Deployment of Grids through Integrated Configuration Management

Prof. R.O. Sinnott, J. Muhammad, W. Yuxiang
National e-Science Centre
University of Glasgow
Glasgow G12 8QQ
r.sinnott@nesc.gla.ac.uk

ABSTRACT
A Grid environment typically comes into existence when several collaborating institutions contribute resources for researchers to solve problems of mutual interest. Such collaboration of personnel and resources are commonly referred to as a Virtual Organization (VO). The process to establish and manage a VO can be a time consuming and laborious process with installation and configuration of VO specific software and data needed across sites. Configuration management technologies can facilitate this process, however to support the establishment of Grid based VOs, it is necessary to align and integrate Grids and configuration management technologies. This alignment should recognize that sites may have their own flavors of configuration management tools and Grid technologies, and allow VO administrators to seamlessly deploy and configure Grid resources across multiple sites. This paper presents the experiences in developing and testing such an integrated Grid and configuration management framework.

Keywords: Grids, Virtual Organizations, Configuration Management.

1. Introduction
Virtual Organizations (VOs) [1] come in to existence when one or more institutions and their research personnel wish to collaborate and potentially share resources, where resources can be a variety of things including hardware such as storage servers/databases, clusters, supercomputers; as well as software and data specific to the needs of that VO [2-4].

The Globus Toolkit [5] is one of the most widely adopted middleware for the establishment and management of Grids today, providing an open source toolkit with software tools targeted to the establishment of Grids. One area in which the Globus middleware and other Grid middleware existing today are lacking is in support for the roll-out of Grid infrastructures themselves. Thus it is currently the case that the establishment and management of Grids requires local administrators to manually install and configure software for the needs of particular local and remote researchers. For large scale Grids involving multi-site collaborations, the installation and configuration of the software infrastructure needs for particular VOs becomes non-scalable for system administrators at a given site. Software targeted to the needs of specific VOs often comes from a variety of sources including remote collaborators. Detailed knowledge of these software tools and their often prototypic nature mean that local administrators are left with a variety of potentially conflicting software and software configuration requirements. This is especially the case when site resources are shared across multiple VOs.

Configuration management tools have evolved to allow for the installation and configuration of software in a given domain and across a closed set of resources. Integration of configuration management tools with Grid technologies is needed to support large scale Grids, where VO administrators are able to install and configure VO specific software across multi-site resources. This is not without challenges however. Perhaps the greatest challenge to overcome is one of security and especially trust. Before any large scale deployment across multiple domains is attempted, it is essential to show that Grid technologies and a variety of configuration management software can be integrated since we recognize that different sites will likely have their own policies and software for configuration management.

In this paper we describe key scenarios needed to support the integration of Grid based VOs and a variety of configuration management tools and experiences in their implementation. Section 2 provides justification of the need for Grid and configuration technology integration, and a review of existing work done in this area. Section 3 provides an overview of key scenarios for integration of Grid and configuration management technologies. Section 4 describes experiences in implementing these scenarios. Finally section 5 draws conclusions and outline the future work we have planned.

2. Configuration Management and the Grid
There are numerous challenges in realizing the integration of configuration management tools with Grid middleware to establish VOs across multiple sites. Arguably the most important to overcome is in security. This is exacerbated by the current models of Grid security.

Currently most major Grid security models existing today are based on Public Key

Infrastructure (PKIs) [6] using X.509 digital certificates issued by a trusted third party certification authority. This model allows sites hosting end resources in a given VO to validate the identity of individuals requesting access to their resources – so called *authentication*. This is limited from several perspectives, e.g. there is no mention of what a user is allowed to do once is gained access to a requested resource. For this reason, Grids have been seen as a potential danger to existing security infrastructures. Although, there have been numerous technological solutions put forward looking towards providing various enhanced security models and solutions such as AKENTI [7], CAS [8], CARDEA [9], PERMIS [10] and VOMS [11], the coupling of Grid security, Grid technologies more generally and configuration management tools has not been addressed. Yet for future secure Grids and VOs it is essential. A single inadequately configured site can endanger all other sites within a given VO if it does not take all appropriate measures to ensure its own configuration and security together. Whilst Grid services might use advanced authorization infrastructures to protect access to given data, these security mechanisms will be made redundant if the basic underlying fabric upon which those services exist is inadequately protected. This might be from numerous perspectives: firewalls that have been left open or incorrectly configured; out of date anti-virus software; operating systems or Grid middleware itself that have not been patched with latest updates to name but a few examples.

Ensuring that all Grid nodes used across a VO have the necessary operating system, Grid middleware and that the underlying fabrics are properly patched and the most up to date antivirus software protection is something that up until now has been left to the individual VO sites. This is something that cannot be left to chance however. It needs to be managed at the VO level. For example, a Grid based systems dealing with medical records or any other highly sensitive data sets demand that resources are protected as far as possible. It is the case that the middleware solutions up to now have been primarily targeted at isolated aspects of Grid security, e.g. advanced authorization, or given e-Research projects have focused on specific aspects of security and tools of relevance to their needs. The wider challenges for an integrated approach to configuration management, security frameworks and associated tools that can encompass and support the spectrum of e-Research community currently being undertaken has not yet been realized.

One of the major challenges in delivering secure VOs is that they can be both formed and changed dynamically. Unlike many previous large scale distributed systems, Grids – or rather the Grid vision – is based upon the idea that collections of resources and researchers constituting a VO can be identified and established dynamically, i.e. without detailed prior negotiations. Furthermore, new resources can be added, old resources removed, new end users added or removed and their roles and responsibilities changed throughout the lifetime of the VO. In this context the identification of roles and responsibilities and their assignment to individuals can be a complex activity.

3. Related Work

Currently, various tools are available to support manual installation and configuration of networks of computers. Those that have focused in particular on configuration and management of grid fabrics include SmartFrog (Smart Framework for Object Groups) [12], LCFG [13], Cfengine [14] and OGSAconfig [15]. SmartFrog has been used to ease the often lengthy process of grid deployment in a dynamic grid environment. A recent effort [16], SmartFrog has been used to ease the often lengthy process of grid deployment in a dynamic grid environment, but this solution lacked the security aspect of the underlying Grid fabric due to the fact that each institution in the VO was required to have its own set of policies that it needed to enforce on its resources and services, e.g. access permissions were granted to each user.

A typical approach adopted by these tools is to use an interactive shell script for installing and configuring software bundles. This script guides the user through the software installation and configuration process across a fixed set of resources. The drawback with this approach in a Grid context is that the user is required to provide Grid configuration data interactively. Scalability is another issue since the script has to be run interactively for each node. HP Labs' GridWeaver project [17,18] with the University of Edinburgh is an application of the SmartFrog Framework that applied LCFG and SmartFrog for Grid installations. This is only suitable for UNIX platforms however. There are other sources and approaches available for configuration management – many of which have adopted a Linux *rpm* like approach. Similarly, ProLiant Essentials [19] also deals with deployment of applications and images.

Dynamic reconfiguration and autonomy of configuration have also been demonstrated through these tools, however, we recognize that large scale Grids are by their very nature heterogeneous. This includes heterogeneity of configuration management tools. Whilst the above mentioned works have shown how configuration management can be incorporated into a Grid, few works have dealt with incorporation of multiple configuration management tools. This is the focus of our work here. In particular, in our work we have focused especially on SmartFrog and CFengine and their

integration with Grid services to support a range of configuration management issues in a Grid context.

4. Configuration Management Tools

4.1. SmartFrog

SmartFrog (Smart Framework for Object Groups) is a technology for describing distributed software systems as collections of cooperating components, and then activating and managing them [20]. The SmartFrog Framework consists of a *description language* for describing component collections and component configuration parameters; a *deployment infrastructure* that activates the application description; a *component model* that manages the components to deliver and maintain running systems, and a *set of components* that provide various application services. SmartFrog has wide applicability across domains ranging from utility computing to large-scale system configuration. SmartFrog and its components are implemented in Java™, though SmartFrog components can easily be written to encapsulate software components based on other technologies.

A typical SmartFrog script used to test out scenarios for copying a Grid archive file (gar file) to a remote machine and then successfully deploying the Grid service on a remote machine is shown in Figure 1.

```
#include "org/smartfrog/components.sf"
#include "org/smartfrog/services/net/telnet.sf"
#include "org/smartfrog/services/net/ftp.sf"

sfConfig extends Compound {
    sampleFTPClient extends FTPClient {
            PasswordProvider:passwordFile "/usr/local/passwd.txt";
            FTP:transferType "put";
            FTP:transferMode "binary";
            FTP:ftpHost "labpc-18.nesc.gla.ac.uk";
            FTP:username "wu";
            FTP:remoteFiles ["/home/wu/org_globus_examples_services_core_first.gar"];
            FTP:localFiles ["/home/wu/org_globus_examples_services_core_first.gar"];
    }

    sampleTelnet extends TelnetSession {
            PasswordProvider:passwordFile "/usr/local/passwd.txt";
            Telnet:commands ["cd /home/wu",
              "globus_deploy-gar ./org_globus_examples_services_core_first.gar"];
        Telnet:host "labpc-18.nesc.gla.ac.uk";
        Telnet:username "wu";
        Telnet:ostype "linux";
  }
}
```

Figure 1: SmartFrog Script

The SmartFrog components shown in Figure 1 show the ftp component for transferring the gar file to a remote machine and the telnet component for deploying the copied (gar file) on the remote machine for grid service deployment.

4.2. CFengine

CFengine is a distributed agent framework for performing policy-based network and system administration that is used on many thousands of Unix-like and Windows systems. CFengine is deployed in an environment which *classifies* its view of the world into overlapping sets. Those tasks which overlap with a particular agent's world view are performed by the agent. The main components of Cfengine are shown in Figure 2 [19].

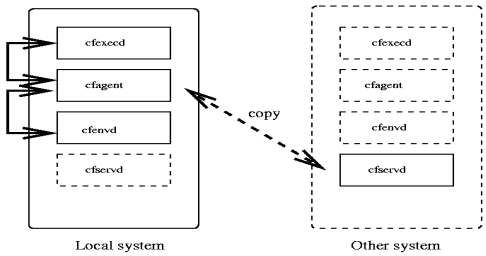

Figure 2: CFengine components

A typical CFengine script used to test out scenarios for copying a Grid archive file (gar file) to a remote machine and then successfully deploying the Grid service on a remote machine is shown in Figure 3.

```
# Simplest version - separate this from main config
control:
   actionsequence = ( copy shellcommands )
   domain = ( nesc.gla.ac.uk )
    policyhost = ( labpc-18.nesc.gla.ac.uk )
    master_cfinput = ( /usr/cfinput  ) #server input directory
    workdir = ( /home/wu )
###############copy gar file from server#####
copy:
 ${master_cfinput}/org_globus_examples_services_core_first.gar
  dest=${workdir}/org_globus_examples_services_core_first.gar
                server=$(policyhost)
                trustkey=true
###############deploy grid service########
shellcommands:
any::
"/usr/local/gt4.0/bin/globus-deploy-gar
$(workdir)/org_globus_examples_services_core_first.gar"
#tidy:
# $(workdir)/outputs pattern=* age=7
```

Figure 3: CFengine Script

This CFengine script shows the control component for setting up the configuration on the remote machine; copying the (gar file) to the remote machine and the shell command showing the grid service deployment copied (gar file) in the working directory on the remote machine.

5. Scenarios and Experiments

To prove that Grid technologies and different configuration management tools could be integrated we have explored numerous scenarios. These were based upon a trial infrastructure comprising three Linux (Fedora Core 6) Pentium-IV machines - whilst not a full blown Grid infrastructure this test bed has allowed for a variety of experiments and scenarios to be explored. Some of the scenarios that were supported included the following: basic data deployment; generic software installation; Grid software installation and Grid service deployment, activation and run time access. For each of these scenarios we describe the basic scenario explored and the associated preconditions and successful post-conditions.

5.1 Data Deployment

This scenario is based around the most trivial configuration management scenario, namely copying data from one machine to other machines in the VO. There are many ways in which data can be copied, moved or replicated across Grid resources, e.g. GridFTP [22], SRB [23], Condor [24] amongst numerous others. The basic scenario here was to show how data from one machine could be seamlessly copied across two other machines in the test bed as part of the behavior of a Grid service using configuration management tools.

The scenario and the data itself were based upon Grid services developed by advanced MSc students at NeSC as part of their Grid Computing module programming assignment which was to develop a GT4 service to search/sort files of different sizes using a Condor pool. The basic idea behind the assignment was to establish at what size of file it was faster to search/sort files using Grid technologies and a Condor pool as opposed to searching and sorting the file on a single PC. The files themselves varied from 512kB up to 1Gb.

This CFengine script shows the control component for setting up the configuration on the remote machine; copying the (gar file) to the remote machine and the shell command showing the grid service deployment copied (gar file) in the working directory on the remote machine. and the data itself were based upon Grid services developed by advanced MSc students at NeSC as part of their Grid Computing module programming assignment which was to develop a GT4 service to search/sort files of different sizes using a Condor pool. The basic idea behind the assignment was to establish at what size of file it was faster to search/sort files using Grid technologies and a Condor pool as opposed to searching and sorting the file on a single PC. The files themselves varied from 512kB up to 1Gb.

The preconditions associated with this scenario were that the client side machine had the data to be transferred, and the two other hosts had CFengine agents and SmartFrog daemons running. The code for the data transferal via CFengine and SmartFrog is shown in Figure 4.

```
# &implest version - separate this from main config
control:
  actionsequence = ( copy shellcommands )
  domain = ( nesc.gla.ac.uk )
  policyhost = ( labpc-18.nesc.gla.ac.uk )
  master_cfinput = ( /usr/cfinput  ) #server input directory
  workdir = ( /home/wu )
##############copy gar file from server######
copy:
$(master_cfinput)/org_globus_examples_services_core_first.gar
  dest=$(workdir)/org_globus_examples_services_core_first.gar
  server=$(policyhost)
  trustkey=true
shellcommands:
#########################
    #include "org/smartfrog/components.sf"
    #include "org/smartfrog/services/net/telnet.sf"
    #include "org/smartfrog/services/net/ftp.sf"

    sfConfig extends Compound {
        sampleFTPClient extends FTPClient {
                PasswordProvider:passwordFile "/usr/local/passwd.txt";
                FTP:transferType "put";
                FTP:transferMode "binary";
                FTP:ftpHost "labpc-18.nesc.gla.ac.uk";
                FTP:username "wu";
                FTP:remoteFiles ["/home/wu/org_globus_examples_services_core_first.gar"];
                FTP:localFiles ["/home/wu/org_globus_examples_services_core_first.gar"]; }
```

Figure 4: Configuration Scripts for Data Transferal

Both the SmartFrog and CFengine scripts were integrated into the functionality of the Grid service. Thus upon invocation by a client, the GT4 search/sort service behavior first replicated the data to the hosts via the appropriate scripts, i.e. the two other machines in the Condor pool before the search and sort took place. Upon completion, the data was returned to the submitting machine. Both SmartFrog and CFengine were (unsurprisingly!) able to transfer data between nodes via execution of scripts invoked from Grid services. One advantage of this approach as opposed to using Condor file staging for example is that it allows for concurrent data transfers to multiple machines.

5.2 Software Deployment

In this scenario, we consider the case where a user wants to deploy software across a set of known Grid resources for a particular application to be run. In this case we focused in particular upon a scenario where a certain version of Java was needed (Java 1.6) across the three nodes of the test-bed for a simple Java 1.6 based application to execute.

The preconditions associated with this scenario include firstly that the user machine has the appropriate version of Java installed and that they have the privilege to install software on the other machines. The successful post-condition for this scenario is that Java 1.6 is installed across the three nodes of the test-bed and that the application could be run.

The SmartFrog and CFengine codes for the transfer and installation of Java 1.6 along with the setting of the appropriate CLASSPATH are shown in Figure 5.

```
control:
  actionsequence = ( copy shellcommands )
  domain = ( nesc.gla.ac.uk )
  policyhost = ( labpc-18.nesc.gla.ac.uk )
  master_cfinput = ( /usr/cfinput  ) #server input directory
  workdir = ( /home/wu )
##############copy gar file from server######
copy:
$(master_cfinput)/jdk-1.6.0_09-fcs
  dest=$(workdir)/ jdk-1.6.0_09-fcs
  server=$(policyhost)
  trustkey=true
shellcommands:
any:   "/usr/sbin/alternatives --install /usr/local/jdk/jre1.6.0/bin/java"
          export JAVA_HOME =/usr/local/jdk/jre1.6.0
#include "org/smartfrog/components.sf"
#include "org/smartfrog/services/net/telnet.sf"
#include "org/smartfrog/services/net/ftp.sf"

sfConfig extends Compound {
    sampleFTPClient extends FTPClient {
            PasswordProvider:passwordFile "/usr/local/passwd.txt";
            FTP:transferType "put";
            FTP:transferMode "binary";
            FTP:ftpHost "labpc-18.nesc.gla.ac.uk";
            FTP:username "wu";
            FTP:remoteFiles ["/home/wu/org_globus_examples_services_core_first.gar"];
            FTP:localFiles ["/home/wu/org_globus_examples_services_core_first.gar"]; }
```

Figure 5: Scripts for Java Software Installation

One issue that arose in testing this application was the interactive nature of software installation. For example, agreeing to the terms and conditions of the licenses was needed during installation and

configuration of the Java 1.6 environment. Scripts were used to overcome this issue, but we recognize that this is something where automation can potentially lead to system administrators being unaware of the agreements and license models of the software being installed, and is perhaps something that should not be automated.

5.3 Grid Middleware Deployment

As an extension to the scenario outlined above in 5.2 we also explored and implemented a scenario automating the installation and configuration of Grid middleware. In particular we attempted to show how a GT4 environment could be installed across remote machines. The preconditions associated with this scenario were that the client node had the various GT4 software components needed for installation and configuration of the GT4 environment. For simplicity we also assumed that the remote machines had needed and associated software tools already installed and configured, e.g. Ant and Java.

We also assumed that the remote machines did not already have a version of Globus installed. We note that previous work has shown that installation of different versions of Grid middleware (especially GT3 and GT4) cause problems when installed on a single machine. Following the installation and configuration of GT4 on the two other machines in the test bed, the containers were started to test for successful installation.

The post condition associated with this scenario was that there should be three machines available with GT4 environments established. The code for the GT4 installation and configuration via CFengine and SmartFrog is shown in Figure 6.

```
#include "org/smartfrog/components.sf"
#include "org/smartfrog/services/net/telnet.sf"
#include "org/smartfrog/services/net/ftp.sf"
sfConfig extends Compound {
        sampleTelnet extends TelnetSession {
                PasswordProvider:passwordFile "/usr/local/passwd.txt";
                Telnet:commands ["mkdir -p /opt/globus-4.0.1
  chown globus.globus /opt/globus-4.0.1
  ls -al /opt | grep globus
  su – globus
   cd
  tar -jxf gt4.0.1-all-source-installer.tar.bz2
  export JAVA_HOME=/opt/jdk1.6.0_06
  export PATH=$JAVA_HOME/bin:$PATH
  export JAVA_HOME=/opt/jdk1.6.0_06
  export PATH=$JAVA_HOME/bin:$PATH
  cd gt4.0.1-all-source-installer
   export GLOBUS_LOCATION=/opt/globus-4.0.1
./configure --prefix=$GLOBUS_LOCATION --disable-rls
make
make install "];
        Telnet:host "labpc-18.neso.gla.ac.uk";
        Telnet:username "wu";
        Telnet:ostype "linux"; } }
control:
  actionsequence = ( copy shelloommands )
  domain = ( neso.gla.ac.uk )
  polloyhost = ( labpc-18.neso.gla.ac.uk )
  master_ofinput = ( /usr/ofinput  ) #server input direotory
  workdir = ( /home/wu )
########deploy grid service########
shelloommands:
any::
  " mkdir -p /opt/globus-4.0.1
  chown globus.globus /opt/globus-4.0.1
  ls -al /opt | grep globus
  su – globus
   cd
  tar -jxf gt4.0.1-all-source-installer.tar.bz2
  export JAVA_HOME=/opt/jdk1.6.0_06
  export PATH=$JAVA_HOME/bin:$PATH
  export JAVA_HOME=/opt/jdk1.6.0_06
  export PATH=$JAVA_HOME/bin:$PATH
  cd gt4.0.1-all-source-installer
   export GLOBUS_LOCATION=/opt/globus-4.0.1
./configure --prefix=$GLOBUS_LOCATION --disable-rls
make
make install
```

Figure 6: Scripts for GT4 Installation

One issue raised in this scenario was concerning use of digital certificates. Whilst it was possible to copy Grid software between machines, the actual installation and configuration of GT4 required recognized digital certificates to be used. To overcome this issue it was necessary to introduce a manual step identifying the root certificate authority and provided the specific certificates needed for a GT4 installation and configuration. Hence the complete automation of this process was not possible.

5.4 Grid Service Deployment

For reasons of reliability and/or fault tolerance it is often the case that Grid services need to be replicated across other hosts. In this scenario we showed how configuration management software allows for the transfer of Grid services (given as Grid archive files) and the copying, extraction, installation and subsequent activation (deployment) of Grid services. This scenario assumes as a precondition that the scenario outlined in section 5.3 has been satisfactorily completed.

To demonstrate this scenario, we showed a a GT4 client interacting with a GT4 Grid service installed on the same machine. A version of this service, or rather the Grid archive file, was then moved to a remote host and extracted through a configuration management script. The remote Grid container was then stopped and restarted. Upon restarting the container, the newly deployed service was available for interactions with the client. A client on the original machine was directed to interact with the newly deployed service and could do so successfully.

Both CFengine and SmartFrog were shown to be capable of supporting this scenario. The script for achieving this scenario is shown in Figure 7.

```
#include "org/smartfrog/components.sf"
#include "org/smartfrog/services/net/telnet.sf"
#include "org/smartfrog/services/net/ftp.sf"

sfConfig extends Compound {
        sampleTelnet extends TelnetSession {
                PasswordProvider:passwordFile "/usr/local/passwd.txt";
                Telnet:commands ["cd /home/wu",
                  "globus_deploy-gar
./org_globus_examples_services_core_first.gar"];
        Telnet:host "labpc-18.neso.gla.ac.uk";
        Telnet:username "wu";
        Telnet:ostype "linux"; }
}

control:
  actionsequence = ( copy shelloommands )
  domain = ( neso.gla.ac.uk )
  polloyhost = ( labpc-18.neso.gla.ac.uk )
  master_ofinput = ( /usr/ofinput  ) #server input direotory
  workdir = ( /home/wu )
########deploy grid service########
shelloommands:
any::
"/usr/local/gt4.0/bin/globus-deploy-gar
$(workdir)/org_globus_examples_services_core_first.gar"
```

Figure 7: Scripts for Grid Service Deployment

6. Conclusions and Future Work

In this paper we have presented an integrated approach using configuration management tools

with Grid middleware – specifically Globus Toolkit 4.0.5 – to dynamically deploy grid infrastructures. In particular we have shown how it is possible to dynamically deploy and subsequently invoke Grid services across multiple nodes. A key aspect of our work is that we have shown how we can use a variety of configuration management tools simultaneously. We have exploited Cfengine and SmartFrog for this purpose.

The work has identified numerous open issues with the installation and configuration of Grid systems most notably the often manual interventions that are needed, e.g. for agreeing to licensing terms and conditions or for requesting and installing certificates needed for Grid middleware. Of course technical solutions to these kinds of issues are possible, e.g. by already having certificates on the machines where Grid middleware is to be deployed. This rather defeats the nature of dynamic deployment as required for dynamic VOs however.

The work described here is still quite nascent and we expect to explore a range of future research directions. Examples of the kinds of research challenges we expect to explore include issues of dependency management, e.g. when conflicting software requirements exist; establishment of a knowledge base for registering software already installed across nodes in a VO. Perhaps our greatest research challenge is in the area of security though. Defining security policies that allow assessment of the risks of establishing VOs comprising a variety of resources is one of our ultimate goals. Similarly, specification of security policies is key to the adoption of this kind of approach and we are exploring standards and technologies that facilitate the expression and enforcement of configuration management policies. One issue we have already identified in this is the level of granularity in expressing software installation and configuration privileges. That is, we wish to allow VO administrators to install software pertinent to their VO, but not allow unrestricted access to install arbitrary code.

7. References

[1] I. Foster, I. C. Kesselman, S. Tuecke, *The Anatomy of the Grid: Enabling scalable virtual organizations.* Lecture Notes in Computer Science, 2001.

[2] I. Foster, C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*. 2nd ed. 1999: Morgan Kaufmann.

[3] F. Berman, G. Fox, and A.J.G. Hey, *Grid Computing: Making the Global Infrastructure a Reality*. 2003, New York, NY, USA: John Wiley & Sons, Inc.

[4] M.L. Parashar, C.A. *Special Issue on Grid Computing*. 2005.

[5] *The Globus Toolkit* http://www-unix.globus.org/toolkit.

[6] R. Housely, T. Polk, *Planning for PKI: Best Practices Guide for Deploying Public Key Infrastructure*, in *Planning for PKI: Best Practices Guide for Deploying Public Key Infrastructure*. 2001, Willey Computer Publishing.

[7] W. Johnston, M. Thompson. *Authorization and Attribute Certificates for Widely Distributed Access Control. IEEE 7th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises - WETICE '98*. 1998: IEEE Computer Society.

[8] L. Pearlman, I. Foster, C. Kesselman, S. Tuecke. *A Community Authorization Service for Group Collaboration*. in *Policies for Distributed Systems and Networks, 2002. Proceedings. Third International Workshop*. 2002. Monterey, CA, USA: IEEE Computer Society.

[9] R. Lepro, R., *Cardea: Dynamic Access Control in Distributed Systems*, in *Technical Report*. 2003, NASA.

[10] D.W. Chadwick, O.O. *The PERMIS X.509 Role Based Privilege Management Infrastructure. ACM symposium on Access control models and technologies SACMAT '02*. 2003: ACM Press.

[11] R. Alfieri, et al, *Managing Dynamic User Communities in a Grid of Autonomous Resources. Computing in High Energy and Nuclear Physics*. 2003. La Jolla, California.

[12] *SmartFrog.* http://www.smartfrog.org, http://sourceforge.net/projects/smartfrog.

[13] P. Anderson, A.S., *LCFG: The Next Generation.* 2002.

[14] *http://www.cfengine.org/*

[15] *http://groups.inf.ed.ac.uk/ogsaconfig*

[16] Sabharwal, R. *Grid Infrastructure Deployment using SmartFrog Technology*. in *International conference on Networking and Services (ICNS'06)*. 2006. Silicon Valley, California, USA: IEEE Computer Society.

[17] G. Beckett, G. Mecheneau, P. Toft, P. Goldsack, P. Anderson, J. Paterson, C. Edwards. *GridWeaver: automatic, adaptive, large-scale fabric configuration for Grid Computing* in *AHM 2003*, Nottingham, UK.

[18] P. Toft, *Large-Scale, Adaptive Fabric Configuration for Grid Computing*, in *AHM UK*, 2003, Nottingham, UK.

[19] *ProLient Essentials*, 18004.www1.hp.com/manage/toolkit.html

[20] P. Goldsack, A. Lain, G. Mecheneau, P. Murray, P. Toft, *SmartFrog: Configuration and Automatic Ignition of Distributed Applications*. 2003, HP Labs Bristol: Bristol, UK.

[21] Burgess, M. *A Tiny Overview of CFengine*, in *1st International Workshop on Multi-Agent Robotic Systems (MARS 2005)*. 2005.

[22] http://www.globus.org/grid_software/data/gridftp.php

[23] http://homepages.nesc.ac.uk/~gcw/NGS/srb.htm

[24] http://www.cs.wisc.edu/condor

Author/s:
Sinnott, R. O.; Muhammad, J.; Yuxiang, W.

Title:
Deployment of grids through integrated configuration management

Date:
2008

Citation:
Sinnott, R. O., Muhammad, J., & Yuxiang, W. (2008). Deployment of grids through integrated configuration management. In H. Burkhart (Ed.), Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Networks: as part of the 26th IASTED International Multi-Conference on Applied Informatics, Innsbruck, Austria.

Publication Status:
Published

Persistent Link:
http://hdl.handle.net/11343/28900

File Description:
Deployment of grids through integrated configuration management