Overview of FutureGrid:
An Experimental Grid Cloud and HPC Test-Bed

Prepared by

Project Collaborators

for the

National Science Foundation

17 January 2011

Table of Contents

1. Introduction 5
2. Management 10
   2.1 Organizational Roles 10
   2.2 Overall Management Structure 15
   2.3 Key Management Personnel 16
   2.4 Team and Committee Structure 16
      2.4.1 Advisory Committee 17
      2.4.2 Operations and Change Management Committee. 17
      2.4.3 Executive Committee. 17
      2.4.4 Operational Teams 17
   2.5 Consensus Management Process 18
   2.6 Maintaining, Refreshing, and Executing the Project Vision 18
   2.7 TeraGrid 18
   2.8 Grid'5000 Collaboration 19
3. Hardware Infrastructure 19
   3.1 Computer Hardware 19
   3.2 Systems Integration and Transition to Operational Status Plan 22
      3.2.1 PY1 Achievements 22
      3.2.2 PY2 and Beyond Plans - Operational Phase 22
   3.3 Network Description 24
   3.4 Services Provided by FutureGrid Network 25
   3.5 Service Levels 26
   3.6 GlobalNOC Support of FutureGrid 26
   3.7 Network Milestones and Status 26
   3.8 Systems Operations and Deployment 26
      3.8.1 Software Support and Deployment 27
      3.8.2 Data Storage 28
4. FG Software Overview 28
   4.1 Goals of the Software Environment 28
   4.2 Software Architecture 32
      4.2.1 Architectural Overview 33

4.2.2 FG Access Services 34
4.2.3 HPC User Tools and Services 38
4.2.4 Additional Tools & Services 39
4.2.5 User and Support Services 41
4.2.6 FG Operations Services 43
4.3 FG Management Services 45
4.3.1 Image Management 45
4.3.2 Image Generation and Creation 46
4.3.3 Image Repository 47
4.4 Experiment Management 47
4.4.1 RAIN 48
4.4.2 Monitoring and Information Services 50
4.5 Current Software Deployment 53
5. Using FutureGrid 53
5.1 General Principles 53
5.2 Early Science Experiences on FutureGrid 55
5.2.1 Educational uses of FutureGrid 55
5.2.2 Grid and Cloud Middleware and Technology users of FutureGrid 56
5.2.3 Interoperability Experiments and Demonstrations on FutureGrid 56
5.2.4 Domain Science Applications of FutureGrid 56
6. User Support 57
6.1 Operations 57
6.2 Support 57
6.2.1 Tier 0: Support through Electronic Documentation 57
6.2.2 Tier 1: Support through Experts and Community 58
6.2.3 Tier 2: Support through staff 58
6.2.4 Tier 3: Advanced user support 59
6.3 Operations 59
6.4 Allocating FutureGrid Usage 60
7. Training Education and Outreach TEO 60
7.1 Discussion of Year 1 Activities 60
7.1.1 Summary 60
7.1.2 Tutorials 61

7.1.3 Virtual appliances and GroupVPN 61
7.1.4 Demonstrations and presentations 62
7.1.5 Class usage 62
7.2 Minority Serving Institution Engagement Plan 63
7.2.1 Introduction 63
7.2.2 Types of MSI Engagement 63
7.2.3 Activities Leveraging Existing MSI Contacts 64
7.2.4 Leveraging Social Networking Technologies 65
7.3 Outreach to New Users 65
7.3.1 Attracting Educators 65
7.3.2 Attracting Grid and Cloud Middleware and Technology Users 65
7.3.3 Attracting Interoperability Users of FutureGrid 66
7.3.4 Attracting Domain Scientists 66
APPENDIX A – List of FG Projects 67
APPENDIX B – Partial List of FutureGrid Projects with significant achievements 73
Results for Project "Windows and Linux performance comparison" 73
Results for Project "Distributed Scientific Computing Class" 73
Results for Project "Fine-grained Application Energy Modeling" 73
Results for Project "B649 Topics on Systems: Graduate Cloud Computing Class" 75
Results for Project "TeraGrid QA Testing and Debugging" 76
Results for Project "Sky Computing" 77
Results for Project "Cumulus" 79
Results for Project "Differentiated Leases for Infrastructure-as-a-Service" 80
Results for Project "Periodogram Workflow Running on FutureGrid Using Pegasus" 82
Results for Project "Big Data for Science Virtual Summer School July 26-30 2010" 86
Results for Project "SAGA" 87
Results for Project "Privacy preserving gene read mapping and hybrid cloud" 90
Results for Project "Cloud Technologies for Bioinformatics Applications" 92
APPENDIX C – Software Achievements and Milestones 94
Software Achievements in PY1 94
Software Milestones in PY2 96

1. Introduction

This document describes the implementation of *FutureGrid—an experimental grid cloud and HPC test-bed*. This document first gives an overview of the project, and then describes the management followed by the key components: hardware, software, use and users, user support and finally Education, Training and Outreach.

Clouds are challenging assumptions about grid computing and providing new technologies such as MapReduce and Bigtable. The goal of FutureGrid is to support the research that is inventing the future of distributed and parallel computing including grid, cloud computing and HPC as well as the integration of these ideas. This is achieved by offering a flexible reconfigurable testbed allowing researchers to test functionality, performance and interoperability of software systems in a reproducible fashion. FutureGrid supports users internationally and its initial design was modeled on the innovative French project Grid5000. Further FutureGrid supports both research and education where latter can exploit both the innovative technologies available and the interactive usage mode of FutureGrid. Interesting features of early use of FutureGrid is the emphasis on computer science systems, interoperability, clouds, education and bioinformatics – a very different spectrum from major TeraGrid resources.

FutureGrid's operating model is different from both TeraGrid, conventional clusters and commercial clouds. FutureGrid achieves its flexibility by dynamically provisioning software as needed onto "bare-metal" as illustrated in Figure 1 below

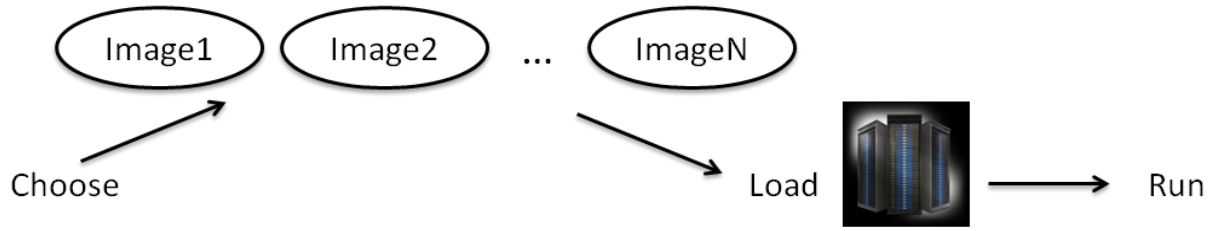


Figure 1: Usage Model for FutureGrid

The images include those for MPI, OpenMP, Hadoop, Dryad, gLite, Unicore, Globus, Xen, ScaleMP (distributed Shared Memory), Nimbus, Eucalyptus, OpenStack, KVM, and Windows. This approach requires some software development described in section 4 and initially flexibility is provided by statically provisioning nodes to a variety of different virtual machine/operating system/middleware configurations; in particular Nimbus, Eucalyptus and a classic HPC configuration. The dynamic approach allows us to support experiments with a variety of environments without the overhead of virtual machines; in particular early use has seen performance studies of Linux vs. Windows (and Hadoop vs. Dryad for MapReduce) and measurement of overheads of virtualization. FutureGrid supports experiments that can be reproduced by use of the same hardware and software between different sessions. As part of FutureGrid, the workflow system Pegasus will be used to develop an Experiment manager that will collect and preserve metadata and invoke dynamically provisioned environments requested by users.

The funded partners in FutureGrid are Indiana University, University of California – San Diego, University of Chicago, University of Florida, University of Texas at Austin – Texas Advanced Computer Center , University of Southern California, University of Tennessee – Knoxville, and University of Virginia with the first 5 institutions supporting hardware distributed as shown in

Figure 2 with details in section 3. This figure also includes unfunded partners Purdue and Dresden.

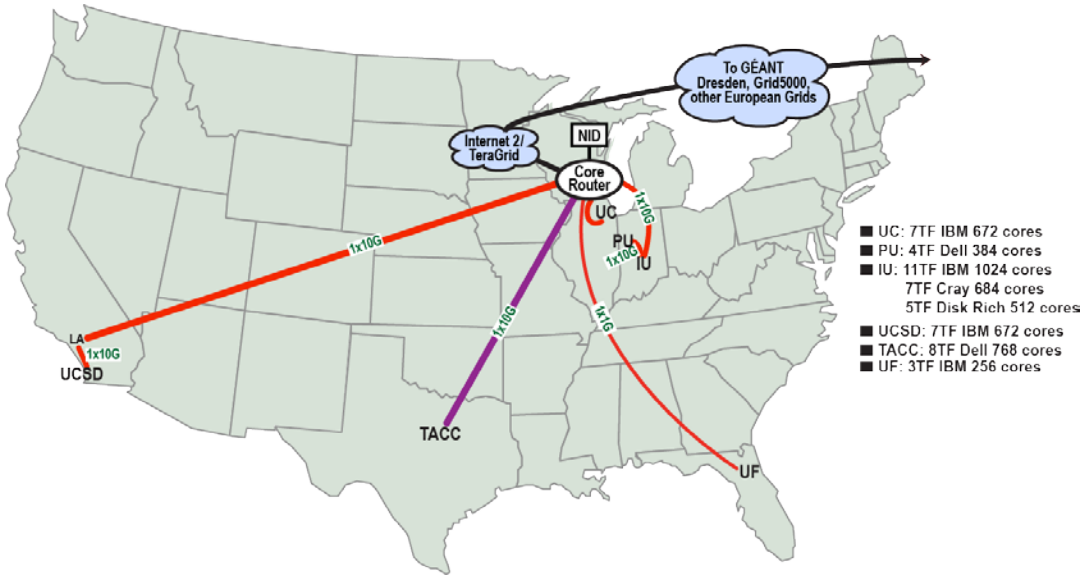


Figure 2: The FutureGrid Distributed Computing Infrastructure showing dedicated (red) and shared (purple, black) network links.

The dedicated network link shown in figure allows us to isolate FutureGrid for experiments requiring this. We also provide a Network Impairment device that's allows one to inject errors or delays into packets between selected FutureGrid nodes. FutureGrid currently consists of approximately 4000 cores that will increase by 20% in year 2 with the addition of a new system that will probably feature large disk space and memory on each node to facilitate data intensive applications. As FutureGrid is a single award all the systems in it are operated according to common principles.

FutureGrid is a partnership between 10 organizations with key people and organization functions shown in Table 1 and Table 2 respectively. Further details are given in section 2. We also have an important collaboration with Grid5000 discussed also in section 2.

Table 1: Key FutureGrid Roles

Roles	**Individuals(Institution)**
PI.	Geoffrey Fox (Indiana University)
Co-PIs.	Kate Keahey (Chicago), Warren Smith (TACC), Jose Fortes (University of Florida), and Andrew Grimshaw (University of Virginia)
Project Manager.	Gary Miksik (Indiana University)
Executive Director/Chair Operations and Change Management Committee	Craig Stewart (Indiana University)
Hardware and Network Team Lead.	David Hancock (Indiana University)
Software Team Lead/ Software Architect.	Gregor von Laszewski (Indiana University)
Systems Management Team Lead.	Gregory Pike (Indiana University)
Performance Analysis	

Team Lead.	Shava Smallen (UCSD/SDSC)
Training, Education, and Outreach Team Lead.	Renato Figueiredo (University of Florida)
User Support Team Lead.	Jonathan Bolte (Indiana University)
Funded Site Leads.	Ewa Deelman (USC-ISI), Jack Dongarra/Piotr Luszczek/Terry Moore (Tennessee), Shava Smallen/Phil Papadopoulos (UCSD/SDSC), Jose Fortes/Renato Figueiredo (University of Florida), Kate Keahey (Chicago), Warren Smith (TACC), and Andrew Grimshaw (University of Virginia)
Advisory Committee.	Nancy Wilkens-Diehr(SDSC), Shantenu Jha(LSU), Jon Weissman(Minnesota), Ann Chervenak(USC-ISI), Steven Newhouse(EGI), Frederic Desprez(Grid 5000), David Margery (Grid 5000), Morris Riedel (Juelich), Rich Wolski (Eucalyptus), Ruth Pordes (Fermilab-OSG), John Towns (NCSA).

Table 2:Institutional Roles and hosted Hardware

Institution	Hardware	Role
Indiana University	1024 Core iDataPlex, 672 core Cray XT5m Large disk/memory TBD	PI, Software Architecture and Dynamic Provisioning, Web Portal, Cloud Middleware, User support of IU and SDSC machines
Univ. of Chicago	672 Core iDataPlex	Nimbus and support UC hardware
University of Florida	256 Core iDataPlex	ViNE Virtual Networking, Training Education and Outreach, support UF hardware
TACC/Univ. Texas	768 Core Dell PowerEdge	Management Portal and support TACC hardware
UCSD/SDSC	672 Core iDataPlex	INCA, Monitoring, Performance
Purdue	384 Core HTC Cluster	Support of Purdue Hardware
Univ. of Virginia	-	Grid Middleware, OGF, User Advisory Board
Univ. of Tennessee	-	Benchmarking, PAPI
USC/ISI	-	Pegasus, Experiment Management
GWT-TUD Dresden	-	VAMPIR Performance Tool

The use of FutureGrid is requested by a convenient online form with over 70 projects listed at https://portal.futuregrid.org/projects. Some projects which are complete or have significant achievements to date are summarized in Table 3. FutureGrid use is described in more detail in section 5 and documented in Appendix A (a list of all projects) and B (some projects achieving significant progress). We have selected here projects across the spectrum of activities supported by FutureGrid; namely Education, Interoperability, Applications, Computer Science Systems and evaluation especially as applied to TeraGrid.

Table 3: Selected FutureGrid Projects

Project	Institution	Details
Educational Projects		
VSCSE Big Data	IU PTI, Michigan, NCSA and 10 sites	Over 200 students in week Long Virtual School of Computational Science and Engineering on Data Intensive Applications & Technologies
LSU Distributed Scientific Computing Class	LSU	13 students use Eucalyptus and SAGA enhanced version of MapReduce
Topics on Systems: Cloud Computing CS Class	IU SOIC	27 students in class using virtual machines, Twister, Hadoop and Dryad
Interoperability Projects		

OGF Standards	Virginia, LSU, Poznan	Interoperability experiments between OGF standard Endpoints
Sky Computing	University of Rennes 1	Over 1000 cores in 6 clusters across Grid'5000 & FutureGrid using ViNe and Nimbus to support Hadoop and BLAST demonstrated at OGF 29 June 2010
Application Projects		
Combustion	Cummins	Performance Analysis of codes aimed at engine efficiency and pollution
ScaleMP for gene assembly	IU PTI and Biology	Investigate distributed shared memory over 16 nodes for SOAPdenovo assembly of Daphnia genomes
Cloud Technologies for Bioinformatics Applications	IU PTI	Performance analysis of pleasingly parallel/MapReduce applications on Linux, Windows, Hadoop, Dryad, Amazon, Azure with and without virtual machines
Computer Science Projects		
Cumulus	Univ. of Chicago	Open Source Storage Cloud for Science based on Nimbus
Differentiated Leases for IaaS	University of Colorado	Deployment of always-on preemptible VMs to allow support of Condor based on demand volunteer computing
Application Energy Modeling	UCSD/SDSC	Fine-grained DC power measurements on HPC resources and power benchmark system
Evaluation and TeraGrid Support Projects		
TeraGrid QA Test & Debugging	SDSC	Support TeraGrid software Quality Assurance working group
TeraGrid TAS/TIS	Buffalo/Texas	Support of XD Auditing and Insertion functions

FutureGrid's success is partly measured by the number of papers published in conferences and journals and it has and will have unusual value to the Computer Science systems community compared to typical TeraGrid resources. The areas of education and interoperability have shown unexpected interest and will be fully supported. Further as seen in table 3, it also has clear value to application community and we expect that to increase for data intensive applications with a probable addition of a high memory (192 GB per node), high disk space (12 TB per node) cluster. Currently large memory is available through a 16 node ScaleMP license. We have just redesigned the FutureGrid portal and we are committed to make projects, projects, tutorials and other useful online resources fully available so all can benefit from work on FutureGrid.

FutureGrid has a complementary focus to both the Open Science Grid and the other parts of TeraGrid. FutureGrid is user-customizable, accessed interactively and supports Grid, Cloud and HPC software with and without virtualization. FutureGrid is an experimental platform with interactive interfaces where computer science applications can explore many facets of distributed systems and where domain sciences can explore various deployment scenarios and tuning parameters and in the future possibly migrate to the large-scale national Cyberinfrastructure. Educators can provide a rich and flexible environment for their students.

An important lesson from early use is that our projects require less compute resources but more user support than traditional machines. As discussed in section 6, we have revamped our user support plans and in particular all new users outside the partners are assigned "FutureGrid experts" to help them get started on our system.

FutureGrid software discussed in Section 4 builds on core components from both the partners (Nimbus, Pegasus, Inca, ViNE) and outside (HPC stack, Hypervisors, Eucalyptus, Moab, xCAT) but requires some development including software for experiment management, image generation and repository and the RAIN dynamic provisioning environment. Further we need to carefully deploy a security environment that ensures that images are properly vetted in addition to traditional functions. Initially we statically deploy operating environments but during year 2, we will switch in a staged fashion to full dynamic reprovisioning on demand as shown in Figure 1. We will also add a usage mode similar to commercial clouds where resources can be assigned from a web page and controlled by an allocation to a "FutureGrid Credit Card". Experiment management is built around Pegasus and will both control user requested provisioning and manage the provenance to enable reproducible experiments. On top of the core environment, important middleware is supported by either the partnership or users. For example Hadoop, gLite, Genesis II, Unicore and Globus are supported by the partners while Twister and Sector/Sphere MapReduce environments by users. Naturally software in the cloud arena is rapidly evolving and so are our plans; for example Eucalyptus is now less important than we originally expected while Nimbus, OpenNebula and OpenStack play a central role in open source cloud environments. Key milestones and timeline for software are given in Appendix C.

The benchmarking work on FutureGrid includes both a test-bed suite specifically designed for grid and cloud test-beds called the FutureGrid Benchmark Challenge, based on the general model of the HPCC challenge suite. However, the core technology PAPI for this area does not function effectively on virtual machines due to inherent limitations of virtual machines' inability to relay hardware counter events. Addressing this problem is outside of scope FutureGrid as it requires research and development not provided in the project. Extensively documenting and limiting the impact of this problem is a goal of FutureGrid.

FutureGrid's Training, Education and Outreach TEO team described in section 7, has coordinated the creation of tutorials for environments available on FutureGrid with an emphasis on cloud-based resources through Nimbus, Eucalyptus, and educational virtual appliances. In addition to tutorials, TEO efforts have focused on the creation of tailored virtual appliances and social group-oriented GroupVPNs that allow users to easily deploy virtual clusters on FutureGrid resources, as well as on their own desktops. Highlights of educational activities in the first year have include the first 3 class/summer school projects in Table 3 and a half-day hands-on Nimbus tutorial at the 2010 CloudCom conference. Highlights of outreach activities have included demos at major conferences, including demonstrations of deployments across FutureGrid and Grid'5000 resources for experiments with inter-cloud computing for bio-informatics applications at CCGrid, OGF and HPDC (see sky computing project in Table 3). We are also involved several HBCU students – especially as summer REU's arranged through the ADMI coalition of MSI computer science departments.

2. Management

The management structure is summarized in Table 1 and Table 2 in introduction and here we give more detail.

2.1 Organizational Roles

Organizational roles are described below for each institution with a focus on participation in the operational teams (Hardware and Network team, Software team, Systems Management team. Performance Analysis team, Training, Education, and Outreach team, User Support team) and project management. The role of the different teams is described later in this section.

Indiana University (IU). IU will be responsible for the overall management of the FutureGrid project. As the home institution of the PI, IU is ultimately responsible for the success of FutureGrid. The largest suite of hardware within FutureGrid will be located at Indiana University, and IU will chair some of the teams as defined below. IU will also lead the interactions between FutureGrid (as an instrument within the TeraGrid) and the TeraGrid (and in the future TeraGrid XD) as a whole. Particular areas of responsibility include

- **Hardware and Network team.** IU will lead the hardware management of FutureGrid. In particular, the chair of the Hardware and Network team will be located at IU (the inaugural chair will be David Hancock). IU will host an IBM iDataPlex, a Cray system, and a new system to be identified. IU will also host a centrally located Spirent network impairment device. IU is responsible for all network support.
- **Systems Management team.** IU leads the systems management team for all hardware in FutureGrid. Indiana University provides primary systems management support for the hardware at Indiana University, Florida University and SDSC.
- **Software team.** IU will chair the Software team (the inaugural chair will be FutureGrid software architect Gregor von Laszewski). IU will lead in software development, particularly as regards development of initial tools for instantiating environments on request. IU will lead the creation of the FutureGrid user portal.
- **Performance Analysis team.** IU will, via matching funds, manage a subcontract with GWT-TUD GmbH for support of Vampir for users of FutureGrid. The extent of this will be evaluated on ongoing basis as current projects have not indicated significant interest in Vampir.
- **User Support team.** IU will provide operational coordination for user support. This will include provision of information to users via an online Knowledge Base, 24 x 7 telephone support (emergency only outside 8am to 8pm Eastern Time), a trouble-ticket management system for FutureGrid, and operational activities between FutureGrid and the TeraGrid (and later TeraGrid XD) as a whole. Indiana University also leads the basic user support (currently implemented as the FutureGrid experts group) and advanced user support.
- **Training, Education, and Outreach team.** IU will have significant activities in this area with tutorials, classes and outreach to Minority Serving Institutions.
- **Project management.** PI Fox will lead this project overall. Executive Investigator Stewart will also serve in a leadership role. IU will be responsible for overall project management, including management of any and all reporting required by the NSF or

TeraGrid (and later TeraGrid XD) leadership. An IU staff member, currently Gary Miksik, will be devoted 0.5 FTE to project management of FutureGrid.

University of California – San Diego (UCSD). UCSD will lead the Performance Analysis team, participate in performance analysis activities, adapt and deploy software for systems monitoring software to aid the operation or FutureGrid, and host an IBM iDataPlex system that will be part of FutureGrid. Particular areas of responsibility include

- **Hardware and Network team.** UCSD will host an IBM iDataPlex system as part of FutureGrid.
- **Systems Management team.** UCSD provides secondary systems support for users of the hardware resource located at UCSD working together with IU as lead in this regard.
- **Software team.** UCSD will adapt and extend Inca as part of the FutureGrid management software.
- **Performance Analysis team.** UCSD will chair the Performance Analysis Committee (inaugural chair will be Shava Smallen).
- **User Support team.** UCSD will provide advanced and basic support for Inca. UCSD will also prepare Knowledge Base entries relevant to Inca.
- **Training, Education, and Outreach team.** UCSD will provide training materials relevant to use of Inca and the performance analysis tests developed by UCSD and used within FutureGrid.
- **Project management.** UCSD will participate in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled.

University of Chicago (UC). UC will be responsible for support of Nimbus for FutureGrid users, will host an IBM cluster as part of FutureGrid, and will participate in TEOS activities. Particular areas of responsibility include

- **Hardware and Network team.** The University of Chicago will host an IBM iDataPlex as part of the FutureGrid test-bed environment.
- **Systems Management team.** The University of Chicago will provide all systems administration and management required for successful operation of hardware at UC. The University of Chicago will be responsible for deployment of Nimbus within FutureGrid.
- **Software team.** The University of Chicago will be responsible for enhancements to Nimbus to fulfill FutureGrid software and deployment needs.
- **User Support team.** The University of Chicago will provide basic and advanced support for Nimbus. UC will also prepare Knowledge Base entries relevant to Nimbus.
- **Training, Education, and Outreach team.** UC will provide training materials relevant to use of Nimbus within FutureGrid.
- **Project management.** UC will participate in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled. UC will also serve as FutureGrid's liaison to the European Grid5000 project. As one of the co-PIs, Kate Keahey will participate in the leadership of the FutureGrid project.

University of Florida (UF). UF will be responsible for deployment of ViNe (Virtual Network) and related technologies within FutureGrid, particularly their use to support educational and training activities. Particular areas of responsibility include

- **Hardware and Network team.** UF will host an IBM iDataPlex system as part of FutureGrid.
- **Systems Management team.** UF provides secondary systems support for users of the hardware resource located at UF working together with IU as lead in this regard.
- **Software team.** UF will enhance the current integration of ViNe, integrating the routing layer with Nimbus so that it is easy to create self-configuring virtual networks and virtual appliances within Nimbus, and then expanding the capabilities of ViNe to also function within other cloud environments. UF will develop appliances to support key cloud and grid technologies on FutureGrid.
- **User Support team.** UF will provide basic and advanced support for ViNe and appliances developed by the Training, Education, and Outreach team. UF will also prepare Knowledge Base entries relevant to ViNe and appliances.
- **Training, Education, and Outreach team.** UF will apply virtual-appliance and social-networking–based systems developed at UF to facilitate dissemination of FutureGrid software for education, development, and testing. In particular, UF will develop self-learning educational modules that will allow teachers and students to download grid software within a virtual appliance and experiment with it on small-scale local hardware. UF will develop a how-to tutorial and support a social networking group related to FutureGrid on Facebook or equivalent social networking sites. UF will lead the Training, Education, and Outreach team.
- **Project management.** UF will participate in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled. As one of the co-PIs, Jose Fortes will participate in the leadership of the FutureGrid project.

University of Southern California (USC). USC will support use of Pegasus within FutureGrid, and work with other developers of FutureGrid software to implement experiments within FutureGrid as workflows executed via Pegasus. Particular areas of responsibility include

- **Software team.** USC will support use of Pegasus by FutureGrid users. USC will integrate Pegasus and other experiment-management systems so that grid experiments can be implemented as a workflow within Pegasus.
- **Performance Analysis team**. Pegasus will be used to collect and consolidate data resulting from performance analysis experiments, and USC will provide second-tier support for researchers who want to do performance experiments with Pegasus particularly.
- **User Support team.** USC will provide basic and advanced support for Pegasus. USC will also prepare Knowledge Base entries relevant to Pegasus.
- **Training, Education, and Outreach team.** USC will participate in outreach activities. These activities will take two forms. First, because Pegasus is capable of integrating and automating complicated workflows, it has considerable potential applicability to a broad array of domain sciences that may or may not currently be heavy users of the TeraGrid. A key component of USC's outreach will encourage domain scientists who are not currently users of the TeraGrid to experiment with Pegasus, creating workflows that automate

work now done by hand. In addition, as a leading woman computer scientist, Ewa Deelman will be involved in activities that focus on encouraging women to pursue careers in computing and science, technology, engineering, and mathematics (STEM) disciplines.
- **Project management.** USC will participate in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled.

University of Tennessee – Knoxville (UTK). UTK will develop and support tools for benchmarking FutureGrid applications. Particular areas of responsibility include.

- **Software team.** UTK has no responsibilities except those specifically related to performance analysis.
- **Performance Analysis team.** UTK will support PAPI on FutureGrid systems. UTK will modify the existing HPC Challenge benchmark test for execution across FutureGrid (and other grid and cloud computing environments). Furthermore, UTK will develop a new test-bed suite specifically designed for grid and cloud test-beds called the FutureGrid Benchmark Challenge, based on the general model of HPCC.
- **User Support team.** UTK will develop Knowledge Base entries related to PAPI, HPCC in grid environments, and the FutureGrid Benchmark Challenge. UTK will provide second-tier support for FutureGrid users making use of these tools.
- **Training, Education, and Outreach team.** UTK will develop training materials relevant to PAPI, HPCC for grid environments, and the FutureGrid Benchmark Challenge.
- **Project management.** UTK will participate in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled.

University of Texas at Austin – Texas Advanced Computer Center (TACC). TACC will host a Dell blade cluster as part of the dedicated FutureGrid hardware environment, and provide access to other systems located at TACC as appropriate. TACC will participate in the development of the FutureGrid user portal, and lead development of test harness software. Particular areas of responsibility include

- **Hardware and Network team.** TACC will manage a Dell blade cluster as part of the hardware dedicated to FutureGrid. In addition, as appropriate and as allocated by the TeraGrid Resource Allocation Committee (TRAC), TACC will make its Ranger and Spur systems available as part of grid experiments. This is not expected to include on-the-fly rebuilding of either Ranger or Spur. However, either or both systems might be used in an experiment using experimental grid workflow systems. For example, a grid experiment might involve computing at scale with Ranger as one element of a larger test. Or, a workflow system test might involve visualization with Spur as one element of a workflow.
- **Systems Management team.** TACC will provide all systems administration and management required for successful operation of hardware at TACC.
- **Software team.** TACC will participate in the development of the FutureGrid user portal. TACC will also be responsible for the creation and support of a test harness for executing experiments on FutureGrid.

- **Performance Analysis team.** No specific responsibilities other than development of the test harness to be used in performance analysis experiments.
- **User Support team.** TACC will develop Knowledge Base entries related to the test harness, and provide basic and advanced support for FutureGrid users making use of the test harness.
- **Training, Education, and Outreach team.** TACC will develop class materials that involve use of FutureGrid.
- **Project management.** TACC will participate in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled. As one of the co-PIs, Warren Smith will participate in leadership of FutureGrid.

University of Virginia (UV). UV will support use of Genesis II, Unicore, and EGEE(gLite) software on FutureGrid. UV will also serve as the primary FutureGrid liaison to the Open Grid Forum and grid-standard working groups. Particular areas of responsibility include

- **Software team.** UV will support deployment of Genesis II, Unicore, and EGEE software on the dynamically configurable FutureGrid nodes. In addition, UV will maintain stable and ongoing endpoint installations of this software on FutureGrid nodes for interoperability testing.
- **User Support team.** UV will develop Knowledge Base entries related to Genesis II, Unicore, and EGEE software, and provide basic and advanced support for FutureGrid users making use of these tools.
- **Training, Education, and Outreach team.** UV is already developing educational materials regarding Genesis II, and these will be made available to users of FutureGrid.
- **Project management.** UV will participate in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled. As one of the co-PIs, Andrew Grimshaw will participate in leadership of FutureGrid, particularly by convening and chairing the Advisory Board.

Purdue University (PU). PU will provide a 96-node high-throughput cluster for use within FutureGrid, and serve as a backup site for hosting hardware. Particular areas of responsibility include

- **Hardware and Network team.** Purdue University will provide a 96-node high-throughput cluster as part of the FutureGrid test-bed connected to FutureGrid systems via the I-light network.
- **Systems Management team.** Purdue will provide all systems administration and management required for successful operation of hardware at Purdue.
- **Software team.** Purdue University will support use of Condor and BOINC on the high-throughput cluster.
- **Project management.** Purdue will participate as requested by FutureGrid, in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled.

Technische Universitaet Dresden (TU-D). TU-D will provide limited use of one of its high performance computing systems for transatlantic distributed system activities, will participate in performance analysis activities, and will serve as a liaison to the German D-Grid project (http://www.d-grid.de). Particular areas of responsibility include

- **Hardware and Network team.** TU-D will provide limited access to its hardware facilities for transatlantic distributed system activities.
- **Systems Management team.** TU-D will provide all systems administration and management required for successful operation of hardware at TU-D.
- **Performance Analysis team.** TU-D will participate in analysis of network and grid performance between the United States and Germany, and collaborate with FutureGrid in trying to establish a suite of official SPEC benchmark applications. TU-D will also provide early access to Vampir and VampirTrace software that will particularly support performance analysis within virtual machines (VMs).
- **Project management.** TU-D will participate as requested by FutureGrid, in project management and reporting so as to ensure that reports are submitted on time and requests for information from the NSF or advisory boards are fulfilled. TU-D will serve as the primary point of contact with the German D-Grid project.

GWT-TUD GmbH. GWT-TUD GmbH will, under a contract with Indiana University funded as part of its match commitment, provide support for FutureGrid users making use of Vampir and VampirTrace software during PY2–4.

2.2 Overall Management Structure

Fox leads overall management of the project. Fox with the co-PIs forms the FutureGrid executive committee. Stewart serves as executive director for the project and Gregor von Laszewski serves as Software Architect and oversee all technical aspects of software development and integration. FutureGrid is operated as a single unified instrument with different capabilities at each site but with uniform policies and operating model. We are not replicating the current TeraGrid Forum model in which participating sites are semi-autonomous.

The management of FutureGrid is set up as a group of key management personnel and a suite of teams shown in Figure 3. each charged with leading a particular area of FutureGrid activities. There are three advisory/review committees for FutureGrid. The external advisory committee gives strategic advice while the operations committee with representatives from all parts of FutureGrid reviews situation biweekly to provide cross-team and site coordination. The executive committee with PI and co-PI's meets in person or by telecon every month. All teams and external (to IU) funded partners prepare biweekly reports that are summarized and made available to NSF.

There are a set of meetings covering individual teams plus weekly meetings for the project covering all members of FutureGrid (Tuesday 3.30pm Eastern) and the Indiana University group (Monday 9am). The Tuesday meetings alternate between All-Hands and the Operations and Change committee.

The different teams make decisions within their area which are reviewed at the weekly meetings. In general, each committee has participants from relevant participating institutions (e.g., all institutions hosting hardware as part of FutureGrid participate in the Hardware and Network and Systems Management teams, but those not hosting hardware generally do not).

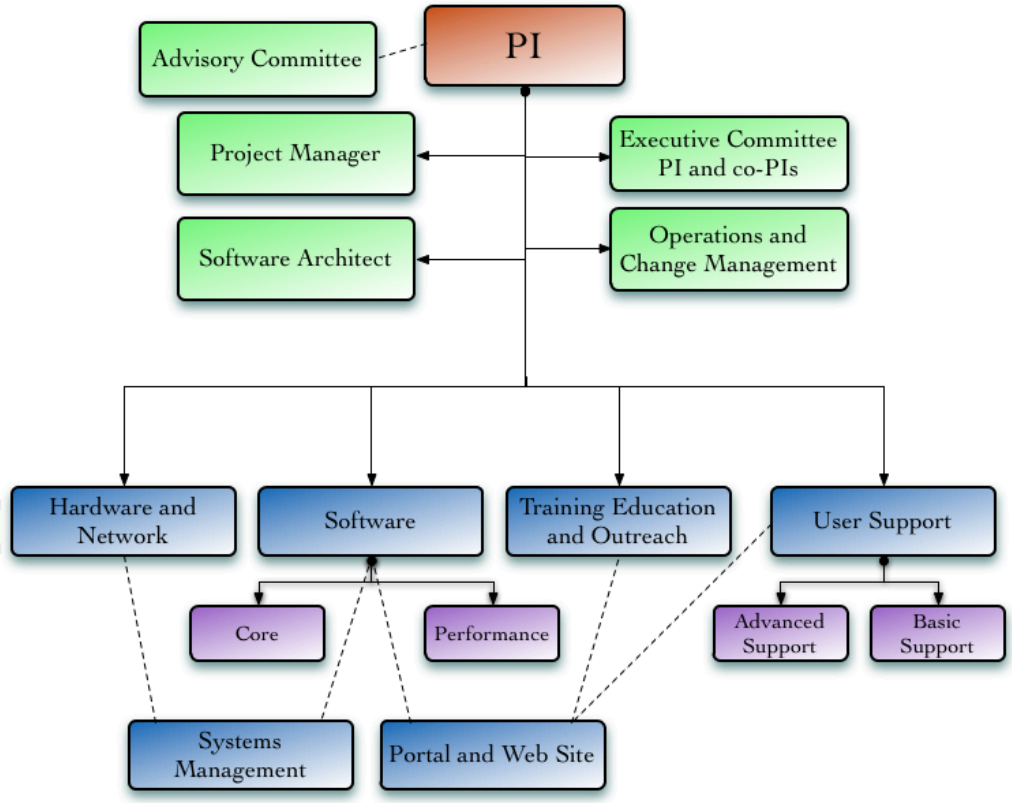


Figure 3. FutureGrid organizational structure.

2.3 Key Management Personnel

PI. Geoffrey Fox is the PI, and has overall responsibility for the project as a whole. Fox is the final arbiter of any decisions that cannot be reached by a consensus approach.

Executive Director. Craig Stewart serves as executive director, responsible particularly for leading the operations committee..

Co-PIs. Kate Keahey, Warren Smith, Jose Fortes, and Andrew Grimshaw serves as co-PIs; each has a particular leadership role within FutureGrid.

Software Architect. Gregor von Laszewski of Indiana University serves as the software architect for FutureGrid..

Project Manager. Gary Miksik serves 0.5 FTE as project manager for FutureGrid, and has management of the WBS, preparation of reports, and collection of responses to requests for information from the NSF as his primary job responsibilities.

2.4 Team and Committee Structure

Each team meets regularly with telecons every one to two weeks.

2.4.1 Advisory Committee

Advisory Committee. An external advisory committee has been set up and chaired by co-PI Andrew Grimshaw. The first meeting was in August 2010 at TG'10 and the committee consists of Nancy Wilkens-Diehr(SDSC), Shantenu Jha(LSU), Jon Weissman(Minnesota), Ann Chervenak(USC-ISI), Steven Newhouse(EGI), Frederic Desprez(Grid 5000), David Margery (Grid 5000), Morris Riedel (Juelich), Rich Wolski (Eucalyptus), Ruth Pordes (Fermilab-OSG), John Towns (NCSA).

2.4.2 Operations and Change Management Committee.

This committee is responsible for operational review of FutureGrid, and is the one committee that will always include at least one member from every participating institution, including those participating without funding. This committee will be responsible for tracking progress against the work breakdown structure (WBS), preparing reports, managing finances, and general coordination. This committee will also include the leads of other teams within FutureGrid. This committee will also serve as a Change Control Board (CCB), meeting biweekly to review and approve changes before they are implemented. (The CCB will be available to meet more often to handle ad hoc requests.) FutureGrid Project Manager Gary Miksik will chair this committee. This committee will also oversee use of the discretional 10% of FutureGrid resource usage reserved for the FutureGrid team.

2.4.3 Executive Committee.

This committee is the second highest authority within the FutureGrid management structure, second only to the PI himself.

2.4.4 Operational Teams

Hardware and Network Team. This team is responsible for all matters related to computer hardware and networking. David Hancock of IU is the chair of this team. This team was very active for first nine months of project when the hardware was being installed and validated. It works closely with systems management team.

Software Team. This team is responsible for all aspects of software design, creation and management. The FutureGrid software architect leads this team. The design of the web portal falls under this team while portal content comes from a variety of sources -- especially user support and training, education, and outreach teams. The software team also works closely with performance and systems management teams.

Systems Management Team. This team is responsible for systems administration including security across all FutureGrid sites. It is led by Gregory Pike of Indiana University.

Performance Analysis Team. This team is responsible for coordination of performance analysis activities. Shava Smallen of UCSD is the chair of this team.

Training, Education, and Outreach Team. This team coordinates Training, Education, and Outreach activities and is chaired by Renato Figueiredo of the University of Florida.

User Support Team. This team is responsible for the management of online help information (knowledge base), telephone support, and basic and advanced user support. Jonathan Bolte of IU chairs this team. The basic user support currently consists of the FutureGrid Expert group -- one

member of which is assigned to each new user. This expert group consists of a number (currently 12) of partner students, staff and postdocs who are experienced in using FutureGrid and can help new users make their initial progress. A new fulltime position has been advertised to lead user support.

2.5 Consensus Management Process

Committees and teams will operate according to a consensus process. Rather than having "yea/nay" votes, there will be four votes: Strongly in favor; in favor; opposed; strongly opposed. Consensus is declared when there is a plurality of votes in the combined categories of "strongly in favor" and "in favor" and there are no "strongly opposed" votes. This process generally works well when there is an across-the-board commitment to success and spirit of collaboration, as we expect within FutureGrid. When it is impossible to reach consensus, committee and team leads will render final decisions. Conflicts may be escalated to the executive committee. Consensus may be reached there, and when consensus even there is impossible, the PI will render a final decision. As a general rule, we expect decisions to be made quickly and do not expect stalemates in discussion.

2.6 Maintaining, Refreshing, and Executing the Project Vision

The proposal to create FutureGrid set out a vision for a cyberinfrastructure for distributed, grid, and cloud computing research. It will be important to maintain that vision and as appropriate refresh it. The Operations and Change Management committee will include representatives of all participating institutions. It is this group that will be most responsible, on a day in–day out basis, for ensuring that project execution is consistent with the annually updated Project Execution Plan PEP which serves as a statement of the vision for FutureGrid. The vision will be updated and refreshed annually by the executive committee. Input for such updating will come from the users and participants in FutureGrid, meetings of the Advisory Board, discussions at conferences including SCxx and TGxx, and discussions with the TeraGrid, as well as NSF staff. This will be documented in the Project Execution Plan updated each year.

2.7 TeraGrid

We will participate in the TeraGrid All Hands meeting, the TeraGrid Forum, the TeraGrid Quarterly Meeting, the activities of the TeraGrid Science Advisory Board, and any TeraGrid Working Groups as required by TeraGrid. Through participation in the TeraGrid Forum we share responsibility with the other TeraGrid partners for developing and implementing TeraGrid policy. Through participation in the activities of the TeraGrid Science Advisory Board we share responsibility with the other TeraGrid partners for receiving and, if appropriate, acting upon input from the national science and engineering community on the strategic planning of the TeraGrid. By participating in any TeraGrid Working Groups required by TeraGrid policy, we share responsibility with the other TeraGrid partners for coordinating user support, security policy and practice, software deployment, and usage accounting across the TeraGrid. Beginning in late 2011, the project is expected to be included in the annual TeraGrid review. Additional reviews of partners may be scheduled as needed.

2.8 Grid'5000 Collaboration

Grid'5000 is a Computer Science experimental testbed established in around 2003 and devoted to the study of large scale parallel and distributed systems. Its goal is to provide a highly reconfigurable, controllable and monitorable experimental platform to its users. The name derives from its aim to reach 5000 processors in the platform (reached during winter 2008-2009). The infrastructure of Grid'5000 is geographically distributed over 9 different hosting sites in France; Porto Alegre in Brazil recently joined Grid'5000 and is now officially the 10th site.

Because of its long-term and wide-ranging experience in the area of experiment support for Computer Science, Grid'5000 is a valuable partner for FutureGrid. Benefits of the partnership also include resource sharing resources giving access to more total resources for each individual user, access to more widely distributed resources than in the case of the individual infrastructure, as well as access to different types of resources and different networking structure. Finally collaboration with Grid'5000 fosters intellectual exchange on experimental methodology for Computer Science and ways to support it as well as provides a forum for research exchange.

To foster the collaboration and leverage the experience in running an experimental testbed accumulated over many years of operation, in 2010 FutureGrid sent a few of its members to attend the annual Grid'5000 workshop. David Hancock and Rich Knepper (IU) attended tutorials on Grid'5000 tools to assess their usefulness for FutureGrid and Kate Keahey (UC) reported on Grid'5000 policies and general project organization. They also gave talks representing FutureGrid to Grid'5000 users.

During the course of the last years several informal efforts took place to capitalize on the possibility of resource exchange. Rafael Bolze (ISI) established a FutureGrid account on Grid'5000, and a French student Pierre Riteau (University of Rennes 1) pioneered a project combining both Grid'5000 and FutureGrid resources as part of one Computer Science experiment. FutureGrid partners are now experimenting with some of the Grid'5000 software, including the Taktuk file distribution tool by TACC.

Building on these early efforts we plan to organize a workshop with the goal of establishing ways for Grid'5000 members and FutureGrid members to use each other's infrastructure. One of the workshop goals is to formalize the currently informal "cycle exchange" that is taking place on the level of individual projects by establishing accounts and allocations for sharing. Other goals include a discussion of tools and interfaces to facilitate such sharing as well as initiating a shared discussion on experimental methodology for Computer Science. Finally, we plan to establish a joint research and development forum: a research exchange venue for experimental computer science and/or ways to support experimental computer science as well as an educational forum: student exchange could be a good practical way of moving expertise back and forth. Structuring student exchange, especially in the context of specific projects leveraging either infrastructure would be good.

3. Hardware Infrastructure

3.1 Computer Hardware

FutureGrid is an unparalleled national-scale grid and cloud test-bed facility that includes a total of at least nine computational resources – six of which are new – from at least three vendors (IBM, Cray, Dell, and one to be determined), four different types of file systems, and a network

that can be dedicated to perform repeatable experiments in isolation, including a network impairment device for repeatable experiments under a variety of predetermined network conditions (see Figure 2). Also, FutureGrid is connected to an archival storage system.

Table 4: FutureGrid Hardware including new machines to be integrated

System type	# CPUs	# Cores	TFLOPS	Total RAM (GB)	Secondary Storage (TB)	Site	Status
IBM iDataPlex	256	1024	11	3072	339*	IU	Operational
Dell PowerEdge	192	768	8	1152	30	TACC	Operational
IBM iDataPlex	168	672	7	2016	120	UC	Operational
IBM iDataPlex	168	672	7	2688	96	SDSC	Operational
Cray XT5m	168	672	6	1344	339*	IU	Operational
IBM iDataPlex	64	256	2	768	On Order	UF	Operational
Large disk/memory system TBD	128	512	5	7680	768 on nodes	IU	New System TBD
High Throughput Cluster	192	384	4	192		PU	Not yet integrated
Total	***1336***	***4960***	***50***	***18912***	***1353***		

Table 5: Internal networks of operational FutureGrid machines

Machine	Name	Internal Network
IU Cray	xray	Cray 2D Torus SeaStar
IU iDataPlex	india	DDR IB, QLogic switch with Mellanox ConnectX adapters Blade Network Technologies & Force10 Ethernet switches
SDSC iDataPlex	sierra	DDR IB, Cisco switch with Mellanox ConnectX adapters Juniper Ethernet switches
UC iDataPlex	hotel	DDR IB, QLogic switch with Mellanox ConnectX adapters Blade Network Technologies & Juniper switches
UF iDataPlex	foxtrot	Gigabit Ethernet only (Blade Network Technologies; Force10 switches)
TACC Dell	alamo	QDR IB, Mellanox switches and adapters Dell Ethernet switches

Table 6: Storage systems used by FutureGrid

System Type	Capacity (TB)	File System	Site	Status
DDN 9550 (Data Capacitor)	339	Lustre	IU	Existing System
DDN 6620	120	GPFS	UC	New System
SunFire x4170	96	ZFS	SDSC	New System
Dell MD3000	30	NFS	TACC	New System

Table 4 provides an overview of the current and future systems integrated into the FutureGrid

project. Table 5 provides further details of system interconnects and names. Table 6 lists the capacity and type of storage systems connected to the FutureGrid systems at the site listed. Figure 2 shows a schematic map of FutureGrid with systems listed in Table 4 while Figure 4 shows an INCA status, which is one of many diagnostics available from this monitoring system. Figure 5 shows a real-time network performance measured by the GlobalNOC.

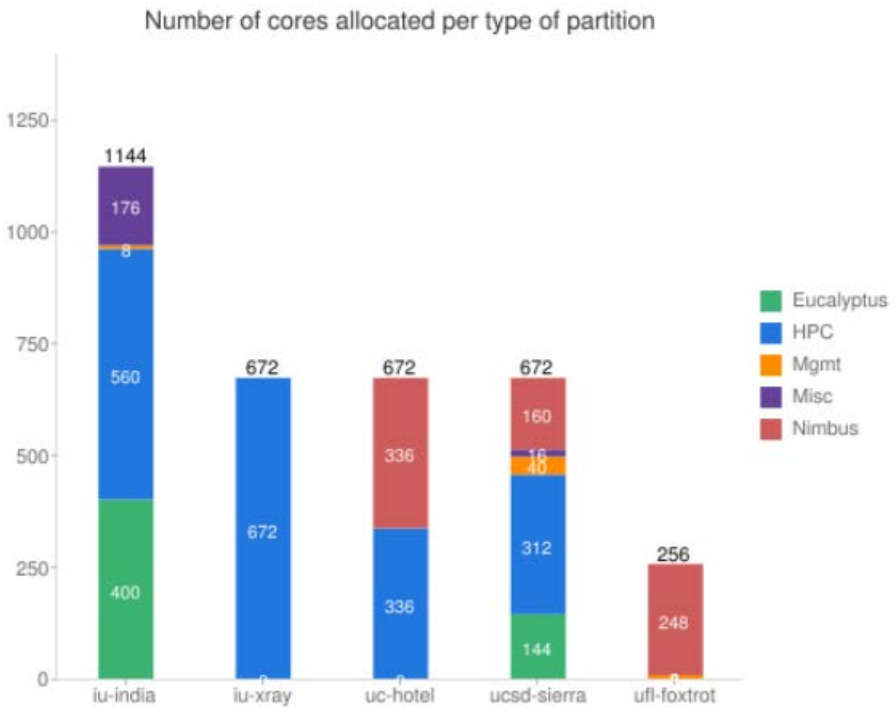


Figure 4: Number of cores partitioned per resource collected by Inca

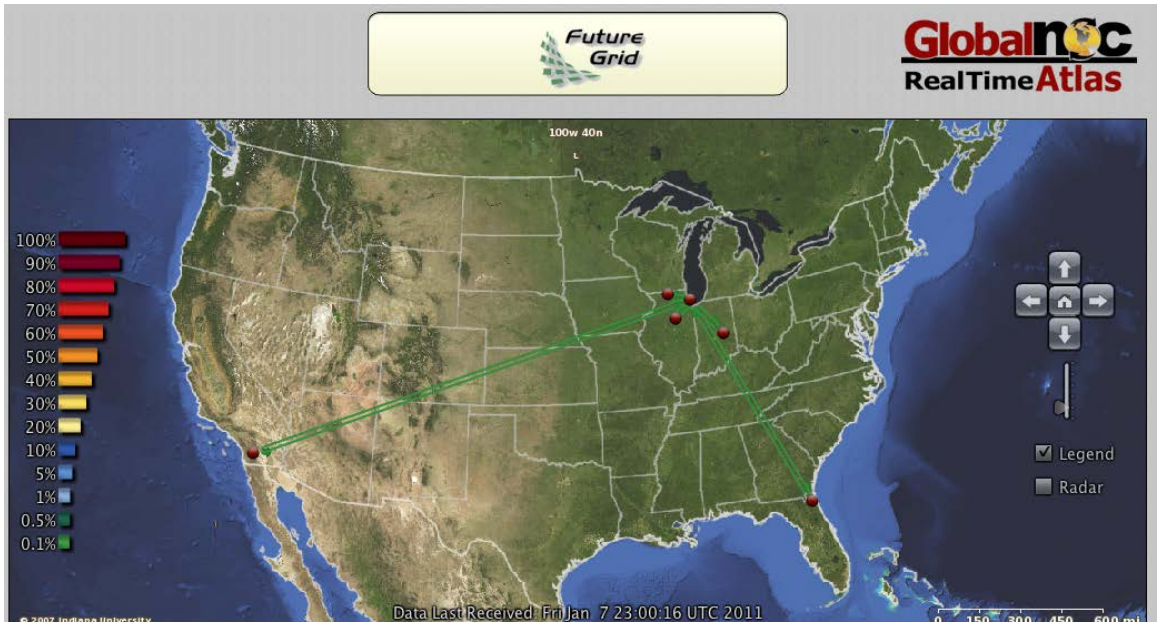


Figure 5: Real-time network performance measured by the GlobalNOC Atlas tool

3.2 Systems Integration and Transition to Operational Status Plan

The initial deployment of FG was conducted in PY1. Corresponding vendors installed systems at all sites and initial hardware diagnostics and acceptance testing were performed. Transitioning this deployment to production was initiated at the beginning of PY2 to a fully operational state. However, this does not mean that development of the facility or software stops at the end of PY1.

Efforts to integrate with the TeraGrid XD infrastructure will be addressed as soon as the XD awards have been made public and the direction of XD is communicated to FG.

3.2.1 PY1 Achievements

System Integration. During this initial implementation year, the systems were integrated into the local software infrastructure (staff accounts, local customization, file system configuration) at each partner institution. Acceptance testing was then performed. Acceptance tests included hardware diagnostics, software functionality testing, performance benchmarking, and stability testing. The exact acceptance testing criteria is discussed in the vendor contracts. The initial deployment of FG included significant problems in the acceptance testing of hardware delivered to IU and UC due to underperforming InfiniBand adapters. All systems have now been accepted and are in production.

Software Infrastructure. After systems were accepted, they were integrated into the FutureGrid-wide software infrastructure. The FG team has identified various phases for the different PY's and have now completed Phase I that was targeted for PY1.

Operations. After acceptance and before general operations, systems were evaluated and hardened for production use. During this transition period, the system was offered for use to a subset of the user community. This transition period allowed for validation of the acceptance test results under a more realistic usage pattern.

3.2.2 PY2 and Beyond Plans - Operational Phase

Table 4, Table 5 and Table 6 list the systems that are integrated into FutureGrid, with one major system to be added in PY2. FutureGrid was brought into operational mode by fulfilling the following criteria:

1. The preexisting systems to be integrated into FutureGrid and systems to be acquired in PY1 are in place and operating as part of FutureGrid;
2. The FutureGrid User Portal is in operation providing basic information and services to FutureGrid users

During the operations phase resources will be allocated as follows:

- 10% of resources will be available at the discretion of the PI.
- 90% of resources will be available for allocation to users (who may include members of the FutureGrid team) through a peer-reviewed resource request and allocation process. This process will evolve over time, as detailed below.

As some of the activities we plan include significant software development on the system, additional reservations may be necessary by the FG team to harden the production services.

As described in section 6, the resource request and resource allocation processes will operate in a preliminary learning phase during Program Years 1 and 2, with that process led by the Project Manager and PI. We anticipate that effective in PY3, it may be both possible and appropriate to transition the process for requesting and awarding resource allocations so that it is somewhat more removed from the FutureGrid team and includes additional formalized peer reviews. It seems unlikely that FutureGrid allocations can easily be mixed in with the allocation requests handled by the TRAC, due to frequency of the meetings in order to deal with the much shorter project lifetimes in FG. An operational external peer review process might meet on a monthly basis, and would most likely meet via teleconference rather than in person. It is also the case that mixing and matching resources and requests may be more complicated for FutureGrid than for the TeraGrid. Depending on the particulars, one request may require all or a very large fraction of the FutureGrid resources. Two other requests may require non-overlapping resources, and so it might be possible to fulfill two different requests simultaneously. As described for PY1 and PY2, all projects using time under the PI's 10% discretionary time will be described in a resource request submitted via the same process as any requestor, but submitted as an FYI rather than an action item request.

A critical component of the FutureGrid plan is that we will continue to enhance services over time, particularly the Actuating Services:

- Program Year 2 will add dynamic provisioning, integrated workflows from Pegasus, a storage repository for virtual environments, and scheduling integration.
- Program Year 3 will add instrumentation with Inca and Vampir for the virtual environments.
- Program Year 4 will focus on maintenance of existing technologies and incorporating user needs.

As a result of the pending TeraGrid eXtreme Digital solicitation, there is more uncertainty about future TeraGrid services and processes than usual. We expect FutureGrid to evolve over time, in response to user needs, technological changes, and the plans, processes, and procedures put into place as TeraGrid eXtreme Digital is implemented. We will develop FutureGrid plans and services so as to best meet the needs of the national science and engineering research community in the context of the TeraGrid and the NSF-sponsored national cyberinfrastructure.

3.3 Network Description

The FutureGrid network provides for interconnections among FutureGrid participants and access to the FutureGrid network impairments device (NID). Figure 6 shows the FutureGrid logical network topology.

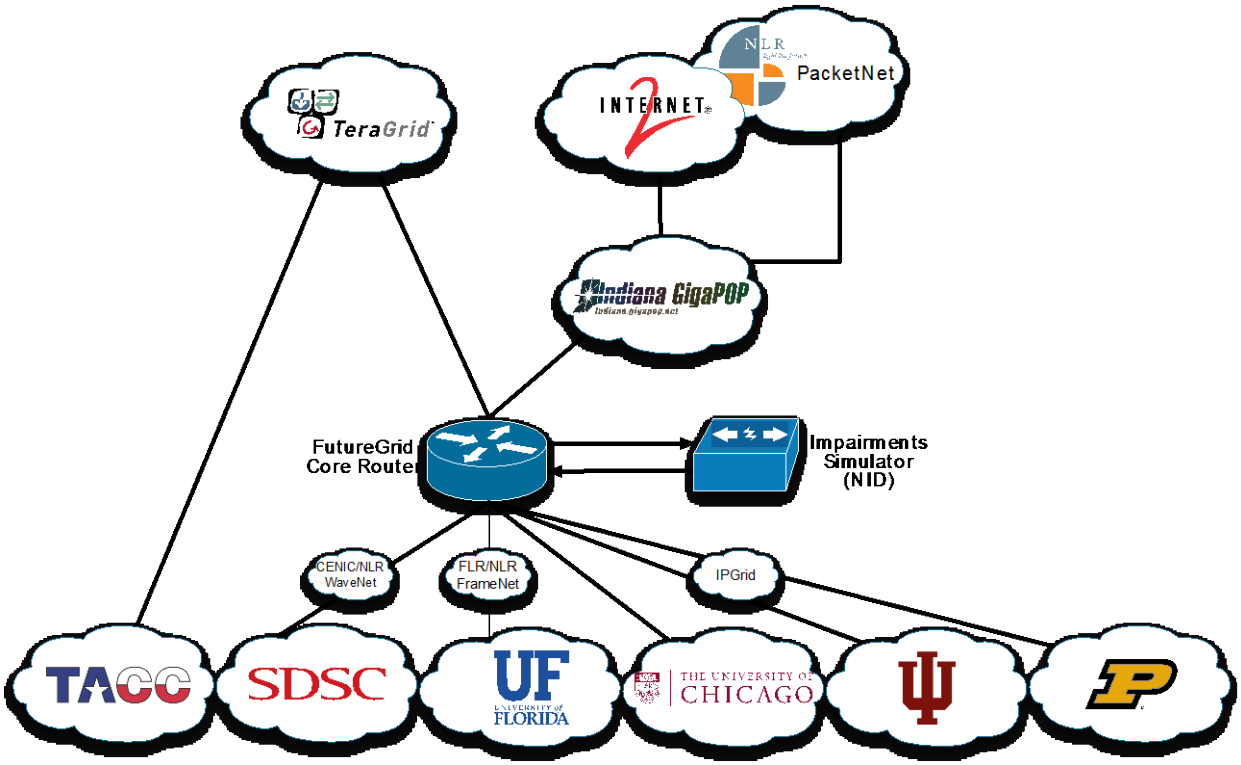


Figure 6. FutureGrid network topology.

The FutureGrid network consists of a core router, a national backbone, a network impairment device, edge networking equipment, and external peerings. The core network consists of a Juniper EX8208 router, located at the Starlight facility in Chicago. A series of dedicated links connect the FutureGrid core router with the FutureGrid participants at IU, SDSC, UC, and UF. TACC uses shared access via TeraGrid for connectivity to FutureGrid.

For IU, the dedicated 10-Gigabit Ethernet network connection to Starlight in Chicago utilizes the IP-Grid network. Purdue also connects through this same dedicated connection by leveraging the IP-Grid network to Indianapolis.

The FutureGrid network uses a 10-Gigabit Ethernet dedicated lambda from National Lambda Rail to connect SDSC to the core router, between the Starlight facility to the National Lambda Rail location in Los Angeles. From the NLR location in Los Angeles, CENIC provides a 10-Gigabit Ethernet dedicated lambda to the SDSC system.

For UC, Starlight provides a 10-Gigabit Ethernet dedicated lambda from the UC system to the location of the core router in downtown Chicago.

FutureGrid connects to National Lambda Rail FrameNet network, located at 111 N Canal Street, Chicago, through a dedicated 10-Gigabit Ethernet lambda provided by National Lambda Rail WaveNet network. FutureGrid uses the 10-Gigabit Ethernet connection to FrameNet to connect

UF via a 1-Gigabit dedicated VLAN to Jacksonville, Florida, with burst capacity up to 10-Gigabit. FLRnet then provides a 1-Gigabit Ethernet dedicated VLAN from the NLR location in Jacksonville to the UF system through a 10-Gigabit Ethernet circuit.

The FutureGrid 10-Gigabit Ethernet connection to FrameNet allows for other National Lambda Rail FrameNet users to provisiong VLAN's to connect to FutureGrid.

For TACC, their existing 10-Gigabit Ethernet connection to Chicago is utilized. The link from TACC is not dedicated, sharing their existing connection to the TeraGrid or using TACC's redundant TeraGrid connection when a dedicated link is required by an experiment.

Network Impairment Device. A Spirent H10 XGEM network impairment simulator is collocated with the FutureGrid core router to simulate the types of network impairments that might be encountered on a production network. This device was chosen because it was the only device on the market that can provide full network impairment simulation of 10Gbps flows of any packet size. This device allows us to introduce delay, jitter, and a number of different types of error and packet loss on traffic flowing through it. The Spirent interconnects with the FutureGrid core router via two 10-Gigabit Ethernet connections, to allow 10Gbps in and out of the device.

The NID is managed by the IU GlobalNOC that also operates the core FutureGrid network. Management access to the NID is restricted to internal subnets and impairments can by requested through phone or e-mail contacts to the FutureGrid NOC.

External Peerings. The FutureGrid network peers with the TeraGrid and also interconnects with the Internet2 IP network to allow access to developers and users who are not directly connected to FutureGrid. The Indiana GigaPoP provides this connectivity via existing Internet2 connections in Atlanta and Chicago.

3.4 Services Provided by FutureGrid Network

The FutureGrid network provides three services.

Isolated Interconnectivity Among Directly Connected FutureGrid Resources. The FutureGrid network's primary service is to provide interconnection between dedicated FutureGrid resources at the various FutureGrid sites. This is performed as simply as possible, using simple switching and routing among the sites, and avoiding complex inter domain routing. However, if FutureGrid users require a different configuration, the network may also be re-provisioned to interconnect sites in other ways, such as using BGP, or at Layer2, making the resources appear to be on the same subnet. Bandwidth and network reservations can also be requested to perform experiments in isolation between FG partners or within a system.

Access to Resources Outside of FutureGrid. FutureGrid also provides, via peering with external networks like Internet2, National Lambda Rail, and TeraGrid, options for sites outside of FutureGrid to provide resources to FutureGrid, when isolation and dedicated bandwidth are not as important.

Network Impairments. Lastly, the FutureGrid network allows users to introduce network impairments by selectively routing traffic through a Spirent XGEM network impairment simulator collocated with the FutureGrid core router. This allows users to introduce jitter, loss, delay, and errors into the network in a fine-grained way using Spirent's built-in TCL interface.

3.5 Service Levels

While FutureGrid is a test-bed environment, it is be crucial that the FutureGrid network perform as expected. The FutureGrid network is treated as a part of a scientific instrument, providing for availability, repeatability, and transparency. Availability of the FutureGrid network is vitally important, and any network impairments should be intentional to allow for increased repeatability of tests.

The FutureGrid network follows standard best practices for maintenance and operations to ensure high availability and predictability for the resource.

3.6 GlobalNOC Support of FutureGrid

The Indiana University Global Research Network Operations Center (GlobalNOC) supports the FutureGrid network with a hierarchy of Service Desk, software, and network engineering personnel.

The GlobalNOC Service Desk provides a 24x7x365 contact for operational aspects of the FutureGrid network, including – pro-active network monitoring, member and peer network communication coordination, incident tracking and response, incident notifications, network impairment scheduling, maintenance tracking, vendor coordination and weekly reporting.

Systems engineers within GlobalNOC provide network measurement and visulization tools, network management tools, reporting tools and performance testing support.

GlobalNOC Network Engineering provides escalation point for incident response, performance troubleshooting, advice on network implementations, coordination with FutureGrid members on site networking issues, network installations and participation in the FutureGrid hardware and networking group.

3.7 Network Milestones and Status

All network milestones were completed in PY1. These milestones include negotiation and execution of network contracts, configuration and installation of all network routing and switching hardware, deployment of the network impairments device, and configuration of external peering with TeraGrid, NLR, and Internet2.

Additionally the GlobalNOC provides all the services and support described in sections 3.4 and 3.6 and will continue to do so for the duration of the project. To date there have been no unplanned core or backbone network outages on the FutureGrid network. The only unplanned network outages have been the result of network component failures at individual sites thus not impacting all FutureGrid resources during those brief outages.

3.8 Systems Operations and Deployment

Information about the services available at each FutureGrid site will be published on a FutureGrid portal. XD will be able to point to this portal or integrate the information through iframes into the XD portal.

The current services provided on each FG hardware contains three classes, namely HPC, and provisioned software stacks, and services that are hosted at the site. Each site is responsible to provide the ability to run HPC software and provisioned software stacks. In fact the HPC

software is just like any other provisioned software stack and requires the creation through provisioned images. Due to our tight security solution while using LDAP for uniform account management we do require that each sites runs its own LDAP replica. Furthermore, each site is responsible for managing their queuing system and allow integration in a Grid based metascheduler managed by IU if needed. Users of HPC images will be able to use modules to load and unload specific enhancements. It is desirable to have an additional software stack that can enhance the ssh based services with GSI based services including GSI-SSH, GridFTP and hope such an image can be created with the help of UC.

3.8.1 Software Support and Deployment

Responsibility for support and deployment of software services on the entire FG is provided by the specific experts in the organizations constituting the FG team. We see the following logical breakdown:

- IU: MOAB, Torque, Bcfg2, xCAT, PerfSonar, Eucalyptus, Portal
- USC: Pegasus
- TACC: Experiment harness
- UC/ANL: Nimbus, CTSS
- UCSD: Inca, PerfSonar
- UF: ViNe, network appliance, Social VPN
- UTK: PAPI
- UV: Genesis II, Unicore, gLite
- Technische Universitaet Dresden: Vampir development, easy access to prerelease versions of Vampir and VampirTrace
- GWT-TUD GmbH: Vampir support

As FG is not just provisioning common HPC services, we will provide two levels of software deployment. In the first level each site will work with the systems manager and the chief architect to assure that the minimal set of services including account management, dynamic provisioning, a queuing system, and backup services are in place. We will be working towards the integration of a central repository of all software to be installed as part of deployable images. Significant changes to the base services such as HPC, Nimbus, Eucalyptus, Inca, and others will follow the FutureGrid change management procedures. One example could be de-emphasizing Eucalyptus and promoting OpenNebula or OpenStack.

We will group test environments into three categories in terms of levels of support:

1. Those for which the FutureGrid team offers extensive support and debugging of applications (TeraGrid CTSS, Eucalyptus, Nimbus, Genesis II);
2. Those for which the FutureGrid team offers some consulting support, but will also depend upon established communities of users or corporate providers for support (e.g., Windows HPC Server, Xen, VMware, EGEE/gLite; Unicore, Condor, BOINC); and
3. User-provided test environments. Those who provide their own test environments will be expected to be self-supporting. We will provide portal support for this.

Depending upon patterns of usage over time, levels of support will be adjusted to match FutureGrid researcher needs. Test environments will be instantiated in accordance with researcher needs and allocations of time on FutureGrid resources. We expect the ambient (default) state of FutureGrid systems to be as follows: The high throughput cluster at Purdue will run Condor and the CTSS. The Cray XT5m will run the vendor-recommended Linux OS and the

current version of CTSS. The rest of the resources are expected to be able to be dynamically configured and "rain" IaaS and PaaS frameworks onto their Fabric. This includes the IBM iDataPlex systems at UF and UC/ANL, the iDataPlex at IU, the iDataPlex at UCSD/SDSC, and the Dell PowerEdge at TACC.

3.8.2 Data Storage

IU

- Statically configured 339TB Lustre WAN file system. The Lustre-WAN file system is theoretically capable of sustained I/O of 3.2 GBps to locally connected FutureGrid systems. Connections from the remainder of the FutureGrid systems are limited by the bandwidth of the connection to IU of 10 Gbps.
- Statically configured 6TB HPSS test instance, 2.8PB HPSS production instance for experiment data

TACC

- Statically configured 30TB NFS file system

UC/ANL

- Statically configured 120TB GPFS file system

UCSD/SDSC

- Statically configured 96TB ZFS file system

IU, TACC, UC, UCSD, UF

- Additional storage facilities will be added during PY2 with UF and IU initial targets.

These storage systems will form a hierarchy for the dynamically configurable computational resources. We need to provide mechanisms for storing images and storing user data. To access images as part of FG we will create a convenient image repository abstraction for the different repositories, and the different file systems used. Furthermore, it allows significant space savings while providing the ability to generate images from specification descriptions rather than storing the image entirely. As part of this hierarchical view of the storage system, we will also be able to integrate an HPSS system seamlessly.
Data transfers will initially rely on a combination of Lustre WAN file system mounts from IU at each site and possibly GridFTP transfers between resources. Data storage to and from dynamic resources will be handled by Pegasus, including archiving to HPSS, after Pegasus is fully integrated into FutureGrid.

4. FG Software Overview

In this section, we will provide a brief overview of the FutureGrid (FG) software activities, which include design, implementation, and deployment. We will first introduce some of the essential motivating goals for FG software. After that, we will provide the overview of the FG architecture. Next, we will present the current status and project our plans for the next year. In addition, we will provide essential milestones.

4.1 Goals of the Software Environment

The goal set by FG is to provide a *"an experimental Grid, Cloud, and HPC testbed"*. This goal has naturally a direct impact on our software design, architecture, and deployment. Hence, we

revisit some of the elementary requirements influenced by the user community, various access models and services, and the desire to be able to conduct a variety of reproducible experiments.

Support a Diverse User Community

As part of our initial investigations, we have identified a number of different user communities that will benefit from a testbed such as FG. Naturally, the desire to support these communities governs our software design and the access to FG in general. We intend to support the following communities:

- ***Application developers*** that investigate the use of software and services provided by FG;
- ***Middleware developers*** that investigate the development of middleware and services for Cloud and Grid computing;
- ***System administrators*** that investigate technologies that they wish to deploy into their own infrastructure;
- ***Educators*** that like to expose their students to software and services to Cloud and Grid computing technologies as offered on the FG; and
- ***Application users*** that like to test out services developed by application and middleware developers.

Support for Shifting Technology Base

In Section 1, we have introduced a number of compelling examples that motivate the creation of a testbed such as FG. One of the observations that motivated FG is the rapidly developing technologies in the Grid and Cloud space that may have profound impact on how we develop the next generation scientific applications keeping these new developments in mind. The introduction of virtualization, Infrastructure as a Service (IaaS), and Platform as a Service (PaaS) paradigms calls for the ability to have access to software tools and services that allow a comparison of these paradigms with traditional HPC methodologies.

Support a Diverse Set of Interface Methods

Based on this technology shift and the interest posed by the various user communities to have easy interfaces to a testbed, we need to develop, as part of our software activities, appropriate interface tools and services. The interfaces include *command line tools, APIs, libraries* and *services*. In addition, many users would like access to these new technologies through portals or GUIs.

Support a Diverse Set of Access Methods

While in previous decades the focus has been to provide convenient libraries, tools, and Web services we currently see an expansion into infrastructure and platform as services. Thus, a new generation tools and services are provided as abstractions to a higher level of services, potentially replacing the traditional OS. Thus, we will not only offer access to Infrastructure as a Service (IaaS) framework, but we will also invest in providing PaaS endpoints, thereby allowing access to a new kind of abstraction.

Support a Diverse Set of Access Services

Due to the rapid development of new tools, services, and frameworks within the Grid and Cloud communities, it is important to facilitate a multitude of such environments. This includes access to Infrastructure as a Service (IaaS) frameworks such as Nimbus, Eucalyptus, OpenNebula, OpenStack; Platform as a Service (PaaS) frameworks such as Hadoop and Dryad; and additional services and tools like Unicore and Genesis II, that are provided and supported by the FG team members. Hence, users will have the ability to investigate a number of different frameworks as part of their activities on FG.

Support of Traditional Services

We provide a number of additional services that users are accustomed to. This includes High Performance Computing (HPC) services, but also access to backup and storage. Naturally, we provide services for user support as part of a portal with access to information including a ticket system.

Support for Persistent Services and Endpoints

One of the potential assets of FG is the ability to expose a number of students and practitioners to the new frameworks offered. However, the entry to such systems may have to be low in order to interest others in using such technologies. Hence, it is important to offer a number of persistent services and endpoints of such frameworks. Furthermore, we must make it easy for the teachers and administrators of FG to manage membership and access rights to such endpoints. In addition, we are interested in providing a number of "standard" images for educational purposes that can be used for teaching about particular aspects.

Support for Raining/Dynamic Provisioning

As we are not only interested in offering pre-installed frameworks exposed through endpoints, we must provide additional functionality to instantiate and deploy them on demand. Therefore, we need to offer dynamic provisioning within FG not only within an IaaS framework, such as Nimbus, Eucalyptus or OpenStack, but also allow for the provisioning of such frameworks themselves. We use the term "*raining*" instead of just dynamic provisioning to indicate that we strive to dynamically provision even the IaaS framework or the PaaS framework. In addition, we also will allow the efficient assignment of resources to services governed by user needs. Thus, if there is no demand for running Eucalyptus staged images, the resources devoted to the Eucalyptus cloud can be de-registered from it and assigned to a different service. An additional aspect of our "rain" tool is that we can specify the mapping onto specific resources, allowing us to compare services on the same hardware.

Support for a Viral User Contribution Model

User contributions are possible at several levels. First, the development of shared images that are instantiated as part of an IaaS framework. Second, the development of middleware, either in the IaaS or PaaS models, that can be rained onto FG. Third, the creation of workflows that utilize a combination of the services offered as part of sophisticated experiment workflows, which we will explain in more detail later. Fourth, through the contribution of educational material.

Differentiation to Existing Services

One of the important features to recognize is that FG distinguishes itself from current available systems. This includes traditional compute centers such as TeraGrid, but also well known IaaS offerings, such as Amazon.

In contrast to Amazon, we provide alternatives to the IaaS framework, but the biggest benefit stems from two unique features of FG. In FG we intend to allow the mapping of specific resources as part of the service instantiation. Thus, we can measure more realistically performance impacts of the middleware and the services developed by the FG testbed user. Furthermore, we allow authorized users a much greater level of access to resources by allowing the creation of images that can not only be placed in a VM but also be run on the "bare" hardware.

A big distinction between TeraGrid and FG is that FG provides a greater breadth of services. Traditionally, supercomputing centers that are part of TeraGrid focus on large scale high performance computing applications while providing a well defined software stack. Access is based on job management and traditional parallel and distributed computing concepts. Virtual machine staging on TeraGrid has not yet deemed to be a major part of its mission. FG is more flexible in providing software stacks which allow the software stack to dynamically adapt to the users' needs.

Provide Management Capabilities for Reproducible Experiments

One of the important concepts of FG is the focus on experiments that ideally lead to results that can be shared with the user community. Hence, activities within FG will be primarily experiment-based. Experiments may vary in complexity. They may include basic experiments, such as to utilize a particular pre-installed service and let a researcher debug an application interactively. They may also include more sophisticated experiments, such as instantiating a particular environment and running a pre-specified set of tasks on the environment. We envision that a direct outcome of having such an experiment-centric approach will be the creation of a collection of software images and experimental data that will provide a reusable resource for application and computational sciences. FG will thus enable grid researchers to conveniently define, execute, and repeat application or grid and cloud middleware experiments within interacting software "stacks" that are under the control of the experiment owner or user. It will also allow researchers to leverage from previous experiences of other users FG will support these pre-configured experiment environments with explicit default settings so that researchers can quickly select an appropriate environment and use it within their specific scenario. To better communicate the scope of the experiment related activities, we introduce a common set of terminology:

> **Project.** A project represents an elementary "execution unit". A project has a particular scientific goal in mind that may need the execution of one or more experiments. An example of a project is to teach Cloud computing as part of a class at a university.
>
> **Experiment.** An experiment contains a number of important metadata: experiment session, the resource configuration, the resources used (apparatus), the images used, deployment specific attributes, the application used, the results of the experiments

(typically files and data), and the expected duration of the experiment. Experiments may be organized in a tree or direct acyclic Graph (DAG) and contain other experiments. An example of an experiment is running a Hadoop job as part of an academic class. If we view the class as a project, then each job submitted by a student could be viewed as an experiment.

Experiment Management. Experiment management refers to the ability of a testbed user to define, initiate, and control a repeatable set of events designed to exercise some particular functionality, either in isolation or in aggregate.

Experiment Apparatus. Often it is desirable to conduct parameter studies or repetitive experiments with the same setup in regards to resources used. We refer to such a configuration as an "experiment apparatus". Such an apparatus allows the users to conveniently reuse the same setup without reconfiguration of the FG resources for different experiments.

Experiment Session. Besides the apparatus, we often find that the apparatus can be used for executing a number of experiments. In addition, the instantiation of experiments may require additional configuration in order to address runtime issues. Together the apparatus and the configuration parameters are building an experiment session that as mentioned can be used throughout multiple experiments.

4.2 Software Architecture

To support such an infrastructure, we have designed a software architecture that is expandable and provides the ability to integrate new resources and services. Our architecture is summarized in Figure 7. The architecture is based on conceptual layers enabling us to gradually introduce new features to the users over time and to assure that teams can develop software and services in parallel. Next, we will give an overview of each of the components that constitute this architecture.

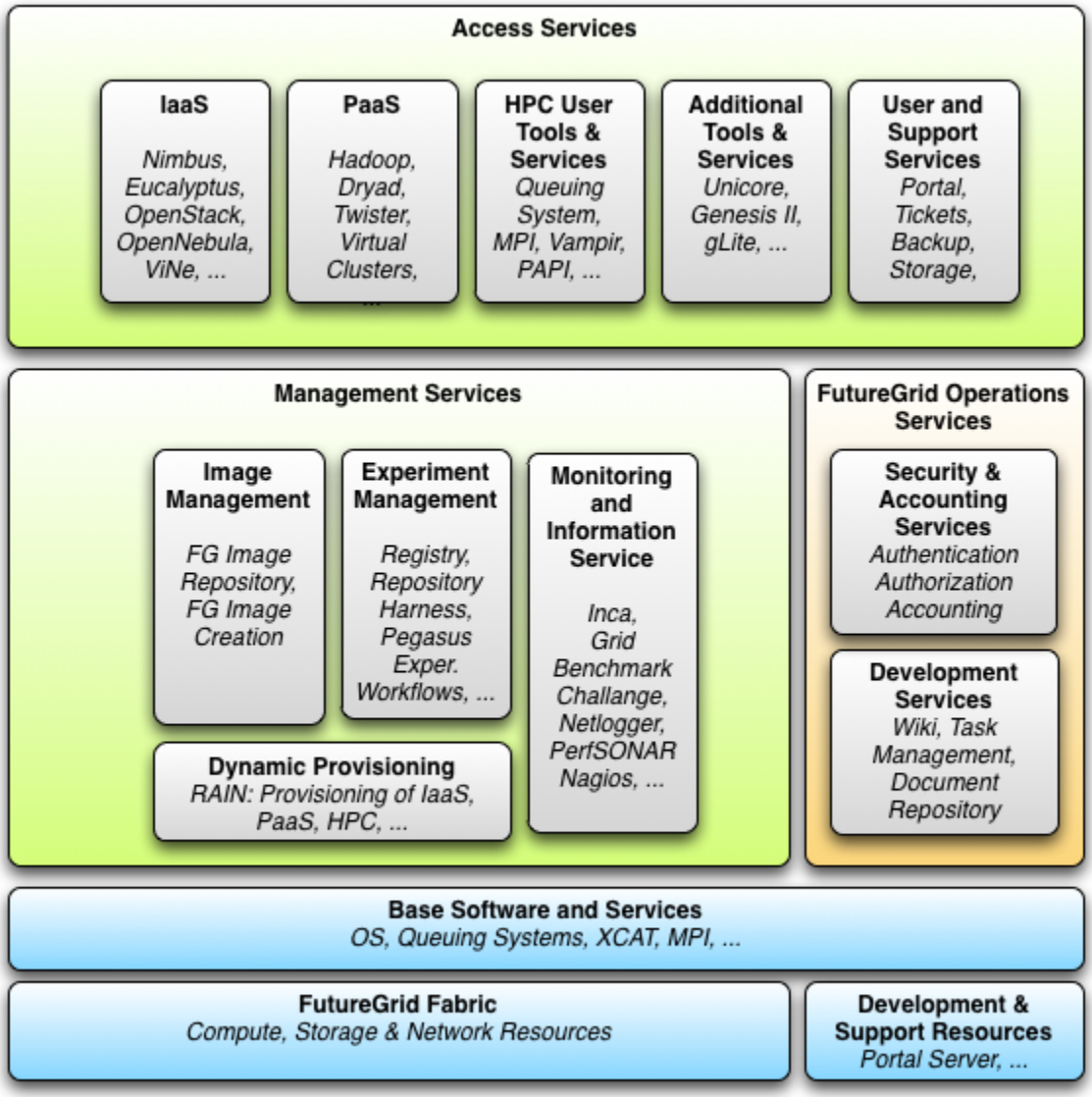


Figure 7: FG Software Architecture

4.2.1 Architectural Overview

We distinguish the following components:

FG Fabric: The Fabric layer contains the hardware resources, including the FG computational resources, storage servers, and network infrastructure including the network impairment device.

FG Development and Support Fabric/Resources: Additional resources are set aside that are helping us with the development and support of operational services. This includes servers for portals, ticket systems, task management systems, code repositories, a machine to host an LDAP server and other services. It is important to recognize that such services should not be hosted on

the "cluster" resources that constitute the main FG Fabric as an outage of the cluster would affect important operational services.

FG Base Software and Services: The FG Base services contain a number of services we rely on when developing software in support of the FG mission. This includes Software that is very close to the FG Fabric and includes tools like MOAB, XCAT, and also the base OS. This category of services will enable us to build experiment management systems utilizing dynamic provisioning.

FG Management Services: The management services are centered around FG experiments and the overall system integration, including information services and raining/dynamic provisioning software stacks and environments on the Fabric.

FG Operations Services: In order to effectively communicate and conduct development effort, the following elementary services have been provided: a website, a development wiki, a task management system to coordinate the software development tasks and a ticket system. In addition, we need to provide security and accounting services to deal with authentication, authorization and auditing.

FG Access Services: FG user services contain variety of services. They include IaaS, PaaS, SaaS, and classical Libraries that provide a service as an infrastructure to the users such as accessing MPI and others.

FG User Contributed Services (as part of additional Services): The architecture image does not explicitly distinguish user contributed services. It is important to note that user contributions take place on many different access levels. This is supported by our architecture by allowing the creation, distribution, reuse and instantiation of user contributed software as part of services or experiments within FG. Thus, we expect that the FG User Contributed Services will grow over time while enhancing areas that we have not explicitly targeted ourselves. Instead, we provide mechanisms for community users to integrate their contributions into FG offered services. The only difference to these services may be the level of support offered in contrast to other FG services.

4.2.2 FG Access Services

4.2.2.1 IaaS

As part of our initial efforts we have deployed two different IaaS frameworks, namely Nimbus and Eucalyptus.

4.2.2.2 Nimbus

Nimbus is an open source toolkit developed at the University of Chicago providing cloud computing tools for scientific and educational projects. The functionality provided by Nimbus focuses around providing the following three capabilities:

(1) An *implementation of Infrastructure-as-a-Service (IaaS):* in this space the Nimbus Workspace Service and Cumulus components provide science-friendly and industry standard compatible implementations of compute and storage cloud respectively

(2) *Higher-level ecosystem tools enabling IaaS clients to leverage and combine IaaS offerings*, via e.g., enabling creation of virtual clusters and scaling tools: the Nimbus Context Broker fulfills this function for many projects.
(3) *High-quality, extensible open source implementation* that enables others to actively participate in Nimbus development experimenting with new features and solutions: in this space we provide not only extensible code but nourish a vibrant contributor community with many successful contributions in the past two years.

FutureGrid uses Nimbus in all three capacities. Our current resource offering has three Nimbus installations: on hotel, sierra and foxtrot. They have been used by multiple early adopter projects, some of them leveraging the opportunity to experiment with federated clouds (see e.g., [FG-P74] in Appendix A) sky computing" early adopter project). These clouds have also been used in various educational activities, e.g. a hands-on Nimbus tutorial at the CloudCom 2010 conference.

In the past year, our main focus area was to support Nimbus installation on FutureGrid and respond to the requirements concerning Nimbus integration into the FutureGrid fabric. The most important highlights from this period include: an improved installer and a "zero->cloud" installation process, both to facilitate Nimbus installation on FutureGrid resources, developed tools and scripts to integrate Nimbus credential distribution process into the FutureGrid credential distribution process, and the development of dynamic node management. We also implemented multiple smaller features resulting from direct requests of our users and collaborators. In addition, we prepared documentation and tutorials for FG users and supported demonstrations, exploration, and early users of FG.

During the early adoption phase we identified two distinct ways in which users want to use Nimbus on FutureGrid: (1) as an IaaS platform to experiment on top of, and (2) as an experimental IaaS implementation to experiment with. We plan to support both and in the following year we plan to particularly focus on the latter: making Nimbus an excellent experimental tool for IaaS. In addition, we also identified multiple requirements provided either by our collaborators on FutureGrid (e.g., multi-cloud to support the experimental harness) or by our users (e.g., specific resource management and debugging features) that we plan to provide in the coming year.

We will also include activities for improving the storage management while taking the FG Fabric into account. Furthermore, we will be integrating the Nimbus image repository with a more generalized image repository that we envision for FG (see Section 4.3.3). Additionally, we will investigate the dynamic provisioning of a Nimbus deployment controlled by a particular user group.

4.2.2.3 Eucalyptus

Eucalyptus is a popular IaaS framework that has been commercialized since this project started. It is very popular and has become available as part of the Ubuntu service offerings. Our current deployment contains two independent services on India and Sierra allowing research in the development of heterogeneous cloud environments such as conducted by the Pegasus team to support a sophisticated experiment management system. However, we would like to replace this setup with a single Eucalyptus install with multiple availability zones. This will simplify the user management as well as the management of resources within the Eucalyptus deployment. It will also offer users with the ability to experiment with disperse resources as part of the deployment.

Additional enhancements will include integration with storage resources, the FG image repository and the better integration into the FG account management. We have delayed the last activity due to the important development that will allow in near future to integrate OpenID into the Eucalyptus security framework. We expect that Eucalyptus, like Nimbus, will be used heavily in educational activities on FG. Hence, we are especially interested to identify mechanisms on simplifying the access. Through a community effort we will collaborate with the iPlant project and identify if we can provide access through FG Eucalyptus installations via the Atmosphere project that provides a convenient graphical user interface. Similar to "Elasticfox" we would like to provide users with the ability to control their IaaS interactions but eliminate the dependency on the browser and enhance it with features needed to support our experiment management. This includes, but is not limited to, providing features to list available images, move them into the Eucalyptus repository, list running instances, launch new instances, manage the security groups manage storage, and record and manage reproducible experiments on FG while using the Eucalyptus cloud.

4.2.2.4 OpenNebula

OpenNebula is an open-source toolkit which transforms existing infrastructure into an Infrastructure as a Service (IaaS) cloud with cloud-like interfaces. It is an important framework, as it receives significant funding from the European Union and is supported by a very active community. The integration of OpenNebula in FG will be done in two different ways. First, we plan to deploy OpenNebula as a service onto FG just as we have deployed Nimbus and Eucalyptus. Initially, user authentication will be done via ssh-keys that we intend to access through our LDAP server. We have therefore started to collaborate with the OpenNebula team to evaluate how to improve this integration, as the current OpenNebula LDAP solution needs extensions to allow LDAP attribute verification. Moreover, we will have to evaluate the use of clear text passwords in OpenNebula and identify how this affects our security infrastructure. Our initial plan may be changed while isolating OpenNebula and requiring users to have an independent account for access to OpenNebula. Additionally, as part of this deployment, we will also provide some default virtual networks and virtual machines that will help users to start working with this framework.

Furthermore, we intend to offer OpenNebula through the experiment management framework while dynamically provisioning it. This would have the advantage of enabling a personalized and isolated version of OpenNebula allowing to utilize it to conduct developments and enhancements to the core OpenNebula services, or experiment with new versions that have not yet been deployed by FG.

4.2.2.5 OpenStack

Most recently OpenStack has been introduced to the community. In contrast to other IaaS frameworks, it has not yet been released into full production mode. However, this is subject to change this year. FG will gain experience with OpenStack while providing an initial deployment. As we expect OpenStack to change rapidly, we will focus our efforts on dynamic provisioning OpenStack on FG. Similar to OpenNebula we will be exploring the integration with our LDAP server.

4.2.2.6 ViNe

ViNe is middleware developed to implement routing and other communication mechanisms needed for virtual networking. ViNe allows the establishment of wide-area virtual networks supporting symmetric communication among public and private network resources (and others behind firewalls or gateways), does not require changes to the physical network, does not require modifications to OS, and has the best performance among wide-area virtual networks. ViNe relies on a set of virtual routers, (implemented as user-level software) to forward its traffic, which are easy to integrate into site infrastructure.

A machine running ViNe software becomes a ViNe router (VR), working as a gateway to overlay networks for the machines connected to the same LAN segment. Delegating the overlay network processing to a specific machine is the recommended deployment of ViNe so that compute cycles are not stolen from compute nodes for overlay network processing, a scenario that can occur if all nodes become VRs.

Currently one machine on Foxtrot and one machine on Sierra are configured as VRs. For Hotel and India, VMs configured as VRs are deployed when needed. Deployment of ViNe on FG has been demonstrated during CCGrid 2010, OGF29, and TeraGrid 2010, when VMs deployed across FG and external (Grid'5000) sites were connected to form a virtual Hadoop cluster to run CloudBLAST (a version of bioinformatics application modified to run on Hadoop). When using over 750 VMs (approximately 1500 cores), it was possible to reduce the time to execute a BLAST computation from over 250 hours (if executed sequentially) to less than 20 minutes.

Currently, ViNe work focuses on improving its management capabilities, so that end users, without the need of networking expertise, can configure overlay networks as needed. Combined with the dynamic provisioning of FG, ViNe will offer FG users the needed communication support to connect FG and external nodes.

4.2.2.7 PaaS

In addition to the IaaS frameworks, we also observe the creation of "Platform as a Service" or PaaS frameworks. They do provide a new way of loking at how to deliver applications while relying on platforms that allow the utilization of convenient programming platforms and the abstraction of the infrastructure. Examples are Hadoop and Dryad.

4.2.2.8 Hadoop

We have developed in PY1 a prototype mechanism of instantiating a Hadoop environment through the HPC queuing system. This allowed the isolation of the instantiated environment to conduct performance experiments to compare them with MPI based applications using a traditional HPC environment. The system has been successfully tested for delivering a fault tolerant implementation of a water threat management application while using the mechanisms provided by Hadoop to avoid duplication within a user provided framework. Our plan is to further develop this prototype and bring the "dynamic deployment" of the Hadoop sandbox environment to the users of FG. Such a setup is suitable for further development of Hadoop and conducting isolated performance studies as opposed to a shared Hadoop environment. Hadoop virtual machines were also developed and deployed on the Eucalyptus cloud, allowing users to instantiate a custom full Hadoop environment with ease.

4.2.2.9 Dryad

Dryad provides a PaaS framework as part of the Microsoft operating system. In PY1, we have demonstrated that such an environment can be instantiated through a dual boot strategy, while on one partitioning hosting the Windows OS and on the other hosting the Linux OS to provide other services. However, although switching between the systems is done relatively fast, this strategy is limited. First, the environment was statically deployed and switching between the OS was made possible while giving the user control over a number of dual bootable servers. Second, due to the dual boot strategy the amount of space that is offered to the user of either one of the systems is significantly impacted. The lessons that we learned from this initial experiment is that we need to investigate who to (a) provide more sophisticated and integrated tools to allow the assignment of resources to experiments form the command line, (b) allow the partitioning to be part of a feature that is exposed within our experiment management, and (c) reduce the work for system staff from offering such features as a one time experiment setup by having the system administrators contribute scripts to the software environment that makes such management easier with tools already available such as Moab and XCAT. These new features are then exposed through a command line script that we call “rain”.

4.2.3 HPC User Tools and Services

Naturally we will also access FG through typical HPC queuing services. Each of the resources in the FG Fabric will have that ability to submit jobs as part of a queuing system. All resources but Alamo use Moab, while TACC is using the Bright Cluster computing solution for Alamo.

To help users analyze the behavior of their application on FG, a set of performance tools will be provided in the user's environment to help them optimize application performance. Support of these tools is divided into two categories: full and best effort. Full support is provided for tools that are supported by the FG team and include Vampir and PAPI. A number of other performance tools have also been identified and support of these tools will be provided at best effort; these tools currently include TAU, IPM, Scalasca, Oprofile, and Marmot. All performance tools will be packaged as .deb and .rpm files so they can be integrated into the FG image generation process and deployed to both bare metal and virtual environments. However, performance information will be limited in virtual environments since performance counters are not yet virtualized. Currently, PAPI is deployed on the Xray resource and will be deployed to other HPC resources after they are upgraded to Redhat 6. The VampirServer runs on the India machine and the VampirTrace libraries are currently deployed on India, Xray, and Hotel.

4.2.3.1 PAPI

PAPI is an acronym for Performance Application Programming Interface. The PAPI Project is being developed at the University of Tennessee’s Innovative Computing Laboratory in the Computer Science Department. This project was created to design, standardize, and implement a portable and efficient API to access the hardware performance counters found on most modern microprocessors. PAPI will be deployed as one of the performance tools for FG. Enabling researchers to explore the power and potential of FG’s distributed and virtualized infrastructure requires performance analysis tools that can help them 1) find where the performance bottlenecks are in their particular applications and 2) establish benchmarks for meaningful performance comparisons between different parts and/or resource configurations of FG. Above we have specified how the work on the two tool sets that UTK will provide for this purpose —

PAPI and the FutureGrid Benchmark Challenge (FBC) — map into the FG management scheme. However, this specification ignores a complication that should be noted: In virtualized environments like FG, tasks 1) and 2) need to be attacked at both a) the level of the underlying hardware and b) the level of the VM's, which virtualize the underlying (heterogeneous) platform infrastructure and offer applications an apparently homogeneous execution environment. Tasks 1a) and 2a) are, for most purposes, solved problems; UTK will help the FG community deploy and use the tools that we have developed to solve them. Tasks 1b) and 2b), however, are not solved problems.

In the case of 1b), for example, the use of PAPI to access and utilize hardware performance counter information on traditional clusters and supercomputers is ubiquitous in the HPC community. But virtual machine platforms generally, and those in use on FG specifically, do not pass on the information on counter events that PAPI relies upon. Some VM software stacks were designed expressly to hide the underlying hardware, whereas PAPI requires them to expose it. Some VM designs are open to exposing/relaying hardware information, but the cost of repeatedly porting such a feature, as hardware platforms continue to evolve, is usually deemed by their designers to be unsustainable and, consequently, this feature tends to be neglected. While addressing this basic problem is outside of the scope our FutureGrid effort, we will work to extensively document its effects and limit its impact on the FG research community. In the case of 2b), a satisfactory benchmark suite for FG, one that builds on standard hardware platform benchmarks, must be able to probe running applications to determine a VM's contribution to an application's performance profile. Thus, as in the previous case, FG's addition of VM environments significantly complicates the problem of achieving the kind of rigor that good, consistent benchmarks require.

4.2.3.2 Vampir

Performance optimization is a key issue for the development of efficient parallel software applications. Vampir provides a manageable framework for analysis, which enables developers to quickly display program behavior at any level of detail. Detailed performance data obtained from a parallel program execution can be analyzed with a collection of different performance views. Intuitive navigation and zooming are the key features of the tool, which help to quickly identify inefficient or faulty parts of a program code. Vampir implements optimized event analysis algorithms and customizable displays which enable a fast and interactive rendering of very complex performance monitoring data. Ultra large data volumes can be analyzed with a parallel version of Vampir, which is available on request. Vampir has a product history of more than 15 years and is well established on Unix based HPC systems. This tool experience is now available for HPC systems that are based on Microsoft Windows HPC Server 2008. This new Windows edition of Vampir combines modern scalable event processing techniques with a fully redesigned graphical user interface.

4.2.4 Additional Tools & Services

4.2.4.1 Unicore

Unicore has been deployed on FG Xray. It is used for interoperability testing of Grid software. Plans will include the update of the software if needed. The installed version supports OGSA-BES & HPC-Basic Profile. Access to this endpoint was enabled through a port so that external clients can connect.

4.2.4.2 Genesis II

Genesis II is an open source, standards-based Grid platform designed to support both high-throughput computing and secure data sharing. Genesis II endpoints have been installed on India, Sierra, and Xray. OGSA-BES endpoints were successfully used by UK group to test their OGSA-BES client for interoperability for Open Grid Forum demonstrations in the Grid Interoperability Now (GIN) working group. Next steps will include the creation of information about the endpoints and how to use them as part of FG.

4.2.4.3 gLite

We intend to deploy gLite on FG in PY2.

4.2.4.4 Pegasus

In the context of FG, Pegasus Workflow Management System (WMS) is of interest to users to facilitate FG resources outside the Experiment Management framework, i.e. pure Pegasus "as a Platform". This may be especially useful to users already familiar with Pegasus. Pegasus has been used to run workflows ranging from just a few computational tasks up to 1 million, in a number of domains ranging from astronomy, biology, earthquake science, to physics and others. One or more workflows can run on as little as a single system or as large as across a heterogeneous set of resources. When errors occur, Pegasus tries to recover when possible using various strategies, including forms of retries, re-planning, and, when all else fails, by providing a rescue workflow containing a description of only the work that remains to be done. It cleans up storage as the workflow is executed so that data-intensive workflows have enough space to execute on storage-constrained resources. Pegasus keeps track of what has been done (provenance) including the locations of data used and produced, and which software was used with which parameters. Pegasus WMS bridges the scientific domain and the execution environment by automatically mapping high-level workflow descriptions onto distributed resources. Pegasus enables scientists to construct workflows in abstract terms without worrying about the details of the underlying execution environment or the particulars of the low-level specifications required by the middleware. Pegasus WMS also bridges the current cyber-infrastructure by effectively coordinating multiple distributed resources provisioned from heterogeneous middlewares. With Pegasus being part of the Experiment Management, it allows us to automatically test various aspects of the FG infrastructure, including VMs, authentication, data and job management services and others. Our plan for the next year includes offering Pegasus as a Platform in production.

User Contributed Community Activities

We have already contacted several groups that are interested in offering their tools and services on FG. This includes the following activities: During OGF a group that implements the standard port types, expressed an interest in deploying endpoints on FG, specifically the SMOA team from Poznan. In addition the CREAM (gLite) developers indicated that they would help deploy CREAM. We will follow up with these requests. Other notable activities include efforts in regards to Sector and Sphere conducted by the community. Furthermore, we identified a plan to offer SAGA as part of the FG software stack. A list of community project activities is provided in Appendix B. We expect more community activities to result in contributions to FG.

4.2.5 User and Support Services

In PY2, we are replacing all systems provided for user and support services and integrating them into a portal. As a first step, the software team has reworked the way ssh keys are managed to be integrated into non-HPC platforms and services. The portal has been designed and includes software enhancements to manage projects and its members. A backup system will be put in place.

4.2.5.1 FG Portal

In order to facilitate information exchange and the use of FG, we are providing a user portal that is tightly integrated with the overall software architecture motivated by the goals of FG. In PY1, we have provided a minimal web site. Integration with the IU knowledge-base was conducted to offer access to "FAQ style" information. However, this deployment has limitations that we need to overcome.

One of the goals of FG is to encourage the community to contribute to the development and use of services that are beneficial to others in the community. As such, it is essential to establish a community portal while focusing on simplicity and functionality. Such functionality is not provided by the TeraGrid (TG) user portal and requires the creation of a specialized portal with focus on the unique characteristics that are as identified different from TG. Hence, In PY2, we are working on a significant enhancement of this Website while transitioning it to a Web 2.0 style portal. We have the following goals in mind for the upcoming year:

1. Shift the user management as much as possible to the portal to further reduce the overhead on our administrative staff.
2. Allow portal access through modern Web 2.0 style SSO solutions such as OpenID.
3. Make it possible that educational users or any project lead can manage membership to their projects easily, similar to modern Web 2.0 sites.
4. Integrate management and sharing of experiments and results with the community through the Web Portal.
5. Establish an editorial workflow of content to be exposed to the portal while all content is managed first through the portal.
6. Project the status of FG through the portal.
7. Support a project approval committee in the task of easily reviewing new project requests.
8. Present a unified search capability on all content related to FG as part of a single search function within the portal, including Web pages, project pages, and FAQ style pages.
9. Allow advanced support features through the portal via Forums and the assignments of experts to projects or areas of expertise pertinent to FG as part of a community building effort.
10. Integrate many of the services and endpoint management into the portal through convenient interfaces, while reusing community efforts.

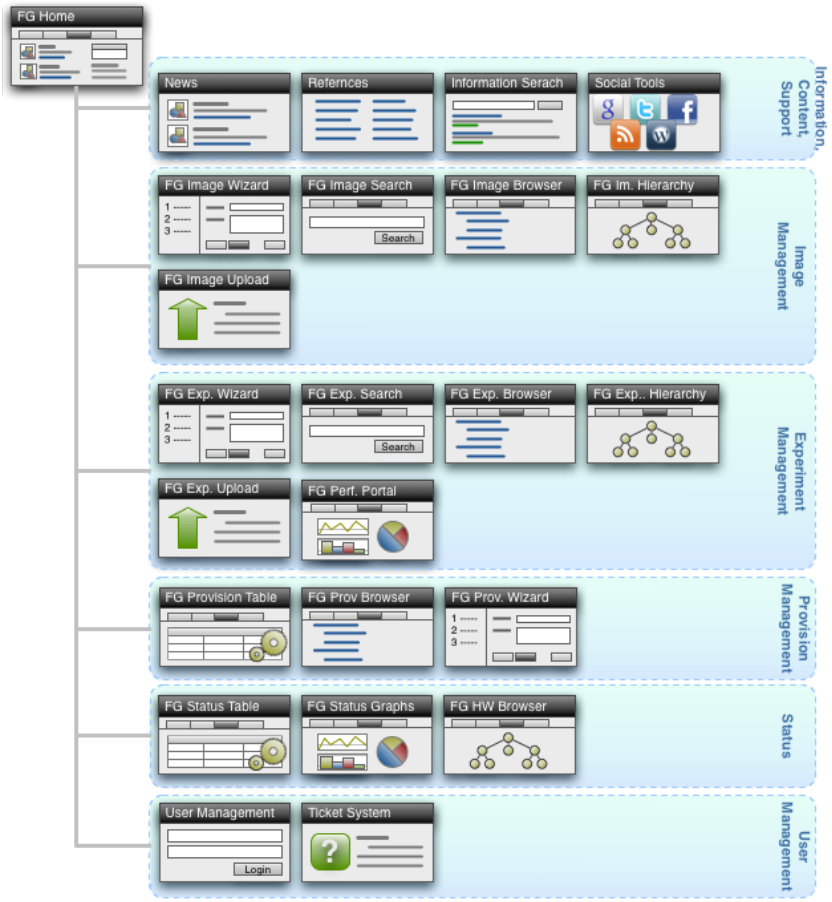


Figure 8: Portal components for conducting FG experiments

We have set up a content management system based on Drupal to implement the highlighted features and made significant progress on all items. In addition, we intend to collaborate with the XD TAS project that provides technology-auditing frameworks and integrate such a framework in the upcoming years into the portal. Furthermore, we have already started to contact other groups such as the iPlant team that has developed a sophisticated portal interface to Eucalyptus. Our intention is to integrate such services in a seamless fashion into our portal.

Figure 8 depicts the major components that will provide high-level experiment management capability through the portal. This includes:

- To facilitate information and content support, we will provide mechanisms to post news, references, conduct searches and integrate information sharing through Web 2.0 social tools.
- To facilitate Image Management, we will provide a wizard that helps generate images, allow the sharing of images through a repository while allowing browsing, searching for features and by category in lists and hierarchical views.
- To facilitate our Experiment Management, we will provide wizards to create experiments, share the experiments with the community and allow searching on various aspects, including the results as part of performance traces.

- To facilitate Dynamic Provisioning, we will provide interfaces with various levels of access controls to privileged users.
- To facilitate the display of the status of FG, our portal will integrate a variety of monitoring tools available as part of the FG system. This includes Inca, Nagios, PerfSONAR, and others as appropriate.

Features of the portal will be gradually enhanced over the next year and are in synchronization with the general software development activities. To showcase an example of an already implemented component, we show in Figure 9 a screenshot of the portal component that displays the current participation of a user in projects hosted on FG.

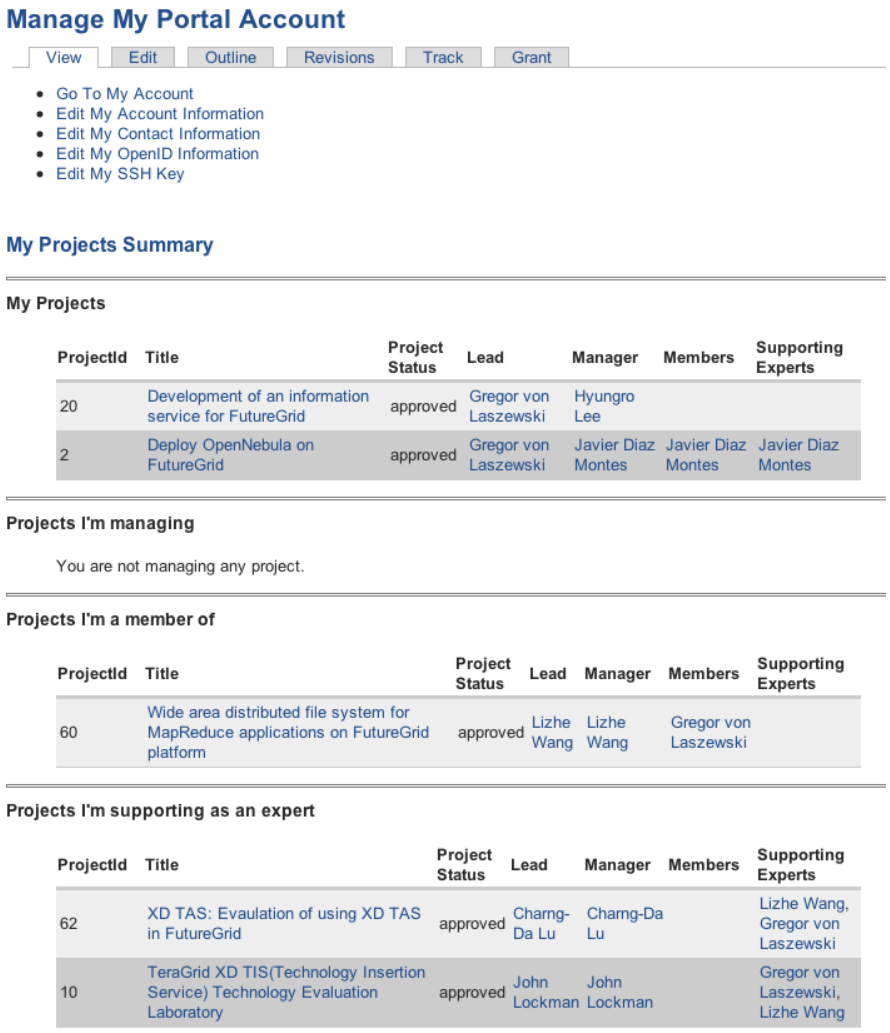


Figure 9: Screenshot of the Portal component that displays the current participation of a user in projects hosted on FG.

4.2.6 FG Operations Services

4.2.6.1 Authentication, Authorization, Accounting, and Auditing Services Activities

This section provides a short overview of security activities in relationship to Authentication, Authorization, Accounting, and Auditing Services activities. Most of our activities are currently

focused on solutions to provide a more adequate authentication and authorization mechanism. We will be replacing the first phase of establishing authentication in FG that is based on the IU TeraGrid authentication solution. This solution did not provide an adequate mechanism in case account access needs to be revoked for images managed through IaaS frameworks. In addition, we integrated OpenID in our portal as we find community Web 2.0 tools in frequent use by our user communities (such as Google). This will allow seamless authentication with OpenID in the portal.

Once InCommon matures and is integrated in TeraGrid XD, we will evaluate available resources and software tasks in regards to its integration. However, we do not anticipate that this task is started in PY2, because the start of XD has been delayed and such an integration has to be evaluated thoroughly once it becomes available.

First, our access model to the FG testbed is slightly different from that of XD, since we are focused on enabling experiment-based result sharing that is an integral part of our account management. So far, we are in the need of governing project approvals through a committee that will meet in an ad-hoc fashion once new applications arrive. The software designed as part of our portal is able to handle this.

Second, in contrast to TeraGrid all of our HPC resources have the benefit of a unified authentication mechanism (supported through replicated LDAP services). Hence, instead of managing different domains we have unified all accounts for HPC resources through a set of distributed replicated LDAP servers.

Third, our testbed is experimental and some of the services that we expose or deploy do not provide mechanisms for single sign-on that are based on GSI or MyProxy. We observe that we can develop compatible technologies for these systems while adapting to SSO via MyProxy. However, it comes with the cost of development and maintenance of the solution. Additionally, we see a shift away from Grid technologies to some of the tools that make cloud computing such an attractive offering. Hence, it is important to not ignore Web 2.0 style technologies such as OAuth and OpenID. As noted, our portal offers integration of OpenID. This is of special importance as even projects such as Eucalyptus provide plans to integrate OpenID. Lessons learned from projects that moved towards frameworks such as OpenID, as for example the Earth Science Grid, have to be considered.

We have furthermore contacted the OpenNebula team and discussed with them the enhancement of their security mechanisms that include clear text password management. Such tools are slowly maturing and it is foreseeable that the initial instantiation of services offered through FG contain different authentication and authorization mechanisms.

Presently, we provide users access to the Nimbus cloud once they apply for an account on a HPC resource. The account application for Eucalyptus is handled separately thus far, as we await the integration with OpenID as a solution to Eucalyptus, hence saving us a lot of development work. As an alternative, we will investigate the solutions provided by iPlant as part of the Atmosphere portal effort.

Accounting and auditing services have to be established in FG. We will identify in PY2 if systems such as AIME are suitable for use or if other systems are better suited. For auditing, we will also collaborate with the XD TAS project.

4.2.6.2 Development Services

We are in the need of certain services to facilitate the development and management of FG. We have established a developer's wiki, a ticket system, set up a google space to collaboratively write documents and presentations, and established a code repository. In the future, we may consider integrating these services closer into the portal if time permits. Furthermore, in PY2, we will deploy the wiki and ticket system onto FG controlled hardware in order to provide better performance of the services and also to better maintain them while transitioning control to the FG systems team.

4.3 FG Management Services

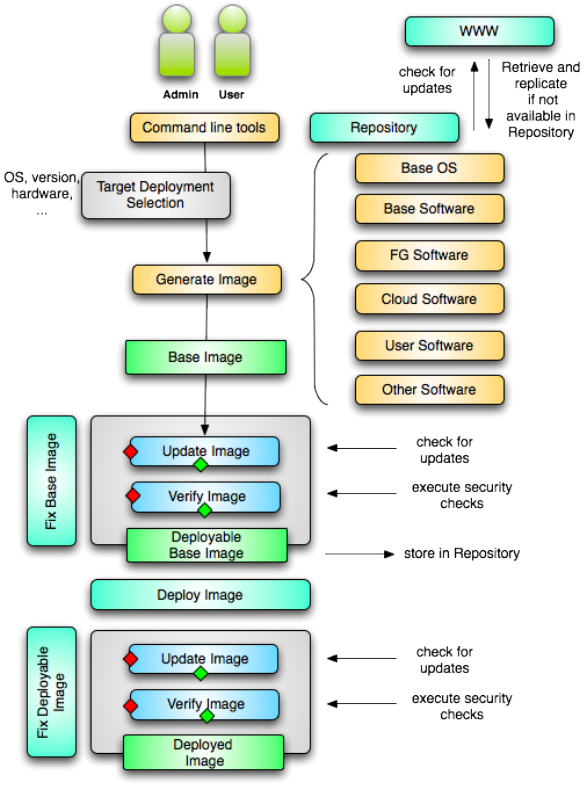


Figure 10: Process to create a vetted FG image.

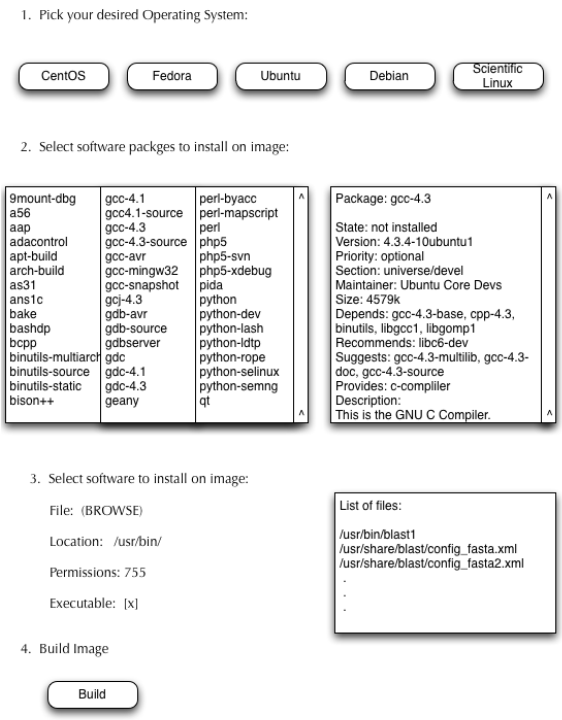


Figure 11: Wireframe mockup of the image generation portal interface

4.3.1 Image Management

One of the capabilities provided by FG will be a catalog and repository of virtual images that will include base images set up by the project as well as images generated by the users if they wish to share their VMs. The images will be described with information about the various

software installed (including versions, libraries, etc), available services, etc. This information will be maintained in the catalog and be searchable by the users and/or other FG services. Users looking for a specific image will be able to discover the available images and find their location in the repository using the catalog interface.

We define image management as the process of creating, storing, sharing, distributing, vetting, and updating the base software stacks for any deployment of IaaS or, perhaps more importantly, PaaS for the users. Various components need to interact to accomplish this task. The storing and distribution of images relates to the FG Image Repository, the creation of images is done through the Image Generator service, and the updating and vetting is done in part through the services rendered via BCFG2 and custom services as part of FG. The eventual goal of the Image Management system is to create and support Platforms architected by the users.

4.3.2 Image Generation and Creation

When creating a cloud-based service deployed on Future Grid resources, a variety of IaaS tools are available. Two of them are Nimbus or Eucalyptus. Here a service image (or multiple images) need to be created to be deployed through these cloud infrastructure services. Instead of leaving the task of creating the image solely to the service developers, FG can help through the use of a guided image creation process, similar to rBuilder for EC2.

A guide can first let the user pick a suitable OS for their service. From there, a minimal installation of that OS (such as Ubuntu or RedHat) can be instantiated. Next, the user can pick packaged software that their service will use or leverage in some way. These packages can be added to a BCFG2 bundle where they are then installed on any/all of their images during deployment. BCFG2 will also update and validate these installations, thereby simplifying the user/developer's experience. Then, the specific developer applications and services can be deployed on the resources given to them using BCFG2. Through the use of the Cfg package in BCFG2, the specific files, folders, and executable files can be managed and validated for each system within the deployment. In addition access rights can be integrated into such an image. This validation process contains multiple phases and is depicted in Figure 10.

This setup is remarkably different from a normal Grid submission system. Many times, you have to "ship" your executable as part of the job submission in a Grid environment to have it propagate into the system and then have it run. Using a configuration management tool such as BCFG2, executable and configuration files are deployed once (during setup), and validated for integrity. Furthermore, if you want to update the executable (or configuration files), you simply add the changeschanges to the configuration file, that will than be propagated automatically. This is similar to the model used within BOINC's application framework.

In order to correctly instantiate images within FG, a specification will need to exist to have a standard for defining FG images. We plan to leverage existing work within the Open Virtualization Format Specification. When new images are created, they are placed in the image repository and attempted to be kept up-to-date. This is important, as they are used across FG resources over time. However, the images will become outdated and need patching or reconfiguration. We have devised a process that will improve the update of these images automatically.

4.3.3 Image Repository

The FG Image Repository provides an environment and service to store images, which are to be dynamically provisioned. Using the FG Image Repository service, users could query image store, upload/register customized image, share images among users, and choose an image for system provisioning. Most related cloud service offerings have similar service. Xcat manages the images through Linux file structure and some table information; Nimbus also uses a Linux file system and symbolic links to manage the images (though now it has a similar back end storage service as S3); AWS, as well as its open source equivalence Eucalyptus, a well-defined functionality set and interface for the image repository service among these. FG will leverage the effort from Xcat, Nimbus, and Eucalyptus to provide cloud services. To justify why we need an image repository service in FG, consider these reasons:

1. We need a unique and common image repository interface that could distinguish image types of VMs for Nimbus, Eucalyptus, or just general HPC. The main reason for this is, due to the different back end storage mechanism used by Nimbus and Eucalyptus (at least at this moment), the ways to retrieve the image have different requirements.
2. By developing a FG repository we can maintain specific data that could assist performance monitoring and user/activity accounting.
3. By introducing a customized image repository we will be able to choose an appropriate storage mechanism that is suitable for the FG platform.
4. By using a mechanism that may actually not just store the image, but rather describe how we generate an image, we can save a significant amount of space. An image is "cached" in the repository or generated upon retrieving based on use patterns.

Currently, the Image Repository is under development.

4.4 Experiment Management

One of the important concepts of FG is the ability to perform repeatable experiments by submitting an experiment plan that will often follow these steps:

- Assemble a set of resources and services (experiment apparatus);
- Execute one or more experiments on the assembled component (experiment session);
- Monitor the progress and status of an experiment;
- Record a description of the assembled components, the steps performed in an experiment, and the results of the experiment; and
- Repeat the experiment while also allowing modifications to the apparatus;
- Release the resources, services, and software gathered for the experiment(s).

More complex features for plans such as hierarchical, iterated, parameter sweep, and DAG-like experiments will also be supported. To accomplish this, we are deploying an infrastructure that supports both a workflow-based experiment management plan and a script-based experiment management plan, while also allowing a combination of the two.

1. **Workflow-based experiment management plan:** A user describes an experiment plan including all of its steps as a workflow, which is then executed as part of the experiment management system. The workflow can also be shared as a template with other users that

like to conduct similar experiments or like to repeat it. Support will be provided for a wide range of experiment plans from simple experiments, as listed above, to much more complex ones.

2. **Script-based experiment management plan:** Convenient command line tools that can either be used directly or as part of workflow and interactive shells will be provided to users. They can use these commands in a familiar shell to create an experiment plan through traditional scripting interfaces or an interactive shell. Hence we will be able to support many different scripting languages and also allow the execution of experiments in an ad-hoc fashion.

3. **Time-based experiment plan:** The ability to provision and reserve resources at predefined times is an important extension to any experiment management framework and will be supported on FG.

4. **Hybrid experiment management plan:** Naturally, it will be possible to combine these approaches.

A toolset will be provided to implement these features and will be built upon existing software tools and extended as necessary with our own software. One such tool is Pegasus, which will be used to implement the workflow-based experiment plans. FutureGrid will leverage several important features of Pegasus such as data staging and replica tracking. Hence, it will be straightforward to expand workflow plans into a μDAG comprising, among other tasks, staging data in, running the application, staging data out, and replica tracking. However, the dynamic provisioning of IaaS, PaaS, and other dynamic provisioning activities will be handled through RAIN. Hence, Pegasus will interface with those FG-specific tools that manage the provisioning of environments and will integrate them into the workflow.

4.4.1 RAIN

In order to provide the concept of "raining" both infrastructure and platforms onto a given set of hardware, we need to develop a convenient abstraction that allows us to hide the many underlying tools and services to accomplish this task. We observed the following needs:

1. In contrast to Grid frameworks such as Globus, we are not only interested in interfacing to the job management system.
2. In contrast to IaaS environments, we are also not just interested in provisioning an image as part of the virtualized environment. Instead, we would like to be able to “provision” even an IaaS framework.
3. In contrast to environments based on a single operating system, we would like to manage the deployment on multiple base OS including the choice of which virtualization technology is used.
4. In contrast to just focusing on virtualized environments, we would also like to enable an environment allowing performance comparisons between the virtualized and the non-virtualized versions of applications, e.g. comparing HPC, vs. IaaS frameworks.

Hence, it is important to recognize that this comprehensive view of "raining" an environment is a significant contribution of FG and allows comparative studies that are otherwise not easily possible.

As a result, this concept is a pivotal process in making the FG deployment unique and applicable to any set of scientific researchers requiring rapid deployment IaaS, PaaS, and HPC environments.
We are currently working on tools and scripts that can simplify these steps to the user. Examples for "raining" a Hadoop environment, a Nimbus cloud, or simply an operating system are given bellow:

- fg-rain -paas hadoop -r india ...
- fg-rain -iaas nimbus -r sierra -n 10-34 ...
- fg-rain -os windows -r sierra -n 10-34 …
- fg-rain -date start end -os windows -r sierra -n 10-34 ...

It is obvious that such a command will be extremely powerful and provide an important mechanism for abstracting the many different tools and services that are needed to accomplish the task. In our environment a user will not even need to know the details of the deployed low-level tools such as XCAT or others. Hence, the command *fg-rain* that will provide the high level interface to the FG fabric will be essential to create workflows in a simple fashion.

Internally *fg-rain* may use a multitude of tools and components suitable to conduct the task indicated by the command line tool. This may include Moab, XCAT, TakTuk, and even IaaS frameworks where appropriate. This tool set is also internally known as "experiment harness" as it allows us to access the low level mechanisms to enable provisioning on multiple scale. It is important to recognize that, in order to allow repeatable experiments, the *fg-rain* command will have to interact with a multitude of services that we offer. These services allow specifically the recording of the experiment apparatus and the experiment conditions as part of a set of metadata to be recoded and logged. This data is then annotated to an experiment that is recorded into an experiment repository. Eventually, the experiment can be shared with other users. The way an experiment can be recorded through *fg-rain* is through the specification of an experiment flag as part of the command interface.

Once we have provided such a command, we are able to build higher-level functions useful for the users. Most recently we have developed a prototype function called *fg-hadoop* that instantiates a user controlled Hadoop environment through the queuing system and allowing to run an application on it.

Together the tools will allow to enable a very comprehensive way of dynamic provisioning, as indicated in Figure 12.

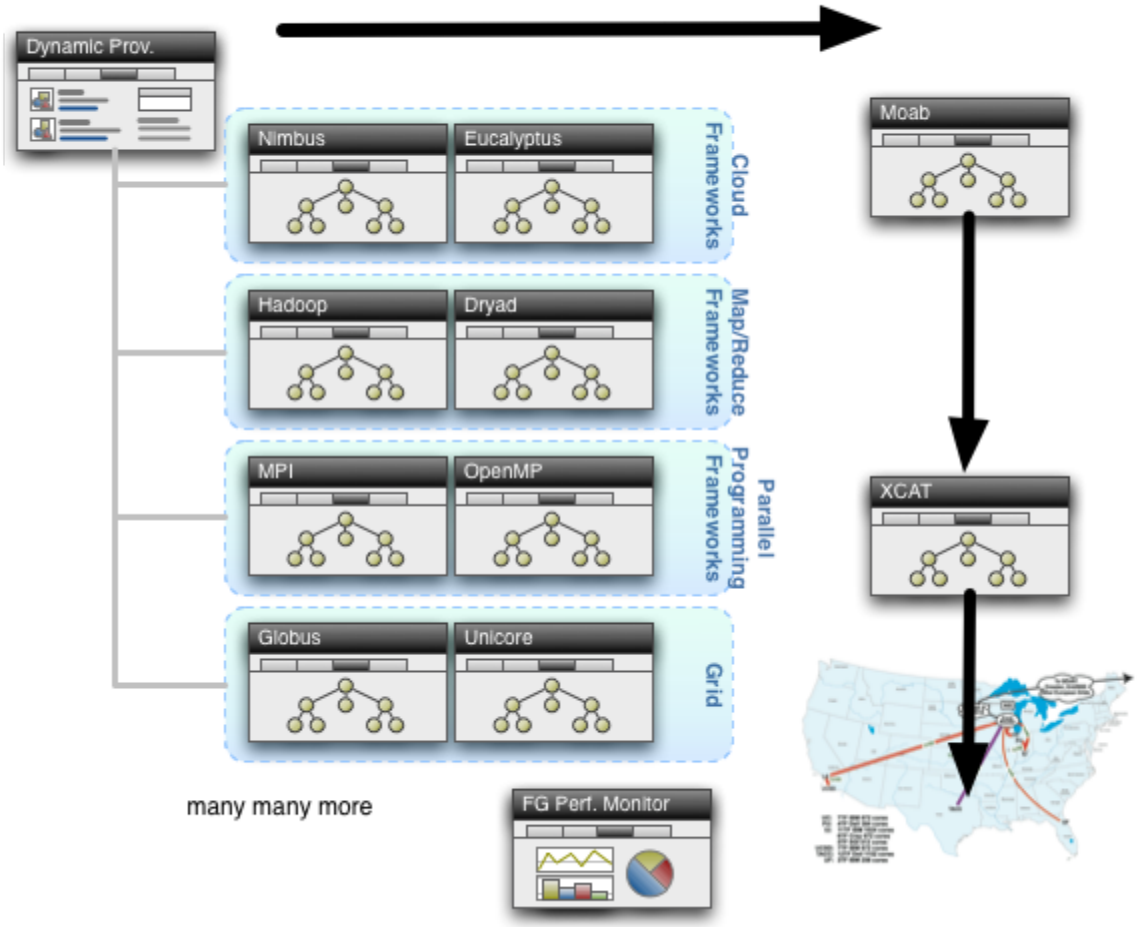


Figure 12: Concept of Raining/Dynamic Provisioning

4.4.2 Monitoring and Information Services

One of the goals of monitoring is to detect functional problems on FG as well as collect and publish information useful to users. Another goal is to detect performance problems on FG by actively and passively measuring the performance of FG components.

To address these goals, we will provide a suite of monitoring tools divided into three main categories: active monitoring, instrumentation and hardware monitors. The data produced by these tools will be integrated through the FG portal. In addition to infrastructure monitoring, we will also plan to eventually make these tools available within a user's experiment.

> **Active monitoring:** We use a user-level active monitoring approach implemented by the Inca monitoring tool. User-level monitoring emulates a regular user and tests and measures the infrastructure in order to detect problems. It typically provides more

comprehensive and end-to-end tests that have a higher system impact and thus run at a coarser-grain than system (hardware) monitoring system tests. User-level functional tests and performance measurements will be identified for every major component of FG and will be collected by Inca (e.g., submit a job, instantiation time for an experiment). This also includes testing of packages that FG develops to generate images.

Also, automated benchmarking will be used to detect performance problems with aspects of the FG infrastructure. A selection of benchmarks will be used to establish a baseline for performance and executed regularly to validate infrastructure improvements or detect degradations of performance. A major part of this work will be to measure the impact of virtualization on experiments as well as UTK's Grid Benchmark Challenge work. Examples of other benchmarks include HPCC, SPEC, and NERSC6 and will leverage similar work done by the DOE Magellan and XD TAS projects. Benchmark results will be stored persistently and made available through the FG portal for users to access.

Instrumentation: Instrumentation will be used to passively collect performance measurements from selected FG components and is based on an existing instrumentation tool called Netlogger. Candidates for instrumentation are components that are developed by FG staff. For these components, the Netlogger API (Java, Perl, Python, or C) will be integrated into the component to collect performance measurements of major events. For components where source code is not available or impractical to work with, a Parsing Tool can be created to iterate over a component's existing logging mechanism to push data into the Logging database. For example, Pegasus already has built-in support for Netlogger whereas a Parsing Tool would need to be developed to grep timing statements out of Moab's log and pushed them into the Logging Database.

Hardware monitors: Hardware monitoring data is another aspect of performance data that can be collected from a system or component. We will integrate existing hardware monitoring tools such as Ganglia or Nagios for clusters and PerfSONAR for the network.

Finally, given that FG will often be a stepping-stone to production Grid/Cloud environments (e.g., TeraGrid, Azure, Amazon), it is natural that users will run performance studies to compare those environments with FG and possibly with one another. FG can serve as a base reference environment for such performance studies, which also makes FG a convenient place to store and display these results as well. This information can be immensely helpful to other users wanting to better understand the differences between these environments and to decide which environment will be most suitable for their application needs. We will work with identified users to publish their results and methodology within the FG portal so others can understand and benefit from their results.

4.4.2.1 Grid Benchmark Challenge

The Grid Benchmark Challenge will be a set of grid benchmarks to measure, characterize, and understand distributed application performance. These benchmarks will include a set of tightly coupled application grid benchmarks based on UTK's well- known HPC Challenge benchmarks and a set of loosely coupled application benchmarks based on real-world scientific workflow

applications. UTK will define appropriate and relevant metrics for the performance, reliability, and variability of grid platforms and tightly coupled grid applications. These metrics will be deployed so that applications, architectures, and middleware implementations can evolve guided by sound engineering principles.

4.4.2.2 Inca

Cyberinfrastructure (CI) aggregates multiple complex and interdependent systems that span several administrative domains. This complexity poses challenges for both administrators who build and maintain CI resources and scientists who use them. In order to provide scientists with persistent and reliable CI, the user-level functionality and performance of the system must be monitored. Developed in 2003, Inca is a tool that provides user-level monitoring for CI systems. It has proven to be a valuable tool for the TeraGrid and eleven other production CI projects, and is currently monitoring over a hundred machines. Inca differs from other monitoring tools with its user-level monitoring approach, flexibility to collect multiple types of monitoring data, and full archiving support. Inca also uses a unique centralized monitoring configuration approach, which is essential for providing consistent monitoring results and a unified and coherent infrastructure for users. The Inca website islocated at http://inca.sdsc.edu.

Inca provides the active monitoring role in Monitoring and Information Services. It has been deployed on FG since March 2010 and is currently collecting ninety-five pieces of test and performance data across FG platforms. Currently, tests are divided into Basic (ping and SSH), Services (drupal, wiki, jira, etc), HPC (MPI, compilers, batch queue), Cloud (Eucalyptus and Nimbus), and Benchmarks (HPCC, Infiniband). In addition new tests and measurements being added to the Inca deployment, Inca will also be used to validate packages developed for image generation and will be modified to easily deploy and install within a user's experiment environment. The Web status pages for the Inca FG deployment can be found at http://inca.FGfuturegrid.org.

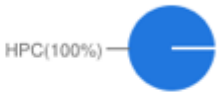
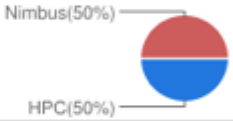
4.4.2.3 Netlogger

Netlogger is a tool for debugging and analyzing the performance of complex distributed applications. It is comprised of an API that allows a user to instrument their application and tools for parsing existing log files. APIs are available in C, Java, Perl, and Python. The overhead of using Netlogger is low and data is collected centrally where it can be queried from a number of APIs. Netlogger has been in development since 1998 and is used in several projects across the world. The Netlogger website islocated at http://acs.lbl.gov/NetLoggerWiki/.

Netlogger provides the instrumentation role in FG Monitoring and Information Services. It will be used to passively collect performance and usage data from FG components. Netlogger is deployed on FG with an AMQP server and Mongo database backend. We developed a simple graphing interface that displays collected Netlogger data. Currently, the number of VMs and active users are being passively collected from Eucalyptus and Nimbus. Data collected from Netlogger can be found at http://portal.futuregrid.org/performance/netlogger.

4.5 Current Software Deployment

The current deployment of services on FG is given in the list bellow. The system administrators can adjust the assignment of resources to IaaS and HPC services.

Resource	HPC	Eucalyptus	Nimbus	
IU-INDIA (1144 cores)	49% (560 cores)	35% (400 cores)		
IU-XRAY (672 cores)	100% (672 cores)			
UC-HOTEL (672 cores)	50% (336 cores)		50% (336 cores)	
UCSD-SIERRA (672 cores)	46.4% (312 cores)	21.4% (144 cores)	23.8% (160 cores)	
UFL-FOXTROT (256 cores)			96.9% (248 cores)	

*A small percentage of nodes may be unavailable or used for management

Figure 13: Screenshot of the assignment of Nimbus, Eucalyptus, and HPC to the FG Resources

(1) Note although TACC-Alamo is in production it has not yet been integrated into the automatic monitoring script. 128 cores of the 768 cores are intended to run Nimbus. The rest are split between Eucalyptus and HPC.

5. Using FutureGrid

5.1 General Principles

This document describes the implementation of *FutureGrid—an experimental, high-performance grid and cloud test-bed*. The goal of FutureGrid is to support the research that will invent the future of distributed, grid, and cloud computing and integrate them with high performance parallel computing. FutureGrid supports the development and early use in science of new technologies at all levels of the software stack: from networking to middleware to scientific applications. This test-bed enables dramatic advances in science and engineering through collaborative evolution of science applications and related software. Table 9 outlines many of the general types of grid and computational science experiments that we plan to support via FutureGrid. Table 7 illustrates requirements of the user project requests during the first year. Many of these are from members of the FutureGrid Expert Group who form the basic user support for FutureGrid users.

The computer and computational science community has a strong need for facilities that enable a more scientific approach to comparison and evaluation of distributed computing software. The critical element of the science plan for FutureGrid is that it will enable rigorous, repeatable

experiments in middleware and distributed computing, facilitating the sort of exactitude for distributed computing systems and performance analysis that has long characterized parallel performance analysis. Repeatability is based on the ability to instantiate a particular environment, in isolation from outside interference, with a particular and repeatable set of initial conditions. Networks will generally be dedicated to particular experiments, and, when network impairments are involved in an experiment, they will be generated through use of a network impairment device, allowing for repeatability. Data stored for any given experiment will include the system images in which an experiment was performed, along with the software actually used and input data. Hence, FutureGrid will be a cyberinfrastructure for the development of new approaches to scientific applications and for distributed computing research.

We expect that the activities that will take place within FutureGrid will be primarily experiment-based, driven by an experiment plan or involving steps that may be viewed as an experiment plan. That plan may be very basic: instantiate a particular environment and let a researcher debug an application interactively, or very sophisticated: instantiate a particular environment and run a pre-specified set of tasks. A direct outcome of this experiment-centric approach is that it will lead to a collection of software images and experimental data that will prove a tremendous resource for application and computational sciences.

Use case	Required to fulfill use case
Testing a new networking protocol or topology, application layer overlays, and peer-to-peer networks	Ability to build system images and propagate them through a test environment Dedicated time in an isolated test environment, with prescribed and repeatable levels of load and error conditions
HCI researchers testing end-to-end productivity of grid computing systems	Variety of software and hardware environments allowing presentation of multiple systems and user interfaces
Testing grid or cloud, particularly end-user applications	Specify a grid or cloud environment and run applications in that environment; compare with other environments Prepare applications for deployment on commercial systems (cloud or grid) Test a complex workflow, which requires a heterogeneous hardware mix
Creating a cloud front end linked to a grid and its resources to enable scientific applications and gateways	Cloud test environment, ability to link to one or potentially many different hardware architectures as back end
Developing data-intensive applications	Link data sources to a grid environment specified by the developer, possibly including supported workflow tools—for example LIGO data flow, medical images, or sensor data
Testing optimization of different layers of parallelism via grid, cloud, and many-core programming models	Grid or cloud test environment that includes systems representing varying levels of core counts per processor
Comparing grid middleware implementations and standards compliance	Persistent endpoints for grid interoperability testing Test-bed to compare grid operating environments
Testing new authentication or authorization mechanism	Ability to run a persistent authentication server in test environment or link to one at the researcher's lab
Hardening of middleware or science application	Security vulnerability ("simulated attack") test service Simulated job load For network- or grid-centric applications, ability to simulate latency, inject errors into network, etc.
Testing performance of applications on non-x86-64 architectures	For resource providers, the ability to place non-x86-64 architectures in a multiuser environment For application developers, the ability to test applications on non-x86-64 architectures to evaluate code performance

Table 7: Experimental grid test-bed requirements matrix. Common needs across all of these use cases include the ability to (1) specify a test environment in advance and use it during a scheduled period of time and (2) create an appropriate record of an experiment

For computer and computational science researchers developing middleware – grid software, cloud software, and HPC – FutureGrid provides a rich and flexible test-bed, and is a platform for computer and computational scientists to use for developing new network, distributed, grid, and cloud applications; and for rigorously evaluating new approaches at all levels, from application science down through the layers of technology to networking.

We will support application science directly and indirectly. Application scientists and software developers can develop and prove new approaches to delivery of their applications. Such applications can then be migrated to other production cyberinfrastructure facilities, enabling better support and delivery of end-user science capabilities to the U.S. research community. We will support network, grid, cloud, and distributed computing directly by providing an environment that supports computer and systems research that will lead to improved cyberinfrastructure that indirectly supports application science. Dedicated networking and 24 x 7 monitoring will provide a secure environment in which new applications can be safely developed, tested, and hardened.

5.2 Early Science Experiences on FutureGrid

FutureGrid is perhaps more unlike the existing TeraGrid resources than any other resource funded through the Track II program. The TeraGrid has added new large systems, experimental hardware, and high-throughput systems. However, no experimental test-bed system has ever been part of the TeraGrid. The history of the TeraGrid suggests that it can take considerable time for the U.S. research community to recognize and make good use of a novel type of resource within the TeraGrid. With a team that brings together some of the very best of leaders in academic grid and cloud research, it will be tremendously important to achieve a good balance between ensuring that FutureGrid is well used early on, and having so much of FutureGrid's use come from our own team that we create a perception that FutureGrid serves the FutureGrid team first and foremost.

Table 9 in the appendix lists the initial projects started on FutureGrid through the end of December 2010 while highlights are given in Table 3. We note that this online summary of users is openly available and has two categories (approved and pending) but we do plan significant web site improvements which will allow us to record confidential projects as well as those that are ongoing and completed. Furthermore we will pro-actively insist that users will populate results within the project as they evolve.

In the overview we described several categories of FutureGrid use and we discuss these in more detail here.

5.2.1 Educational uses of FutureGrid

With the emergence of dense multicore and similar architectures for personal computing, the proliferation of smart devices and sensors on the real-time Internet, and the evolution of large-scale production instruments, it is important to provide a new and forward-looking teaching environment that integrates seamlessly with large-scale cyberinfrastructure. Achieving this goal requires programmers and domain scientists who understand grid, distributed, and parallel programming. Current production cyberinfrastructure such as the TeraGrid is not ideal for teaching – a student might even crash the system while learning to program it. In addition, students learning to program grids may introduce real and severe security vulnerabilities; even seconds of exposure may be all it

takes for a malicious actor to gain unauthorized access to a computing system. In order to let students program in a safe and encapsulated environment, we have created an environment that will allow the creation of a virtual grids, clouds and HPC systems in which students can experience the full complexity of grid computing for writing and debugging grid software, allowing students to use a variety of cloud and grid computing environments. These appliances are described in https://www.futuregrid.org/tutorials while http://salsahpc.indiana.edu/tutorial/index.html exemplifies FutureGrid used as a support for a class – a week long summer school with several hands on sessions teaching MapReduce. Classes at Indiana University (https://www.futuregrid.org/Qiu/classroom), and Louisiana State University have used FutureGrid this fall and we will evaluate this experiment in next month.

5.2.2 Grid and Cloud Middleware and Technology users of FutureGrid

A significant part of initial FutureGrid activities fall in the "computer science systems" research area with grid and cloud computing research and software development. Examples in Table 9 include work on the SAGA software, grid and HPC scheduling, MapReduce, monitoring and resource discovery. Given the current interest in security (and expertise of new CISE director), we note one project looking at hybrid clouds (linking FutureGrid to an IBM cloud) so one separates applications into privacy sensitive and insensitive components running respectively on private and public clouds. There is significant interest in data intensive systems with study of Lustre and Bigtable style data storage. An interesting project involves the European Grid Initiative (EGI) which will explore virtualization on FutureGrid and establish an experimental node of EGI on FutureGrid.

There are also some TeraGrid related experiments proposed for FutureGrid including work of the TeraGrid XD TIS(Technology Insertion Service) Technology Evaluation Laboratory. This was broken out as a separate category "Evaluation and TeraGrid Support" in Table 3 of the Introduction. It is similar in topic to Computer Science systems but aimed at a infrastructure not research goal.

5.2.3 Interoperability Experiments and Demonstrations on FutureGrid

We had anticipated the importance of interoperability for FutureGrid as explicit tasks for co-PI Andrew Grimshaw to support key standards compliant Grid software including gLite, Unicore and Genesis II. However we have found the ability of FutureGrid to support general endpoints (due to virtualization) very important and seen significant interest in interoperability work on FutureGrid. This includes work with Grid5000 http://www.isgtw.org/?pid=1002832, OGF with SAGA and BES standards and could extend to clouds with a collaboration with IEEE Cloud Computing Standards Study Group (http://salsahpc.indiana.edu/CloudCom2010/ccsccc2010.html). Co-PI Jose Fortes is leading this cloud interoperability work. It is worth noting that work of the GIN (Grid Interoperability Now) working group at OGF was often stymied by inconsistent software stacks in different grids – something that is addressed with virtualization available on FutureGrid.

5.2.4 Domain Science Applications of FutureGrid

Initial work on FutureGrid has not seen substantial interest from classic HPC applications involving particle dynamics and partial differential equation solution. Rather we find data-intensive and Life Sciences applications. This probably reflects that traditional fields have well developed codes and may not immediately be interested in FutureGrid. However interest in biological and data intensive applications is growing rapidly and many new codes need to be

developed. We realized that several biology problems require large memory and deployed a 16 node ScaleMP distributed shared memory environment funded by Indiana University. This is currently being used in two applications for gene assembly

As a result of the FutureGrid project, we create an open-source, integrated suite of software to instantiate and execute repeatable experiments targeting grids and clouds. Experiments can be coordinated in workflows and instantiate services provided as part of the FG systems or through dynamic provisioning of software stacks. We are leveraging from open source tools to avoid replication of functionality that is already provided through existing software. A portal is provided that allows easy access to FG information, and allows generation and management of experiment and images. One of the key services we provide is the access to performance tools and services allowing users to assess performance impacts of the software stacks and the associated programming paradigms. To support the later, we are developing a grid version of the widely used HPCC benchmark suite, and are then develop a new Grid Benchmark Challenge application suite.

6. User Support

6.1 Operations

All FutureGrid sites are coordinating their activities across sites, which are expected to supply 24x7 availability with set preventative maintenance windows. In case of severe security vulnerabilities or system issues that impact the availability of the system, emergency maintenance will be undertaken in order to correct the issue. Both preventative maintenance and outages will be communicated via the FG portal and through additional mechanisms once integration into TeraGrid XD is completed. FutureGrid network systems will be monitored at the IU Global Research Network Operations Center.

6.2 Support

We adopt the tiered support model that includes the following:

6.2.1 Tier 0: Support through Electronic Documentation

FG will be developing a set of documentation that serves as an immediate entry point for users of FG. All information will be managed through the Web site and accessible through a search service. The primary information about FG will be structured as a manual, but will also be available if needed as part of a KnowledgeBase (KB). The IU KB team will be responsible to provide editorial help for the development of the manual, tutorials and has the option to integrate this material through automatic electronic inclusion into the KB. It is important to note that we also provide the community to participate in the development and improvements of this material through passive comments such as left through ratings, and active contributions through comments. Comments are vetted as part of the FutureGrid Portal and are allowed by authenticated FG users that have an active project on FG.

6.2.2 Tier 1: Support through Experts and Community

Support through Experts: To facilitate the support of projects, FG has established an expert team. Each project will be assigned an expert that can be consulted in case of questions or technical issues. If the expert cannot answer the question, he will consult with other experts. The communication with the expert is initially conducted simply via e-mail. In future, we will have forums and a dedicated ticket system available that logs interactions with these experts. On general topics a forum is used that is monitored by the experts. Responsibilities of an expert include

- help projects through the application process if contacted
- help on technical questions related to FG services
- help creating manual pages from the information they have been asked by FG users
- help on gathering results from the projects
- help on establishing a web presence of the project on the FG Web site

In case of complex problems experts may communicate with their designates via phone. The expert team will interface with the editorial team managed by the KB staff. Through our expert team members we will also ensure that at all times users have a single point of contact within FutureGrid and know who that point of contact is.

Support through the community: Based on advice from the Grid 5000 project we will integrate community members as part of our support infrastructure. We will use Web 2.0 services to allow users to share experiences, and to enable one-to-many and many-to-many discussions in resolving problems and enabling new capabilities. For a resource that serves a community of leading grid experts, enabling users to share expertise should be particularly beneficial. The interactions can be managed through Forums on the Portal.

6.2.3 Tier 2: Support through staff

Support through Network Experts. IU will provide 24x7 phone support delivered from the Global Research Network Operations Center (GRNOC). The GRNOC will provide 24x7 system status information, immediate handling of security concerns and incidents, and limited technical support, and will either forward phone calls to second-level technical experts (between 8 am and 8 pm Eastern Time) or initiate a trouble ticket (between 8 pm and 8 am Eastern Time). All support for the Network Impairment Device is handled through GRNOC in tight collaboration with the Systems Management team.

Support through technical experts at IU and partner organizations: Technical experts at IU will provide second-tier support to users via email or phone (in response to email or web form queries). For some systems problems, it may regularly be the case that second-tier problems are referred to systems management personnel at sites hosting FutureGrid hardware when a problem appears to be specific to a particular machine. Problems related to systems provided by our partners such as Nimbus, Pegasus, Vampir, PAPI, and others will be forwarded to these organizations. The organizations are expected to participate in gathering useful information from this support and integrate it in Tier 0 support as appropriate.

6.2.4 Tier 3: Advanced user support

Top technical experts anywhere within the FutureGrid team will provide third-tier support. Such experts may also be involved in advanced user support provided via the TeraGrid or TeraGrid XD. Personnel supporting software and applications that execute on another site's hardware will be provided privileged access in accordance with best practices for each operating system using the principle of least privilege (http://hissa.ncsl.nist.gov/rbac/paper/node5.html).

Throughout the tiered user support/problem resolution process, we will use a ticket system to ensure that user issues are promptly addressed. Together the tiered model provides a strong dual support model by the use of electronic documents and FG experts that allow the integration of the community (see Figure 14).

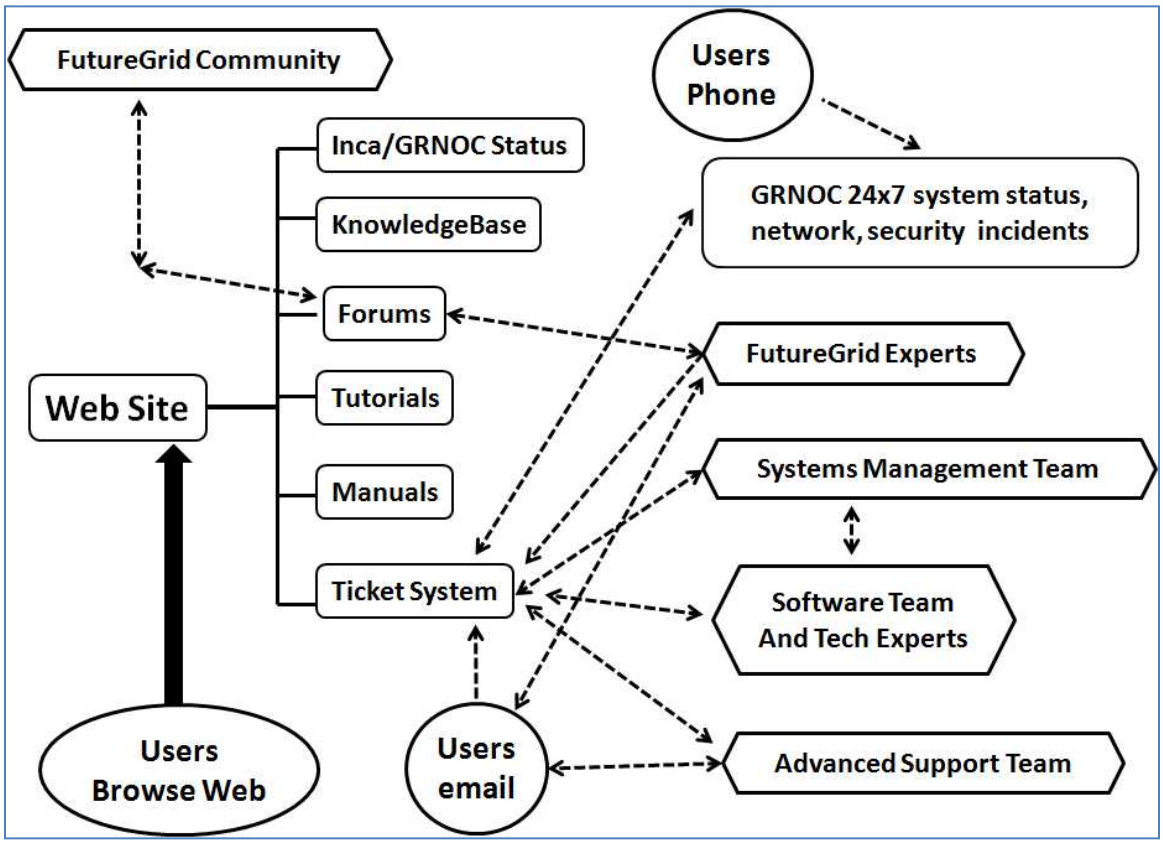


Figure 14: The FutureGrid support model

6.3 Operations

All FutureGrid sites are coordinating their activities across sites, which are expected to supply 24x7 availability with set preventative maintenance windows. In case of severe security vulnerabilities or system issues that impact the availability of the system, emergency maintenance will be undertaken in order to correct the issue. Both preventative maintenance and outages will be communicated via the FG portal and through additional mechanisms once integration into TeraGrid XD is completed. FutureGrid network systems will be monitored at the IU Global Research Network Operations Center.

6.4 Allocating FutureGrid Usage

The TeraGrid allocation process represents the outcome of 20 years of experience and refinement. However, while it is regarded as much improved, it is still perceived as difficult to negotiate by many. Rather than start with a complicated acceptance process, we implemented a resource request in the form of an explanation of the experiment they wish to perform with FutureGrid and a list of the resources and software capabilities they will need. This asks that requestors attach a standard NSF-format two-page biosketch for the PI and details typical NSF information (intellectual merit and broader impact). This approach minimizes barriers to adoption while at the same time allowing us to learn over time how best to structure later, more formal, resource requests. A summary of this form is presented in Appendix A.

We have learnt that FutureGrid use is controlled by different constraints from TeraGrid. Namely most of our requests are for small total time. So the constraint is not machine availability but rather the needed level of user and systems management support. Realizing this, we have instituted the FutureGrid Expert group to provide basic user support and plan expanded systems administration support. Two positions are currently advertised; one to lead the FutureGrid Expert group and one (shared with another project) to provide additional systems admin staffing.

We are generally heavily biased in favor of fulfilling early requests in particular, in the belief that by so doing we can best facilitate the development of new computational tools (middleware and application software), and best learn how to develop more refined and formal templates for resource requests during the latter half (PY 3 and 4) of the project. The later evolution of the allocation process is described in detail in section 3.2.2. Further many of these early users are candidates for expert group who can help later users.

Currently allocation decisions are made by the PI with the Operations committee involved when special issues come up. The latter is illustrated by requests that led to ScaleMP software being supported and discussion of issues involved in industry use of FutureGrid.

We use the merit of proposal in making decisions and do not require applicants come from USA or have NSF support. We are interested in establishing links with key international collaborators as illustrated by Grid5000 and EGI.

7. Training Education and Outreach TEO

7.1 Discussion of Year 1 Activities

7.1.1 Summary

FutureGrid TEO team has facilitated and coordinated the creation of various tutorials that provide a basis for users to understand and use resources and the core middleware functionality available on FutureGrid. Emphasis has been given to cloud-based resources through Nimbus, Eucalyptus, and educational virtual appliances, as they provide the greatest flexibility in configuration end use for educational and training activities. In addition to tutorials, TEOS efforts have focused on the creation of tailored virtual appliances and social group-oriented GroupVPNs that allow users to easily deploy virtual clusters on FutureGrid resources, as well as on their own desktops. Highlights of educational activities in the first year have included a week-long tutorial organized by Indiana University, use in class modules at Indiana University and Louisiana State University, and a half-day hands-on tutorial at CloudCom conference. Highlights

of outreach activities have included demos at major conferences, including demonstrations of deployments across FutureGrid and Grid'5000 resources for experiments with inter-cloud computing for bio-informatics applications (CloudBLAST). The FutureGrid TEO team has also generated requirements in terms of user accounts based on educational use case scenarios that have served as input for software and security teams.

7.1.2 Tutorials

The following FutureGrid tutorials have been created and are available on the FutureGrid portal. These are available from http://futuregrid.org/tutorials

Tutorial topic 1: Cloud Provisioning Platforms

FutureGrid Tutorial NM1: Using Nimbus on FutureGrid

FutureGrid Tutorial NM2: Nimbus One-click Cluster Guide

FutureGrid Tutorial GA6: Using the Grid Appliances to run FutureGrid Cloud Clients

FutureGrid Tutorial EU1: Using Eucalyptus on FutureGrid

Tutorial topic 2: Cloud Run-time Platforms

FutureGrid Tutorial HA1: Introduction to Hadoop using the Grid Appliance

FutureGrid Tutorial HA2: Running Hadoop on FG using Eucalyptus (.ppt)

FutureGrid Tutorial HA2: Running Hadoop on Eualyptus

Tutorial topic 3: Educational Virtual Appliances

FutureGrid Tutorial GA1: Introduction to the Grid Appliance

FutureGrid Tutorial GA2: Creating Grid Appliance Clusters

FutureGrid Tutorial GA3: Building an educational appliance from Ubuntu 10.04

FutureGrid Tutorial GA4: Deploying Grid Appliances using Nimbus

FutureGrid Tutorial GA5: Deploying Grid Appliances using Eucalyptus

FutureGrid Tutorial GA7: Customizing and registering Grid Appliance images using Eucalyptus

FutureGrid Tutorial MP1: MPI Virtual Clusters with the Grid Appliances and MPICH2

Tutorial topic 4: High Performance Computing

FutureGrid Tutorial VA1: Performance Analysis with Vampir

FutureGrid Tutorial VT1: Instrumentation and tracing with VampirTrace

7.1.3 Virtual appliances and GroupVPN

Development activities focused on development, testing and improvements of the baseline educational virtual appliance image, with emphasis on integration with Nimbus and Eucalyptus and tests on FutureGrid hardware (foxtrot and india). Grid appliance-based virtual clusters have been successfully deployed on both foxtrot (using Nimbus) and India (using Eucalyptus). The Grid appliance was successfully instantiated and automatically connected, via the GroupVPN

virtual network, to a PlanetLab overlay and other Grid appliances running Condor and Hadoop virtual clusters. Grid appliance images are available both for user download to their own resources, as well as on image repositories of FutureGrid. These are available from http://www.grid-appliance.org

7.1.4 Demonstrations and presentations

Outreach activities have included various demonstrations and presentations given by the FutureGrid partners. Presentations and videos from tutorials given at the CloudCom 2010 conference are available from http://futuregrid.org/presentations and http://futuregrid.org/tutorials

7.1.5 Class usage

FutureGrid has been used in support of several educational activities. Highlights include classes ranging from a half-day hands-on tutorial at CloudCom, a week-long "Big Data for Science" workshop held across various institutions through video-conference, a Cloud topics class at Indiana University, and a new, innovative semester-long class offered Fall 2010 at Louisiana State University.

In the half-day tutorial at CloudCom, given by Dr. Kate Keahey from U. Chicago, approximately 30 attendees were given hands-on access to FutureGrid resources through Nimbus, and were able within a short session to create and manage virtual machines running on cloud FutureGrid resources.

In the "Big Data for Science" workshop, over 200 students across 10 institutions (Arkansas High Performance Computing Center, University of Arkansas, Fayetteville; Electronic Visualization Laboratory, University of Illinois at Chicago; Indiana University, Bloomington; Institute for Digital Research and Education, University of California, Los Angeles; Michigan State University, East Lansing; Pennsylvania State University, University Park; University of Iowa, Iowa City; University of Minnesota Supercomputing Institute, Minneapolis; University of Notre Dame, Notre Dame, Indiana; and University of Texas at El Paso). Additionally 100 additional students attended via streaming video. Students in the workshop used FutureGrid in hands-on activities that covered, among others, Hadoop/MapReduce, Twister, Grid Appliance, and GroupVPN.

At Louisiana State University, FutureGrid supported a new class focusing on a practical and comprehensive graduate course preparing students for research involving scientific computing. Module E (Distributed Scientific Computing) taught by Shantenu Jha used FutureGrid in hands-on assignments on: Introduction to the practice of distributed computing; Cloud computing and master-worker pattern; and Distributed application case studies.

At Indiana University, Judy Qiu taught a graduate level Cloud Topics class with class activity of 27 graduate students supported on FutureGrid (https://www.futuregrid.org/Qiu/classroom). In Spring 2011, she will follow this with the core distributed systems class with an enrollment of over 60 Undergraduate and Graduate students.

In 2011, we will exploit a new book on Distributed and Cloud computing written by Hwang, Fox and Dongarra to provide a coherent framework for some of our educational material.

7.2 Minority Serving Institution Engagement Plan

7.2.1 Introduction

The FutureGrid team leverages extensive, pre-existing activities at FutureGrid partners and the TeraGrid to involve MSIs in our project. This allows us to offer virtual clusters and test-beds focused on teaching and developing FutureGrid applications. These capabilities are a consequence of expected operations and require no additional effort. In order to make MSIs aware of FutureGrid capabilities available to them, we are engaging in an outreach and for example in December 2010, Fox presented FutureGrid at an Association of Computer/Information Sciences and Engineering Departments at Minority Institutions (ADMI) meeting at Elizabeth City State University.

MSI activities include providing resources for MSI faculty to teach systems programming on individual machines and clusters as well as preconfigured, dynamically instantiated environments for teaching parallel programming, web programming, grid and cloud programming, and computational science. We also suggest providing designated FutureGrid experts to support particular MSI groups.

Principal Investigator Geoffrey Fox has an established track record of working with MSIs. Similarly, Dr. Jose Fortes and his colleague Dr. Renato Figueiredo have expertise in use of social networking techniques for engaging individuals from traditionally underserved groups. We note the distinction between engaging with MSIs as opposed to engaging with a few students from MSIs. This plan is for engagement at the institutional level. It is based on two key strategies – leveraging MSI contacts such as ADMI and the MSI Cyberinfrastructure Empowerment Coalition (MSI-CIEC) and using social networking tools.

7.2.2 Types of MSI Engagement

We have already noted several successful educational uses of FutureGrid and it is natural to build on this for MSI's as FutureGrid is probably better suited for outreach and "wide computing" than the high-end TeraGrid facilities.

Our goals for collaborating with MSIs include the following:

- Teaching faculty how to use FutureGrid resources (virtual machines and virtual clusters) to teach basic distributed computing, systems programming, and system administration in the classroom. FutureGrid provides a secure sandbox that allows each student to have his/her own test-bed in isolation from other students and operational facilities.
- Providing MSI faculty with preconfigured environments for teaching parallel, web, distributed, and grid computing.
- Enabling teaching and research collaborations between MSI institutions and experts in grid and cloud technologies and research.
- Teaching faculty how to build test-bed versions of FutureGrid out of resources at their institutions for classroom use.
- Teaching students how to use FutureGrid tools through internships.
- Ultimately, ensuring that computational sciences in particular and STEM disciplines in general have the benefit of the talents of the best and brightest individuals. Conversely, we wish to engage such students through FutureGrid and expose them to a scientific instrument shaping the future.

- Helping MSI faculty participate as an equal partner in Cyberinfrastructure enhanced research.

7.2.3 Activities Leveraging Existing MSI Contacts

Fox is currently a principal member and founder of the MSI Cyberinfrastructure Empowerment Coalition (MSI-CIEC), which has been funded by the NSF CI-TEAM and other awards. MSI-CIEC's primary theme is to "teach the teachers" at MSIs so that they can incorporate cyberinfrastructure into their research and involve students and staff at their home institutions. MSI-CIEC's current principal activity is the organization of Cyberinfrastructure Days at various MSIs. These daylong workshops feature prominent speakers who discuss the application of cyberinfrastructure to research and education.

In addition to the MSI-CIEC, Fox and the FutureGrid team work closely with Maureen Biggers, Indiana University's assistant dean for diversity and education. Biggers' qualifications include acting as project manager for the National Science Foundation's Broadening Participation in Computing Alliance for the Advancement of African-American Researchers in Computing, and as a member of the leadership team for the National Center for Women and Information Technology. We are working with Biggers to organize outreach and pursue REU funding to bring MSI students to IU for summer internships and to coordinate education and training workshops. Fox is co-PI of a funded REU program related to his work on ice sheet dynamics with the NSF Science and Technology center CReSIS led by Kansas University. In the summer of 2010, 4 HBCU students from ADMI institutions were funded by this NSF project.

Finally, FutureGrid involves students from Historically Black Colleges and Universities (HBCUs) through Indiana University's STEM Initiative (http://www.stem.indiana.edu/). This program provides travel, housing, and support for HBCU students to intern at Indiana University during the summer. We particularly expect to engage the MSIs listed in Table 8, with which Indiana University has already established formal collaborative agreements.

Institution	Location
Alabama A&M	Normal, AL
Bennett College for Women	Greensboro, NC
Clark Atlanta University	Atlanta, GA
Hampton University	Hampton, VA
Jackson State University	Jackson, MS
Langston University	Langston, OK
Morgan State University	Baltimore, MD
Morehouse College	Atlanta, GA
Xavier University	New Orleans, LA
Tennessee State University	Nashville, TN
North Carolina Central University	Durham, NC
Clark Atlanta University	Atlanta, GA

Table 8. Minority Serving Institutions with which Indiana University has a formal collaborative agreement, and which we expect to engage in using FutureGrid.

Finally we will apply for an REU supplement for FutureGrid each year and this funded 2 HBCU students in summer 2010. Note that in total for the summer of 2010, a total of 10 HBCU

undergraduates were hosted by the Pervasive Technology Institute at Indiana University and many of these used FutureGrid resources.

We are planning with Elizabeth City State University a summer school for ADMI faculty and students on cloud computing with a short "teaser tutorial" at the 2011 ADMI symposium at Clemson which already has a Cloud Computing emphasis.

7.2.4 Leveraging Social Networking Technologies

U. Florida is applying virtual appliance and social networking systems developed at U. Florida (http://www.grid-appliance.org, http://www.socialvpn.org) to facilitate the dissemination of the grid test-bed software for education, training, and development. This allows MSI educators and students to quickly (within minutes) gain hands-on access to a system that has the same software stack of the grid test-bed but runs on their own resources – without worrying about software installation, configuration, or the time taken to request and process an account.

The system we propose allows an individual or groups of users to easily deploy an ad-hoc virtual private network of virtual machines that would run the same software that runs in the grid test-bed. All they need to do is download a VM image that runs out-of-the-box in a free VM monitor (e.g., VM Player or VirtualBox), create a group in a social network infrastructure (e.g., Facebook), and turn on the appliances to create an ad-hoc virtual cluster. This enables interesting usage scenarios in education and training.

7.3 Outreach to New Users

The issue of attracting scientists to use FutureGrid can be broken down into the four categories of section 5 – attracting educators, attracting computer/computational science researchers; attracting interoperability users; and attracting domain scientists to use FutureGrid. We note that FutureGrid presentations and FutureGrid user success stories are two major ways we attract new users and we are currently improving the web site to highlight this better. FutureGrid leaders have given approximately 20 significant presentations on FutureGrid with venues including OGF, HPDC, TG'10, HPCC, SC10 and CloudCom. This activity will continue. We also expect increased activity with TeraGrid as latter transitions to XD and FutureGrid itself moves further into production status.

7.3.1 Attracting Educators

We believe that the key to attracting educators will be having early exemplars of successful use of FutureGrid in education, and high-quality curriculum materials that educators can adapt and reuse. We already have a good start here described earlier in this section. We intend to evaluate initial uses of FutureGrid in courses this fall and adjust our support based on this feedback. We will document this well on the FutureGrid web site with extensive hands-on material. Natural pro-active efforts in this area include working with TeraGrid XD TEOS and interactions with NSF education (EHR) and outreach (CISE BPC, OCI Citeam) programs.

7.3.2 Attracting Grid and Cloud Middleware and Technology Users

Computer and Computational scientists are attracted to FutureGrid through a variety of mechanisms – talks and posters at conferences, articles in such publications as HPCwire, announcements on NSF, TeraGrid, and Open Science Grid web pages, etc. We believe it is relatively straightforward to generate interest in this area especially right now, with so many

claims and counterclaims regarding performance, efficacy, and ease of use of many new grid and cloud environments. In fact computer science systems research describes a significant number of projects described in section 5. We believe there is tremendous interest in being able to do grid and cloud research with the sort of rigor that in past years has characterized parallel scalability research and that the computer/computational science community will be highly motivated to conduct high-quality research in a configurable test-bed. We plan to enhance motivation to use FutureGrid by making it convenient for researchers to deposit results in open repositories such as http://www.myexperiment.org/. We plan visits to NSF including both CISE and OCI. Our work with Grid5000 (whose main users are computer scientists) is relevant here and we plan a joint workshop in 2011. We will also extend our support of data-intensive computing with both the new system and additional disk space for existing systems.

7.3.3 Attracting Interoperability Users of FutureGrid

As discussed in section 5, interoperability is an interesting use of FutureGrid. Many interoperability activities use testbeds and already we are positioned well in Grid and cloud space and can expect to expand these areas. There are other possibilities that we will explore including the Open Geospatial Consortium OGC where we already have contacts. In general interoperability is usually addressed through consortium organizations (like OGF, OGC, SNIA, DMTF, IEEE Clouds etc.) and we need to reach out to these through leadership contacts and attendance at meetings. In the case of IEEE Clouds for example, we hosted a panel and workshop for this group at the IEEE CloudCom conference.

7.3.4 Attracting Domain Scientists

We expect it to be somewhat more challenging to attract domain scientists to FutureGrid, but we already have substantial interest from Life Science applications that we intend to expand. These applications come from institutions (Oregon, North Texas, Vermont) outside the partners as well as San Diego and Indiana. We will focus on generating successes here that can be used as exemplars for further users. We believe two factors will be critical in attracting domain scientists to FutureGrid: making the process of applying for usage simple and sending domain scientists to domain science conferences to discuss the value of the facility to the science domain. We plan to do both. We will visit the application directorates at NSF and also reach out to NIH and NSF projects including the big projects such as iPlant (which has a major cloud initiative), OOI and NEON. These are all addressing data-intensive applications and are attractive early users of FutureGrid. Support of MSI (Minority Serving Institutions) scientists will also be an important possibility here and we have for example interest from Chemistry at University of Houston Downtown and remote sensing at Elizabeth City State. FutureGrid with its emphasis on broad use and education is well matched to MSI's. We are discussing collaboration with OSG (Open Science Grid) which will bring in new applications. Finally we note that once XD is established it will be natural to target traditional TeraGrid applications wanting to develop new codes.

APPENDIX A – List of FG Projects

Table 9: Projects on FutureGrid through January 10, 2011. Some early work before we set up proper tracking are missing. This can be found dynamically at http://portal.futuregrid.org/projectssummary with ability to click down for detailed information.

Project Ref ID	User	Title	Institution	Date Started	Keywords
[FG-P72]	Judy Qiu	B534 Distributed systems Graduate/Undergraduate Class	Indiana University, School of Informatics and Computing	01/08/2011	Education MapReduce Virtual Machines MPI
[FG-P69]	Jiaan Zeng	Investigate provenance collection for MapReduce	Indiana University, Pervasive Technology Institute	01/03/2011	MapReduce, Twister, Hadoop, Provenance
[FG-P67]	Aniket Rastogi	Cloud Security and Forensics	Symbiosis Centre for Information Technology	01/03/2011	Cloud, Security, Incidents
[FG-P49]	JunWeon Yoon	Experiments for Science Cloud	SuperComputing Center, KISTI(Korea Information Science and Technology Institute)	01/03/2011	Science, Climate, Astronomy, Biology, Nimbus, Eucalyptus
[FG-P66]	Christopher Hemmerich	Collaborative Genomic Analysis Software Development	Indiana University, Center for Genomics and Bioinformatics	12/19/2010	TIGR, Twister, HPC, Amazon
[FG-P65]	Yang Ruan	Collaborative Genomic Analysis Software Development	Indiana University, Pervasive Technology Institute	12/16/2010	Bioinformatics, Twister, TIGR, Genomics, Eucalyptus
[FG-P63]	Aaron Buechlein	Collaborative Genomic Analysis Software Development	Indiana University, Center for Genomics and Bioinformatics	12/15/2010	TIGR, Twister, HPC, Amazon
[FG-P64]	Masoud Valafar	Storage security for Clouds	University of Oregon, Computer Science	12/15/2010	Clouds, Storage, Security
[FG-P62]	Charng-Da Lu	XD TAS: Evaulation of using XD TAS in FutureGrid	University at Buffalo, SUNY, Center for Computational Research	12/14/2010	TeraGrid, XD, Auditing, Clouds, HPC
[FG-P61]	Albert Everett	Hadoop Evaluation	University of Arkansas at Little Rock	12/13/2010	Hadoop, MPI, OpenMP, Evaluation
[FG-P60]	Lizhe Wang	Wide area distributed file system for MapReduce applications on FutureGrid platform	Indiana University, Pervasive Technology Institute	12/12/2010	Hadoop, MapReduce, Distributed, File, System
[FG-P59]	Alexandra Carpen-Amarie	Scalable data management for cloud services	INRIA Rennes - Bretagne Atlantique Research Center	12/10/2010	Nimbus, Cumulus, BlobSeer
[FG-P58]	Jong Youl Choi	Parallel Performance of GTM Dimension Reduction	Indiana University, Pervasive Technology Institute	12/09/2010	MPI, GTM, Benchmarking, Parallel

Project Ref ID	User	Title	Institution	Date Started	Keywords
[FG-P57]	Greg Cross	Use of xray and hotel	University of Chicago, Computation Institute	12/08/2010	xray, hotel, systems, administration
[FG-P55]	Adam Hughes	Collaborative Genomic Analysis Software Development	Indiana University, Pervasive Technology Institute	12/08/2010	genomics, biosequence, pipeline, Twister, Hadoop
[FG-P54]	Randall Sobie	Investigating cloud computing as a solution for analyzing particle physics data	University of Victoria, Dept of Physics and Astronomy	12/07/2010	Nimbus, particle, physics, cloud, computing, data, preservation, BaBar
[FG-P53]	Xiaoyong Zhou	Combining public cloud and private cloud	Indiana University, School of Informatics and Computing.	12/06/2010	Privacy, Hybrid, Clouds, MapReduce
[FG-P52]	David Lowenthal	Cost-Aware Cloud Computing	The University of Arizona, Dept. of Computer Science	12/06/2010	iPlant, MapReduce, Clouds
[FG-P51]	Thomas William	Vampir	Technische Universität Dresden	12/06/2010	Performance, Analysis, HPC
[FG-P50]	Yunhi Kang	Performance evaluation of MapReduce applications	Indiana University, Pervasive Technology Institute	12/03/2010	MapReduce, Performance, Evaluation, Virtual, Machine
[FG-P48]	Thilina Gunarathne	Cloud Technologies for Bioinformatics Applications	Indiana University, Pervasive Technology Institute	12/02/2010	Hadoop, DryadLINQ, Bioinformatics
[FG-P47]	Michael Wilde	Parallel scripting using cloud resources	Argonne National Laboratory, Computation Institute	12/01/2010	Swift, parallel, scripting
[FG-P44]	David Chiu	Managing an Adaptive Cloud Cache for Supporting Data-Intensive Applications	Washington State University, School of Engineering and Computer Science	11/25/2010	Clouds, Cache, Dataintensive
[FG-P43]	Robert Henschel	ScaleMP Performance Evaluation	Indiana University	11/24/2010	Bioinformatics, ScaleMP, Gene, Assembly, Large, Memory
[FG-P37]	Michel Drescher	EGI-InSPIRE	EGI.eu, EGI.eu	11/21/2010	Quality, Assurance, Grid, Software, EGI
[FG-P5]	Sumin Mohanan	Policy based distributed computing	University of Minnesota, Department of Computer Science and Engineering	11/17/2010	SAGA, Grids, Clouds, Policy
[FG-P36]	Ryan Hartman	Advanced Technology for Sensor Clouds	UITS PTI CGL, Indiana University	11/14/2010	Sensors, Clouds, Grids
[FG-P34]	Yonggang Liu	Parallel File System	Indiana University, Pervasive Technology Institute	11/11/2010	Parallel, I/O, Parallel, File, Systems, Nimbus, QoS

Project Ref ID	User	Title	Institution	Date Started	Keywords
[FG-P32]	Mariusz Mamonski	Interoperability tests between OGF HPC-BasicProfile endpoints	Poznan Supercomputing and Networking Center	11/09/2010	OGSA-BES, Interoperability, Genesis, II, SMOA, Unicore
[FG-P3]	Tak-Lon Wu	Survey of Open-Source Cloud Infrastructure using FutureGrid Testbed	Indiana University, PTI	11/09/2010	survey, cloud, iaas
[FG-P33]	Jenett Tillotson	Comparing Moab metascheduling to Condor and MCP (Modified Critical Path)	Indiana University, Research Technologies	11/09/2010	Moab, Nimbus, Condor, Metascheduling
[FG-P31]	Anthony Chronopoulos	Integrating High Performance Computing in Research and Education for Simulation, Visualization and RealTime Prediction	University of Texas at San Antonio, Department of Computer Science	11/05/2010	Education, Research, Clouds, MapReduce
[FG-P8]	Gideon Juve	Running workflows in the cloud with Pegasus	University of Southern California, Information Sciences Institute	11/05/2010	Running workflows in the cloud with Pegasus
[FG-P30]	Warren Smith	Publish/Subscribe Messaging as a Basis for TeraGrid Information Services	Texas Advanced Computing Center	11/05/2010	Nimbus, TeraGrid, Information, Services, Publish/Subscribe
[FG-P29]	Kashi Revanna	Metagenomics	University of North Texas, Department of Biology	11/04/2010	Metagenomics
[FG-P28]	Adam Hughes	Biosequence Alignment Studies	Indiana University, Pervasive Technology Institute	11/03/2010	Bioinformatics, Twister, MapReduce,
[FG-P4]	Jonathan Klinginsmith	Word Sense Disambiguation for Web 2.0 Data	Indiana University, Computer Science	11/02/2010	MapReduce, Natural Language Processing
[FG-P27]	Gerald Guo	Scientific application performance test on Hadoop/HBase	Indiana University, Pervasive Technology Institute	10/28/2010	Hadoop, Hbase, Dataintensive
[FG-P25]	James Vincent	Collaborative Research: North East Cyberinfrastructure Consortium	University of Vermont, Vermont Genetics Network	10/22/2010	Bioinformatics, Clouds, Workflow
[FG-P24]	Kyungyong Lee	Resource discovery in an asynchronous grid and cloud	University of Florida, ACIS Lab.	10/21/2010	Resource, Discovery, Grid, Cloud
[FG-P23]	Shirley Moore	Hardware Performance Monitoring in the Clouds	University of Tennessee	10/19/2010	Clouds, PAPI, Performance, Monitoring
[FG-P22]	Ole Weidner	SAGA	Louisiana State University, Center for Computation & Technology	10/15/2010	SAGA, Grids, Clouds
[FG-P20]	Hyungro Lee	Development of an information service for FutureGrid	PTI, Indiana University	10/14/2010	Information, Service, FutureGrid
[FG-P21]	Lucas Wilson	FutureGrid - experiment harness	Texas Advanced	10/14/2010	experiment, harness,

Project Ref ID	User	Title	Institution	Date Started	Keywords
			Computing Center		FutureGrid
[FG-P19]	Yunhong Gu	PIRE: Training and Workshops in Data Intensive Computing Using The Open Science Data Cloud	University of Illinois at Chicago	10/12/2010	Sector, Dataintensive, Clouds
[FG-P7]	Mats Rynge	SDCI NMI Improvement: Pegasus: From Concept to Execution- - -Mapping Scientific Workflows onto the National Cyberinfrastructure	USC, ISI	10/08/2010	Pegasus, Workflow, Nimbus, Eucalyptus, Elastic Resources
[FG-P18]	Yangyi Chen	Privacy preserving gene read mapping using hybrid cloud	Indiana University, School of Informatics and Computing	10/04/2010	Security/Privacy, Bioinformatics, Hybrid, Cloud, Hadoop
[FG-P2]	Gregor von Laszewski	Deploy OpenNebula on FutureGrid	Indiana University, Community Grids Lab at the Pervasive Technology Institute	10/04/2010	OpenNebula, IaaS
[FG-P17]	Hui Li	Comparison of MapReduce Systems	Indiana University, CGL at PTI	09/22/2010	Twister, Hadoop, Dryad
[FG-P16]	John Naab	FutureGrid and Tempest/Madrid/Storm Support	Indiana University, Pervasive Technology Institute	09/22/2010	FutureGrid, Tempest/Madrid/Storm, Support
[FG-P15]	Panoat Chuchaisri	Grid Appliance	University of Florida, CISE Department	09/15/2010	Grid, Appliance, Clouds, Nimbus, Eucalyptus
[FG-P12]	Xiaoming Gao	The Virtual Block Store system	Indiana University, Pervasive Technology Institute	09/15/2010	Virtual, Block, Store, Eucalyptus, Nimbus, Lustre, Dataintensive
[FG-P6]	Randy Heiland	Parameter sweeps for multi-cell models on FutureGrid	Indiana University, Pervasive Technology Institute	09/14/2010	Biocomplexity, Clouds, TeraGrid, HPC
[FG-P11]	Pradnya Kakodkar	CSCI B649 (Topics in Systems/Cloud Computing)	Indiana University, Computer Science	09/13/2010	Education, Clouds, MapReduce
[FG-P10]	John Lockman	TeraGrid XD TIS(Technology Insertion Service) Technology Evaluation Laboratory	University of Texas at Austin, Texas Advanced Computing Center	09/10/2010	TeraGrid, Technology, Insertion, Service
[FG-P1]	Renato Figueiredo	Peer-to-peer overlay networks and applications in virtual networks and virtual clusters	University of Florida	09/08/2010	P2P, HPC, Virtual Networking, Cybersecurity, Resource Discovery
[FG-P77]	Jens-S. Vöckler	Periodogram Workflow Running on FutureGrid Using Pegasus	University of Southern California, Information Sciences Institute	08/01/2010	Sky computing, Periodogram Workflow, Pegasus
[FG-P71]	Judy Qiu	B649 Topics on Systems: Graduate Cloud Computing Class Fall 2010	Indiana University, School of Informatics and Computing	08/01/2010	Education, MapReduce, Hadoop, Twister, Dryad

Project Ref ID	User	Title	Institution	Date Started	Keywords
[FG-P76]	Paul Marshall	Differentiated Leases for Infrastructure-as-a-Service	University of Colorado at Boulder	07/01/2010	IaaS
[FG-P75]	John Bresnahan	Cumulus	University of Chicago	07/01/2010	Cloud, Storage, Cumulus
[FG-P74]	Pierre Riteau	Sky Computing	University of Rennes	07/01/2010	sky computing, federated clouds
[FG-P73]	Shava Smallen	TeraGrid QA Testing and Debugging	UC San Diego, San Diego Supercomputer Center	07/01/2010	TeraGrid, GRAM5, GridFTP5, QA
[FG-P70]	Judy Qiu	Big Data for Science Virtual Summer School July 26-30 2010	Indiana University, School of Informatics and Computing	07/01/2010	Education, Hadoop, Twister, Data Intensive
[FG-P68]	John Duer	CFD and Workload Management Experimentation	Cummins Inc., Combustion Research	05/01/2010	CFD, HPC, Cray, India
[FG-P46]	Amy Apon	Data Analysis in the Cloud	University of Arkansas	05/01/2010	Virtual, Machines, Data-intensive, Minorities, Education
[FG-P45]	Shantenu Jha	Experiments in Distributed Computing	Louisiana State University, Center for Computation & Technology	05/01/2010	Education, Research, SAGA, OGF, Standards, Dataintensive
[FG-P42]	Andre Merzky	SAGA	Louisiana State University, Center for Computation and Technology	05/01/2010	SAGA, API, Distributed, Applications, Interoperability
[FG-P40]	Shava Smallen	Inca	UC San Diego, San Diego Supercomputer Center	05/01/2010	INCA, Monitoring
[FG-P39]	Shava Smallen	TeraGrid QA testing and debugging	UC San Diego, San Diego Supercomputer Center	05/01/2010	TeraGrid, Quality, Assurance
[FG-P38]	Catherine Olschanowsky	Fine-grained Application Energy Modeling	San Diego Supercomputer Center at UCSD	05/01/2010	Energy, GreenIT
[FG-P26]	John Conery	Bioinformatics and Clouds	Oregon University, Computer Science Department	05/01/2010	Clouds, Bioinformatics
[FG-P14]	Shantenu Jha	Distributed Scientific Computing Class	Louisiana State University, Center for Computation & Technology	05/01/2010	dataintensive, SAGA, Research, and, Education, Eucalyptus
[FG-P13]	Andrew Younge	FutureGrid Systems Development and Prototyping	Pervasive Technology Institute, Indiana University	05/01/2010	Grid, Cloud, ScaleMP
[FG-P9]	Ilkay Altintas	Distributed Execution of Kepler Scientific Workflow on Future Grid	UCSD, SDSC	05/01/2010	Distributed Execution of Kepler Scientific Workflow on Future Grid

Project Ref ID	User	Title	Institution	Date Started	Keywords
[FG-P56]	Robert Henschel	Windows and Linux performance comparison	Indiana University, UITS RT	01/01/2010	sierra, Linux, Windows, Benchmarking
[FG-P41]	Kiruba Karan	Cloud Computing	BIT (Bannari Amman Institute of Technology) Sathyamangalam, Tamil Nadu	pending	Grids, Clouds, Scheduling
[FG-P35]	Ahmed Alothman	Software Engineering and VM's	Canberra Australia	pending	Education, Masters, Software, Engineering

APPENDIX B – Partial List of FutureGrid Projects with significant achievements

---

Results for Project "Windows and Linux performance comparison"

Robert Henschel
Indiana University, UITS RT

A collection of performance benchmarks have been run on the FutureGrid IBM System X iDataPlex cluster using two different operating systems. Windows HPC Server 2008 (WinHPC) and Red Hat Enterprise Linux v5.4 (RHEL5) are compared using SPEC MPI2007 v1.1, the High Performance Computing Challenge (HPCC) and National Science Foundation (NSF) acceptance test benchmark suites. Overall, we find the performance of WinHPC and RHEL5 to be equivalent but significant performance differences exist when analyzing specific applications. We focus on the results from the application benchmarks and include the results of the HPCC microbenchmark for completeness.

SPEC 2010 workshop paper available at http://hdl.handle.net/2022/9919

Results for Project "Distributed Scientific Computing Class"

Shantenu Jha
Louisiana State University, Center for Computation & Technology

FutureGrid supported a new class focusing on a practical and comprehensive graduate course preparing students for research involving scientific computing. Module E (Distributed Scientific Computing) taught by Shantenu Jha used FutureGrid in hands-on assignments on:

- Introduction to the practice of distributed computing;
- Cloud computing and master-worker pattern;
- and Distributed application case studies.

A paper is in preparation and will be posted when complete.

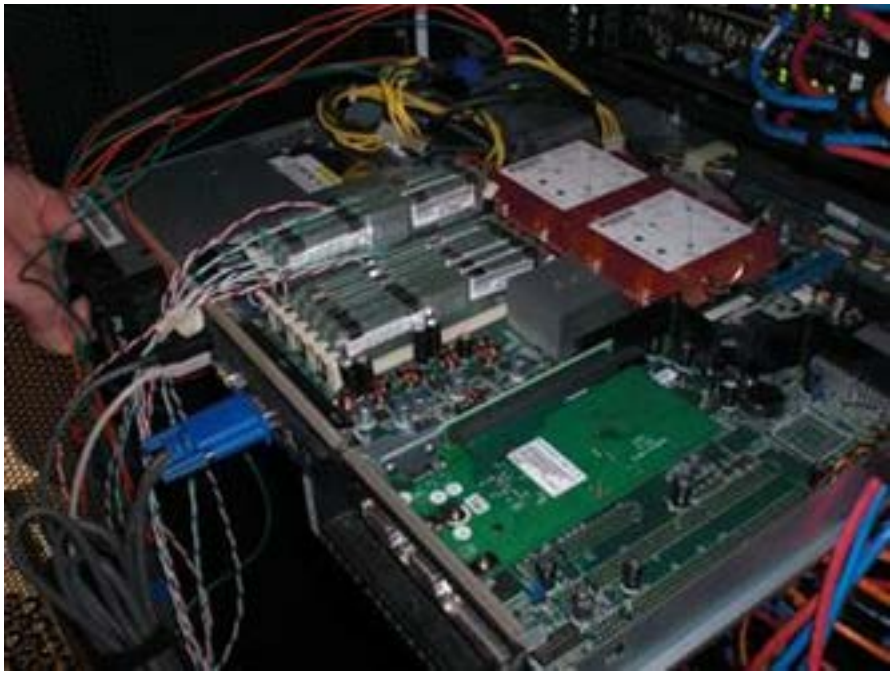
Results for Project "Fine-grained Application Energy Modeling"

Catherine Olschanowsky
San Diego Supercomputer Center at UCSD

The following success story illustrates bare-metal access to FutureGrid where the user's experiment required physical access to a FutureGrid node to attach a device needed to gather data for their research.

As with performance, energy-efficiency is not an attribute of a compute resource alone; it is a function of a resource-workload combination. The operation mix and locality characteristics of the applications in the workload affect the energy consumption of the resource. Data locality is the primary source of variation in energy requirements. The major contributions of this work include a method for performing fine-grained DC power measurements on HPC resources, a benchmark infrastructure that exercises specific portions of the node in order to characterize operation energy costs, and a method of combining application information with independent energy measurements in order to estimate the energy requirements for specific application-resource pairings.

During August 2010, UCSD allocated a single node of the Sierra cluster to Olschanowsky for two weeks. During that time Olschanowsky attached a custom-made power monitoring harness to the node as shown in the next Figure.

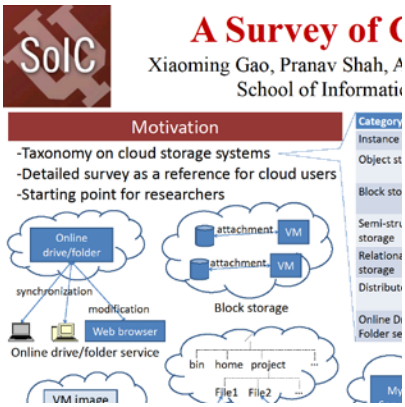


Fine-grained power measurements of components were taken by measuring the current close to each component; this is done using a custom harness to intercept the DC signals. Both CPUs and each memory DIMM were measured this way. The CPUs are measured by intercepting the signal at the power socket between the power supply and the motherboard; the DIMMs are measured using extender cards. In addition to the DC measurements course-grained power measurements are taken using a WattsUp device (a readily available power analyzer). Once installed a series of benchmarks were run to gather needed data for their models. This data will be included as part of Olschanowky's PhD dissertation. Olschanowsky is a PhD candidate for the Department of Computer Science and Engineering at UC San Diego.

The node that Olschanowsky used was tested and returned to service and was recertified by IBM.

Results for Project "B649 Topics on Systems: Graduate Cloud Computing Class"

Judy Qiu
Indiana University, School of Informatics and Computing



This class involved 27 Graduate students with a mix of Masters and PhD students and was offered fall 2010 as part of Indiana University Computer Science program. Many current FutureGrid experts went to this class which routinely used FutureGrid for student projects. Projects included

- Hadoop
- DryadLINQ/Dryad
- Twister
- Eucalyptus
- Nimbus
- Sector/Sphere
- Virtual Appliances
- Cloud Storage
- Clustering by Deterministic Annealing (DAC)
- Multi Dimensional Scaling (MDS)
- Latent Dirichlet Allocation (LDA)

See class web page http://salsahpc.indiana.edu/b649/. All students in class attended rthe CloudCom conference in Indianapolis November 30-December 3 and entered posters. The latter were judged by the attendees and the poster shown won top prize in the emerging research section.

Results for Project "TeraGrid QA Testing and Debugging"

Shava Smallen
UC San Diego, San Diego Supercomputer Center

This success story illustrates collaboration with TeraGrid and the ability to acquire short-term access to FutureGrid resources in order to perform QA testing of software.

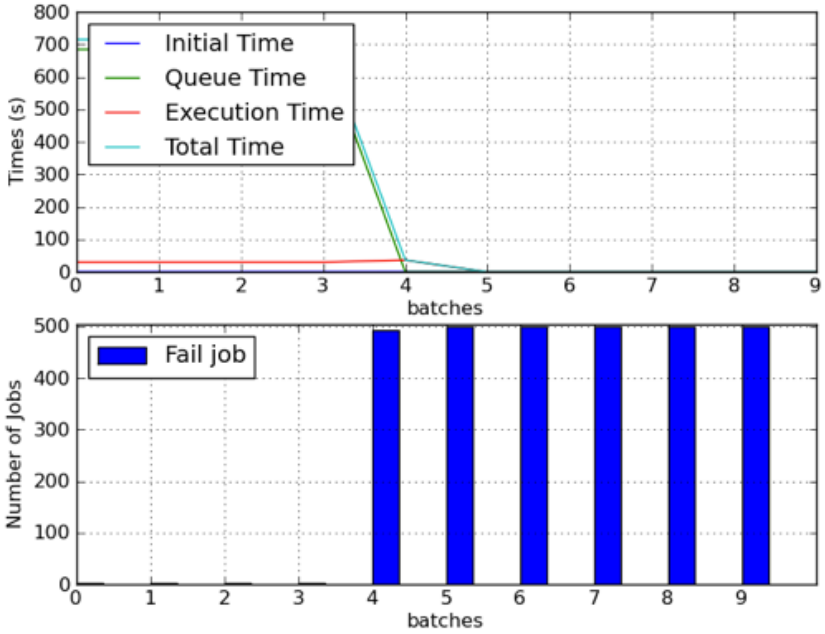
The mission of the TeraGrid Quality Assurance (QA) working group is to identify and implement ways in which TeraGrid software components/services and production deployments can be improved to reduce the number of failures requiring operator intervention that are encountered at TeraGrid resource provider sites. The TeraGrid QA group utilized FutureGrid in the below experiments:

GRAM 5 scalability testing: The TeraGrid Science Gateway projects tend to submit large amounts of jobs to TeraGrid resources usually through the Globus GRAM interfaces. Due to scalability problems with GRAM, members of the Science Gateway team at Indiana University extracted code from their GridChem and UltraScan Gateways anddeveloped a scalability test for GRAM. When GRAM 5 was released, GRAM 5 was deployed to a TACC test node on Ranger and scalability testing was started. Due to the possibility that the Ranger test node might be re-allocated, the group created an alternate test environment on FutureGrid in July 2010. A virtual cluster running Torque and GRAM 5 was created on UF's Foxtrot machine using Nimbus. Access to the virtual cluster was provided to the Science Gateway team as well. One problem that was

debugged on the virtual cluster was numerous error messages showing up in a log file in the user's home directory. This did not effect job execution but took up space in the user's home directory and was reported to the Globus developers. The effort is summarized in the following Wiki page at http://www.teragridforum.org/mediawiki/index.php?title=GRAM5_Scalability_Testing

GridFTP 5 testing: In order to test the newest GridFTP 5 release, the TeraGrid QA group again turned to FutureGrid and instantiated a single VM with GridFTP 5 on UCSD's Sierra and UF's Foxtrot machine in October 2010. They then verified several of the new features, such as data set synchronization and the offline mode for the server. No major problems were detected in this testing, though a bug related to the new dataset synchronization feature was reported. The results are summarized on the TeraGrid Wiki at http://www.teragridforum.org/mediawiki/index.php?title=GridFtp5_Testing.

Some results:



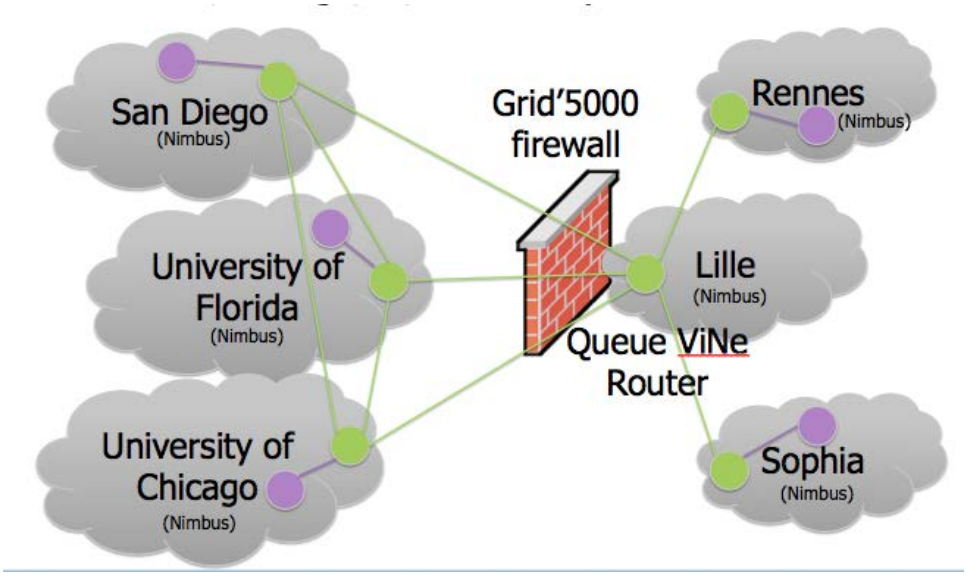
Results for Project "Sky Computing"

Pierre Riteau
University of Rennes

Problem: Scientific problems are often large and distributed by nature as they combine processing of data produced at various sources. In the context of cloud computing this leads to a question of what problems would arise if we were to use resources obtain from not just one IaaS cloud, but a federation of multiple geographically distributed infrastructure clouds. Such multi-cloud federations have been called “sky computing” (see [1]) and involve challenges of standardization, security, configuration, and networking.

Project: Pierre Riteau from the University of Rennes 1 proposed one solution in this space by creating a virtual cluster combining resources obtained from six geographically distributed Nimbus clouds: three hosted on Grid’5000 and three hosted on FutureGrid. Experimenting with distribution and scale he succeeded in creating a geographically distributed virtual cluster of over 1000 cores. His solution overcame firewall and incompatible network policy problems by using the ViNe overlay to create a virtual network and secure creation of a virtual cluster by using the Nimbus Conext Broker. Further, in order to overcme the image distribution problem which becomes a significant obstacle to fast deployment at this scale Pierre developed a QOCW system which uses copy-on-write techniques to speed up image distribution.

Widely distributed compatible cloud resources needed for this experiment would have been impossible to obtain with the existence of resources such as Grid’5000 and FutureGrid and their close collaboration. In addition to experimenting with research problems at unprecedented scale, this project was also a proof-of-concept and a trail blazer for a close collaboration between Grid’5000 and FutureGrid. Because of its integrative nature, this project was demonstrated at OGF 29 in June 2010.



References:

- Sky Computing on FutureGrid and Grid'5000, Pierre Riteau, Mauricio Tsugawa, Andrea Matsunaga, José Fortes, Tim Freeman, David LaBissoniere, Kate Keahey. TeraGrid 2010, Pittsburgh, PA. August 2010
  http://www.nimbusproject.org/files/riteau_tg10.ppt
- ISGTW article: Reaching for sky computing

www.isgtw.org/?pid=1002832

- *ERICM article: Large-Scale Cloud Computing Research: Sky Computing on FutureGrid and Grid'5000*

http://ercim-news.ercim.eu/en83/special/large-scale-cloud-computing-research-sky-computing-on-futuregrid-and-grid5000

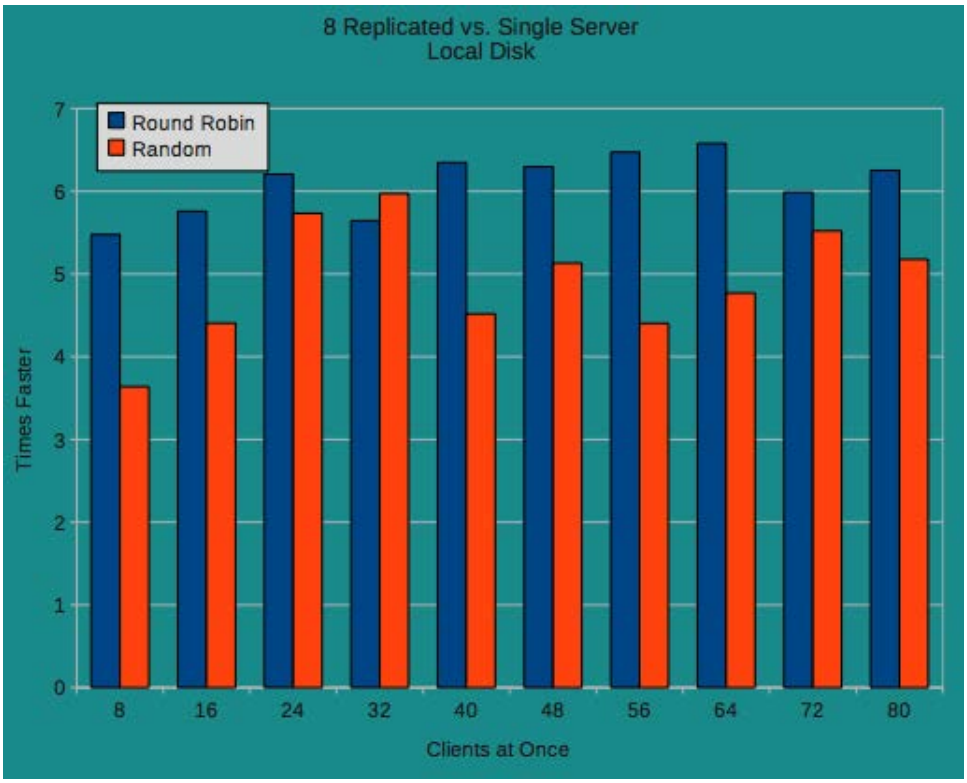
Results for Project "Cumulus"

John Bresnahan
University of Chicago

Problem: The advent of cloud computing introduced a convenient model for storage outsourcing. At the same time, the scientific community already has large storage facilities and software. How can the scientific community that already has accumulated vast amounts of data and storage take advantage of these data storage cloud innovations? How will the solution compare with existing models in terms of performance and fairness of access?

Project: John Bresnahan at the University of Chicago developed Cumulus, an open source storage cloud and performed a qualitative and quantitative evaluation of the model in the context of existing storage solutions, and needs for performance and scalability. The investigation defined the pluggable interfaces needed, science-specific features (e.g., quota management), and investigated the upload and download performance as well as scalability of the system in the number of clients and storage servers. The improvements made as a result of the investigation were integrated into Nimbus releases.

This work, in particular the performance evaluation part was performed on 16 nodes of the FutureGrid hotel resource. It was important to obtain not only dedicated nodes but also a dedicated network for this experiment because network disturbances could affect the measurement of upload/download efficiency as well as the scalability measurement. Further, for the scalability experiments to be successful it was crucial to have a well maintained and administered parallel file system. The GPFS partition on FutureGrid's Hotel resource provided this. Such requirements are typically hard to find on platforms other than dedicated computing resources within an institution.



References:

1. Cumulus: Open Source Storage Cloud for Science, John Bresnahan, Kate Keahey, Tim Freeman, David LaBissoniere. SC10 Poster. New Orleans, LA. November 2010.
2. http://www.nimbusproject.org/files/cumulus_poster_sc10.pdf

---

Results for Project "Differentiated Leases for Infrastructure-as-a-Service"

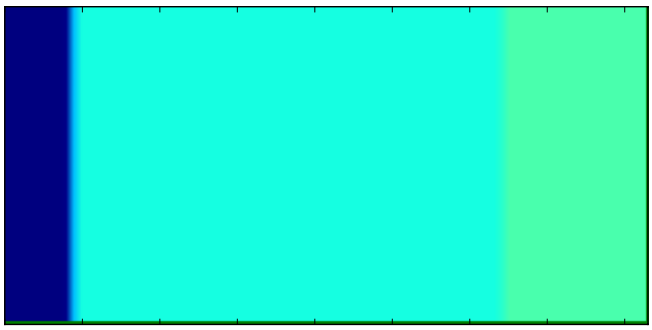
Paul Marshall
University of Colorado at Boulder

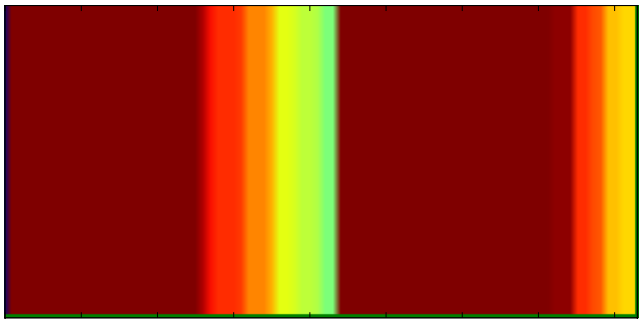
Problem: A common problem in on-demand IaaS clouds is utilization: in order to ensure on-demand availability providers have to ensure that there are available resources waiting for a request to come. To do that, they either have to significantly overprovision resources (in which case they experience low utilization) or reject a large proportion of requests (in which case the cloud is not really on-demand). The question arises: how can we combine best of both worlds?

Approach: Paul Marshall from the University of Colorado at Boulder approached this problem by deploying always-on preemptible VMs on all nodes of an IaaS cloud. When an on-demand request comes, the preemptible VMs are terminated in order to release resources for the on-demand request; when the nodes again become available the preemptible VMs are redeployed. Using this method, Paul was able to solve the utilization problem described above and demonstrate cloud utilization of up to 100%.

Since sudden preemption is typical in volunteer computing systems such as SETI@home or various Condor installations, this solution was therefore evaluated in the context of a Condor system measure its efficiency for the volnteer computing execution which was shown to be over 90%.

In order to evaluate his system experimentally, Paul first modified the open source Nimbus toolkit to extend its functionality to supports the backfill approach. He then had to deploy the augmented implementation on a sizable testbed that gave him enough privilege (root) to install and configure the augmented Nimbus implementation -- Such requirements are typically hard to find on platforms other than dedicated local resources. In this case however, this testbed was provided by the FutureGrid hotel resource (specifically we used 19 8-core FG nodes on hotel).





References:

1. Improving Utilization of Infrastructure Clouds, Paul Marshall, Kate Keahey, Tim Freeman, submitted to CCGrid 2011.

Results for Project "Periodogram Workflow Running on FutureGrid Using Pegasus"

Jens-S. VöcklerGideon JuveBruce BerrimanEwa DeelmanMats Rynge

USC-ISI USC-ISI IPAC-CalTech USC-ISI USC-ISI

The Periodogram workflow searches for extra-solar planets, either by “wobbles” in the radial velocity of a star, or dips in the star’s intensity. In either case, the workflow searches for repeating variations over time in the input “periodogram” data, a sub-set of the light curves released by the Kepler project. The workflow in this scenario only executed the “plav-chan”[1] algorithm, which is the computationally most intense. A full search needs to execute all three algorithms.

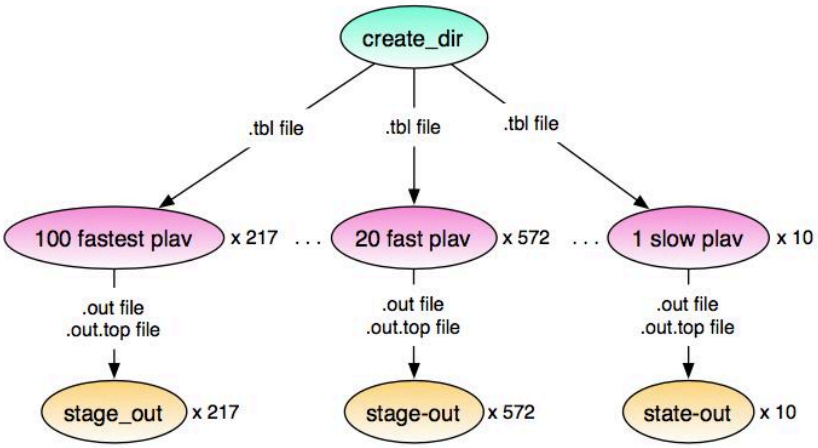


Figure 1: Workflow processing first release with one algorithm.

Figure 1 shows the complete workflow of 1,599 Condor jobs and 33,082 computational tasks, computing every periodogram twice.[2] The top (root) node in the workflow graph is an ancillary job, creating the necessary directory to receive the computational output.

In the first level under the top, the 33k computational tasks were binned into 799 Condor jobs depending on their run-time estimate: extremely fast (sub-second), fast (minutes) and slow (hours). The last bin for extremely fast and fast jobs was not completely filled. Each Condor job stages in all “.tbl” files that the job requires, and stages back all “.out” and “.out.top” result files. The staging happens through Condor-I/O between the submit machine at ISI, and the remote resources in FutureGrid. The “heavy lifting” with regards to staging happens this point.

In the last level, 799 Condor staging jobs, ancillary jobs that Pegasus generated, copy a file on the submit host between directories. This seemingly redundant stage takes almost no time, and is not reflected in the timings except total run-time. We are working on removing this stage from the workflow plan.

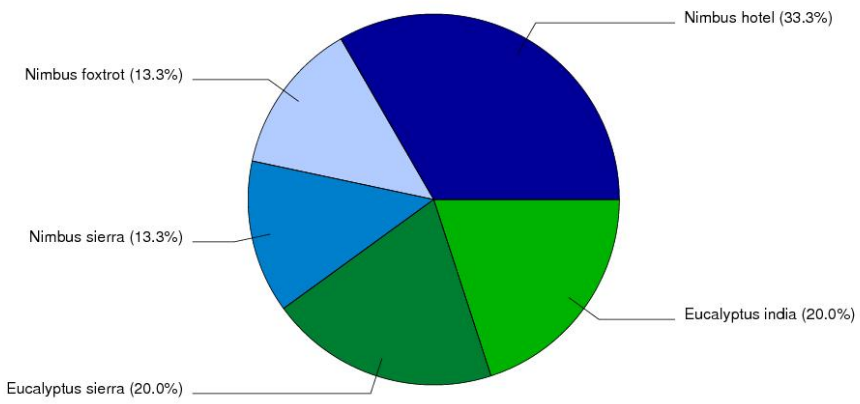


Figure 2: Requested Resources per Cloud.

Figure 2 describes the resource request setup. We requested 90 resources from Nimbus clouds (blues), and 60 from Eucalyptus clouds (greens). 1/3 of combined resources were provided by *sierra* (SDSC), 1/3 by *hotel* (UofC), and the final 1/3 shared between *india* (IU) and *foxtrot* (UFl). 150 machines in five clouds at four sites with two cloud middleware systems justify the term *Sky Computing* for this experiment.

The resources boot a Pegasus VM image that has the Periodogram software installed. Each provisioned image, based on a CentOS 5.5 system, brings up a Condor *startd*, which reports back to a Condor *collector* at ISI. As much as possible, we tried to request non-public IP modes, necessitating the use of the Condor connection broker (CCB).

On the provisioned image, each host came up with 2 cores, and each core had 2 Condor *slots* assigned to it. This computational over-subscription of the remote resources is considered not harmful for the periodogram workflow. Further experimentation will be required to validate this decision.

The provision requests were entered manually, using the Nimbus- and Eucalyptus client tools. After the first resources started reporting back to the Condor *collector*, the Pegasus-planned workflow was started, resulting in an instance of Condor DAGMan executing the workflow. Once the workflow terminated successfully, the resources were manually released.

Figure 3 shows a combination of available Condor slots and jobs in various states for the duration of the workflow. The blue line shows the provisioned slots as they become available over time, thus starting in negative time with regards to the workflow. The start of the workflow indicates 0 in the x-axis.

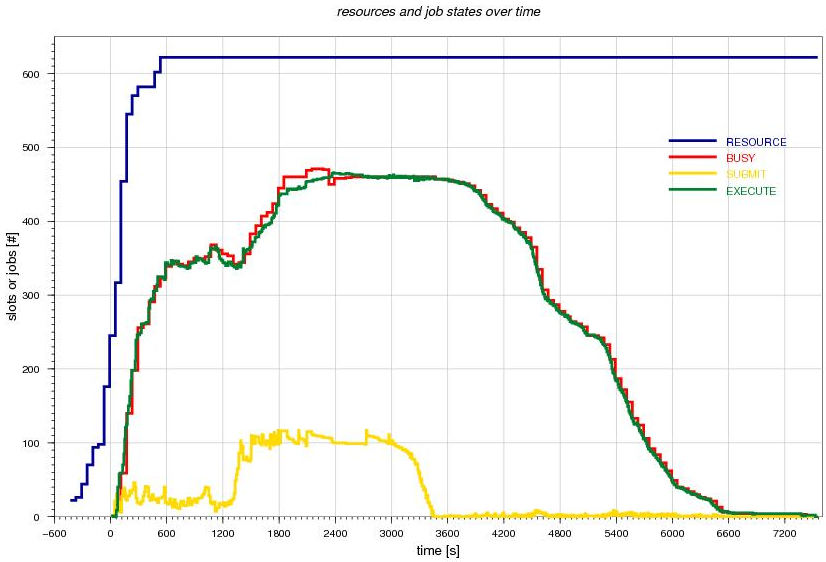


Figure 3: Slots and Job State over time.

The blue line tops out at 622 resource slots. However, since this is a total derived from *condor_status*, the submit host slots (6) and any other non-participating slots (20) need to be subtracted, bringing the total to 596 slots, or 298 participating cores, or 149 remote hosts. It also shows that a single remote resource never reported back properly.

For this workflow, partial success for a resource request is not a problem. However, other workflows do rely on the all-or-nothing principle, and the middleware should never provision a partial success, unless expressly permitted.

The red line in Figure 3 shows the slots that Condor perceived to be busy. This curve is over-shadowed by the tasks in state *executing* found in the Condor queue. At one point during the workflow, the number of executing tasks topped out at 466 parallel executing tasks.

The yellow line shows the number of *idle* tasks in the Condor queue. The workflow manager DAGMan was instructed to only release more jobs into the queue, if there were less than 100 *idle* jobs. It does not make sense to drop hundreds of jobs into the queue, if only a limited number of them can run. While a maximum of 117 *idle* jobs does not hit the target perfectly, it is quite sufficient to balance between saturation and scalability.

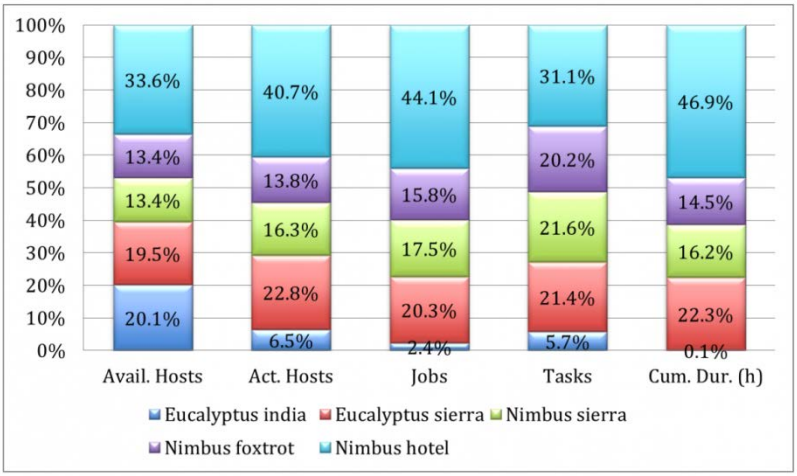


Figure 4: Display of Table 1.

Site	Avail. Hosts	Active Hosts	Jobs	Tasks	Cumulative Duration (h)
Eucalyptus *india*	30	8	19	1900	0.4
Eucalyptus *sierra*	29	28	162	7080	119.2
Nimbus *sierra*	20	20	140	7134	86.8
Nimbus *foxtrot*	20	17	126	6678	77.5
Nimbus *hotel*	50	50	352	10290	250.6
TOTAL	149	123	799	33082	534.5

Table 1: Statistics about Sites, Jobs and Tasks.

Table 1 and Figure 4 summarize the hosts that, according to Condor, were actually participating in the workflow. With only 123 actively participating hosts that received work from the Condor scheduler, the maximum number of job slots is 492, over 100 slots less than we requested.

Even though the Eucalyptus resources on *sierra* were only participating with 8 hosts, they managed to deal with 1,900 tasks. The amount of tasks computed per site reflects the number of resources closely, albeit not the time taken.

Overall, the workflow contained over 22 days of computational work, including staging of data. The workflow executed in a little more than 2 hours total workflow duration.

Even though every periodogram was computed twice, input files were staged from separate locations, with 33,082 compressed files totaling 3.4 GB over Condor-I/O. The output totals 66,164 transfers of compressed files with over 5.8 GB size in transferred volume.

Size range		Input .tbl.gz	Output .top.gz	Output .out.gz
1,024	2,047		606	
2,048	4,095		32,476	
4,096	8,191			
8,192	16,383	8,457		
16,384	32,767	1,065		
32,768	65,535	1,297		21,638
65,536	131,071	5,665		
131,072	262,143	52		
262,144	524,287	1		11,434
524,288	1,048,575	4		10
1,048,576	2,097,152			

Table 2: Ranges of compressed input and output sizes.

[1] Binless phase-dispersion minimization algorithm that identifies periods with coherent phased light curves (i.e., least “dispersed”) regardless of signal shape: Plavchan, Jura, Kirkpatrick, Cutri, and Gallagher. ApJS, 175,19 (2008)

[2] We will fix this in future re-runs.

Results for Project "Big Data for Science Virtual Summer School July 26-30 2010"

Judy Qiu
Indiana University, School of Informatics and Computing

The workshop was successfully delivered with good reviews.
In this “Big Data for Science” workshop, over 200 students across 10 institutions (Arkansas High Performance Computing Center, University of Arkansas, Fayetteville; Electronic Visualization Laboratory, University of Illinois at Chicago; Indiana University, Bloomington; Institute for Digital Research and Education, University of California, Los Angeles; Michigan State University, East Lansing; Pennsylvania State University, University Park; University of Iowa, Iowa City; University of Minnesota Supercomputing Institute, Minneapolis; University of Notre Dame, Notre Dame, Indiana; and University of Texas at El Paso). Additionally 100 additional students attended via streaming video. Students in the workshop used FutureGrid in hands-on activities that covered, among others, Hadoop/MapReduce, Twister, Grid Appliance, and GroupVPN. See http://salsahpc.indiana.edu/tutorial/index.html

Figure: Virtual School with 10 student sites (white) and 5 Presenter only sites (pink)

Results for Project "SAGA"

Shantenu Jha

Louisiana State University

Summary: The design and development of distributed scientific applications presents a challenging research agenda at the intersection of cyberinfrastructure and computational science. It is no exaggeration that the US Academic community has lagged in its ability to design and implement novel distributed scientific applications, tools and run-time systems that are broadly-used, extensible, interoperable and simple to use/adapt/deploy. The reasons are many and resistant to oversimplification. But one critical reason has been the absence of infrastructure where abstractions, run-time systems and applications can be developed, tested and hardened at the scales and with a degree of distribution (and the concomitant heterogeneity, dynamism and faults) required to facilitate the transition from "toy solutions" to "production grade", i.e., the intermediate infrastructure.

For the SAGA project that is concerned with all of the above elements, FutureGrid has proven to be that *panacea*, the hitherto missing element preventing progress towards scalable distributed applications. In a nutshell, FG has provided a persistent, production-grade experimental infrastructure with the ability to perform controlled experiments, without violating production policies and disrupting production infrastructure priorities. These attributes coupled with excellent technical support -- the bedrock upon which all these capabilities depend, have resulted in the following specific advances in the short period of under a year:

1. Use of FG for Standards based development and interoperability tests:

Interoperability, whether service-level or application-level, is an important requirement of distributed infrastructure. The lack of interoperability (and its corollary -- applications being tied to specific infrastructure), is arguably one of the single most important barriers in the progress and development of novel distributed applications and programming models. However as much as interoperability is important, it is difficult to implement and provide. The reasons are varied, but some critical elements have been the ability to provide (i) Persistent testing infrastructure that can support a spectrum of middleware -- standards-based or otherwise (ii) Single/consistent security context for such tests.

We have used FutureGrid to alleviate both of these shortcomings. Specifically, we have used FG as the test-bed for standards-compliant middleware for extensive OGF standards based testing as part of the Grid Interoperability Now (GIN) and Production Grid Infrastructure (PGI) research group efforts. As part of these extended efforts, we have developed persistent and pervasive experiments, which includes ~10 different middleware and infrastructure types -- most of which are supported FG, including Genesis, Unicore, BES and AWS (i.e. Eucalyptus) and soon OCCI. The fact that the FG endpoints are permanent has allowed us to keep those experiments "alive", and enable us to extend static interoperability requirements to dynamic interoperability requirements. Being relieved of the need to maintain those endpoints has been a critical asset.

See the following URL for visual map on the status of the experiments:
http://cyder.cct.lsu.edu/saga-interop/mandelbrot/demo/today/last/

2. Use of FG for Analysing & Comparing Programming Models and Run-time tools for Computation and Data-Intensive Science

What existing distributed programming models will be applicable on Clouds? What new programming models and run-time abstractions will be required to enable the next-generation of data-intensive applications? We have used FG in our preliminary attempts to answer some of these questions.

In Ref [http://www.cct.lsu.edu/~sjha/select_publications/2010_fgcs_mapreduce.pdf] published in Future Generation Computing Systems, we compare implementations of the word-count application to not only use multiple, heterogeneous infrastructure (Sector versus DFS), but also to use different programming models (Sphere versus MapReduce).

There is a fundamental need to support dynamic execution of tasks and data in extreme-scale systems. The design, development and experimentation of the abstractions to support this requirement isthus critical; FG has been used for this. In Ref [http://cct.lsu.edu/~sjha/select_publications/bigjob-cloudcom10.pdf
And http://www.cct.lsu.edu/~sjha/select_publications/interop_pilot_job.pdf] we (i)

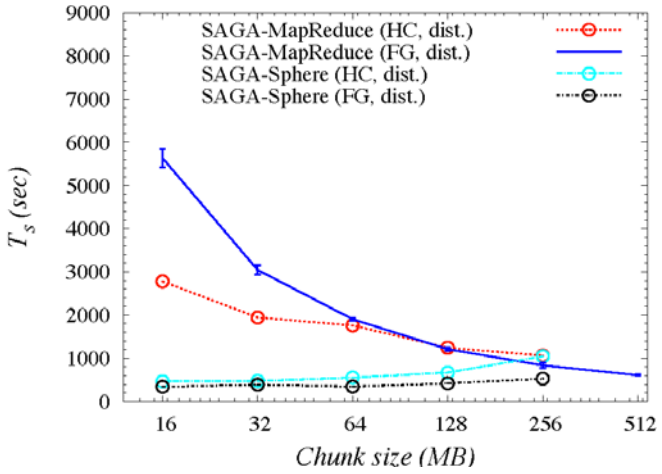
extended the Pilot-Job abstraction for Cloud environments, (ii) understand the basic roles of "system-level" abstractions. There is ongoing but mature work in developing run-time abstractions for data-intensive applications that can be used across the distributed infrastructure -- virtualized or otherwise. Although under development, these efforts rely on FG as a critical component for their testing, performance characterisation & deployment at scale and degrees of distribution that are not possible otherwise.

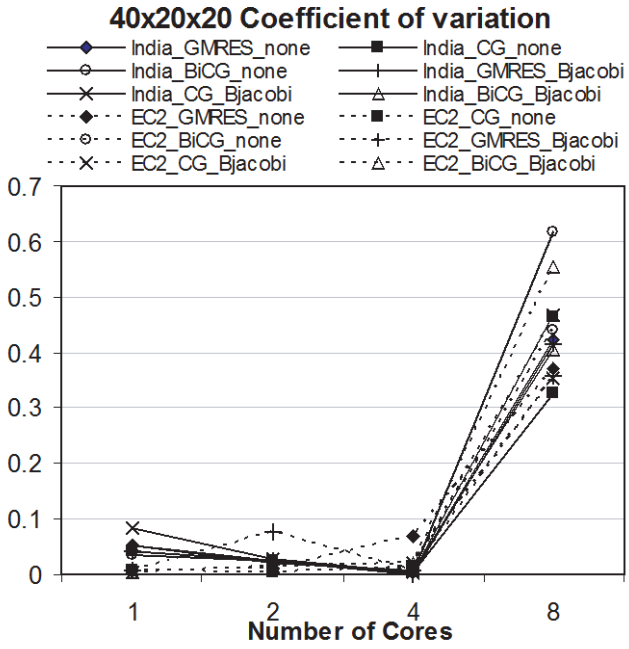
3. Use of FG for Developing Hybrid Cloud-Grid Scientific Applications and Tools (Autonomic Schedulers) [Work in Conjunction with Manish Parashar's group]

Policy-based (objective driven) Autonomic Scheduler provide a system-level approach to hybrid grid-cloud usage. FG has been used for the development and extension of such Autonomic Scheduling and application requirements. We have integrated the distributed and heterogeneous resources of FG as a pool of resources which can be allocated by the policy-based Autonomic Scheduler (Comet). The Autonomic Scheduler dynamically determines and allocates instances to meet specific objectives, such as lowest time-to-completion, lowest cost etc. We also used FG supplement objective driven pilot jobs on TeraGrid (ranger).

Additionally, during our investigations, we encountered inexplicable variations in our results. These has led to another strand of work that attempts to explore and characterize run-time fluctuations for a given application kernel representative representative of both a large number of MPI/parallel workloads and workflows. Fluctuation appears to be independent of the system load and a consequence of the complex interaction of the MPI library specifics and virtualization layer, as well as operating environment. Thus we have been investigating fluctuations in application performance, due to the cloud operational environment. An explicit aim is to correlate these fluctuation to details of the infrastructure. (See Fig: 40x20x20_coefVariation.pdf). As it is difficult to discern or reverse engineer the specific infrastructure details on EC2 or other commercial infrastructure, FG has provided us a controlled and well understood environment at infrastructure scales that are not possible at the individual PI/resource level.

Initial results from this work can be found at:
More info: - http://cct.lsu.edu/~sjha/select_publications/hybrid-autonomics-sciencecloud.pdf





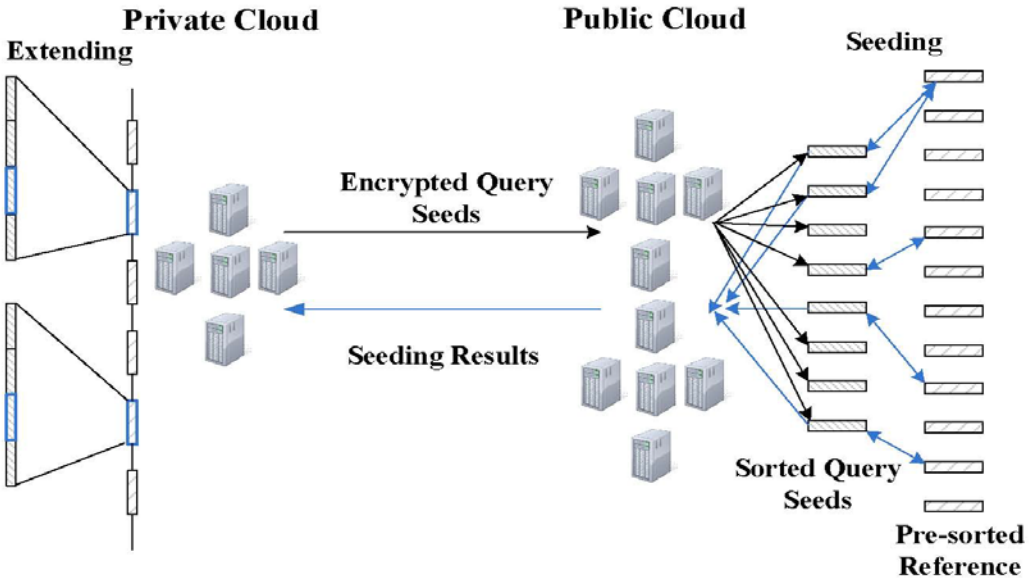
Results for Project "Privacy preserving gene read mapping and hybrid cloud"

Yangyi Chen

Indiana University, School of Informatics and Computing

One of the most important analyses on human DNA sequences is read mapping, which aligns a large number of short DNA sequences (called reads) produced by sequencers to a reference human genome. The analysis involves intensive computation (calculating edit distances over millions upon billions of sequences) and therefore needs to be outsourced to low-cost commercial clouds. This asks for scalable privacy-preserving techniques to protect the sensitive information sequencing reads contain. Such a demand cannot be met

by the existing techniques, which are either too heavyweight to sustain data-intensive computations or vulnerable to re-identification attacks. Our research, however, shows that simple solutions can be found by leveraging the special features of the mapping task, which only cares about small edit distances, and those of the cloud platform, which is designed to perform a large amount of simple, parallelizable computation. We implemented and evaluated such new techniques on a hybrid cloud platforms built on FutureGrid. In our experiments, we utilized specially-designed techniques based on the classic "seed-and-extend" method to achieve secure and scalable read mapping. The high-level design of our techniques is illustrated in the following figure: the public cloud on FutureGrid is delegated the computation over encrypted read datasets, while the private cloud directly works on the data. Our idea is to let the private cloud undertake a small amount of the workload to reduce the complexity of the computation that needs to be performed on the encrypted data, while still having the public cloud shoulder the major portion of a mapping task.



We constructed our hybrid environment over FutureGrid in the following two modes:

1. Virtual mode:

   We used 20 nodes on FutureGrid as the public cloud and 1 node as the private cloud.

2. Real mode:

   We used nodes on FutureGrid as the public cloud and the computing system within the School of Informatics and Computing as the private cloud. In order to get access to the all the nodes on public cloud, we copied a public SSH key shared by all the private cloud nodes to the `authorized_keys` files on each public cloud node.

   Our experiments demonstrate that our techniques are both secure and scalable. We successfully mapped 10 million real human microbiome reads to the largest human chromosome over this hybrid cloud. The public cloud took about 15

minutes to do the seeding and the private cloud spent about 20 minutes on the extension. Over 96% of computation was securely outsourced to the public cloud.

Results for Project "Cloud Technologies for Bioinformatics Applications"

Thilina Gunarathne
Indiana University, Pervasive Technology Institute

This project is ongoing with the current intermediary results:

For the first step of our project, we performed an in-detail performance analysis of different implementations of two popular bio-informatics applications, namely sequence alignment using SmithWaterman-GOTOH algorithm and sequence assembly using CAP3 program. These applications were implemented using cloud technologies such as Hadoop MapReduce and Microsoft DryadLINQ as well as using MPI. The performance comparison consisted of comparing the performance scalability of the different implementations, analyzing the effects of inhomogeneous data on the performance of cloud technology implementations and comparing the performance of cloud technology implementations under virtual and non-virtual (bare metal) environments. We also performed an auxiliary experiment to calculate the systematic error of these applications in different environments.
We used Apache Hadoop on 33 bare metal Linux Futuregrid nodes as well as on 33 future grid Linux virtual instances (deployed using Eucalyptus). We also used Microsoft DryadLINQ on 33 bare metal Windows HPCS cluster on Futuregrid.The results are published in the following paper.

- J. Ekanayake, T. Gunarathne, J. Qiu, and G. Fox. "Cloud Technologies for Bioinformatics Applications", Accepted for publication in Journal of IEEE Transactions on Parallel and Distributed Systems, 2010

The following graphs present few selected results from our project. For more information refer to the above paper.

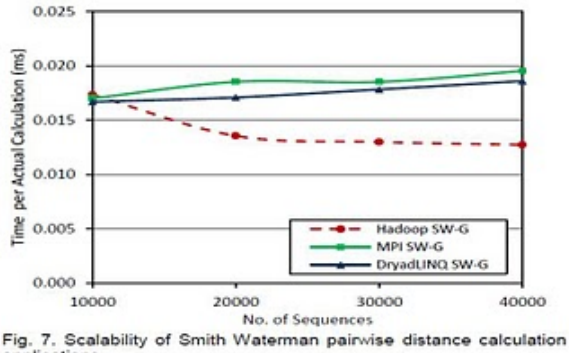


Fig. 7. Scalability of Smith Waterman pairwise distance calculation applications.

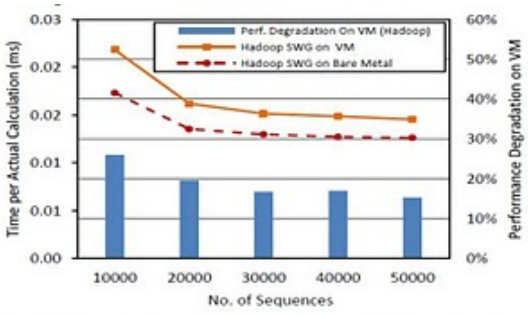


Fig. 13. Virtualization overhead of Hadoop SW-G on Xen virtual machines.

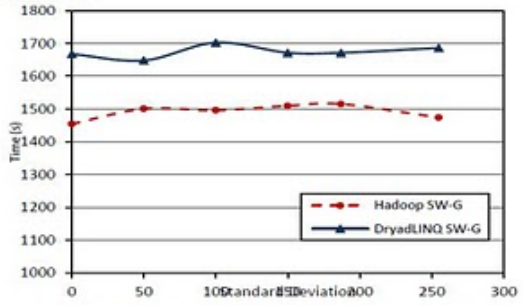


Fig. 9. Performance of SW-G pairwise distance calculation application for randomly distributed inhomogeneous data with '400' mean sequence length.

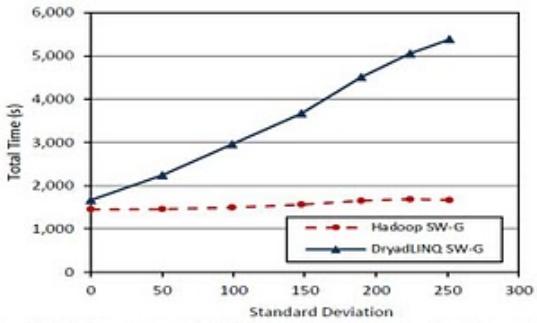


Fig. 10. Performances of SW-G pairwise distance calculation application for skewed distributed inhomogeneous data with '400' mean sequence length.

For the second step of our project, we implemented few pleasingly parallel bio-medical applications using cloud technologies, Apache Hadoop MapReduce and Microsoft DryadLINQ, and using cloud infrastructure services provided by commercial cloud service providers, naming it the "Classic Cloud" model. The applications used were sequence assembly using Cap3, sequence alignment using BLAST, Generative Topographic Mapping (GTM) interpolation and Multi Dimensional Scaling (MDS) interpolation. We used Amazon EC2 and Microsoft Windows Azure platforms for obtaining the "Classic Cloud" implementation performance results, while we used FutureGrid compute resources to obtain the Apache Hadoop and Microsoft DryadLINQ performance results. The results were published in the following papers.

- Thilina Gunarathne, Tak-Lon Wu, Judy Qiu, and Geoffrey Fox, Cloud Computing Paradigms for Pleasingly Parallel Biomedical Applications March 21 2010. Proceedings of Emerging Computational Methods for the Life Sciences Workshop of ACM HPDC2010 conference, Chicago, Illinois, June 20-25, 2010.

- Thilina Gunarathne, Tak-Lon Wu, Jong Youl Choi, Seung-Hee Bae, Judy Qiu Cloud Computing Paradigms for Pleasingly Parallel Biomedical Applications, Submitted for publication in ECMLS special edition of Concurrency and Computations Journal (invited).

APPENDIX C – Software Achievements and Milestones

This section includes a number of selected milestones for PY1 and PY2 in order to outline our development time plan for the Software section. A more detailed list of milestones for the overall project is provided as part of the PEP plan.

Q1: Oct 01 to Dec 31
Q2: Jan 01 to Mar 31
Q3: Apr 01 to Jun 30
Q4: Jul 01 to Sep 30

Software Achievements in PY1

Year/Quarter	Description
	HPC
PY1	Provided an elementary HPC queuing system on all compute resources.
PY1	Provided statically provisioned experiments through system administrative staff.
PY1	Demonstrated the concept of static provisioning using dual boot through XCAT.
	FG Fabric
PY1	Deployment of elementary queuing systems on all computational resources.
PY1	Deployment of network infrastructure by GRNOC.
PY1	Deployment of the network impairment (NIP) device and its software.
PY1	Deployment of development resources on hardware hosted outside of the control of FG as part of the Indiana campus infrastructure.
	Image Repository
PY1	Evaluated related technologies and completed software architecture design.
PY1	Designed APIs and developed a prototype which supports image upload/registration, query, etc. Security is achieved through SSH/SCP..
	Image Management
PY1	Designed and Prototyped an Image Generation Service.
PY1	Prototyped BCFG2 configuration management system for use with bare metal and virtual machines.

	Experiment Management Services
PY1	Experiment management data structure is defined and the framework is planned, which will be easily integrated with the accounting service and portal access.
PY1	Demonstrated heterogeneous cloud executions using Pegasus (see success story).
PY1	Improved Pegasus guided by the FG environment: Removed historical constraints about shared filesystem between head-node and worker-node to facilitate head-less execution. Launch worker-node jobs through *seqexec* wrapper to facilitate easier debugging.
	Dynamic Provisioning
PY1	Provided concepts for RAIN and worked on workflow use cases to utilize multiple clouds with Nimbus and Pegasus.
PY1	Deployed Website, Wiki; deployed task management system to coordinate the software development tasks; deployed ticket system; developed simple SSH key copy mechanism across sites; implemented limited backup services.
PY1	Integrated the task management system with the wiki system to display queries and trees of tasks making monitoring of progress of WBS tasks easier across all software development personnel.
PY1	Developed command line tools to manage tasks.
PY1	Deployed an external open source code repository at Sourceforge.net, and deployed an internal one for management tools, configurations etc.
	FG Access Services
PY1	Deployed various Nimbus service on Sierra, Foxtrot, and Hotel. Deployed ViNe: Sierra, Foxtrot, and Hotel through Nimbus. Connected via ViNe resources on FG (Sierra, Foxtrot, Hotel) and Grid5000 to run bioinformatics applications. Integrated account application with FG account application. Deployed various Eucalyptus installations: Sierra, India. Deployed on all machines HPC services including queues. Deployed Unicore 6 endpoint. Deployed Genesis II endpoint. Collaborated with the SAGA team to develop a strategy to distribute SAGA as part of the HPC image. Collaborated with the ScaleMP group to provide Scale MP solutions in FG.
PY1	Improved Nimbus based on FG requests (e.g. security, dynamic Fabric control).
	Web Site and Portal
PY1	A basic web site was deployed for users and developers.
PY1	An FG category has been added to KB.
	Security, Accounting, and Auditing
PY1	Replication of a non-scalable SSH copy based authentication mechanism from TG.

	Performance Tools
PY1	PAPI installed as part of default Cray environment on Xray.
PY1	Initial architecture document completed (performance architecture).
PY1	Vampir workshop at IU.
PY1	Script written to automate installation of Vampir, Marmot, and Scalasca.
PY1	Vampir deployed to India and Xray. Vampir documentation written.
	Monitoring and Information Services
PY1	Initial architecture document completed (performance architecture).
PY1	Deployed Inca server and Inca clients to Xray, India, and Sierra -- provides basic monitoring of available software and services.
PY1	Automated HPCC deployed to India and Xray. Inca deployed to Foxtrot.
PY1	Inca deployed to Hotel. Netlogger server installed. Collected and displayed machine partitioning information.

Software Milestones in PY2

Q	**Description**
	Dynamic Provisioning
PY2 Q1	Setup India and Sierra to dynamically provision bare metal nodes with stateful and stateless operating systems.
PY2 Q1~Q2	Providing dynamic provisioning through the queuing system on each of the resources.
PY2 Q2~Q3	Providing dynamic provisioning through the queuing system across distributed resources.
PY2 Q4	Investigate the use of dynamically provisioned Windows HPC services including Dryad.
	Image Management
PY2 Q1	Deliver and test an alpha release of the image generation tools.
PY2 Q2	Deliver an image repository on each of the resources. Synchronize the images in the distributed image repository based on user demand. Integrate LDAP authentication into image management services.
PY2 Q3	Develop simple command line interfaces to the image management services.

	Provide an updated image generation service in beta release.
PY2 Q4	Develop simple portal interfaces to the image management services.
	Image Repository
PY2 Q2	Extend the Image Repository capabilities to use a cloud-like technology as backend in the data persistent layer.
PY2 Q3	Evaluate plans to provide Web Service interfaces, especially the mechanism to secure the service.
PY2 Q3~4	Provide command line tools to interact with the services. Collaborate with the portal team to provide functionality to access the FG image repository through the portal.
PY2 Q4	Study how to create plugins to use the Image Repository directly from IaaS frameworks like OpenNebula, Nimbus, Eucalyptus.
	Web Site and Portal
PY2 Q1~Q2	Integrate community building features into the portal, including forums, news, references, blogs, comments, and a rating system.
PY2 Q2	Integrate the portal with the FG account management system.
PY2 Q2	Integrate the ability of authenticated users to manage their projects through the portal.
PY2 Q3	Support the dissemination of information.
PY2 Q4	Explore collaboration with XD TAS to provide auditing views for FG deployed services.
PY2 Q2~3	Establish a workflow in the portal allowing review of contributed contents for the portal.
PY2 Q3	Encourage community participation in the development of contents and manual entries.
PY2 Q2	Include performance measurement indicators into the portal such as the ability to create polls, rating of content, commenting on content.
PY2 Q4	Establish the inclusion of a more sophisticated search engine such as Apache Solr.
	HPC
PY2 Q1	Created a LDAP directory for users with custom FG schema.
	FG Fabric
PY2 Q1	Replacement of all hardware that hosted our previous development services, moving the services under control of FG system administrative staff.
PY2 Q1~Q4	Providing backup software solutions for users and systems in conjunction with the FG system administrative staff.
PY2 Q2~Q4	Tutorials and use cases demonstrating the use of the NIP.
PY2 Q1~Q3	Providing better storage resource capabilities for the users of FG and integrate them with backup

	solutions.
PY2 Q3	Elementary exposure of the network information through the portal (Q3).
PY2 Q4	Investigate reservation through the queuing system (Q4).
PY2 Q1~4	Deploying other than XCAT and MOAB solutions at TACC.
	Experiment Management Services
PY2 Q1~Q2	Provide streamlined application process for projects.
PY2 Q2~Q3	Integrate experiment management with account management.
PY2 Q3	Provide information browsing capabilities for user experiments. Provide the ability to share experiments through the portal.
PY2 Q4	Provide the ability to reproduce an experiment. Develop simple command line interfaces to the experiment.
PY2 Q2~Q4	Develop Pegasus-based workflows for resource provisioning and de-provisioning. Develop initial capabilities to include times steps in FG workflows. Provide deployment templates for image generation, provide templates for configuration management, use the FG repository for the distribution of Pegasus workflows.
	Dynamic Provisioning (Duplicate with the previous one. Need to be merged)
PY2 Q2	Provide a command line tool RAIN. Enhance the ability to *rain* images on FG hardware.
PY2 Q3	Enhance the ability to *rain* services such as Hadoop on FG hardware. Develop prototypes to *rain* across different resources. Explore to *rain* file systems. Explore *to rain* MS services.
PY2 Q4	Develop simple command line interfaces to the RAIN services. Develop simple portal interfaces to the RAIN services.
	FG Operational Services
PY2 Q1~Q2	To facilitate project agility, all FG infrastructure services will be transferred to dedicated hardware locate on the FG network.
PY2 Q2	Re-evaluation of the ticket system used for user tickets.
PY2 Q1~Q4	Re-implementation of the portal system with tight integration into a SSO solution developed by the FG core team.
PY2 Q2	Re-deployment of the wiki service with significant performance improvements.
PY2 Q2~3	Implementation of a documented backup strategy.
PY2 Q2~3	Implementation of a QA strategy by FG systems staff for all services. Establish best practices for QA of the code including code reviews, and automated testing where needed.

PY2 Q3~4	Collaboration with the Project Manager on using of the task management system for all FG related tasks to manage the PEP plan.
PY2 Q2	Establish the use of the svn with partners outside of IU.
	FG Access Services
PY2 Q1	Collaborated with the OpenNebula team to work towards the installation of an OpenNebula service in FG.
PY2 Q2	Integration of the Eucalyptus service into a single Eucalyptus install with zones. Continue the collaboration with the community and work towards deploying services for OpenNebula, Sector/Sphere, ScaleMP, OpenStack, and others based on user needs. Work on an Hadoop service that allows setting up modified versions of Hadoop. Develop a plan for the delivery of a Windows HPC service.
PY2 Q3	Explore the integration of the Eucalyptus account management with the FG account management. Evaluate the user services needed by the community and determine priorities. Study how to integrate OpenNebula authentication with FG LDAP authentication server (FG account management).
PY2 Q4	Identify mechanisms for reassigning resources to various services on user demand. Study how to integrate the OpenNebula Image Repository with the FG Image Repository.
PY2 Q2~Q4	Improve Genesis II for FG.
PY2 Q2~Q4	Improve Nimbus for FG in regards to security, storage management, image repository integration with the FG image repository, provide an ad-hoc deployable version of Nimbus via deployment and configuration management templates. Improve ViNe in regards to management so that FG users can easily manipulate ViNe operating parameters.
PY2 Q3~Q4	Provide OpenNebula users with a web portal. Study how to create Federate Clouds in OpenNebula.
PY2 Q1-Q2	Design of ViNe management APIs.
PY2 Q2-Q3	Implement the designed ViNe management features.
PY2 Q3-Q4	Deploy ViNe management-enhanced services on FG.
	Performance Tools
PY2 Q1	VampirTrace deployed to Hotel. PAPI documentation written. Vampir and PAPI tests deployed to Inca.
PY2 Q2-Q4	Provide images enhanced with PAPI, benchmark tools. Add step-by-step user tutorials for PAPI and Vampir. Integrate performance tools into image generation by providing deployment configuration templates and templates that can be used in other images.
	Monitoring and Information Services
PY2 Q1	Inca deployed to Alamo.

	Enhanced Inca Web status overview page. Inca tests added for Nimbus and Eucalyptus. Inca and Netlogger documentation written. Usage data collected from Nimbus and Eucalyptus.
PY2 Q2~Q4	Testing of image packages and monitoring of image generator and image repository. Add additional tests and collect performance data from Eucalyptus and Nimbus and other FG components as they are developed. Maintain user documentation. Assist set up monitoring via Nagios and Ganglia as needed.
	Security, Accounting and Auditing
PY2 Q2	Deployment of a scalable authentication mechanism based on replicated LDAP servers.
PY2 Q4	Identifying and deploying an auditing service.
PY2 Q3	Identifying and deploying an elementary accounting service.
PY2 Q2	Evaluation of InCommon and possible integration mechanisms.