values);
(62)
(63)         String query = "/*/tns:GridProperty";
(64)         String dialect = XPathUtils.NAMESPACE_URI;
(65)         client.queryResourceProperties(query, dialect);
(66)         client.getResourceProperty(name);
(67)     }
(68)
(69)         public static void main(String[] args)
(70)         {
(71)             try
(72)             {
(73)                 int producer_port = Integer.parseInt(System.
                                 getProperty("producer_port","8080"));
(74)                 int consumer_port = Integer.parseInt(System.
                                 getProperty("consumer_port","8080"));
(75)
(76)                 //Create WsResourceFactory and ping it to create SManRes01
(77)                 //
(78)              String factroot = "/SManRes01/services/WsResourceFactory";
(79)              URI addressf = getLocalAddress(factroot, 8080);
(80)              EndpointReference factory = new EndpointReference(addressf);
(81)              WsResourceClient factoryClient = new WsResourceClient(factory);
(82)              factoryClient.getEndpointReference();
(83)
(84)                 //Create EPR for GridResource01
(85)              //
(86)              String webAppRoot = "/SManRes01/services";
(87)                 String contextPath = webAppRoot + "/GridResource01";
(88)                 URI address = getLocalAddress(contextPath, producer_port);
(89)                 EndpointReference ProdEPR = new EndpointReference(address);
(90)
(91)                 //Create EPR for ACManager
(92)                 //
(93)                 webAppRoot = "/AManager/services";
(94)                 contextPath = webAppRoot + "/ACManager";
(95)                 address = getLocalAddress(contextPath, consumer_port);
(96)                 EndpointReference ACManERP = new EndpointReference(address);
```

```
(97)
(98)                //Create new WsResourceClient for the Autonomic Manager
(99)                //
(100)               WsResourceClient AManager01 = new WsResourceClient(ACManERP);
(101)   //          AManager01.setTrace(true);
(102)
(103)               // Initialise Notification Producer
(104)               //
(105)               NotificationProducerClient producer =
                                  new NotificationProducerClient(ProdEPR);
(106)   //            producer.setTrace(true);
(107)               setResourceProperties(producer);
(108)
(109)               //Subscribe SManRes to AMan
(110)               //
(111)               producer.subscribe(ACManERP, new TopicFilter(_TOPIC_NAME),
                                                                        null);
(112)
(113)
(114)           }
(115)
(116)           catch (Throwable error)
(117)           {
(118)               error.printStackTrace();
(119)           }
(120)       }
(121) }
```

Appendix J – GLAParser.java File

The GLAParser reads the configuration file, sets the output file and initiates the parser.

```
(01)     package thshome.hlinux.vres.muse.touch.impl;
(02)
(03)     import java.util.logging.Logger;
(04)
(05)     import org.eclipse.hyades.logging.adapter.Adapter;
(06)     import org.eclipse.hyades.logging.adapter.AdapterException;
(07)     import org.eclipse.hyades.logging.adapter.IComponent;
(08)     import org.eclipse.hyades.logging.adapter.IContext;
(09)     import org.eclipse.hyades.logging.adapter.IStatus;
(10)     import org.eclipse.hyades.logging.adapter.outputters.CBEFileOutputter;
(11)     import org.eclipse.hyades.logging.adapter.sensors.SingleOSFileSensor;
(12)     import org.eclipse.hyades.logging.adapter.impl.Status;
(13)
(14)     public class GLAParser {
(15)
(16)        public Boolean glaCompleted = false;
(17)
(18)        public void runGLAParser()
(19)        {
(20)
(21)           Logger mylog = Logger.getLogger("global");
(22)           mylog.info("Starting GLA Parser");
(23)
(24)            String adapterConfigFile =
                                    "C://Touch/GLA/gridService/regex.adapter";
(25)
(26)             // Create an Adapter instance and set config file
(27)           //
(28)             Adapter adapter = new Adapter();
(29)
(30)             adapter.setContextConfigPath(adapterConfigFile);
(31)             adapter.setComponentConfigPath(adapterConfigFile);
(32)
(33)             // Create a file outputter and set
(34)             //
(35)             CBEFileOutputter fo = new CBEFileOutputter();
(36)
(37)             // Set the outputter to capture the GLA log messages
(38)             //
(39)             adapter.setLogOutputter(fo);
(40)
(41)             // Get the adapter configuration
(42)             //
(43)             IContext [] contexts = null;
(44)             try {
(45)              contexts = adapter.getConfiguration();
(46)             }
(47)             catch (AdapterException e) {
(48)              System.err.println("Exception occurred getting configuration
```

```
                                        of the adapter " + adapterConfigFile);
(49)              System.err.println(e.getMessage());
(50)              return;
(51)             }
(52)             mylog.info("Received configuration from adapter " +
                                                    adapterConfigFile);
(53)             mylog.info("Number of contexts in configuration is " +
                                                    contexts.length);
(54)
(55)             // Get the component from the first context
(56)             //
(57)             IComponent [] components = contexts[0].getComponents();
(58)
(59)             /* Get the sensor and update the directory and file name
                                                          of the log to parse */
(60)             SingleOSFileSensor sensor = (SingleOSFileSensor)components[0];
(61)            sensor.setDirectory("C://Touch/GLA/gridService/");
(62)            sensor.setFileName("gridResource01.log");
(63)
(64)             // Get the outputter and set directory and file name
(65)             //
(66)             CBEFileOutputter outputter =
                           (CBEFileOutputter)components[components.length-1];
(67)             outputter.setFileName("gridResource01CBEs.xml");
(68)             outputter.setDirectory("C://Touch/GLA/gridService/");
(69)
(70)             // Validate the adapter configuration file
(71)             //
(72)             try {
(73)                 adapter.validate();
(74)             }
(75)             catch (AdapterException e) {
(76)                 System.err.println("Adapter configuration file is invalid.");
(77)                 System.err.println(e.getMessage());
(78)                 return;
(79)             }
(80)
(81)             System.out.println("Adapter configuration was validated.");
(82)
(83)             // Set the logging level so the Adapter logs Warning,
                                          Critical and Fatal log messages
(84)             //
(85)             adapter.setLoggingLevel((short)30);
(86)
(87)             // Start the adapter as a separate thread
(88)             //
(89)             try {
(90)                 adapter.start(true, true);
(91)             }
(92)             catch (AdapterException e) {
(93)                 System.err.println("GLA Exception occurred");
(94)                 System.err.println(e.getMessage());
(95)                 adapter.stop();
(96)                 return;
(97)             }
(98)             mylog.info("Adapter was started.");
```

```
(99)
(100)            // Wait 30 seconds
(101)            //
(102)            try {
(103)                Thread.sleep(30000);
(104)            }
(105)            catch (InterruptedException e) {
(106)
(107)            }
(108)
(109)            // Get the adapter status
(110)            //
(111)            IStatus status = adapter.getStatus();
(112)
(113)            if (status != null) {
(114)
(115)                printAdapterStatus(status);
(116)            }
(117)
(118)            // If the adapter is still running then stop it
(119)            if (status != null && status.isActive()) {
(120)                adapter.stop();
(121)                System.out.println("Adapter was stopped.");
(122)            }
(123)
(124)          //Set completion flag to true
(125)            glaCompleted = true;
(126)          return;
(127)       }// End runGLAParser()
(128)
(129)
(130)       public Boolean glaFinished()
(131)       {
(132)          return glaCompleted;
(133)       }
(134)
(135)       public void printAdapterStatus(IStatus status) {
(136)            /* print the current adapter status out */
(137)            if (status.isActive()) {
(138)                System.out.println("Adapter is active!  It has processed "
(139)                    + status.getItemsProcessedCount() + " items in "
(140)                    + ((Status)status).getElpasedTimeInSeconds()
(141)                    + " seconds.");
(142)            }
(143)            else {
(144)                System.out.println("Adapter is not active!  It processed "
(145)                    + status.getItemsProcessedCount() + " items in "
(146)                    + ((Status)status).getElpasedTimeInSeconds()
(147)                    + " seconds.");
(148)            }
(149)
(150)            /* Get the status of the contexts */
(151)            IStatus [] contexts = status.getChildrenStatus();
(152)
(153)            if (contexts != null && contexts.length > 0) {
(154)                for (int i=0; i < contexts.length; i++) {
```

```
(155)                     /* For each context, print its status */
(156)                     if (contexts[i] != null) {
(157)                         IStatus cstatus = contexts[i];
(158)                         if (cstatus.isActive()) {
(159)                             System.out.println("Context " + cstatus.getName()
(160)                                 + " is active!  It has processed "
(161)                                 + cstatus.getItemsProcessedCount()
(162)                                 + " items in "
(163)                                 + ((Status)cstatus).getElpasedTimeInSeconds()
(164)                                 + " seconds.");
(165)                         }
(166)                         else {
(167)                             System.out.println("Context " + cstatus.getName()
(168)                                 + " is not active!  It processed "
(169)                                 + cstatus.getItemsProcessedCount()
(170)                                 + " items in "
(171)                                 + ((Status)cstatus).getElpasedTimeInSeconds()
(172)                                 + " seconds.");
(173)                         }
(174)
(175)                         /* Get the status of the components */
(176)                         IStatus [] components = cstatus.getChildrenStatus();
(177)                         if (components != null && components.length > 0) {
(178)                             for (int j=0; j < components.length; j++) {
(179)                                 /* For each component, print its status */
(180)                                 IStatus cpstatus = components[j];
(181)                                 if (cpstatus.isActive()) {
(182)                                     System.out.println("  Component "
(183)                                         + cpstatus.getName()
(184)                                         + " is active and has processed "
(185)                                         + cpstatus.getItemsProcessedCount()
(186)                                         + " items in "
(187)                                         + ((Status)cpstatus).
                                                    getElpasedTimeInSeconds()
(188)                                         + " seconds.");
(189)                                 }
(190)                                 else {
(191)                                     System.out.println("  Component "
(192)                                         + cpstatus.getName()
(193)                                         + " is not active and processed "
(194)                                         + cpstatus.getItemsProcessedCount()
(195)                                         + " items in "
(196)                                         + ((Status)cpstatus).
                                                    getElpasedTimeInSeconds()
(197)                                         + " seconds.");
(198)                                 }
(199)                             }
(200)                         }
(201)                     }
(202)                 }
(203)             }
(204)         }
(205) }
```

Appendix K – WEFTransform.java File

The WEFTransform file sets the input and transformation xst files, and returns the results of the transformation.

```
(01)     package thshome.hlinux.vres.muse.touch.impl;
(02)
(03)     import java.io.File;
(04)     import java.util.logging.*;
(05)
(06)     import org.w3c.dom.*;
(07)     import javax.xml.parsers.*;
(08)     import javax.xml.transform.*;
(09)     import javax.xml.transform.dom.*;
(10)
(11)     public class WEFTransform
(12)     {
(13)
(14)        public static Node getMyElement()
(15)        {
(16)           Document docSource;
(17)           Document docResult;
(18)           Document docXSLT;
(19)           DOMSource dsSource;
(20)           DOMSource dsXSLT;
(21)           DOMResult drResult;
(22)           Node AllEvents = null;
(23)
(24)           Logger mylog = Logger.getLogger("global");
(25)           mylog.info("XSL Transformation");
(26)
(27)           try
(28)           {
(29)                 // Create a document builder factory and the builder
(30)              //
(31)              DocumentBuilderFactory dFactory =
                                  DocumentBuilderFactory.newInstance();
(32)              dFactory.setNamespaceAware(true);
(33)              DocumentBuilder dBuilder = dFactory.newDocumentBuilder();
(34)              dBuilder.setErrorHandler(new SAXErrorHandler());
(35)
(36)              // Load and parse the source and xsl documents
(37)              //
(38)              docSource = dBuilder.parse(new
                        File("C:/Touch/GLA/gridService/gridResource01CBE.xml"));
(39)              docXSLT = dBuilder.parse(new
                        File("C:/Touch/GLA/gridService/100108.xsl"));
(40)
(41)              // Create the result document
(42)              //
(43)              docResult = dBuilder.newDocument();
(44)
(45)              // Create DOMSource objects source and xsl
```

```
(46)              //
(47)              dsSource = new DOMSource(docSource);
(48)              dsSource.setSystemId
                        ("C:/Touch/GLA/gridService/gridResource01CBE.xml");
(49)              dsXSLT = new DOMSource(docXSLT);
(50)              dsXSLT.setSystemId("C:/Touch/GLA/gridService/100108.xsl");
(51)
(52)              // Create a DOMResult object for transformation result
(53)              //
(54)              drResult = new DOMResult(docResult);
(55)
(56)              // Create a transformer factory and transformer
(57)              //
(58)              TransformerFactory tFactory = TransformerFactory.newInstance();
(59)              Transformer transformer = tFactory.newTransformer(dsXSLT);
(60)
(61)              // Perform the transformation and get the DOM result tree
(62)              //
(63)              transformer.transform(dsSource, drResult);
(64)
(65)              //Place results in return Node
(66)              //
(67)              AllEvents = drResult.getNode();
(68)              mylog.info("XSL Transformation Completed");
(69)           }//EndTry
(70)           catch (Throwable t)
(71)           {
(72)              t.printStackTrace ();
(73)              System.exit(1);
(74)           }//EndCatch
(75)
(76)           //Return results
(77)           //
(78)           return AllEvents;
(79)        }//End getMyElemnt()
(80)
(81)  }//End WEFTransform
```