FutureGrid Software

Acknowledgement:This presentation has been collaboratively developed with all of the members of the Software team

We asked the team members to put their names here, the order is random:

Gregor von Laszewski, Fugang Wang, Archit Kulshrestha, Andrew Younge, Gregory Pike, Javier Diaz, Warren Smith, Shava Smallen, Ewa Deelman, Jens Voeckler, Andrew Grimshaw, Terry Moore, Piotr Luszczek, Kate Keahey, David LaBissoniere, Renato Figueiredo, Mauricio Tsugawa, José Fortes, Robert Henschel, Geoffrey Fox

Overview of the FutureGrid Software

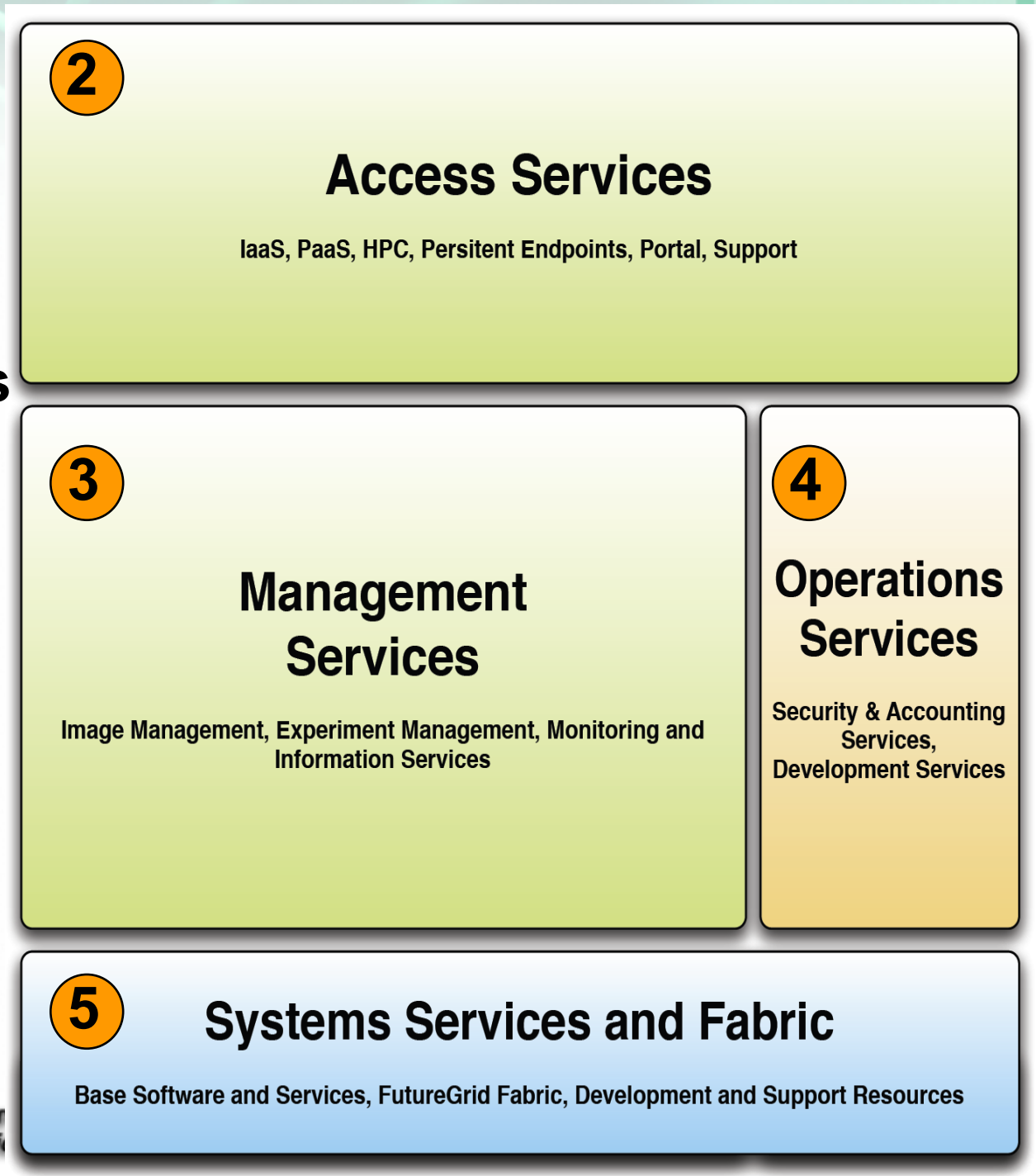
Presented by
Gregor von Laszewski
8 minutes

FutureGrid Software Architect
Community Grids Laboratory
Pervasive Technology Institute

Outline

1. Overview
2. Access Services
3. Management Services
4. Operations Services

5. We will not much go into:
- *Base Software and Services*
- *Fabric*
- *Software for Development & Support Resources*

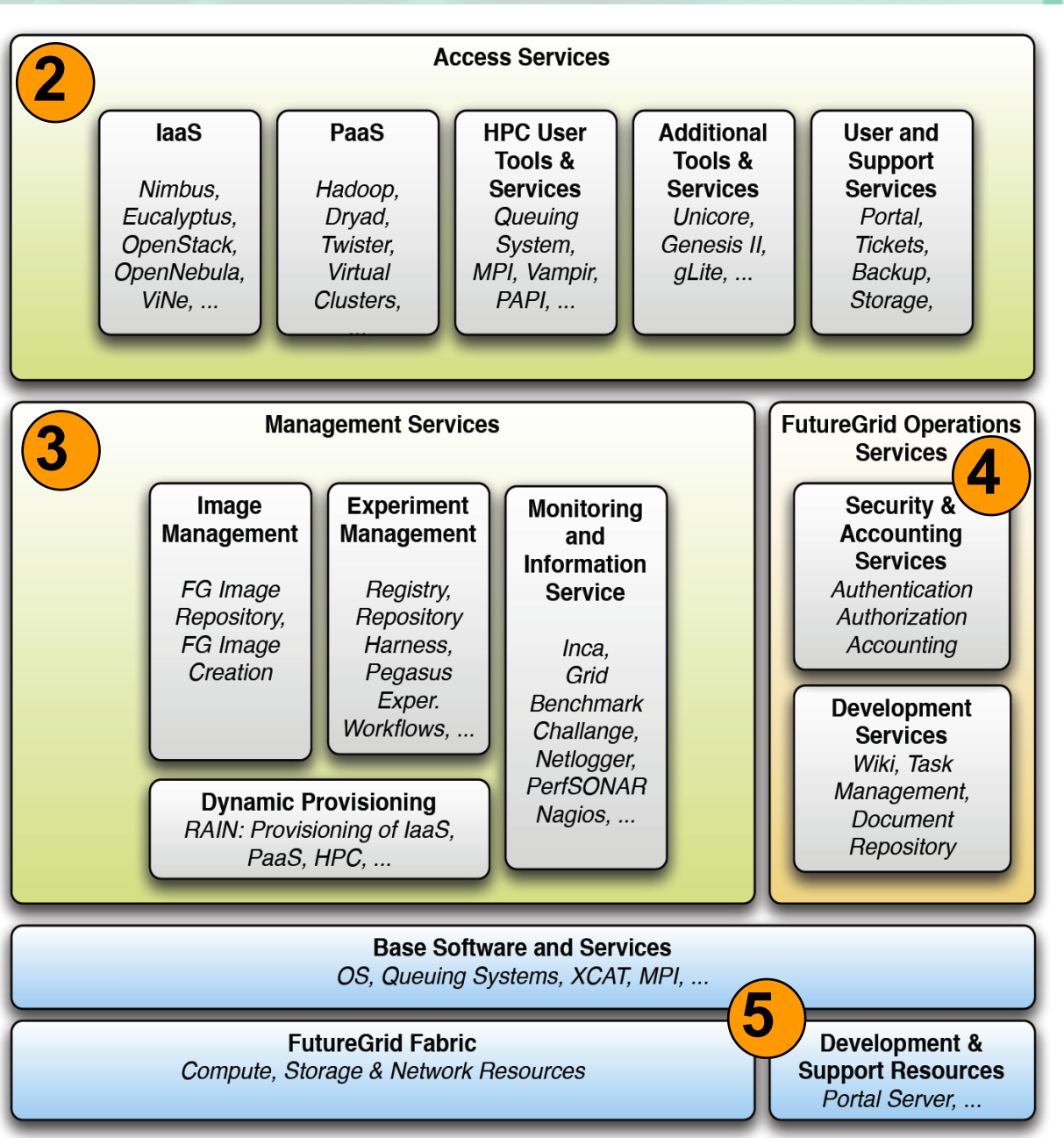


Outline

1. Overview
2. Access Services
3. Management Services
4. Operations Services

5. We will not much go into:

- *Base Software and Services*
- *Fabric*
- *Software for Development & Support Resources*



Software Engineering Approach

Approach

- Spiral Process
- Requires tight integration of software and systems management teams
- Task Management
- Integrated with WBS
- QA and QC
- Weekly calls

Team

- All partners participate
- Tasks assigned by expertise
- Collaborative development

Risks

- Collaboration is large
- Technology is new
- Systems are diverse
- Software/System best practices do not exist for FG
- Tradeoff between Services and Software

Future Grid

Goals of the Software

- **Support Diverse User Community**
  - *Application developers, Middleware developers, System administrators, Educators*, *Application users*
- **Support for Shifting Technology Base**
  - Infrastructure as a Service (IaaS), and Platform as a Service (PaaS) paradigms
  - In IaaS we see less important role of Eucalyptus
    - Nimbus: Our main IaaS framework. Rapidly evolving
      - Several releases a year, our funded partner!
    - OpenNebula: Important project in Europe
    - OpenStack: Expected to take large share of user base from Eucalyptus due to strong partners and open source philosophy
  - PaaS are rapidly evolving

Goals of the Software

Support of Diverse Access Models

- **Persistent Endpoints:** Unicore, gLite, Genesis II, Nimbus, Eucalyptus, OpenStack, OpenNebula, HPC
  - User just wants to use a preinstalled framework
  - User wants to compare HPC with framework x
- **Dynamically Provisioned Frameworks:** install cloned versions with modifications of the above + my own framework
  - Middleware developer provides next generation software
- **Community:** I want to showcase my service
  - Enable viral contribution model to services offerend in FutureGrid

Differentiation

FG vs. Amazon

- Multiple alternative IaaS frameworks
- Control of resource mapping
- Development of middleware, not just using it
- OS level work possible not just virtualized environment
- Windows and Linux
- Performance comparison

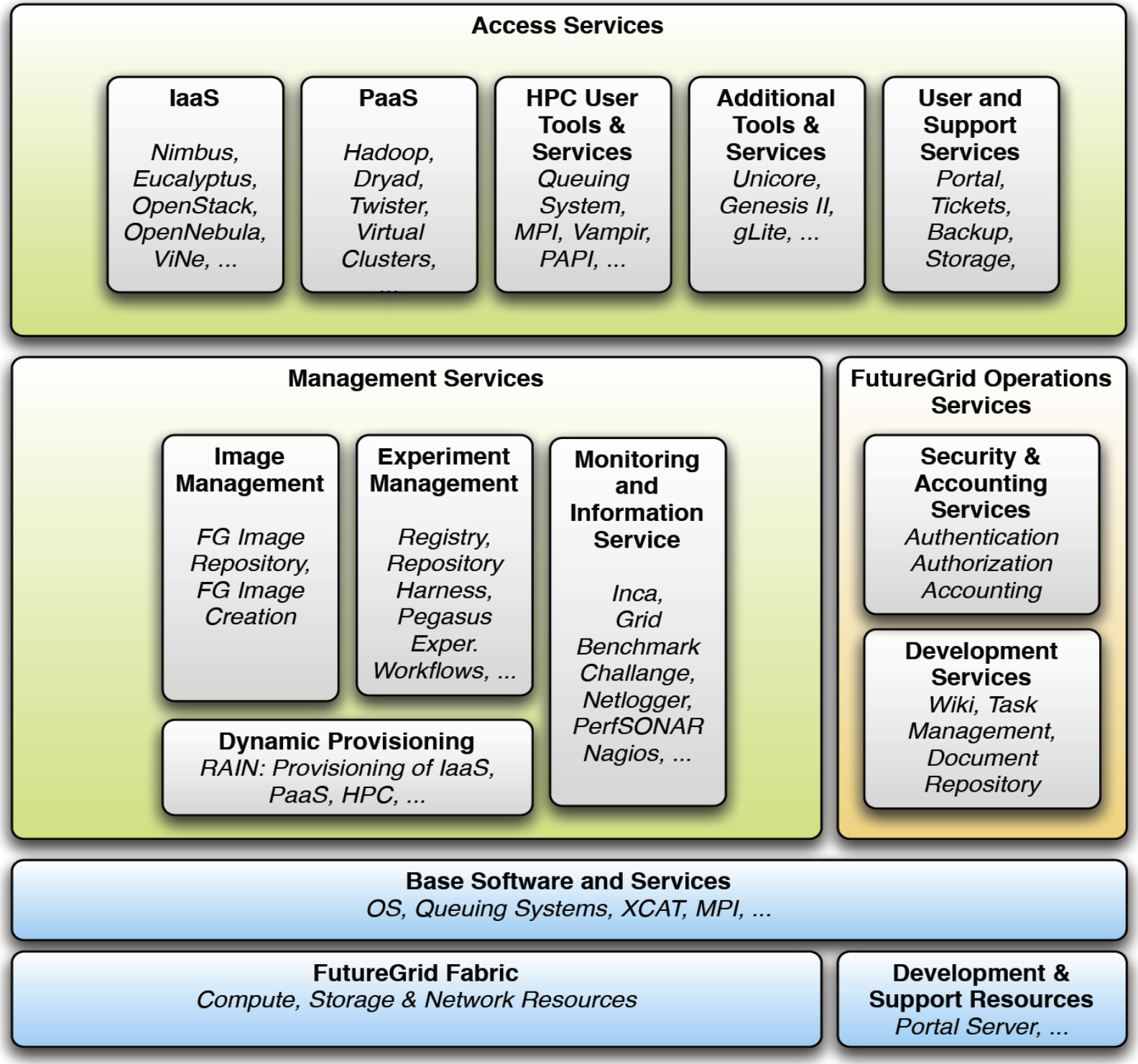
FG vs.TeraGrid » XD

- Environment is customizable
  - Dynamically provisioning software as needed onto “bare-metal”
  - exploit both the innovative technologies available and the interactive usage mode of FutureGrid
- Richer environment, not just traditional HPC
  - TG software + IaaS, PaaS & HPC
- Different spectrum of use
  - computer science systems, interoperability, clouds, education and bioinformatics

Goals of Software

- **Provide Management Capabilities for Reproducible Experiments**
  - Conveniently define, execute, and repeat application or grid and cloud middleware experiments within interacting software “stacks” that are under the control of the experimenter.
  - Leverage from previous experiments.
  - Terminology: Experiment Session & Apparatus

Architecture



Software Roadmap

PY1:

- Enable general services: HPC, Nimbus, Eucalyptus
- Explore dynamic provisioning via queuing system
- Explore raining an environment (Hadoop)

PY2:

- Provide dynamic provisioning via queuing system
- Deploy initial version of fg-rain, fg-hadoop, ...
- Explore replication of experiments
- Allow users to contribute images
- Deploy OpenNebula, OpenStack

PY3:

- Deploy replication of experiments
- Deploy replication of comparative studies

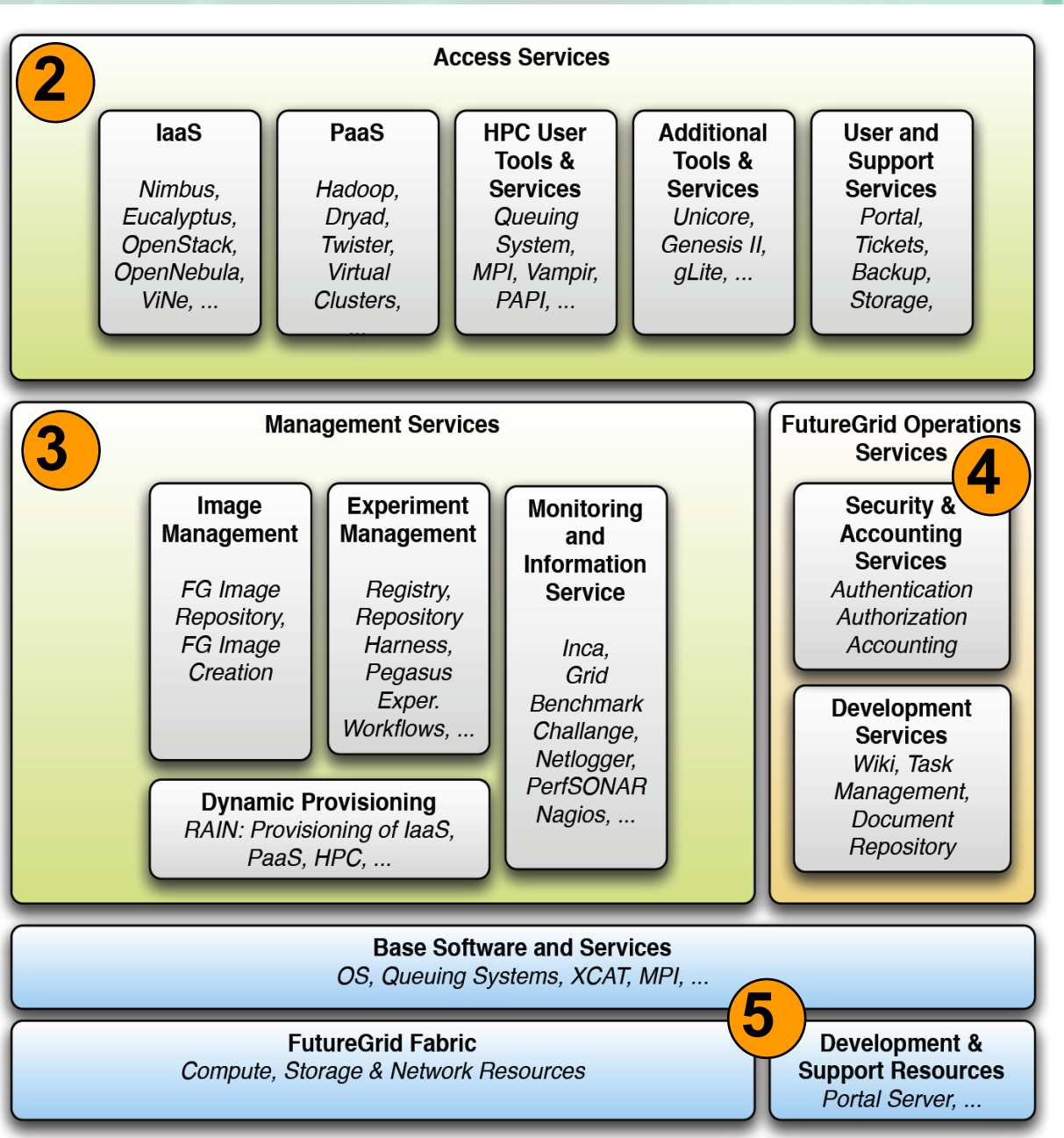
PY4:

- Harden software for distribution

Outline

1. Overview
2. Access Services
3. Management Services
4. Operations Services

5. *We will not much go into:*
   - *Base Software and Services*
   - *Fabric*
   - *Software for Development & Support Resources*



Grid Standards & Interoperability

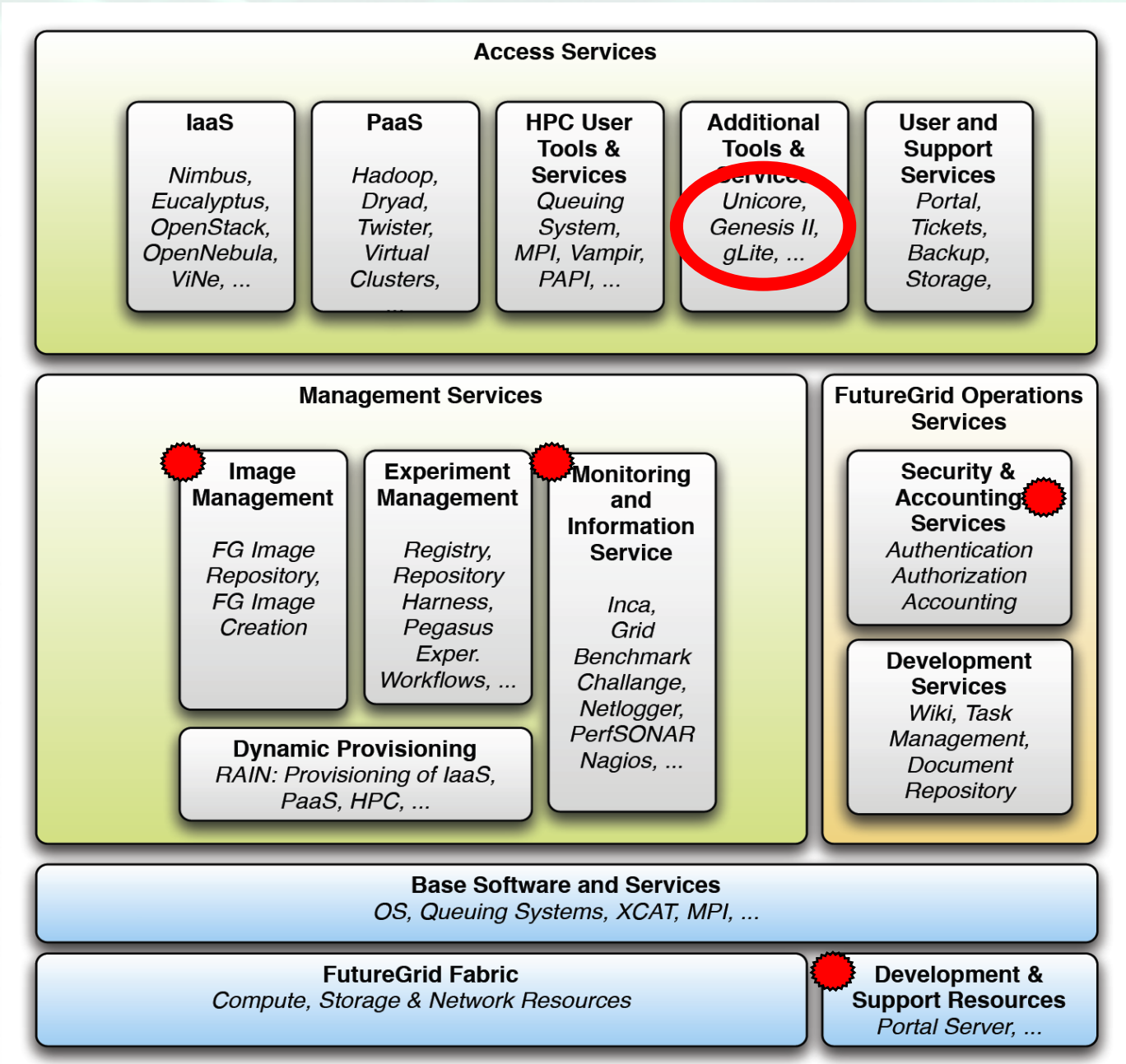
Andrew Grimshaw
University of Virginia

Presenter:

- Andrew Grimshaw
- University of Virginia
- 4 minutes

Key Points

- Interoperability
- Unicore 6
- gLite
- Genesis II



Requirements

- Provide a persistent set of standards-compliant implementations of grid services that clients can test against
- Provide a place where grid application developers can experiment with different standard grid middleware stacks without needing to become experts in installation and configuration
- Job management (OGSA-BES/JSDL, HPC-Basic Profile, HPC File Staging Extensions, JSDL Parameter Sweep, JSDL SPMD, PSDL Posix)
- Resource Name-space Service (RNS), Byte-IO
- Provide a place where Grid middleware developers can stress-test their systems without impacting production systems.

Usecases

- Interoperability tests/demonstrations between different middleware stacks
- Development of client application tools (e.g., SAGA) that require configured, operational backends
- Develop new grid applications and test the suitability of different implementations in terms of both functional and non-functional characteristics
- Many faults only occur under heavy load. Need a place to stress test, and fail without impacting production users

Implementation

- UNICORE 6
  - OGSA-BES, JSDL (Posix, SPMD)
  - HPC Basic Profile, HPC File Staging
- Genesis II
  - OGSA-BES, JSDL (Posix, SPMD, parameter sweep)
  - HPC Basic Profile, HPC File Staging
  - RNS, ByteIO
- *EGEE/g-Lite (in progress)*
- *SMOA (in progress)*
  - OGSA-BES, JSDL (Posix, SPMD)
  - HPC Basic Profile

Deployment

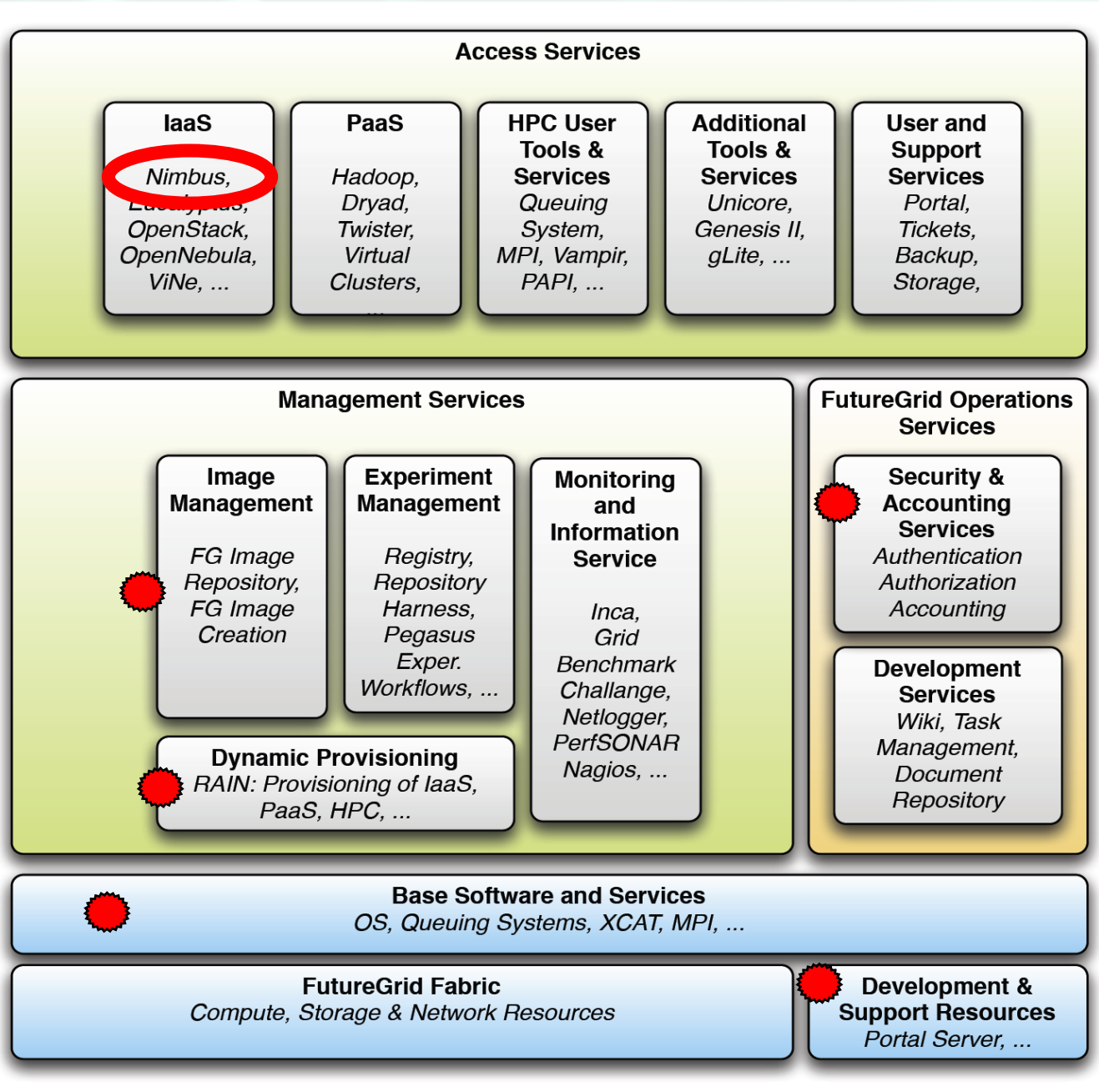
- UNICORE 6
  - Xray
  - Sierra
  - India
- Genesis II
  - Xray
  - Sierra
  - India
  - Eucalyptus (India, Sierra)

Nimbus

Kate Keahey
University of Chicago
12 minutes

Key Points

- IaaS
- We can directly impact development of Nimbus



Nimbus Overview

High-quality, extensible,
customizable,
open source implementation

Higher-level IaaS Tools

- Context Broker
- Nimbus Clients
- Elastic Scaling Tools

Enable users to use IaaS clouds

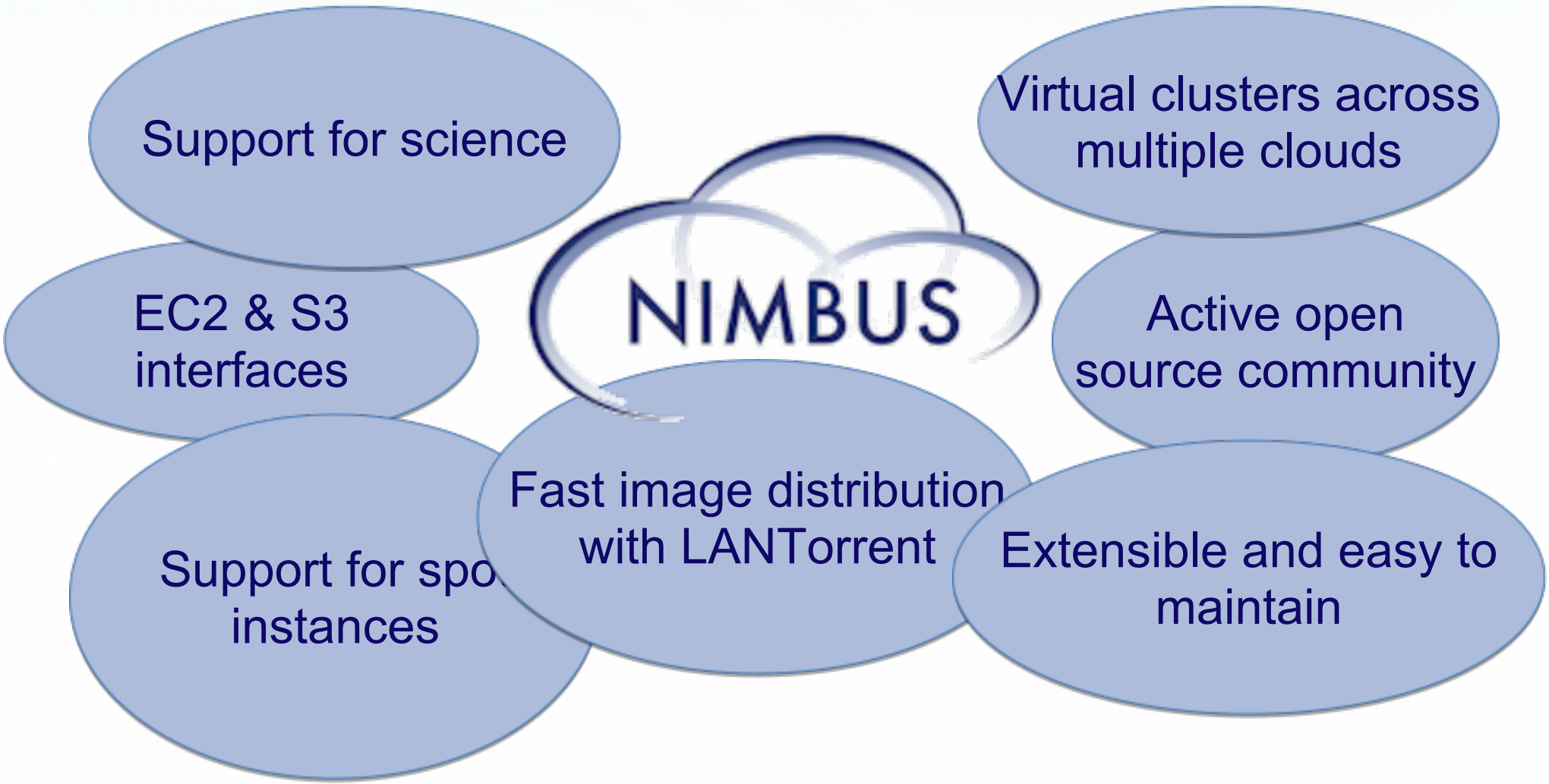
Infrastructure-as-a-Service Tools

- Workspace Service
- Cumulus

Enable providers to build IaaS clouds

Enable developers to extend, experiment and customize

Nimbus Key Features



Nimbus in FutureGrid

Requirements

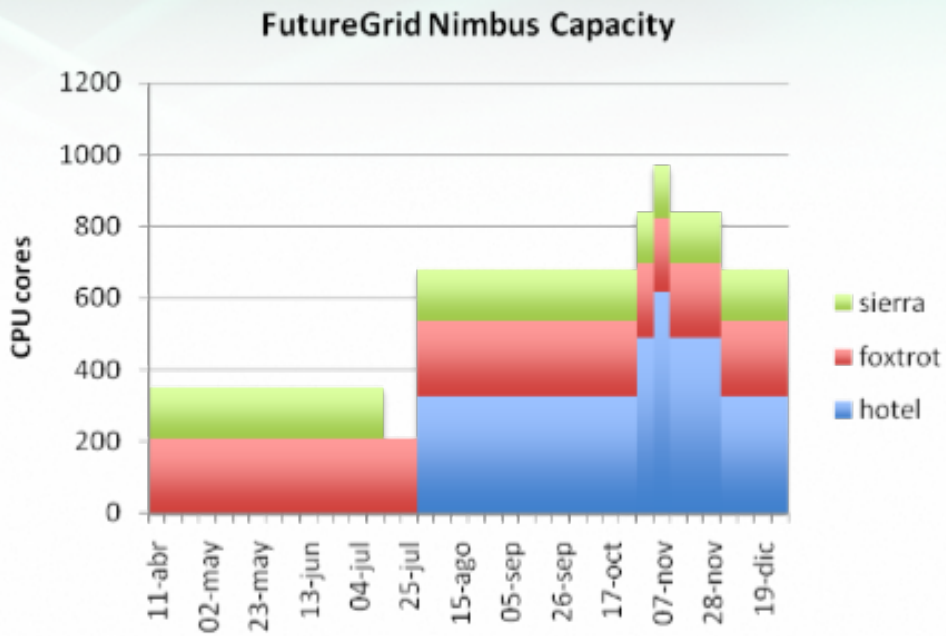
- IaaS Infrastructure:
  - IaaS infrastructure to experiment on top of: feature-rich and easy to use
  - IaaS infrastructure to experiment with: modular and extensible
- Higher-level services:
  - Virtual ensembles
  - Multi-cloud support
- Integration, user and exploration support

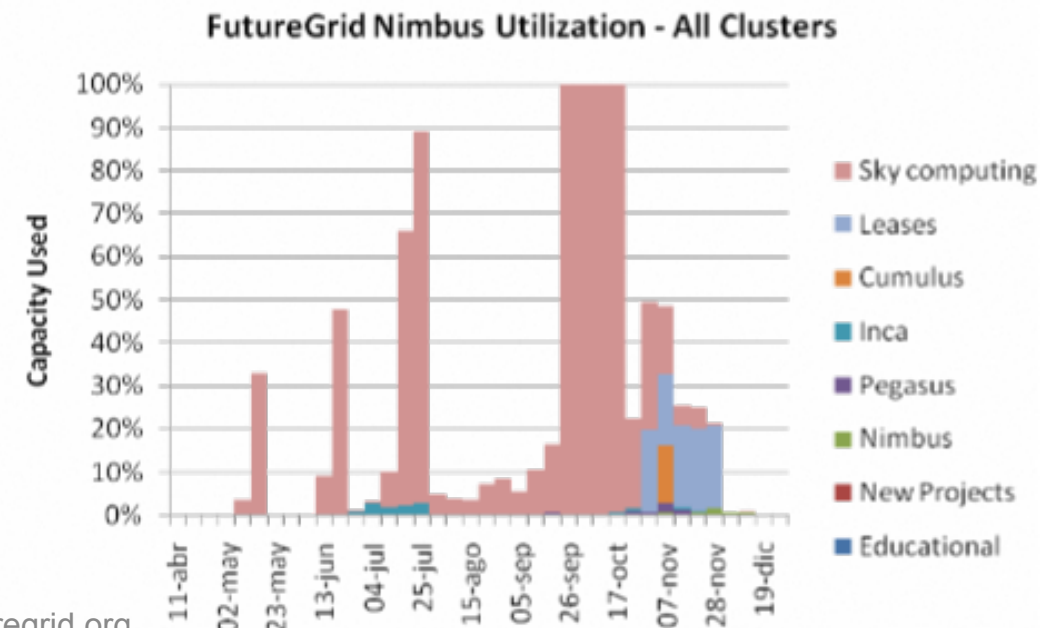
Use Cases

- Can a user:
  - Deploy a (group of) VMs or create a storage objects?
  - Modify or instrument IaaS to experiment with new capabilities?
  - Create a virtual cluster?
  - Create a multi-cloud experiment?
- Are those things easy to do and cost-effective?

Nimbus Deployment

- Resources:
  - Hotel (UC) 328 cores
  - Foxtrot (UFL) 208 cores
  - Sierra (SDSC) 144 cores
  - Alamo (TACC), in preparation
- Usage types so far:
  - Projects using IaaS
  - Projects modifying IaaS
  - Using higher-level tools
  - Educational (using IaaS)





Nimbus PY1 Milestones

- Nimbus deployed on FutureGrid sites
- Collected Nimbus requirements for improvements, integration and analysis, and to support new projects
- Nimbus releases containing FG-driven features:
  - Nimbus installer (Nimbus 2.4 in 05/10)
  - Zero -> cloud installation process and user management tools (Nimbus 2.5 in 07/10)
  - Partial: dynamic node management (Nimbus 2.6 in 11/10)
- Other major features:
  - Tools and scripts to integrate Nimbus credential distribution process into the FutureGrid credential distribution process
- Prepared documentation and tutorials for FG users
- Supported demonstrations, exploration, and early users

Nimbus PY2 Milestones

- Ongoing requirements gathering process
- Continue to respond to requirements
  - Current requirements: integration of Nimbus credential distribution into FG (completed), RM enhancements (fine-grained instances + dynamic provisioning), additional VM monitoring, FG image format integration, debugging features (get-console-output), multi-cloud support, make Nimbus a **better experimental tool** (specific extensibility enhancements), maintainability enhancements (admin "sanity check" scripts)
- Exploration and integration support
- Continue documentation and educational outreach work
- User and Project support

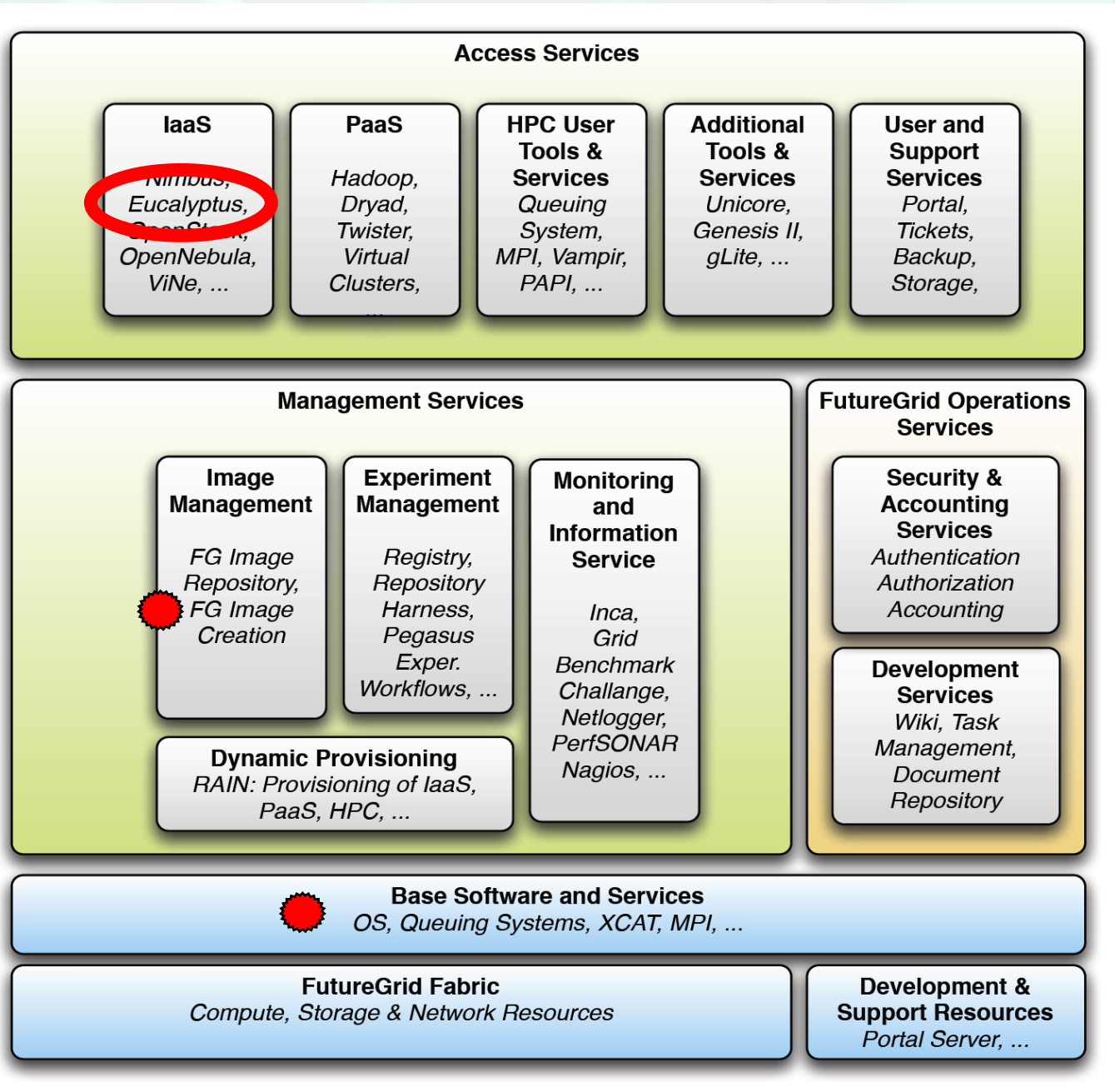
Risks

- Many changing requirements

Eucalyptus

Archit Kulshrestha
Indiana University
5 minutes
Key Points

- Elastic Utility Computing Architecture Linking Your Programs To Useful Systems
- EC2 interface for deploying user images on virtualized hardware.
- Deployment on Sierra and India
- Atomic allocation for VMs, storage and networking



FG Eucalyptus Testbed

- Eucalyptus is available to FutureGrid Users on the India and Sierra clusters.
  - Xen Based Virtualization
  - Users can make use of a maximum of 50 nodes on India and 21 on Sierra. Each node supports up to 8 VMs.
  - Different Availability zones provide VMs with different compute and memory capacities.

```
AVAILABILITYZONE        india   149.165.146.135
AVAILABILITYZONE        |- vm types     free / max   cpu   ram  disk
AVAILABILITYZONE        |- m1.small     0400 / 0400   1    512    5
AVAILABILITYZONE        |- c1.medium    0400 / 0400   1   1024    7
AVAILABILITYZONE        |- m1.large     0200 / 0200   2   6000   10
AVAILABILITYZONE        |- m1.xlarge    0100 /
0100   2   12000   10
AVAILABILITYZONE        |- c1.xlarge    0050 / 0050   8   20000   10
```

```
AVAILABILITYZONE        sierra  198.202.120.90
AVAILABILITYZONE        |- vm types     free / max   cpu   ram  disk
AVAILABILITYZONE        |- m1.small     0160 / 0160   1    512    5
AVAILABILITYZONE        |- c1.medium    0160 /
0160   1   1024    7
AVAILABILITYZONE        |- m1.large     0080 / 0080   2   6000   10
AVAILABILITYZONE        |- m1.xlarge    0040 /
0040   2   12000   10
AVAILABILITYZONE        |- c1.xlarge    0020 /
0020   8   30000   10
```

Management Interfaces and Clients

- FG provides Euca2ools to interact with Eucalyptus.
  - Available on India and Sierra via modules
- Account creation and credential management interfaces for requesting accounts and obtaining credentials.
  - https://eucalyptus.india.futuregrid.org:8443/
  - https://eucalyptus.sierra.futuregrid.org:8443/
- The Eucalyptus installations on India and sierra will be integrated into one umbrella with different availability zones.

Eucalyptus - Milestones and Risks

PY1:

- Eucalyptus on India and Sierra; Basic Image Library; Classroom use Grid Computing Class

PY2:

- Unified Install across India and Sierra; evaluation of iPlant Atmosphere (major NSF project ~$50 Mil.); integration with image and experiment management

PY3:

- Rich Web Interface for EC2 APIs using iPlant Atmosphere

Risks

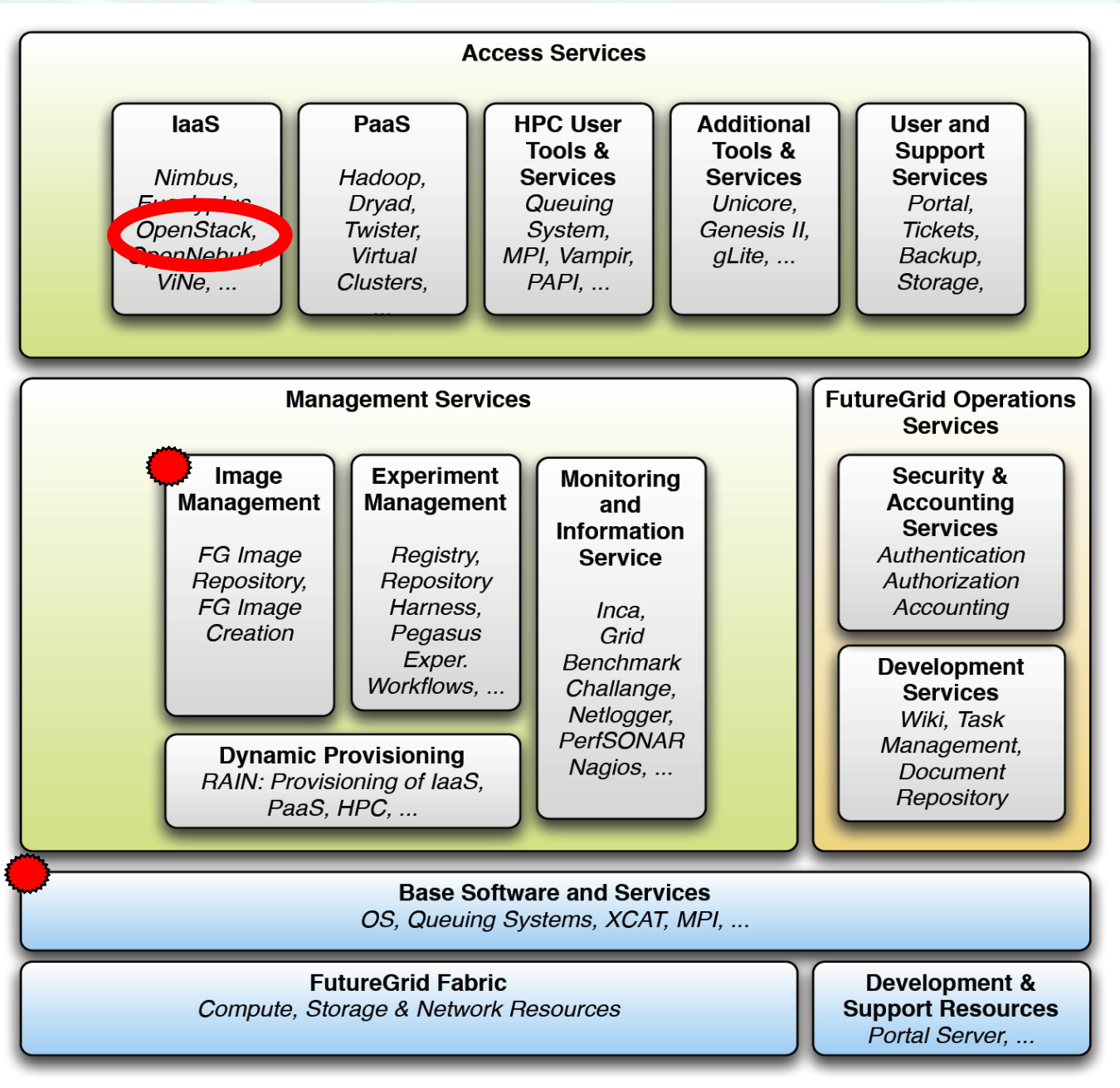
- Limited Public IP Address Pool
  - Potentially very large number of VMs possible
- Transient Errors
  - Image transfer errors, network disruption, etc.

OpenStack

Archit Kulshrestha
Indiana University

Key Points

- EC2 interface for VM management
- Alternative to Eucalyptus et al.
- Growing open source community - NSF + Rackspace
- Tutorial at CloudCom helped increase interest and understanding.



Milestones

- OpenStack evaluated for deployment on FutureGrid.
  - Test installation on the FG mini cluster for evaluation - PY2 Q1
  - Scaling tests - PY2 Q2
- A plan will be developed on how to provide both OpenStack and Eucalyptus as production services on FG in PY2 Q4
  - Dynamically add and remove nodes.

Risks

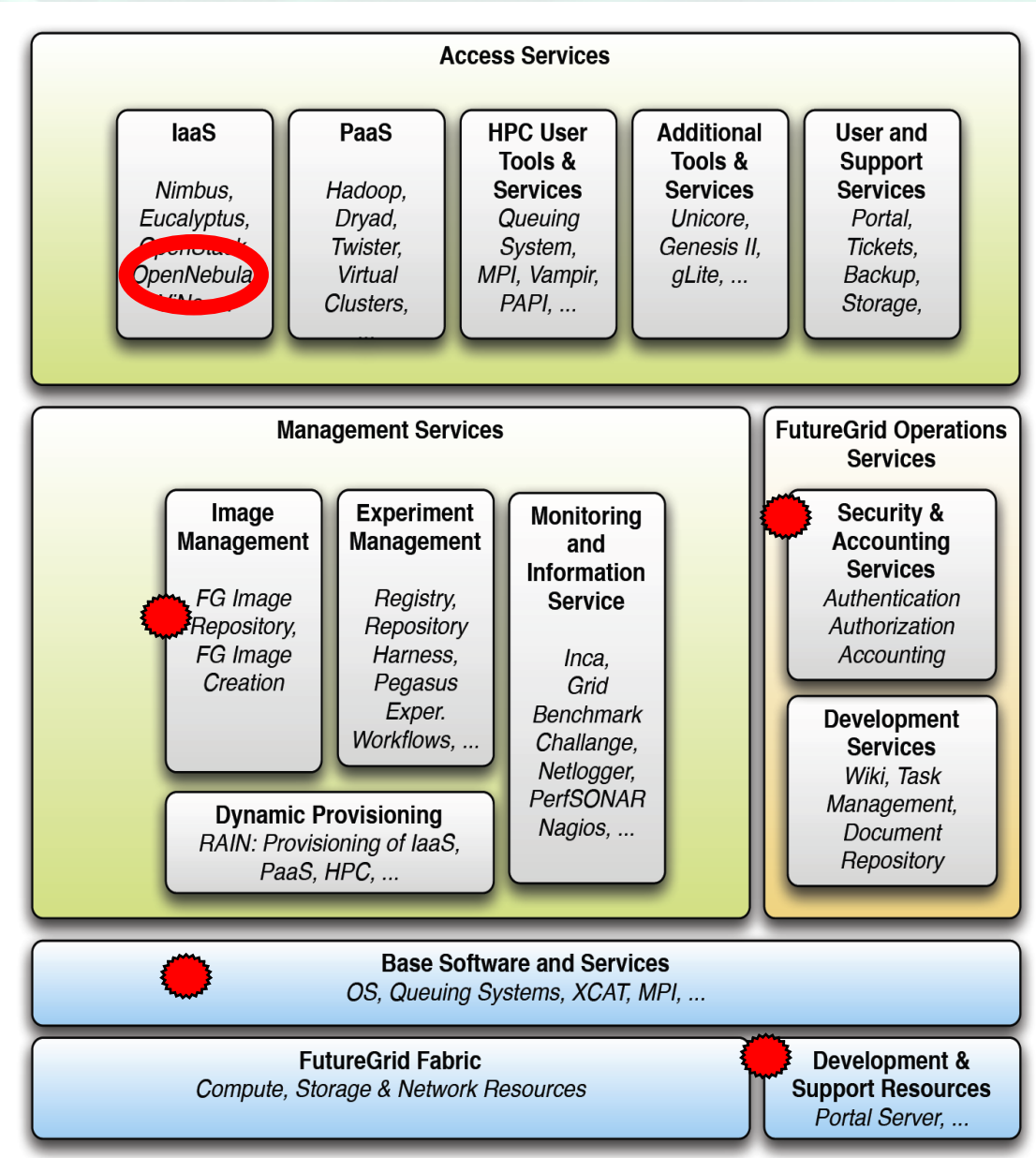
- Limited Public IP Address Pool
- No Web/GUI Management interface for users
  - Will not be needed with FG SSO - needs dev work

OpenNebula

Javier Diaz
Indiana University
Presenter: Archit Kulshrestha

Key Points

- Dominant European Effort
- Important for collaboration with European Initiatives like Reservoir or EGI
- Adaptability: Private, Public and Hybrid cloud
- Different authentication methods (password, ssh, LDAP)
- Performance and Scalability
- Customizable drivers for different components like Scheduling, Authentication, Storage or Hypervisor



Milestones

- Q1 2011
  - Deploy OpenNebula (Authenticate Users through ssh-keys)
  - Create User Manuals
- Q2 2011
  - Study how to integrate OpenNebula authentication with FutureGrid LDAP authentication server
- Q3 2011
  - Study how to integrate the OpenNebula Image Repository with the FutureGrid Image Repository
- Q4 2011
  - Provide users with a web portal
  - Study how to create Federated Clouds in OpenNebula

Risks

- Clear text passwords to access OpenNebula and the database
  - Appropriate file and system level permissions to avoid exposure to these passwords

ScaleMP

Andrew J. Younge,
Robert Henschel

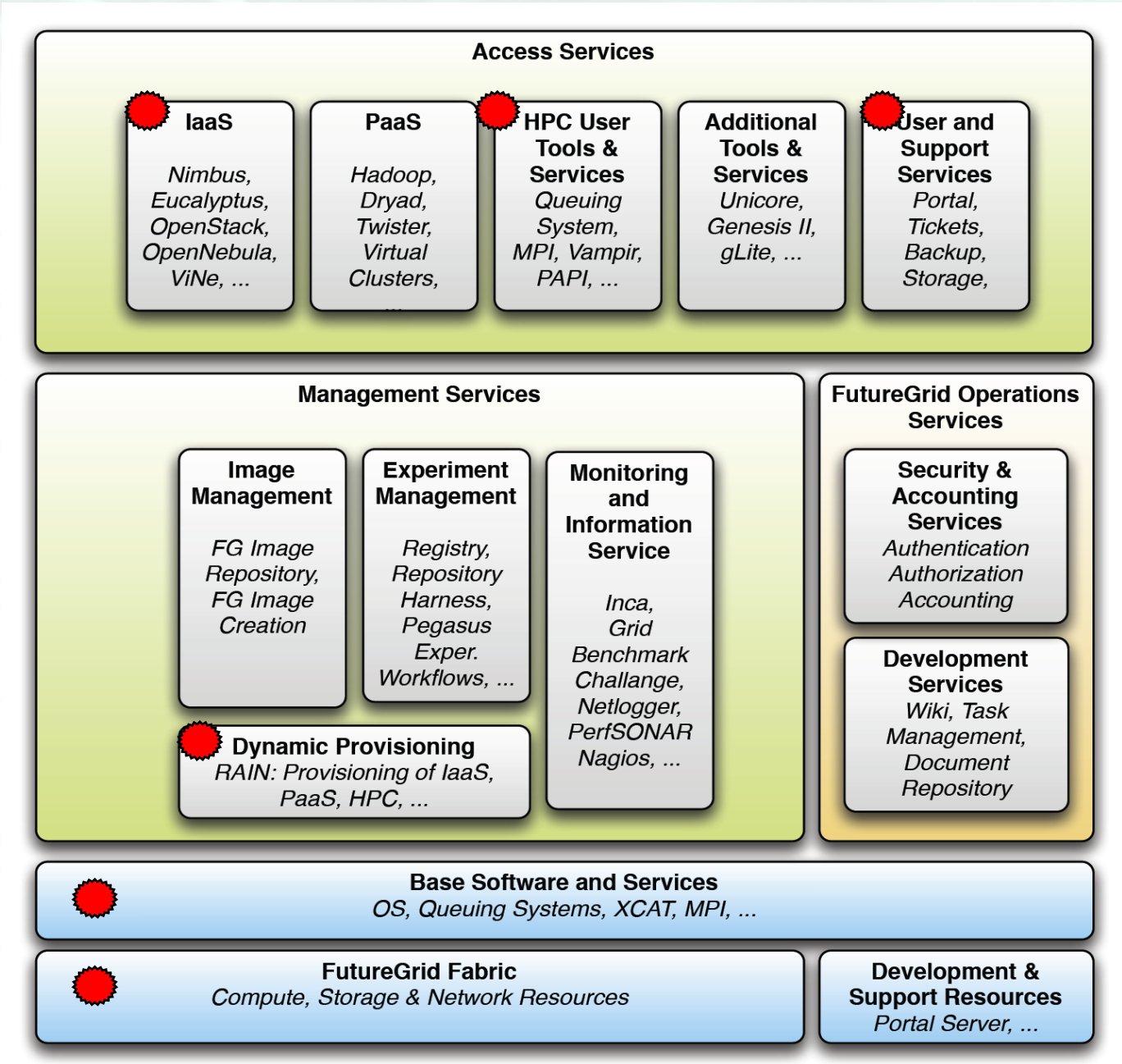
Indiana University

Presenter:

- Andrew J. Younge
- 2 minutes

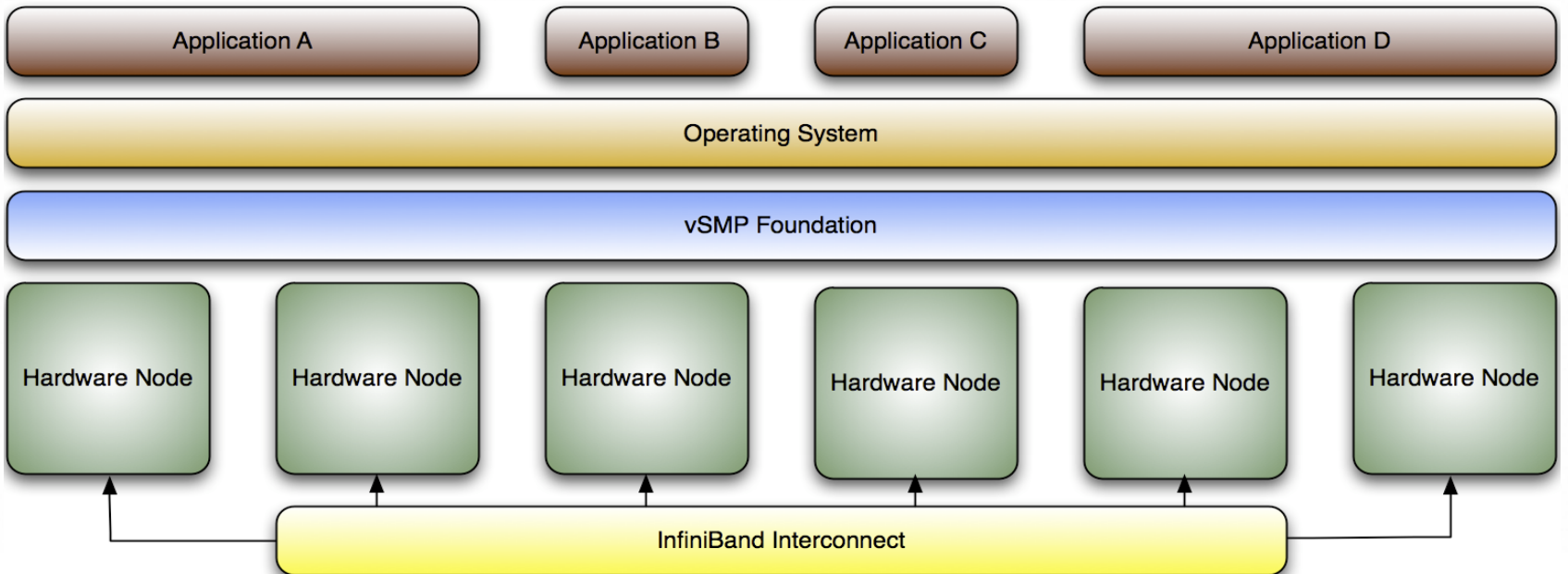
Key Points

- vSMP Foundation
- Usability of vSMP in FG



ScaleMP

- vSMP Foundation is a virtualization software that creates a single virtual machine over multiple x86-based systems.
- Provides large memory and compute SMP virtually to users by using commodity MPP hardware.
- Allows for the use of MPI, OpenMP, Pthreads, Java threads, and serial jobs on a single unified OS.
- Available today on the India IBM IDataPlex.
  - Currently used for Genome Assembly.
  - vSMP is also deployed on SDSC's Gordon.



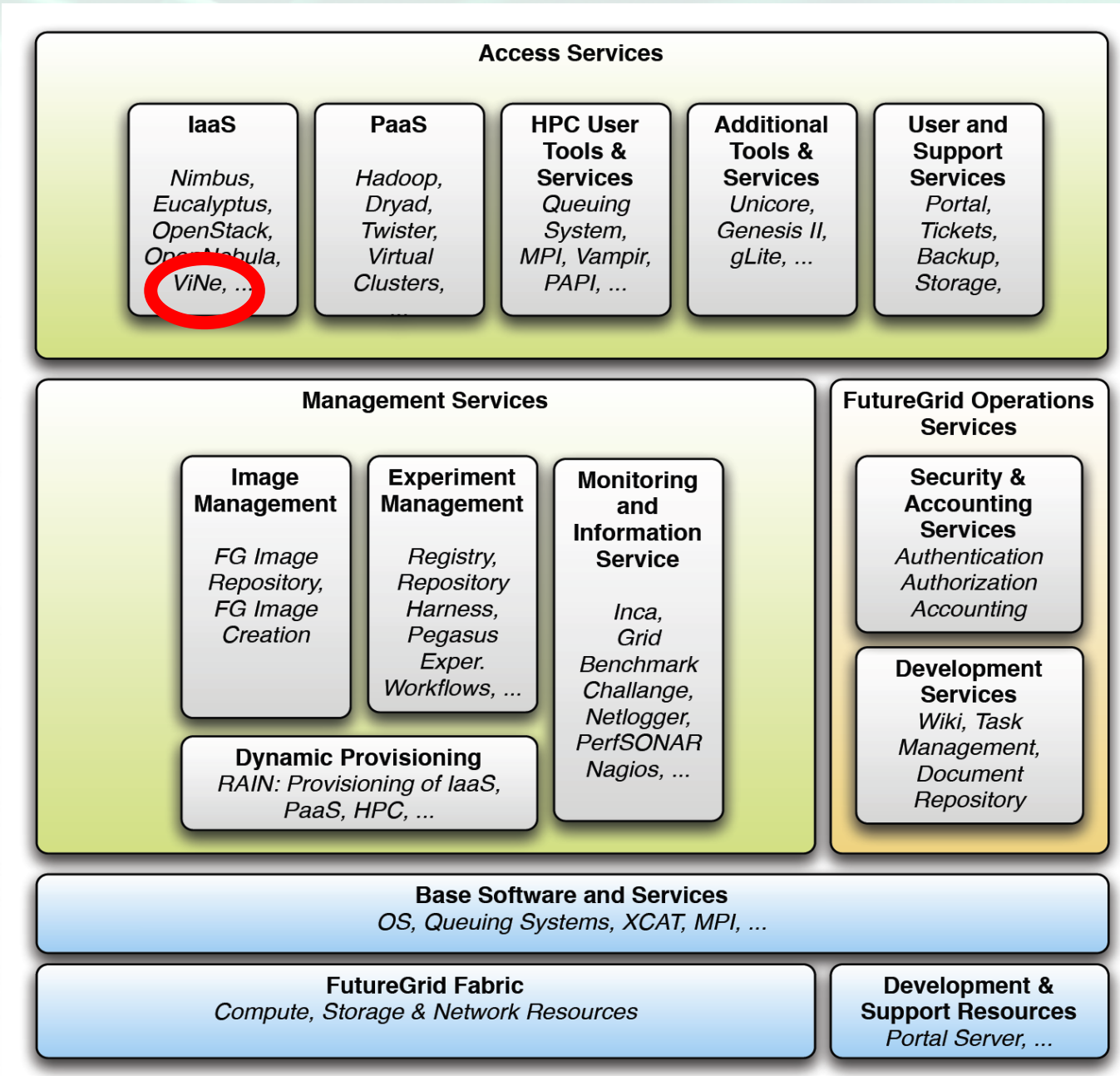
Vine

University of Florida

7 minutes

ViNe

- University of Florida
- Presenter: José Fortes



ViNe (Cont.)

Requirements

- Connectivity among FG and external machines
- Mutually exclusive overlay networks
- Easy management
- Configurable network parameters (e.g., delay, loss rate, bandwidth)

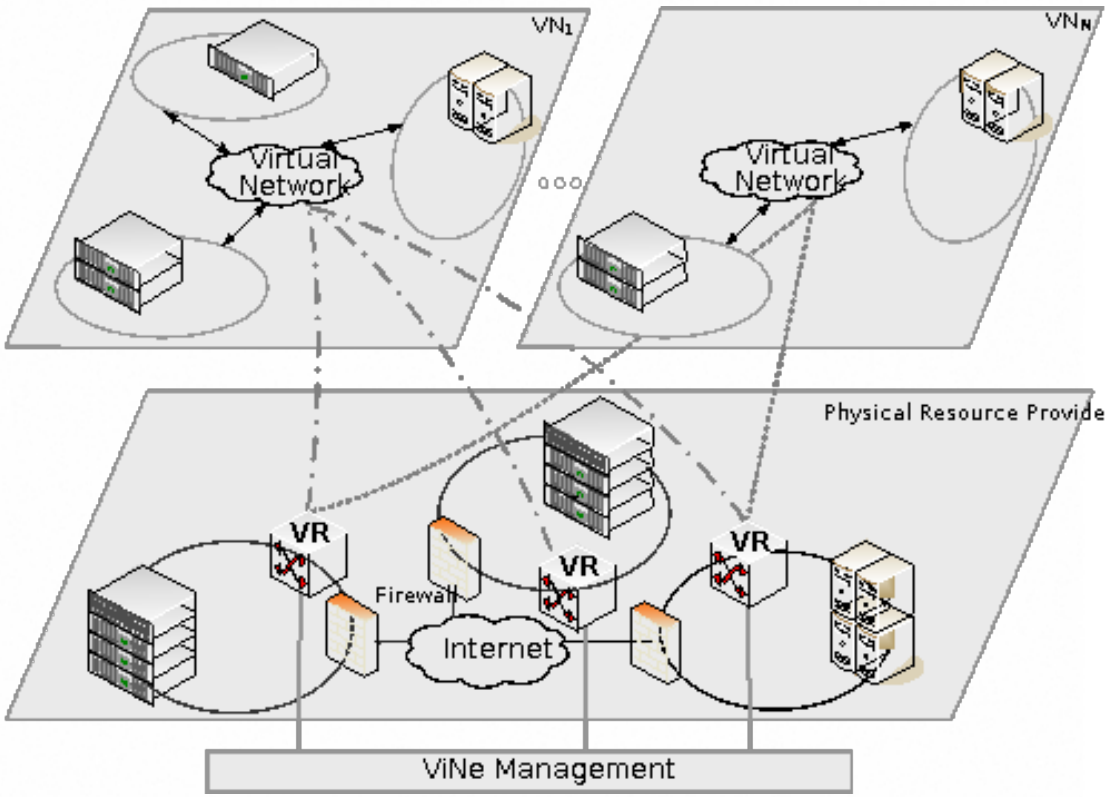
Usecases

- Deployment of virtual clusters spanning multiple sites
- Isolated networks minimize negative effects of misconfigured VMs
- Virtual clusters with appropriate connectivity should be easily started
- Deployment of experimental networks

ViNe (Cont.)

Design

- User-level network routing software (no hardware or kernel dependency), which creates overlay networks using the Internet infrastructure



ViNe (Cont.)

Implementation

- Routing logic implemented in Java (version 1.6+)
- Low-level network access implemented in C
- Built-in firewall/NAT traversal
- Routing capacity of 900 Mbps measured on foxtrot

Deployment

- On each site, a machine running ViNe software becomes a ViNe router (VR), working as a gateway to overlay networks for other nodes connected to the same LAN segment
- Deployed on sierra, foxtrot, hotel and 3 Grid'5000 sites, to offer full connectivity among VMs on all 6 sites

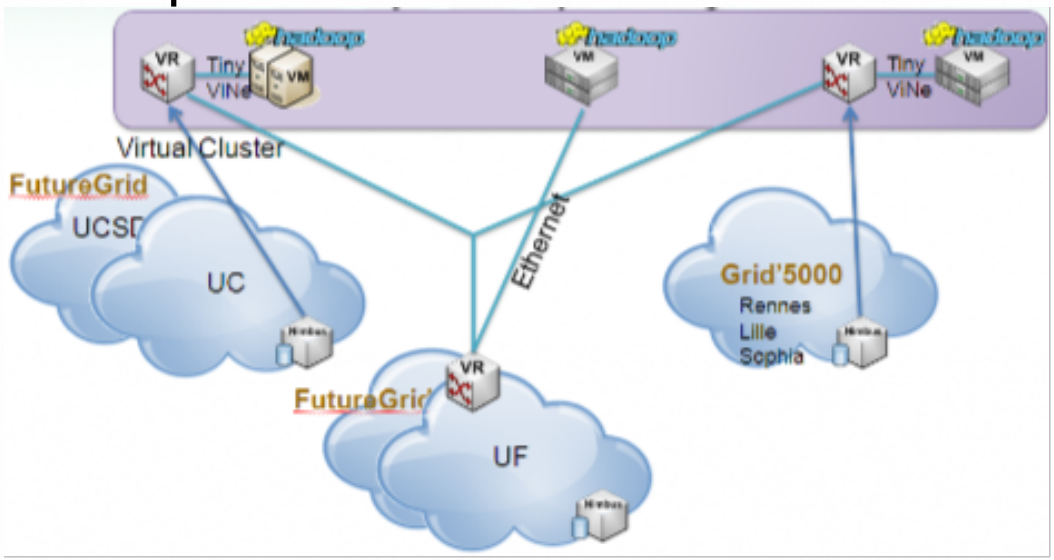
ViNe (Cont.)

Milestones

- Year 1 (completed)
  - Collect requirements to ViNe-enable FG sites
  - Run experiments to assess FG inter- and intra-site communication performance
  - Deploy ViNe on FG sites
  - Deploy Virtual Clusters across multiple FG sites, connect via ViNe, and run experiments/jobs
- Year 2
  - Q1/Q2 Design ViNe management APIs
  - Q2/Q3 Implement and Test ViNe management features
  - Q3/Q4 Deploy improved version of ViNe

Risks

- Not all FG sites have a physical machine running ViNe software. Deploying ViNe router in a VM has a negative impact on performance.

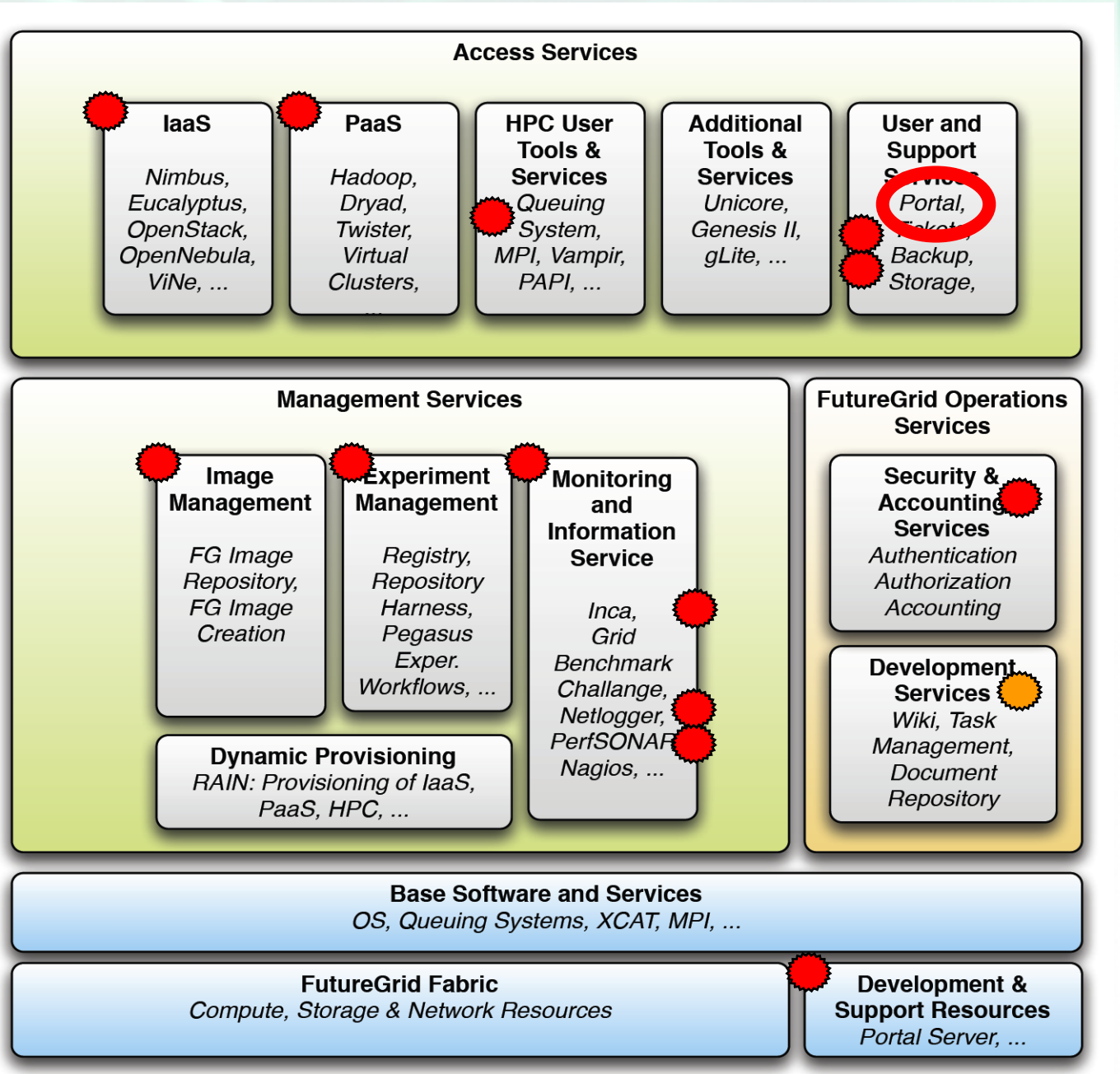


User Portal

Fugang Wang
Gregor von Laszewski
Indiana University
5 minutes

Key Points

- Entry point for obtaining help, support, training materials, etc.
- Enable FG community
- Web client for set of important services, like image management, project/experiment management, etc.



FutureGrid Web Portal: Requirements

- **Present information from diverse sources**
  - Status of the Resources, Software, and offered services: Inca, PBS, XCAT, ...
  - Information on how to use FG: Manual, FAQ/IU Knoweledge Base, General information about the project
  - Unified search: All relevant material integrated in a single search function
  - Role based access: user, sysadmin, approval committee, editor

- **Support FG specific processes**
  - Project Management**:** List/create/join/approve projects, provide personal view, list/report results
  - Experiment Management**:** List/create/monitor experiments; image Management: manage images used in experiments,; share/clone/verify images
  - Account Management**:** Integrate with the FG account management processes, allow interface with SSO services (manage SSH key, OpenID, certificates, ...)
  - Information Dissemination Management**:** through manual, FAQ/IU Knowledgebase, project & experiment information, editorial workflows, mailinglists/forum, RSS feeds, News, References

FutureGrid Web Portal: Status

- **Implementation**
  - Based on Drupal: proven open CMS with access control
  - Use of proven Drupal community extensions: no development needed for them, but configuration
  - New deployment: re-deployment, with FG processes in mind, not just web site
- **Available Features (PY1, PY2 ...)**
  - Drupal: forum, news, polls, information tables, page management, user management, theme, book layout (for manual), FAQ, references, OpenID
  - FG specific: supporting FG processes: account management; project management including FG experts, project approval committee; information dissemination to support FG these processes; SSH key management
- **Future Features (**FG specific, **PY...)** :
  - Eucalyptus: Support SSO management features for Eucalyptus (PY2 Q4); Integration of iPlant Atmosphere (PY2 Q2-3)
  - Experiment Management**:** List/create/monitor experiments (v0 PY3 Q1); image Management: manage images used in experiments (v0 PY2 Q3); share/clone/verify images (v0 PY2 Q4)
  - Account Management**:** Verify account data in the LDAP server (PY2 Q2)
  - Information Dissemination Management**:** improve editorial workflows (v1 PY2 Q3), extend RSS feeds (v1 PY2 Q4); unified KB search (PY3)
  - Integration with TG user portal: identify path once XD plan is available to us

FG Web Portal - Risks

- Modules used from the community may no longer be supported
  - use modules that are hugely popular
  - be aware of the Drupal roadmap

FutureGrid Web Portal

FutureGrid Portal

To Do | About | Help | Jobs | Developer | Admin | Projects | Accounts | Login | Experts | Editor | Search this site: | Search

News
- FutureGrid Boosts Indiana in IT
- IU to lead FutureGrid
- FutureGrid Workshop at SC09
- Director Fox Honored

Recent Publications
- PolarGrid and FutureGrid Panel Presentation
- FutureGrid Software Architecture
- FutureGrid User Portal
- Introduction

More...

New forum topics
- I have a portal account, why can I not login on the HPC resources?
- Which IaaS do you prefer?
- Why using a SSH key without password is wrong!
- Managing your Portal account
- High Failure Rates of Eucalyptus

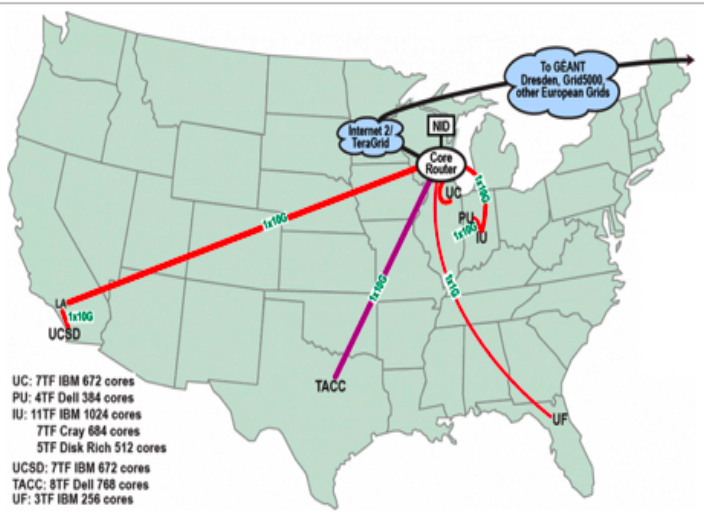
more

andrew.younge
- Sign In/Sign Up
- Biblio
- My account
- Popular content
- Ask a question
- Create content
- Frequently Asked Questions
- Recent posts
- Log out

Who's new
- grimshaw
- nalam
- kauschan
- shivjana
- conery

Home Page

View | Edit | Outline | Revisions



FutureGrid is a project to develop a high-performance grid test bed that will allow scientists to collaboratively develop and test innovative approaches to parallel, grid, and cloud computing. The test bed will be composed of a high-speed network connected to distributed clusters of high-performance computers. FutureGrid will employ virtualization technology to allow the test bed to support a wide range of operating systems. The goal is to build a system that helps researchers identify cyberinfrastructure that best suits their scientific needs. The FutureGrid project is funded by the National Science Foundation (NSF) and is led by Indiana University with University of Chicago, University of Florida, San Diego Supercomputing Center, Texas Advanced Computing Center, University of Virginia, University of Tennessee, University of Southern California, Dresden, Purdue University, and Grid 5000 as partner sites.

In case you like to apply for a portal account please use this link. A portal account allows you to apply for a project.

Rate This

Average: 4.8 (5 votes)

Your rating: 4

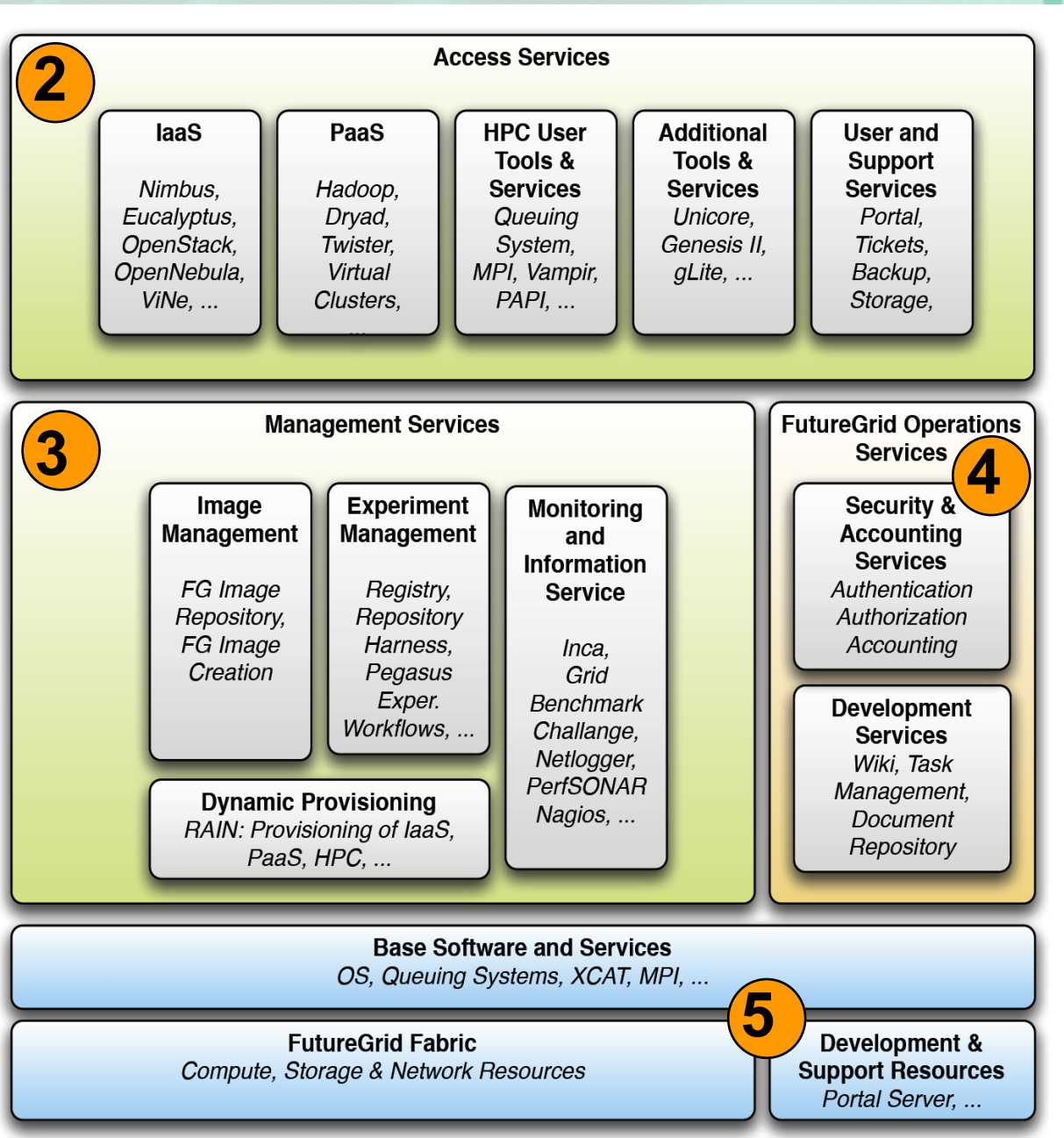
Subscriptions
- Subscribe

Book navigation
- Polls
- User Manual

Outline

1. Overview
2. Access Services
3. **Management Services**
4. Operations Services

5. We will not much go into:
- *Base Software and Services*
- *Fabric*
- *Software for Development & Support Resources*

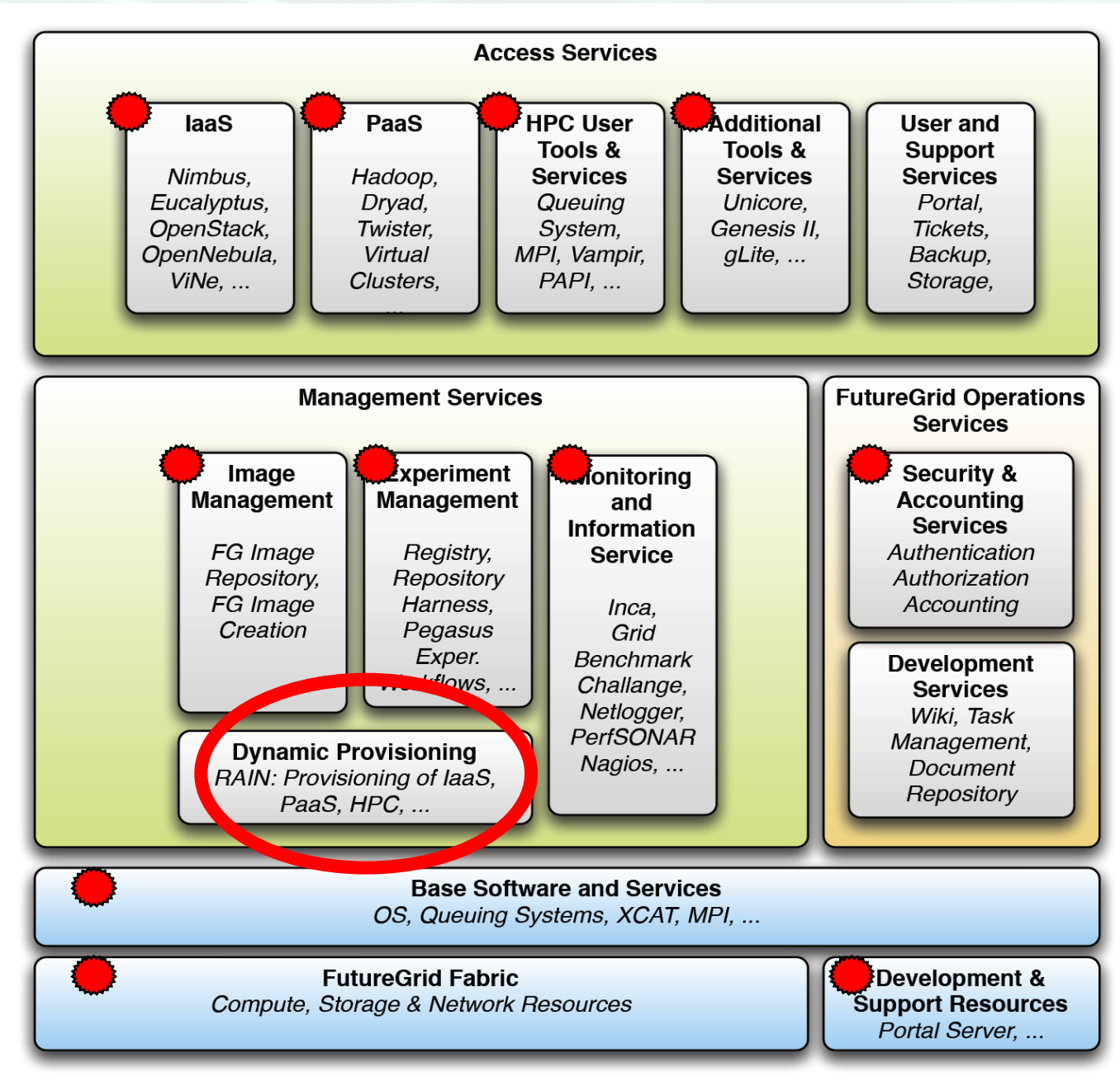


Dynamic Provisioning

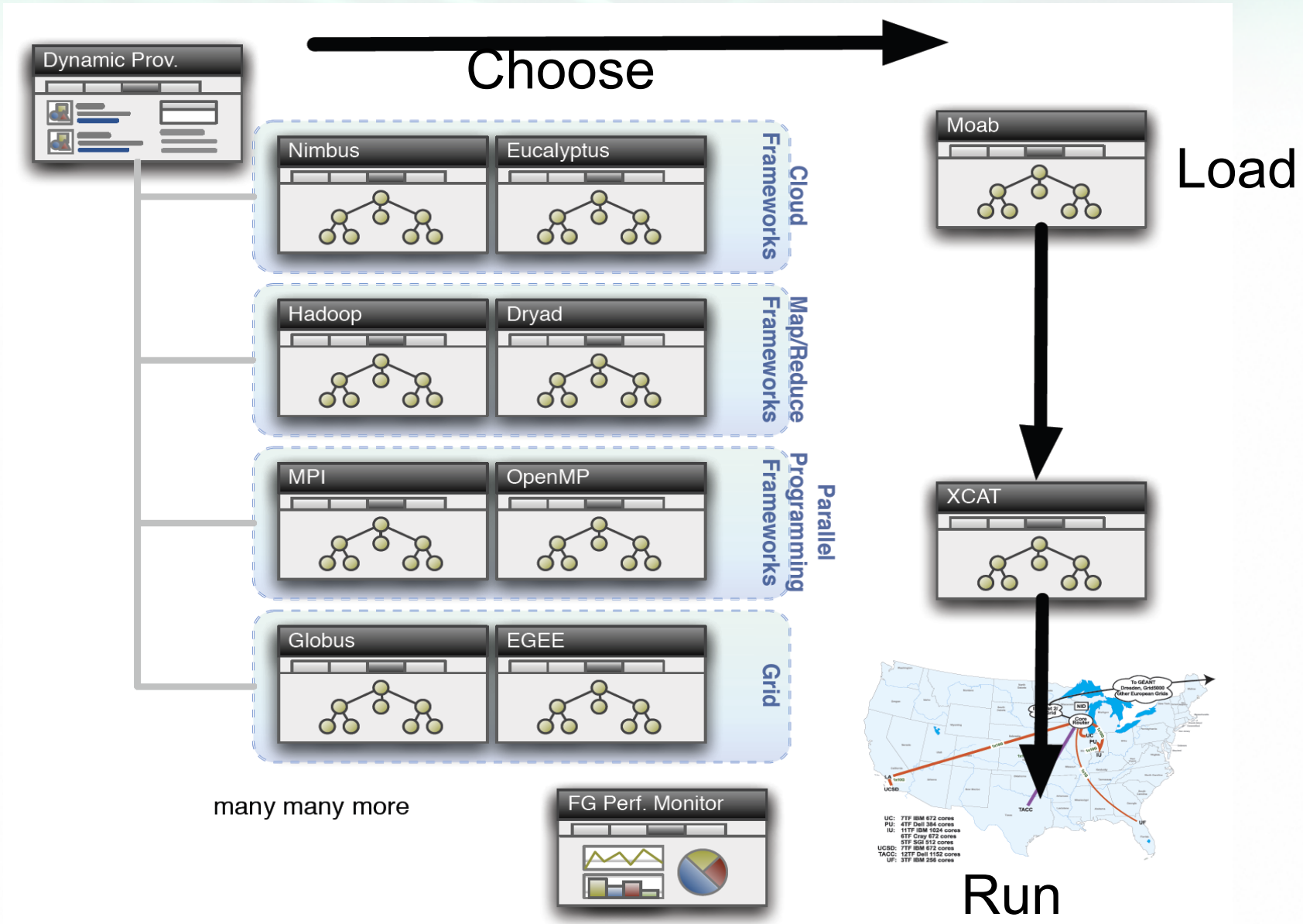
Gregor von Laszewski
Greg Pike
Fugang Wang
Archit Kulshrestha
Warren Smith
Indiana University
TACC
5 minutes

Key Points

- Customizable environment
- Not just images on IaaS
- Operating system level



Dynamic Provisioning



FG RAIN Command

- fg-rain –h hostfile –iaas nimbus –image img
- fg-rain –h hostfile –paas dryad …
- fg-rain –h hostfile –image img
  - the default way if I do not care about IaaS
- fg-rain –h hostfile –paas hadoop …

|

- fg-hadoop ....

Image Management

Fugang Wang
Andrew Younge
Gregor von Laszewski
Indiana University
5 minutes

Key Points

- Abstraction layer that deals with all FG images.
- Service oriented architecture so interaction with other modules could be easily achieved.
- Layered design so the choose of concrete implementation is flexible. E.g., provide alternative data storage mechanism.

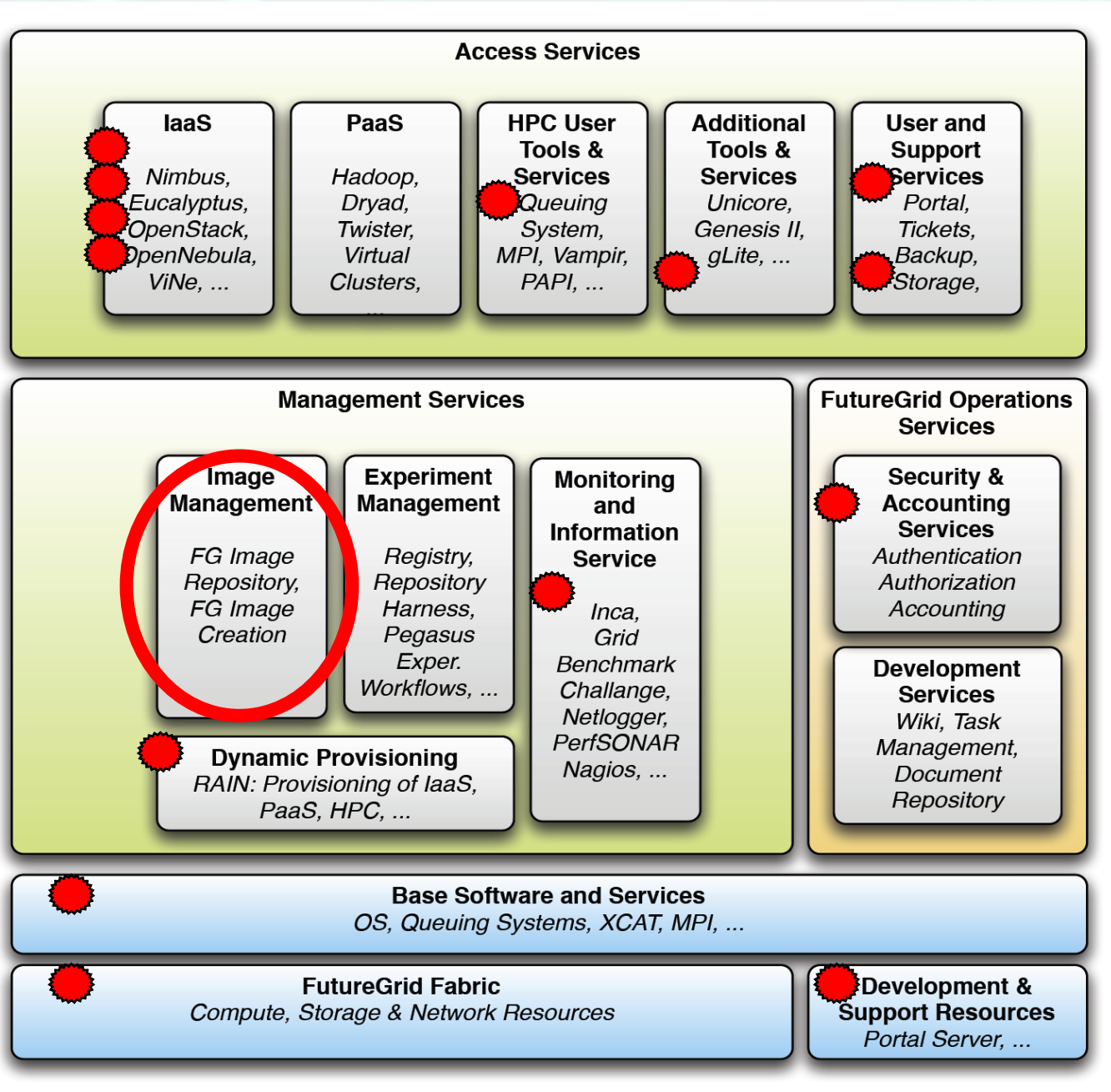


Image Management

Requirements

- Generate Images
  - Needed as part of security architecture
  - Consistency
  - Provide assistance to users
  - Provide integration with LDAP
- Store Images
  - Integrate with different image repository systems
  - Integrate with image creation module, and dynamic provisioning
- Access Interfaces
  - Commandline, portal, and REST interfaces

Use Cases

- Upload, search, clone, ... standard format
- Security review
- Access images with the same functionality but run on different IaaS frameworks
- Share Images with colleagues
- Create an image for me with features x,y,z, allow my FG project team members to login

Image Creation Process

- Creating deployable image
  - User chooses one base mages
  - User decides who can access the image; what additional software is on the image
  - Image gets generated; updated; and verified
- Image gets deployed
- Deployed image gets continuously
  - Updated; and verified
- Note: Due to security requirement an image must be customized with authorization mechanism
  - We are not creating NxN images as many users will only need the base image
  - Administrators will use the same process to create the images that are vetted by them
  - An image gets customized through integration via a CMS process

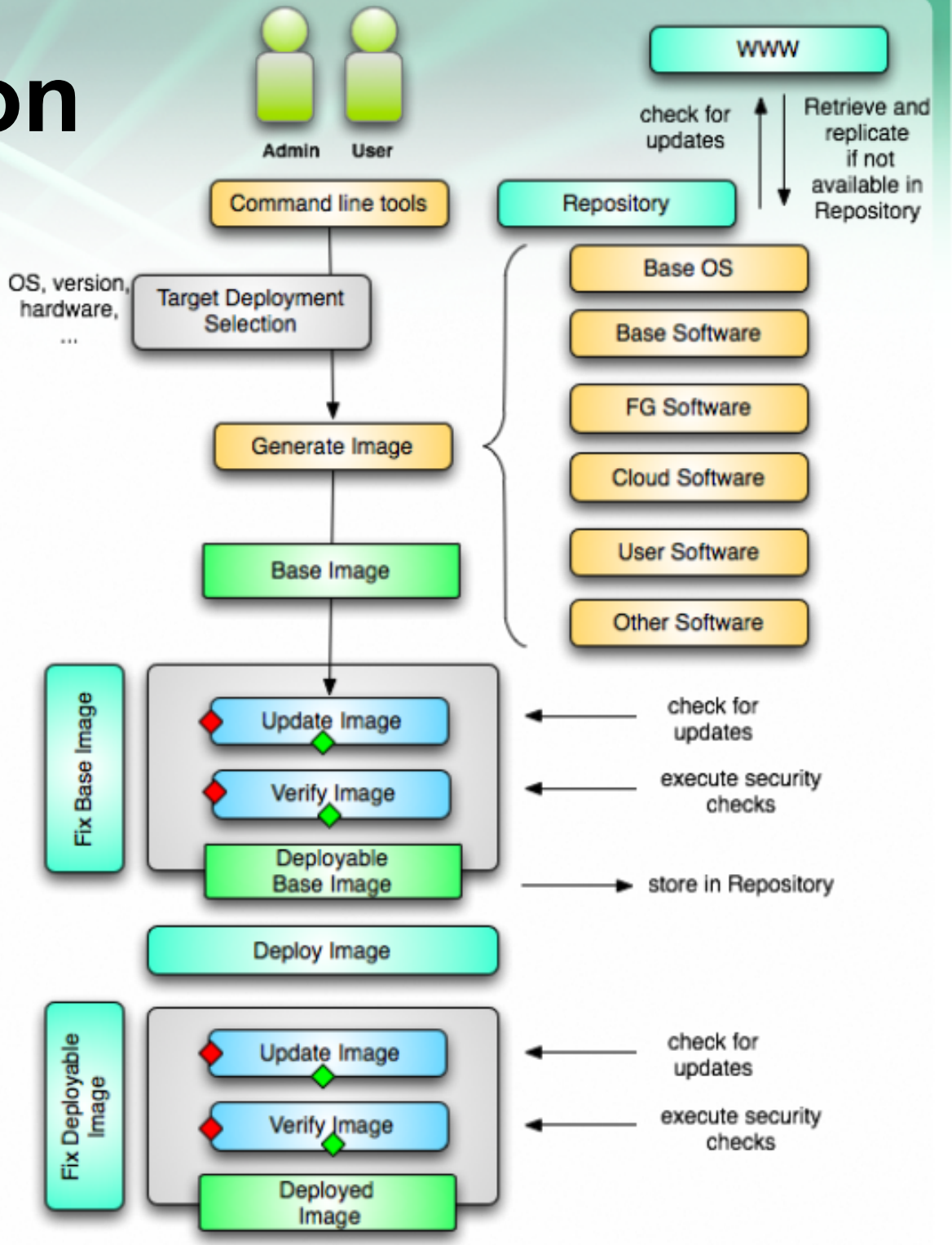


Image Management

Implementation

- Layered architecture; Web Services; Data access abstraction; Command line interface; Python; Integration with FG security framework

Deployment

- First deploy a centralized repository store based solution; then expand to provide distributed/replicated based one.
- First deploy a number of base images and test mechanism
- Integrate community contributed images

Review

- Continue to work with security experts (Von Welch formerly NCSA security expert was just hired by IU, ...).

Image Management

Milestones

PY1

- Designed and prototyped an Image Repository & Generation services
- Prototyped configuration management system for use with bare metal and virtual machines

PY2

- Q1 Deliver and test an alpha release of the image generation tools
- Q2 Deliver repository on each resource
- Q2 Integrate LDAP authentication into image management services
- Q3 Distributed repository database
- Q3 Provide an updated image generation service in beta release

PY3

- REST interfaces & Portal interface

PY4

- Dynamic user pattern governs image creation

Risks

- There will never be a secure image regardless which technology we use
- High level of integration with the various IaaS technologies
- Standards are under development
- Some users may want to bypass the mechanism
  - I have my code developed 30 years ago, please run it ....
  - but ... what about all the exploits ....

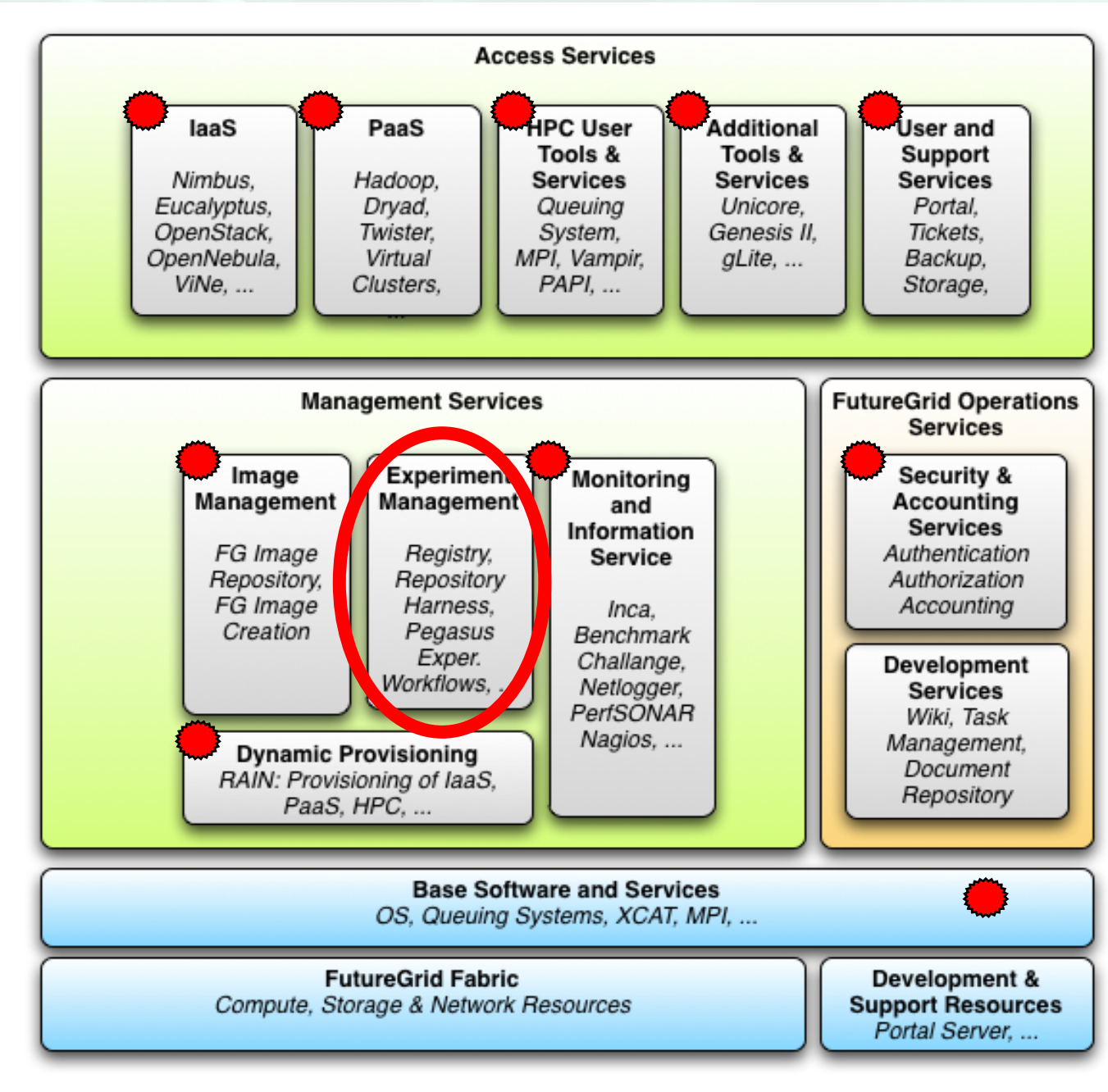
Experiment Management

Warren Smith[1]
Luke Wilson[1]
Ewa Delman[2]
Jens Voeckler[2]
Gregor von Laszewski[3]
Fugang Wang[3]
Greg Pike[3]
Archit Kulshrestha[3]
7 minutes

[1]TACC
[2]USC ISI
[3]IU

Key Points

- enable reproducible experiments



Experiment Management

Requirements

- Assemble and release resources
- Execute actions on assembled resources
- Monitor actions and results
- Record and archive information about an experiment
- Allow experiments to be repeated as run or with modifications

Use Cases

- Workflow-based experiment management
- Interactive experiment management
- A mix of the two

Experiment Management

Design

- Provide tools to coordinate experiment execution
  - Interact with a number of FutureGrid services
- Support several usage models
  - Workflow
  - Interactive
  - Hybrid
- Store experiment information for later use
  - Service

Experiment Management Components

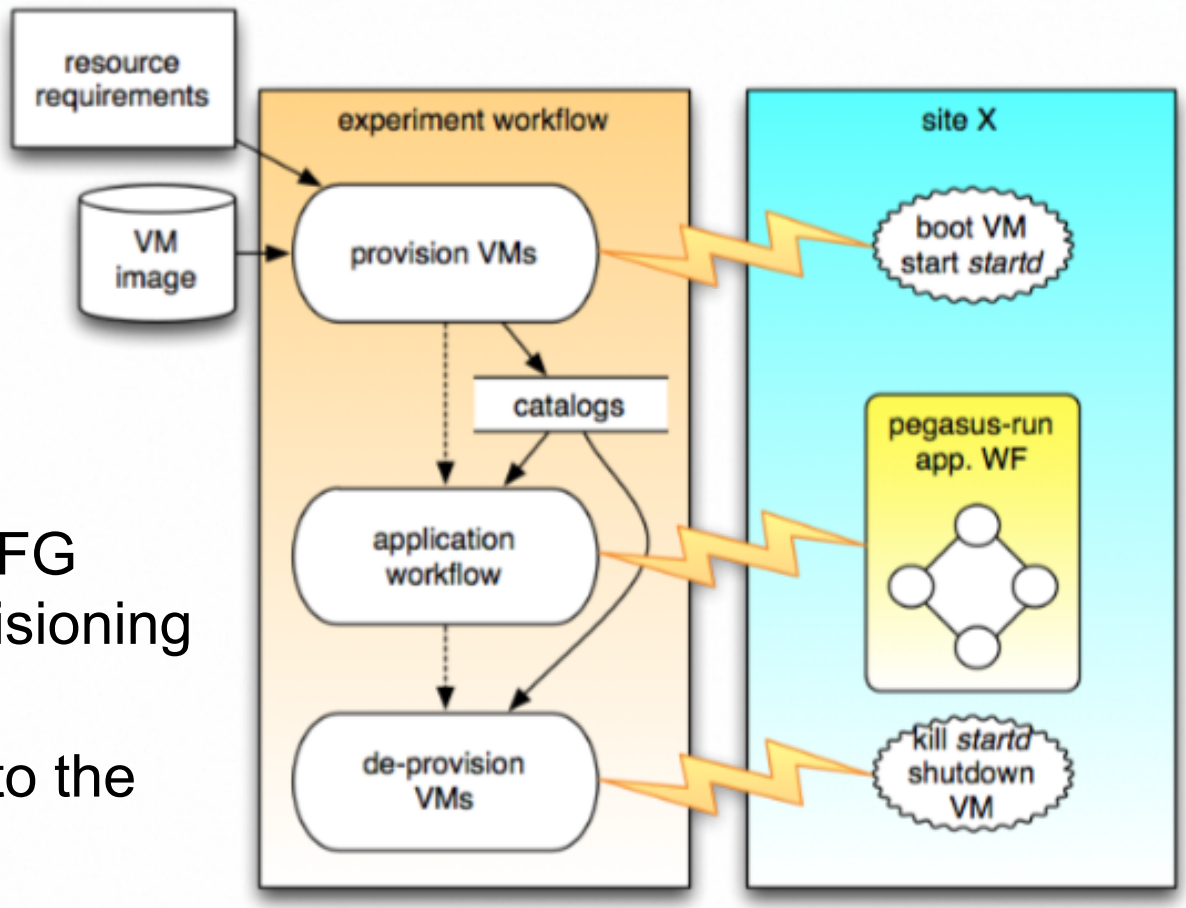
Implementation

- Pegasus
  - Workflow-based experiment management
  - Enhance existing tool
- TakTuk
  - Basic interactive experiment management
  - Reuse tool deployed on Grid 5000
- Messaging-based Execution and Monitoring System (MEMS)
  - More sophisticated interactive experiment management
- Experiment Repository
  - Store and retrieve information about experiments
- Integration with the Portal

Experiment Management

Pegasus Deployment

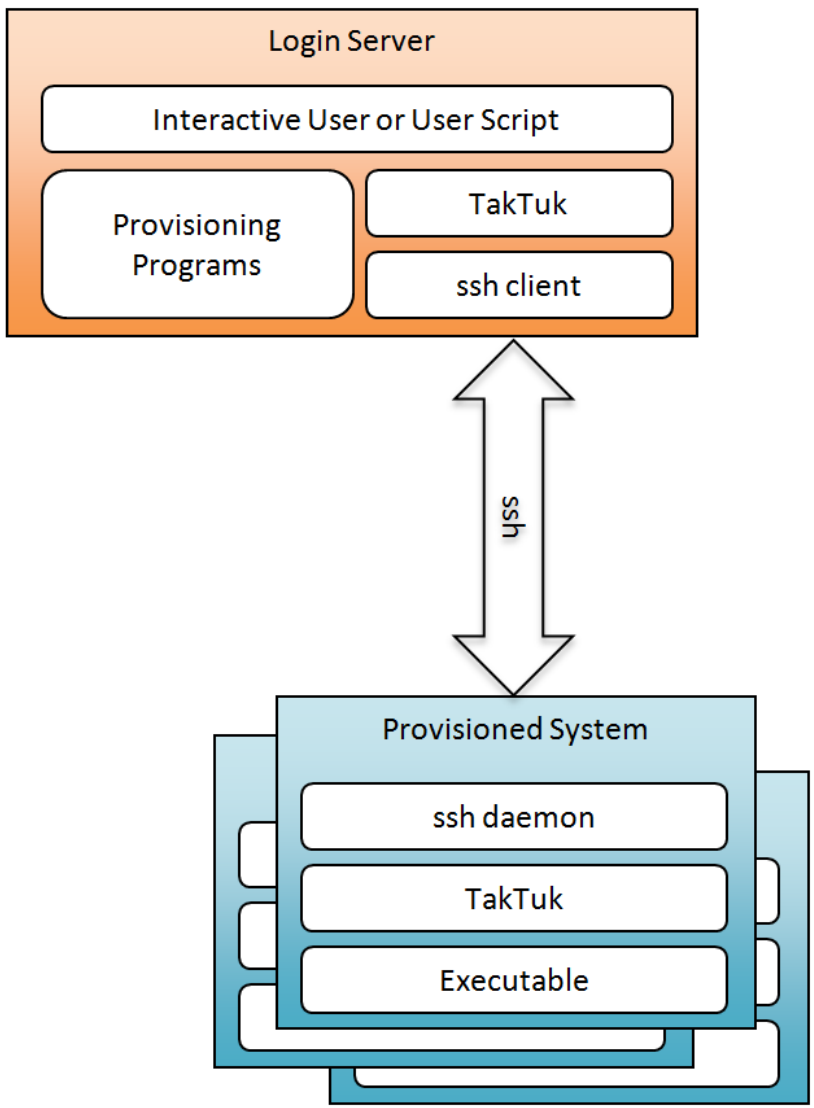
- Existing standard workflow management, deployed on FG
- Enhancing to meet FutureGrid needs:
  - Adding timing support
  - Developing interfaces to FG provisioning and de-provisioning capabilities
  - Implementing interfaces to the image repositories
  - Defining reproducibility—same logical experiment vs same exact experiment



Experiment Management

TakTuk Deployment

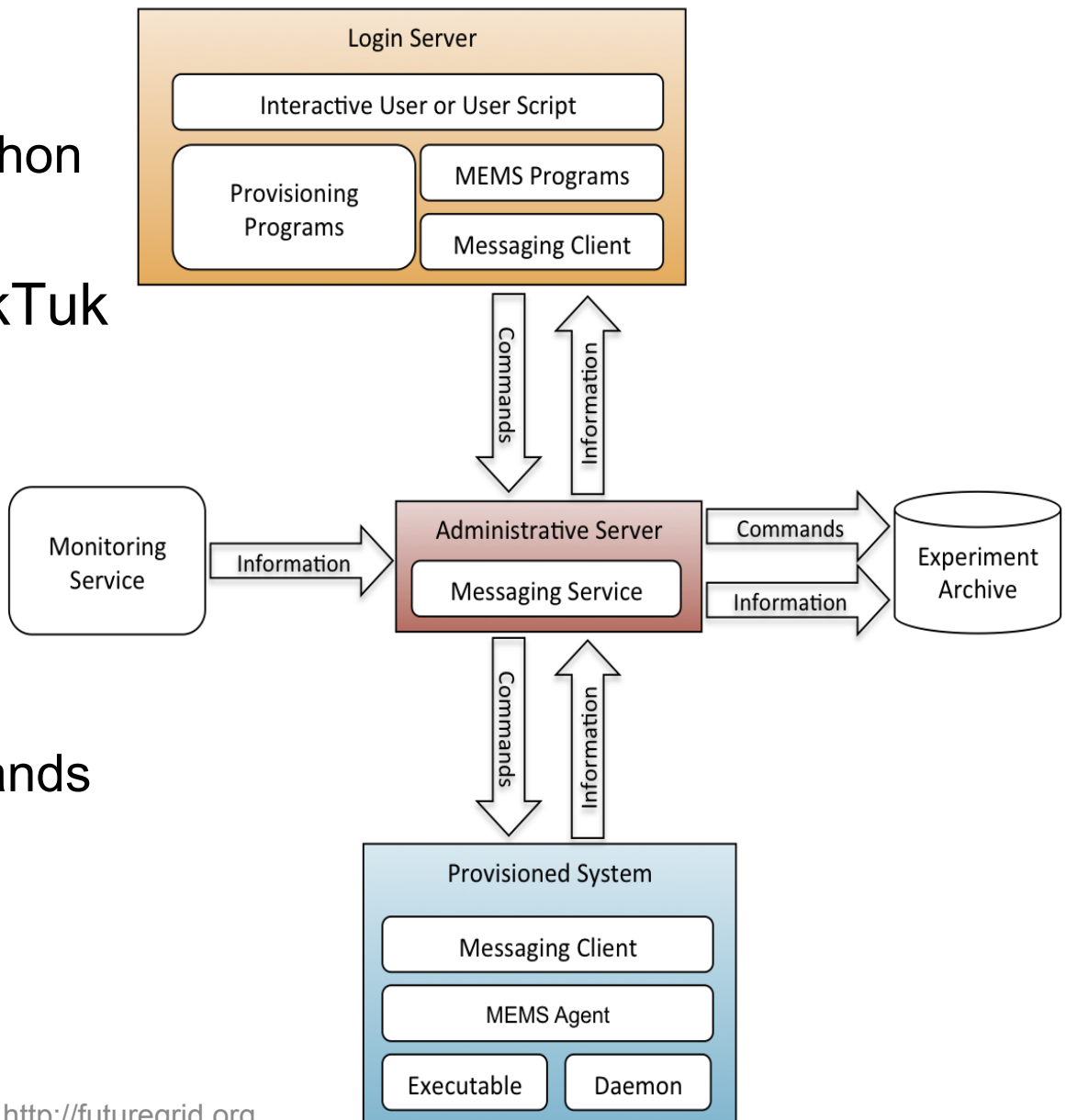
- Cluster-fork/parallel shell type tool
- Deployed on Grid 5000
- Minimal requirements
  - Written in Perl
  - Only other dependency is ssh
  - Self deploys any necessary components to provisioned systems
- Optimized execution
  - Arranges provisioned systems into a tree
- Partially deployed on FutureGrid



Experiment Management

MEMS Deployment

- Minimal requirements
  - Programs and agent in Python
  - Python messaging client
- Additional features over TakTuk
  - Automatic logging of commands, results, other information
  - Provide information about provisioned systems and FutureGrid
  - Execute distributed commands simultaneously
  - Built in support
- Under development



Experiment Management

Experiment Archive Deployment

- Gathering requirements
  - Interfaces
    - Command line and web. Messaging to support MEMS. APIs?
  - Functionality
    - Insertion, querying, searching
    - Provenance & metadata
    - Grouping? Annotation?
- Exploring design options
  - Information format/organization
  - One archive or many?
    - A number of existing archives are relevant: Image repository, Inca, Netlogger

Experiment Management

Milestones

- Develop provisioning workflows
- Develop initial timed workflow solution
- Complete TakTuk deployment
- Finish MEMS development
- Deploy MEMS

Risks

- Multiple tools could be confusing to users
  - Document differences well
- Many dependencies may cause deployment delays
  - Provide partial (but useful) functionality

Pegasus as an application management capability

Presented by

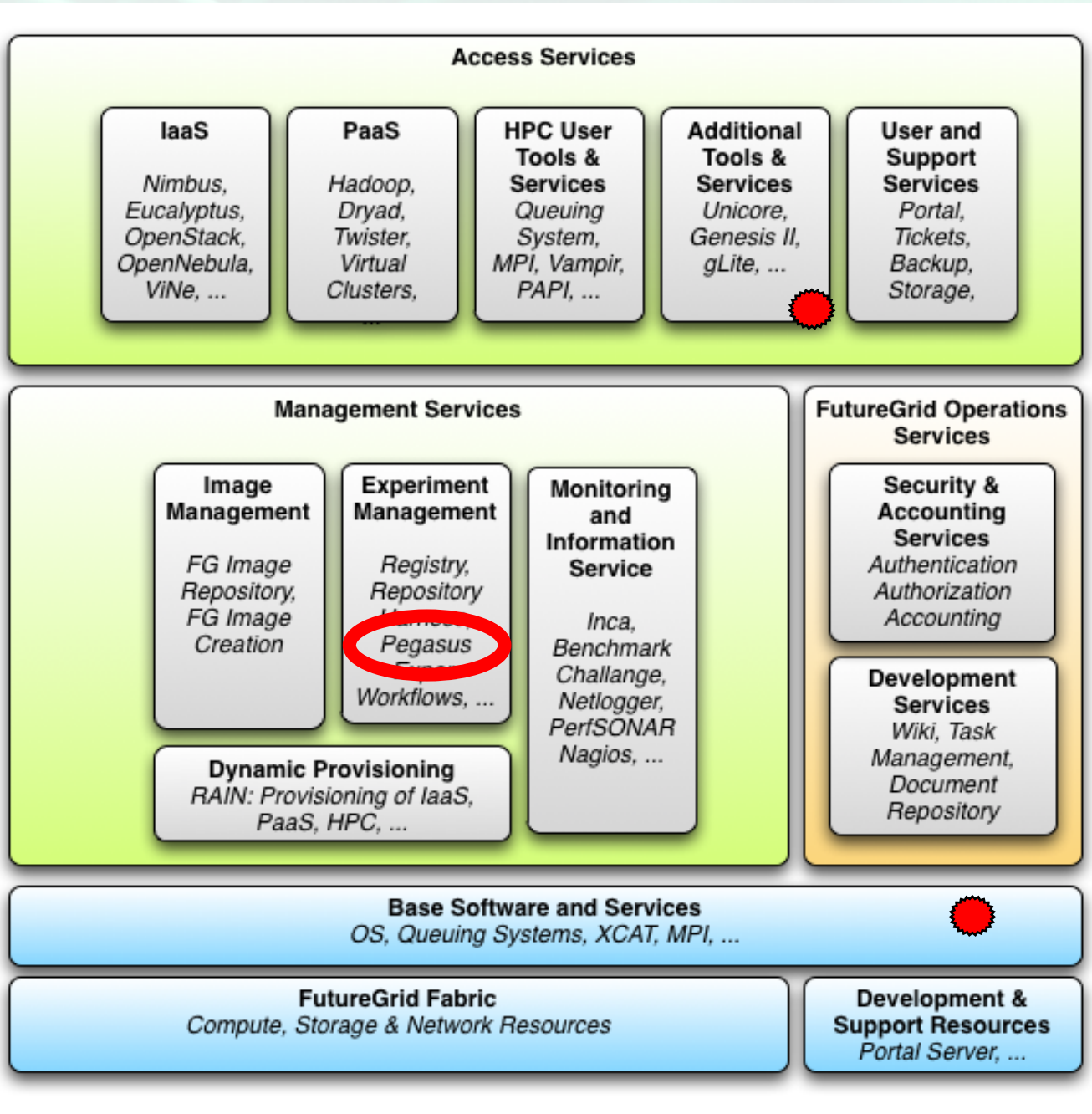
Jens Vöckler

Pegasus

Ewa Deelman
Jens Vöckler
USC Information Sciences Institute

Presenter:

- Jens Vöckler
- 7 minutes



Pegasus managing workflow applications on FutureGrid

Use-cases

- User familiar with Pegasus wants to run existing workflows on FG resources.
- Provide and environment for tutorials.

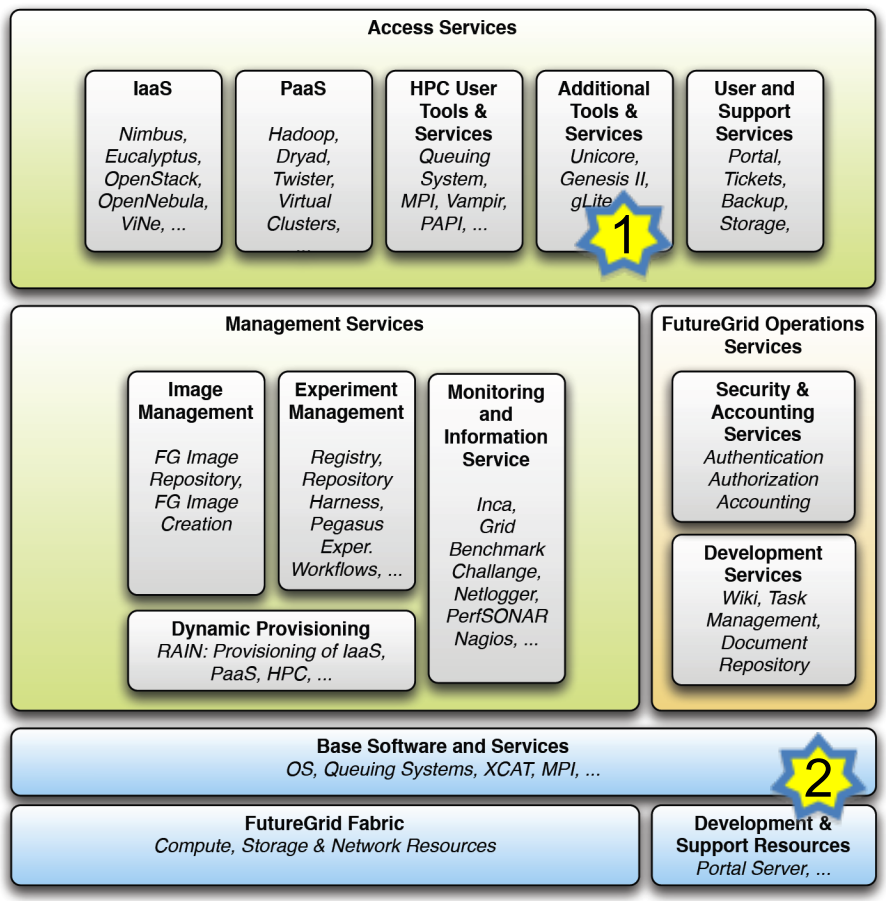
Requirements

- Provide Pegasus VM as submit host to users familiar with Pegasus
- Complementary to Experiment Management
- Develop new capabilities to address FG environment
- [optional] Pre-installed Pegasus run-time tools.

Pegasus

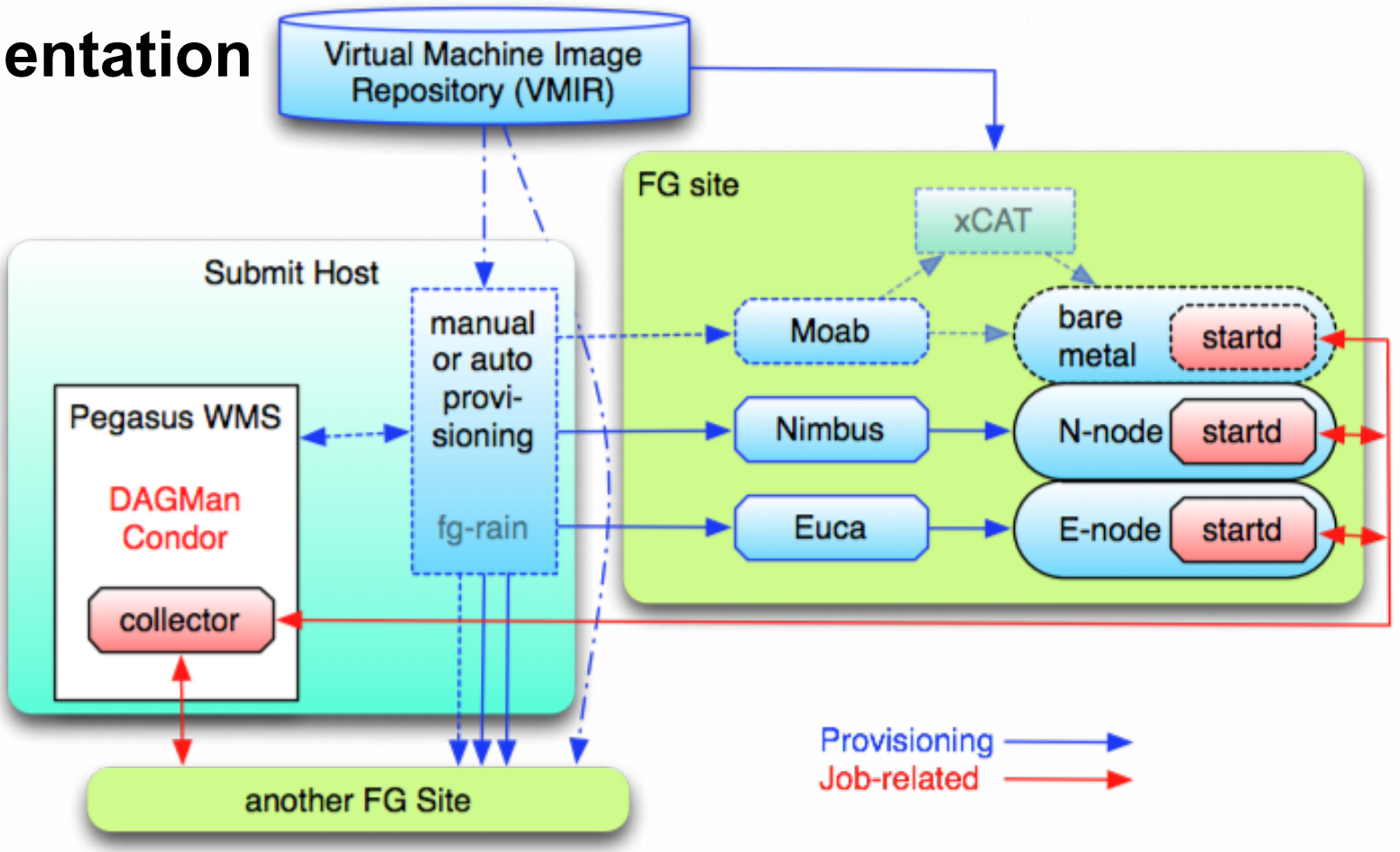
Design

1. Provide VM with Pegasus WMS to interested users.
   - Runs the planner.
   - Manages workflow(s).
   - Aggregates resource VM (s).
2. Provide Pegasus run-time tools (optional).



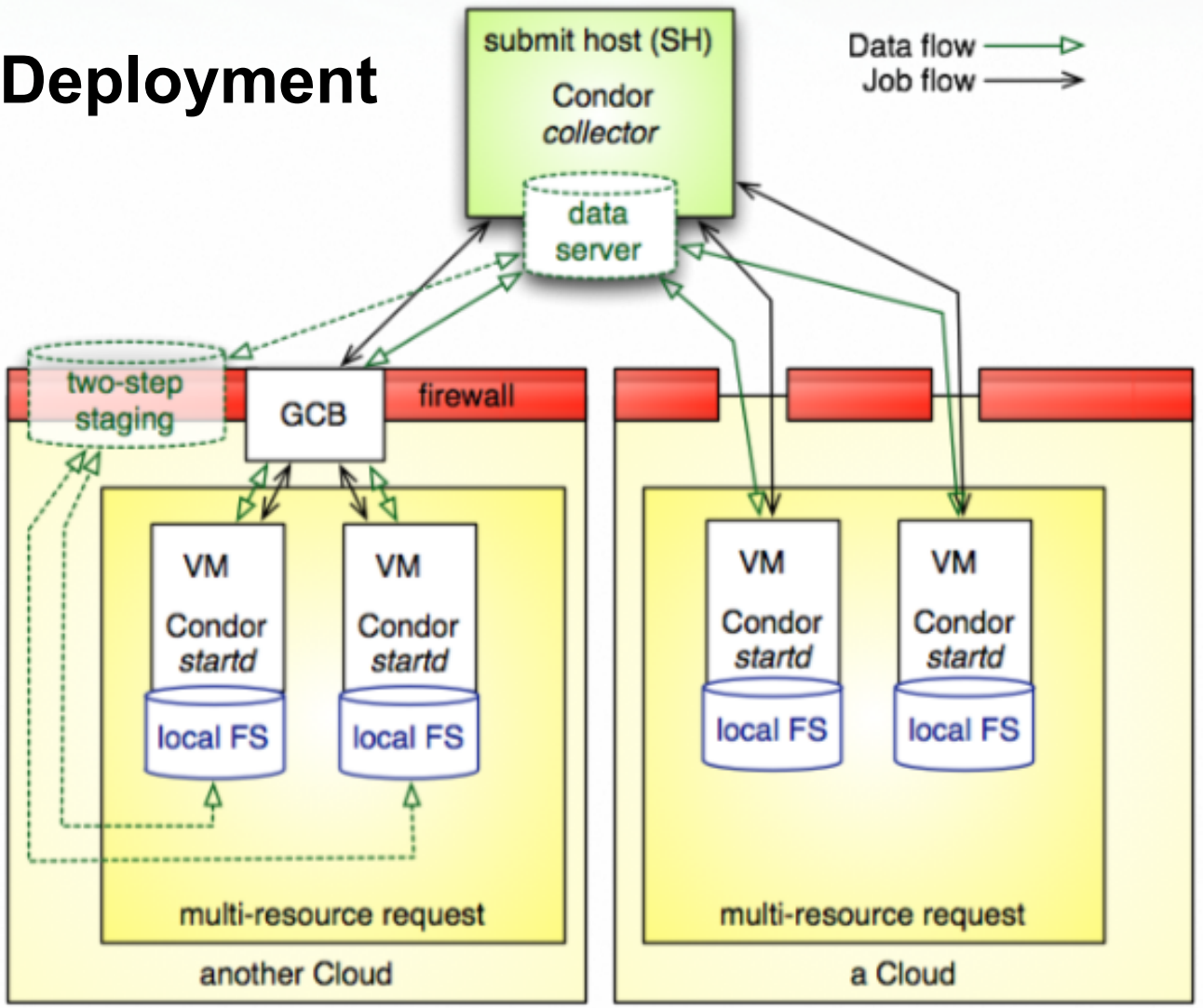
Pegasus

Implementation



Pegasus

Example Deployment



Pegasus

Milestones

- Provide “Planner VM”
  - Includes Condor tunings.
  - Manages prov. resources.
- Improve “2nd-level staging”
  - Permit multiple protocols to stage large data files.
  - Make transfers in head-less execution 1st class citizens.
- Include “bare-metal” execution using Moab.

Risks

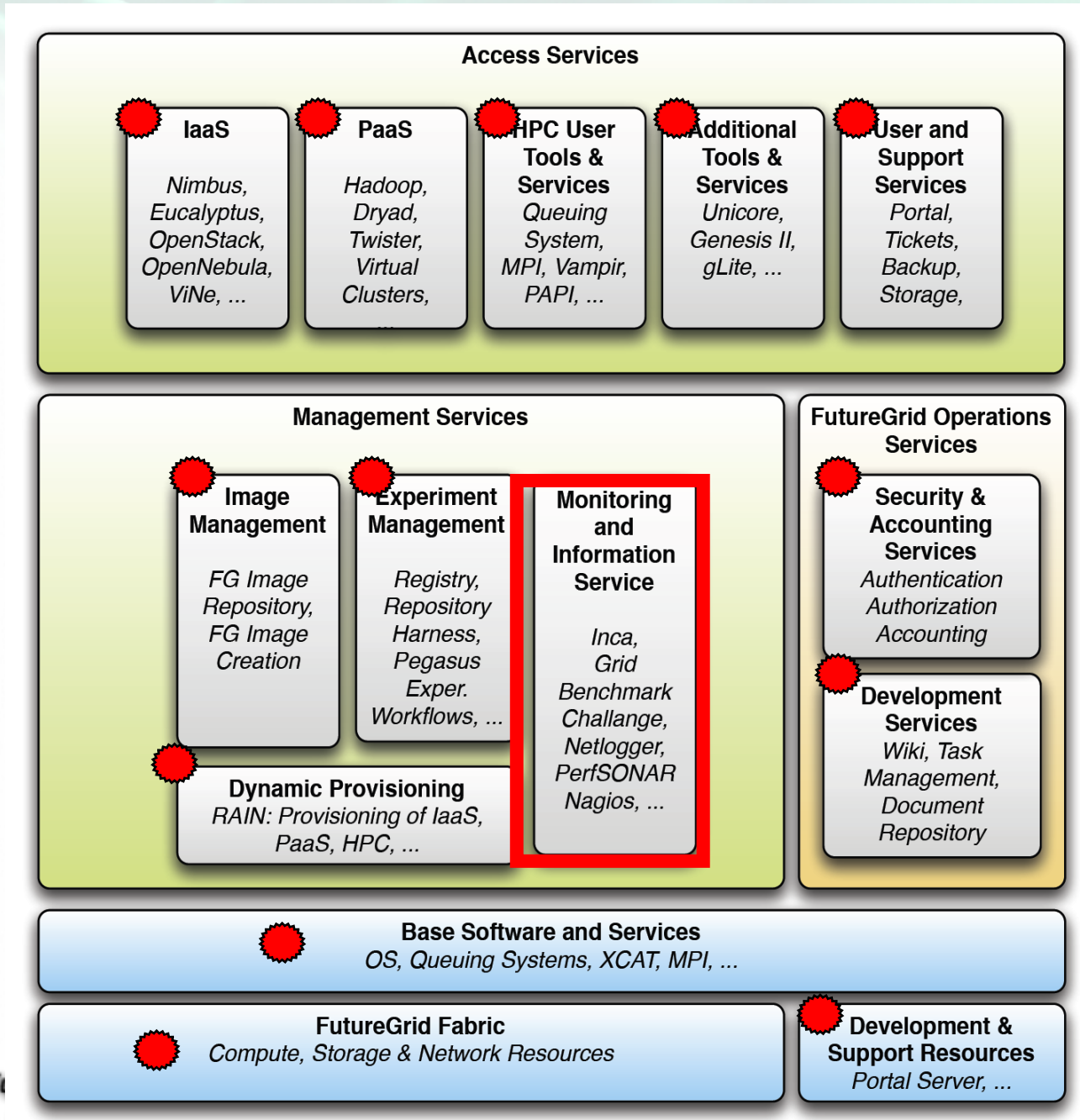
- Requiring a too specialized infrastructure.
  - Dependencies on too many 3rd-party software pieces.
- Too many auxiliary nodes in the generated workflow possibly negatively impact execution turn-around.

Monitoring and Information Services

Presenters:

Shava Smallen (SDSC)
Piotr Luszczek (UTK)

9 minutes



Monitoring and Information

Requirements

- Detect functional and performance problems on FutureGrid
- Collect basic information and usage about components
- Compare the performance of FG to other systems
- Re-use existing components
- Measurement results are stored historically
- Minimal system impact
- Flexible query interface

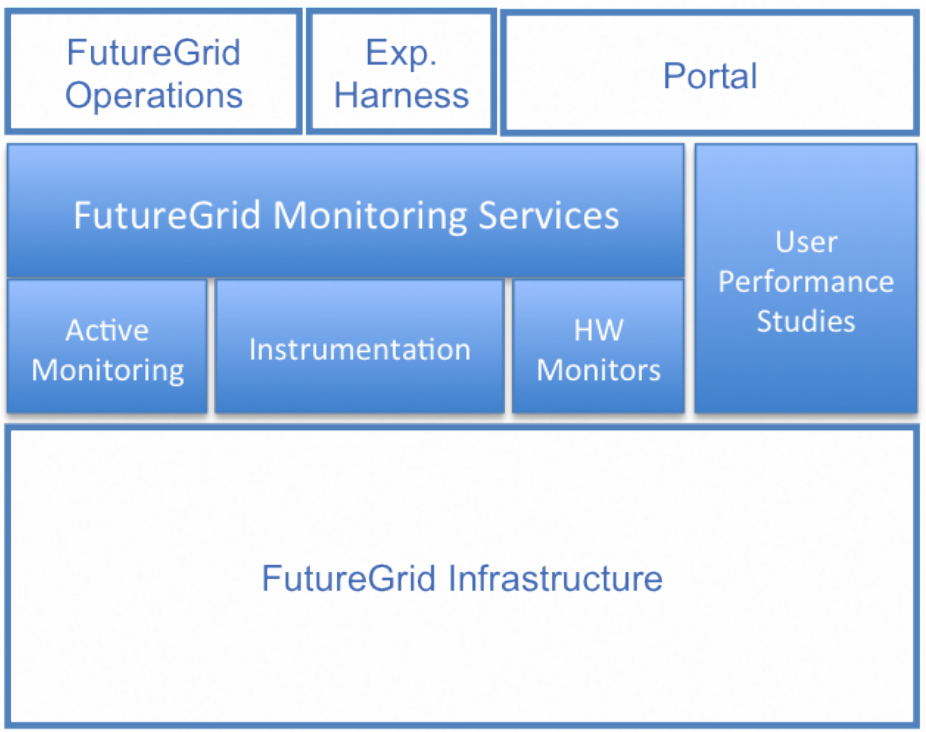
Use cases

- Can a user submit a job to each HPC resource?
- How much time does it take for a user to create an experiment?
- What is the number of VM instances deployed in Nimbus and Eucalyptus?
- How many users are utilizing the system?
- What is the machine performance (HPCC, SPEC, etc.)
- What is the utilization of machines?
- What is the network performance?

Monitoring and Information

Design

- Actively test and measure the infrastructure as a user (Inca, GBC - Grid Benchmark Challenge)
- Passively collect usage and performance information from infrastructure (Netlogger)
- Leverage system monitoring tools (Nagios, Ganglia, PerfSONAR, GlobalNOC tools, …)
- Repository to host user performance studies
- Interface to other FG components (experiment harness, portal)

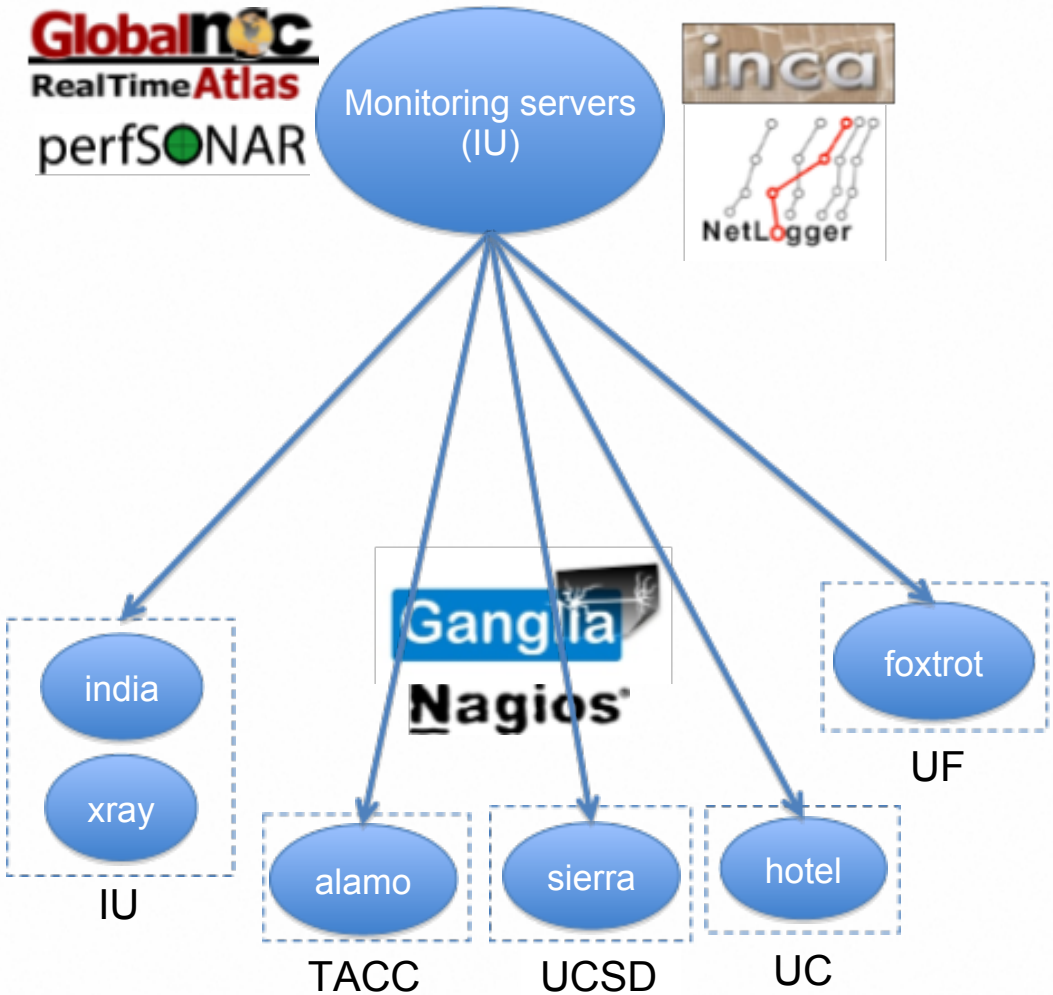


Monitoring and Information

Implementation

- **Inca**
  - **Server** – Three Java server processes with Postgres backend and reporter repository (Perl, Python)
  - **Client** – Perl daemon
- **Netlogger**
  - **Server** – Two processes with either TCP or AMQP interfaces and MongoDB backend
  - **Client** – AMQP or TCP APIs (C, Perl, Python, Java) and parse script
- …

Deployment



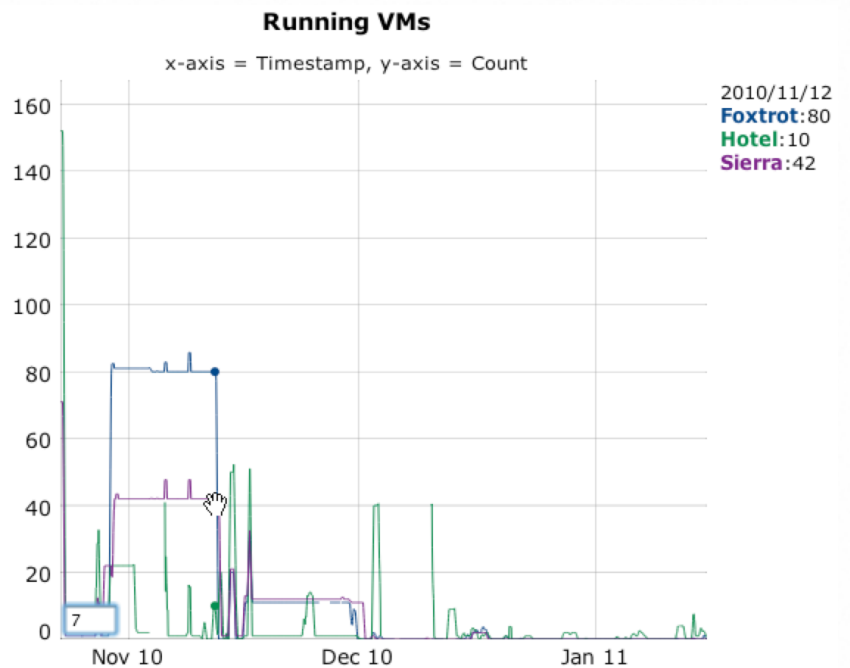
Monitoring and Information

Milestones

- **Year 1** (completed)
  - **Q1-Q2**: Initial architecture document completed
  - **Q2**: deployed Inca server and Inca clients to Xray, India, and Sierra -- provides basic monitoring of available software and services
  - **Q3**: automated benchmarking with HPCC deployed to India and Xray, Inca deployed to Foxtrot
  - **Q4**: Inca deployed to Hotel, Netlogger server installed, collect and display machine partitioning information
- **Year 2**
  - **Q1**: completed
    - 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
  - **Q2-Q4**:
    - Testing of image packages and monitoring of image generator and image repository
    - Add additional tests
    - Deploy Nagios
    - Begin development of GBC

Risks

- Dependent on other software components being ready (image generation, dynamic partitioning, image repository, experiment harness, …)



History of running VM counts in Nimbus deployments collected by Netlogger

Inca

http//inca.futuregrid.org

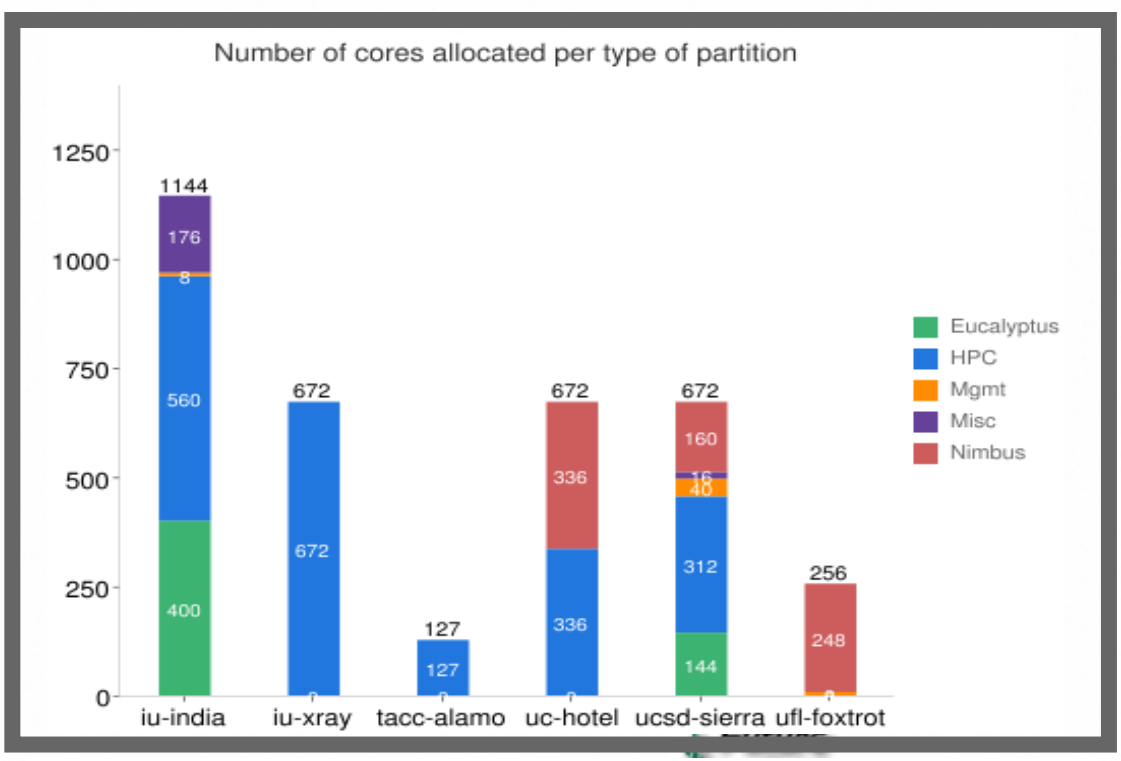
Cloud (view legend)

	iu-india	iu-xray	tacc-alamo	uc-hotel	ucsd-sierra	ufl-foxtrot
eucalyptus-clientStatus	✓	n/a	n/a	n/a	✓	n/a
eucalyptus-storage	✗	n/a	n/a	n/a	✓	n/a
eucalyptus-webpage-loads	✓	n/a	n/a	n/a	✓	n/a
nimbus-clientStatus	n/a	n/a	n/a	✓	✓	✓
nimbus-external-telnet	n/a	n/a	n/a	✓	✓	✓
nimbus-storage	n/a	n/a	n/a	✓	✓	✓

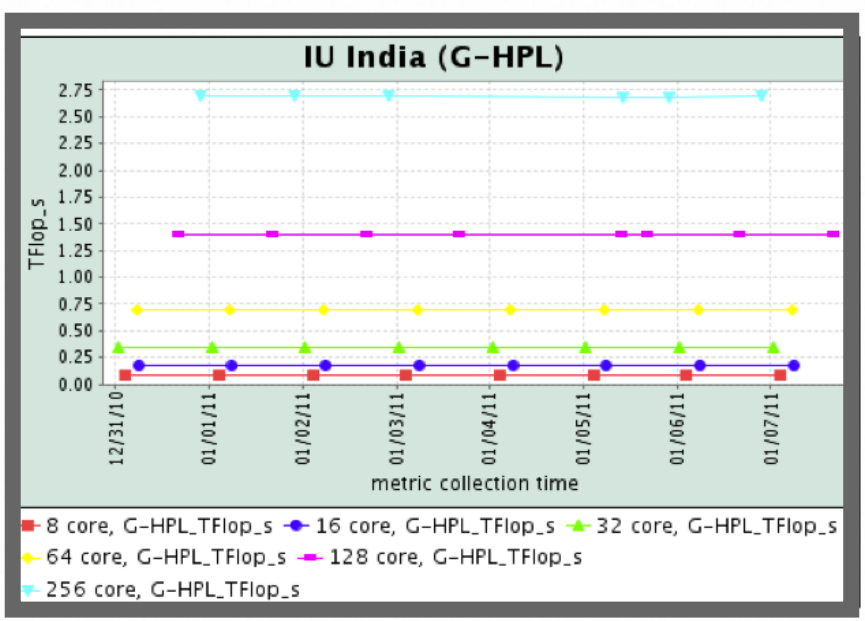
Status of basic cloud tests

iu-india		
hpcc8	EP-STREAM_Triad_GB_s	3.744
	G-STREAM_Triad_GB_s	29.954
	Wall_Mins	13.033
	EP-DGEMM_GFlop_s	12.116
	G-HPL_TFlop_s	0.090
	Random_Ring_Latency_usecs	0.641
	ProcSpeed_GHz	2.93
	Cores	8.000
	HPL_percent	96.293
	G-Random_Access_GFlop_s	0.120
	G-PTRANS_GB_s	3.314
	Random_Ring_Bandwidth_GB_s	1.055
	G-FFTE_Gup_s	6.661

Statistics displayed from HPCC performance measurement

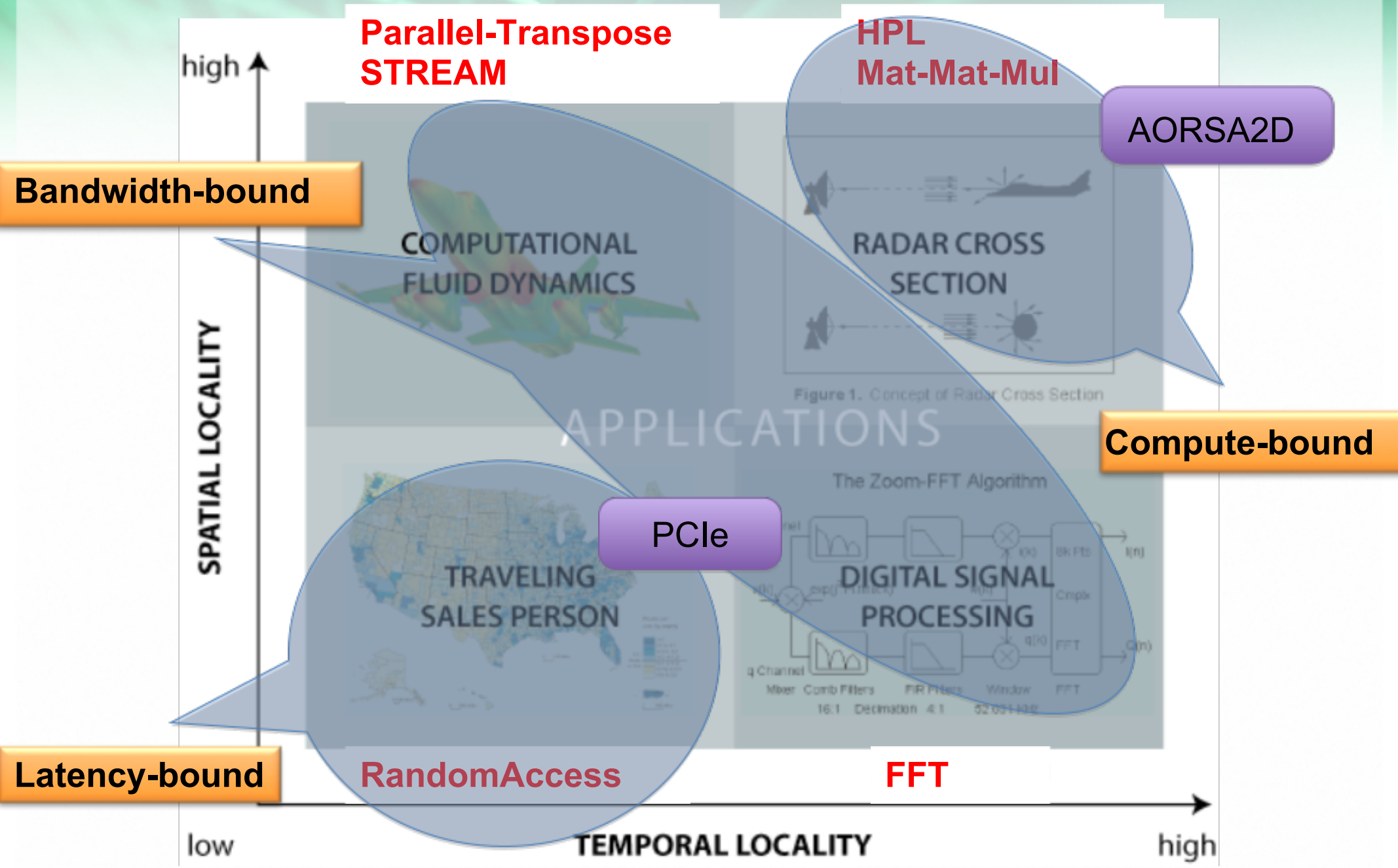


Information on machine partitioning



History of HPCC performance

Grid Benchmark Challenge: Feature Space

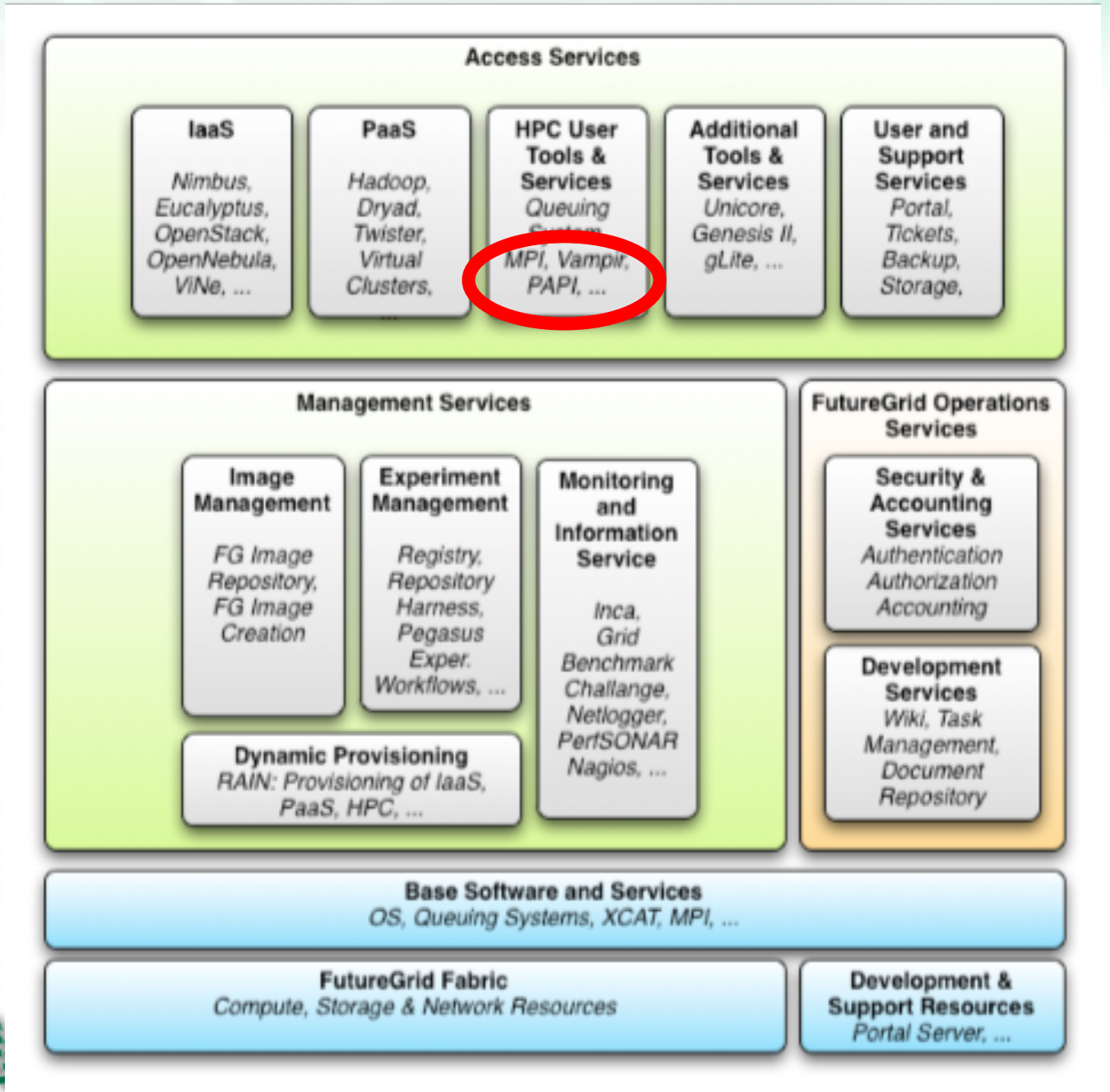


HPC Performance Tools

Presenters:

Shava Smallen (SDSC)
Piotr Luszczek (UTK)

9 minutes



Performance Tools Summary

Requirements

- Help users analyze the behavior of their application
- Re-use existing tools

Use cases

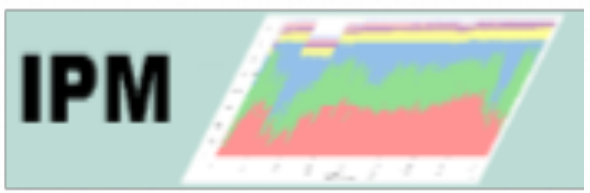
- What is the performance of my application on different machines?
- What is the performance of my application using different compiler optimizations?
- What is the I/O performance of my application using different file systems?
- What is the performance of my application on a physical machine and in a cloud?
- What is the performance of my application on different clouds?

Performance Tools Summary

Design

- Provide full support of partner tools

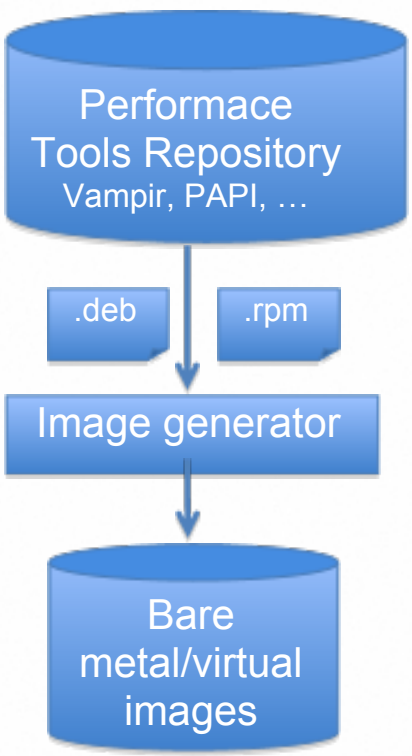
- Provide best effort support of external tools



Performance Tools Summary

Implementation

- Deploy to bare-metal and virtual machine thru image generation process



Deployment

Key:

Currently Deployed
Planned Deployment

IU
- india — VampirServer, VampirTrace
- xray — VampirTrace, PAPI

- alamo — TACC — VampirTrace, PAPI
- sierra — UCSD — VampirTrace, PAPI
- hotel — UC — VampirTrace, PAPI

Performance Tools Summary

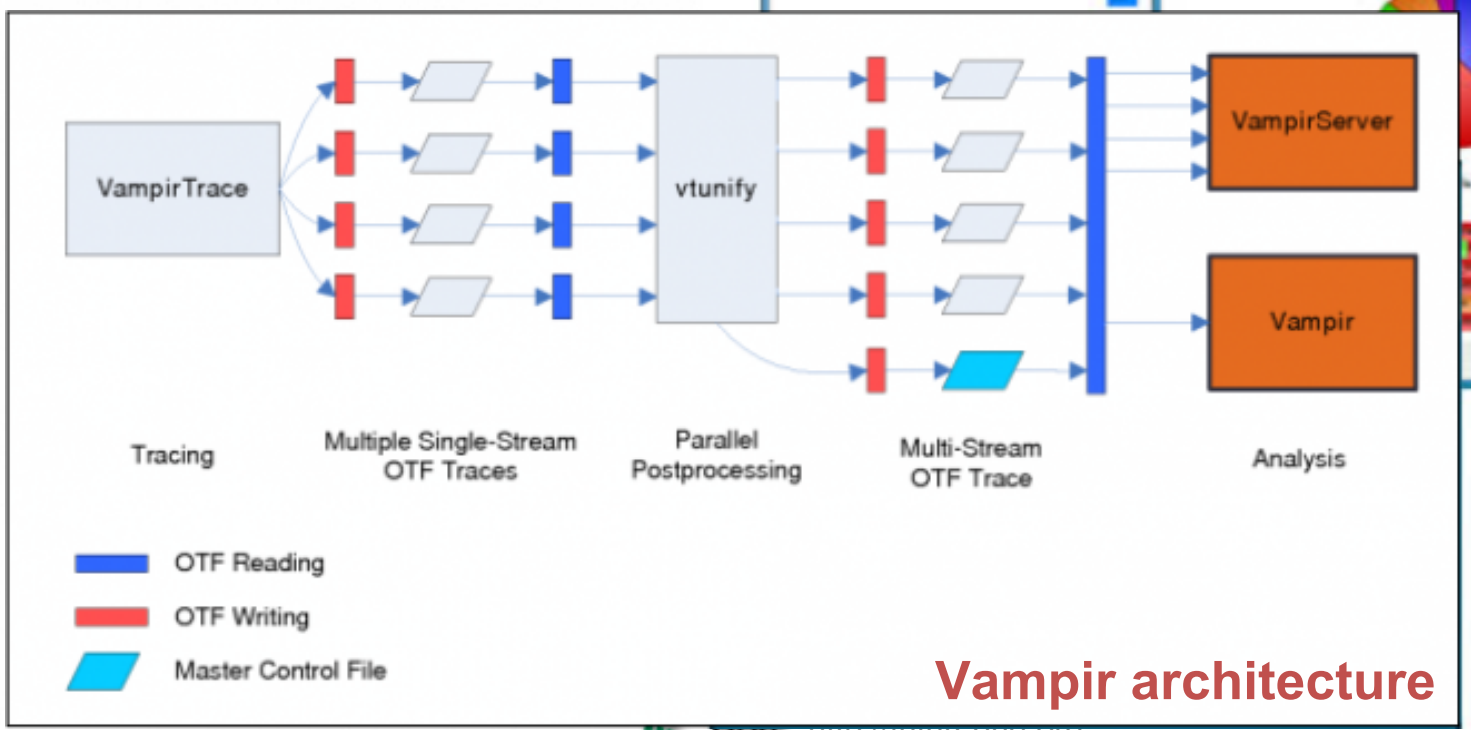
Milestones

- **Year 1** (completed)
  - **Q1**: PAPI installed as part of default Cray environment on Xray
  - **Q1/Q2**: Architecture document completed (performance architecture)
  - **Q2**: Vampir workshop at IU
  - **Q3**: Script written to automate installation of Vampir, Marmot, and Scalasca
  - **Q4**: Vampir deployed to India and Xray; Vampir documentation written
- **Year 2**
  - **Q1**: VampirTrace deployed to Hotel; PAPI documentation written, Vampir and PAPI tests deployed to Inca (complete)
  - **Q2-Q4:** Integrate performance tools into image generation, add step-by-step user tutorials for PAPI and Vampir

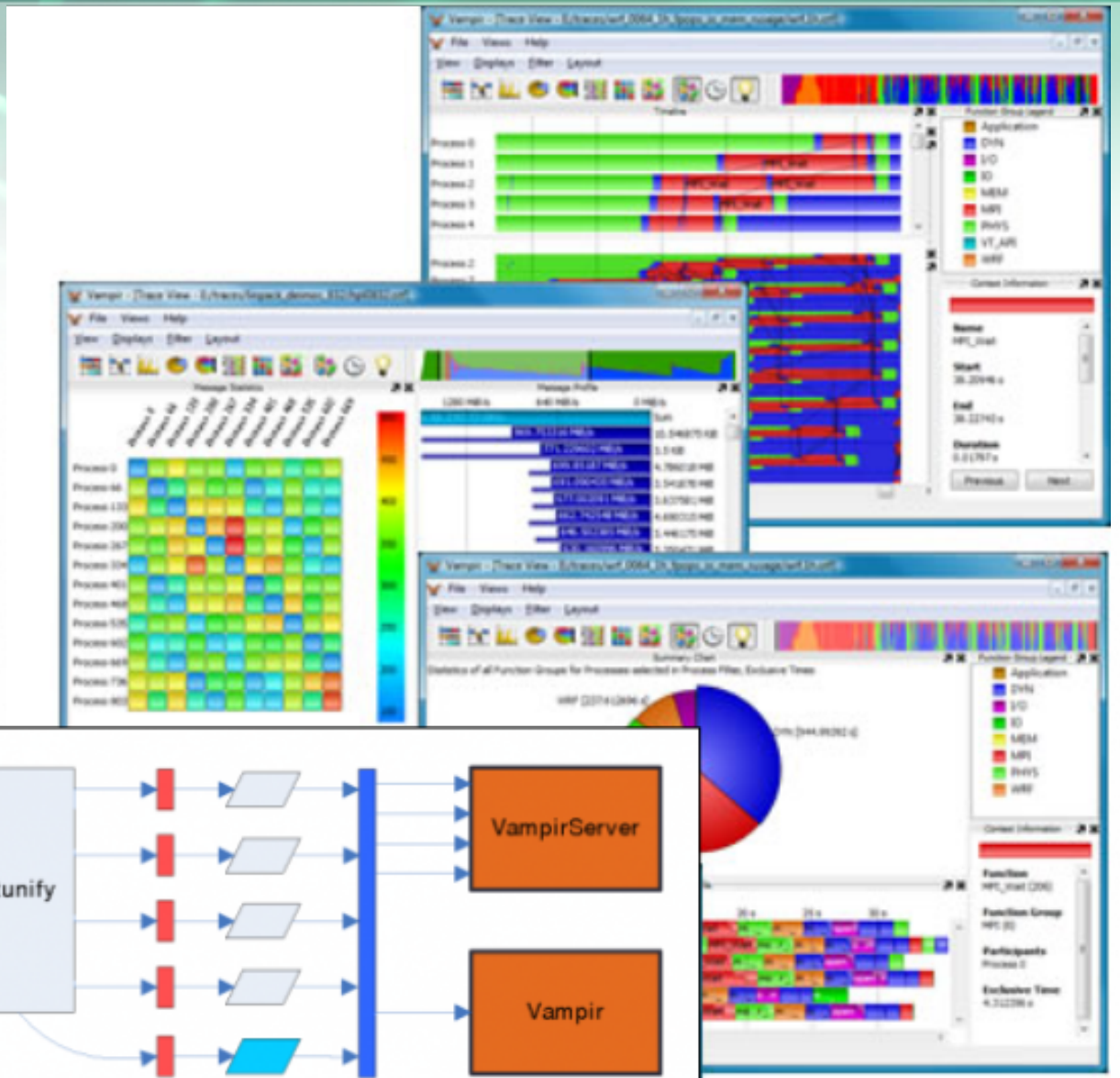
Risks

- Deployment dependent on image generator and Redhat 6 deployment

Vampir (TU-D)



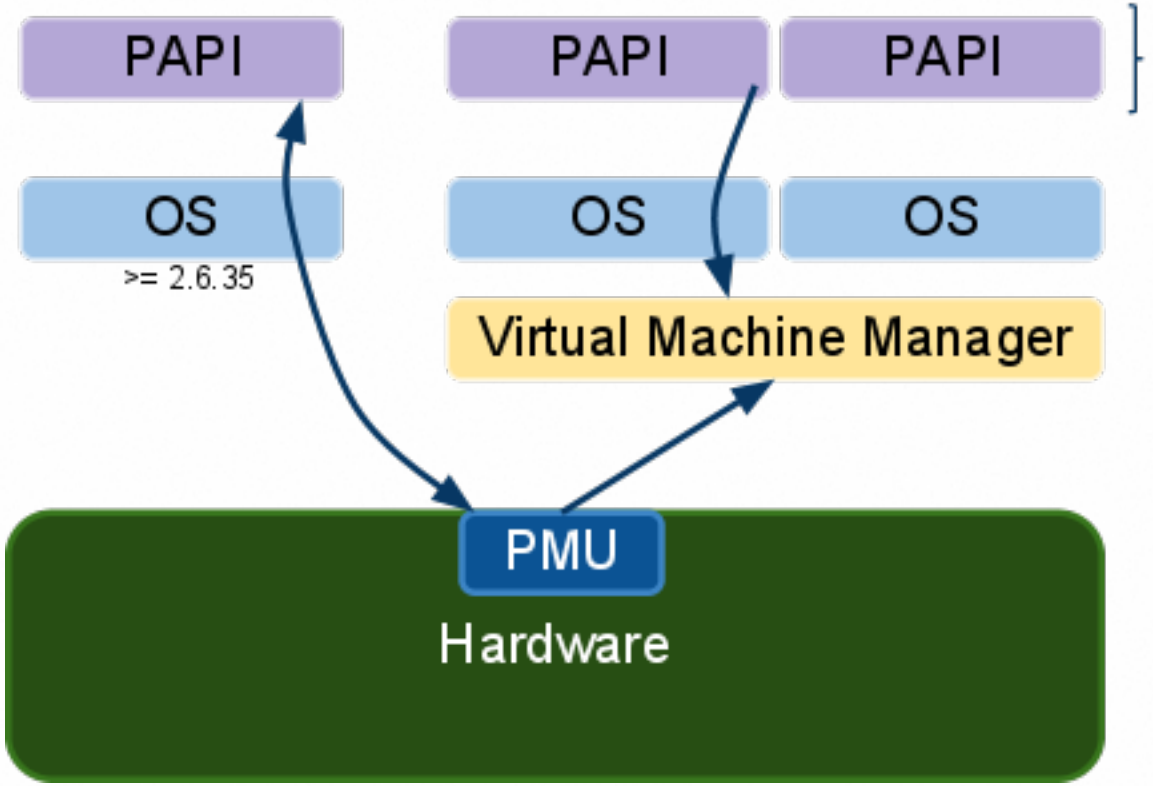
Vampir architecture



Vampir GUI screenshots

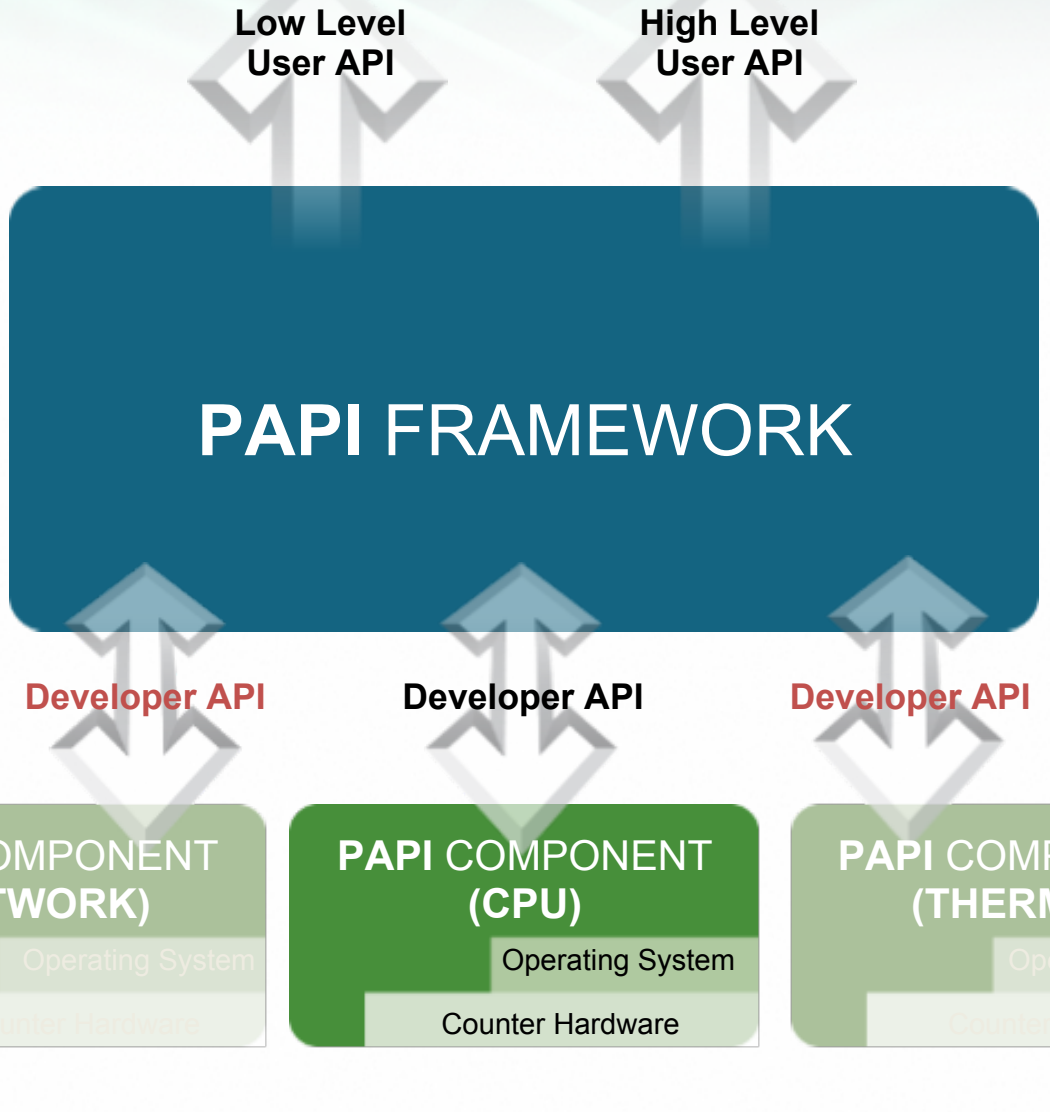
PAPI in Virtualized Environment

No PAPI support in any VMM ()



PAPI-V as an FG user project (separate proposal under review)

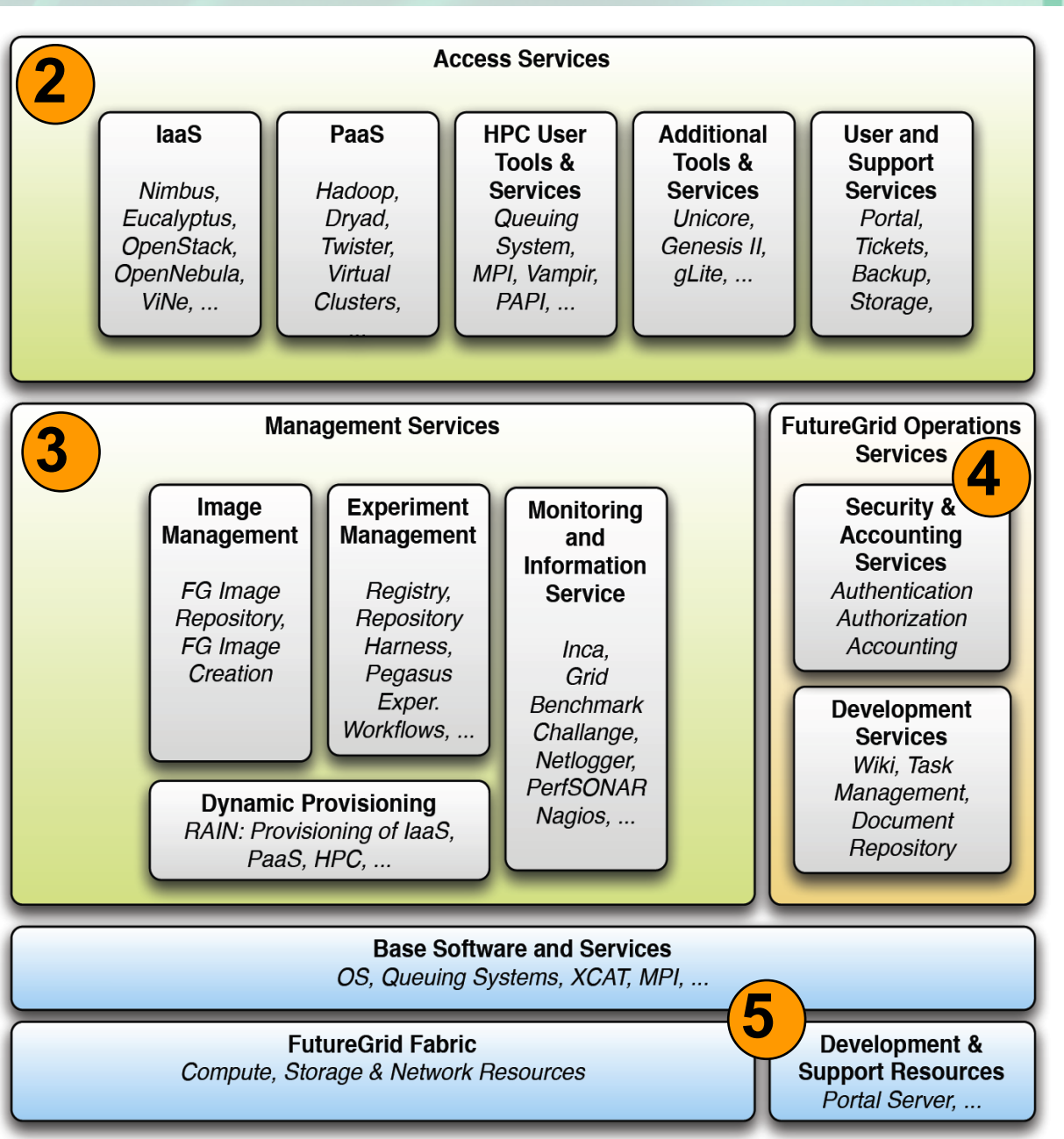
Component **PAPI**



Outline

1. Overview
2. Access Services
3. Management Services
4. **Operations Services**

5. *We will not much go into:*
   - *Base Software and Services*
   - *Fabric*
   - *Software for Development & Support Resources*



Security and Account Management

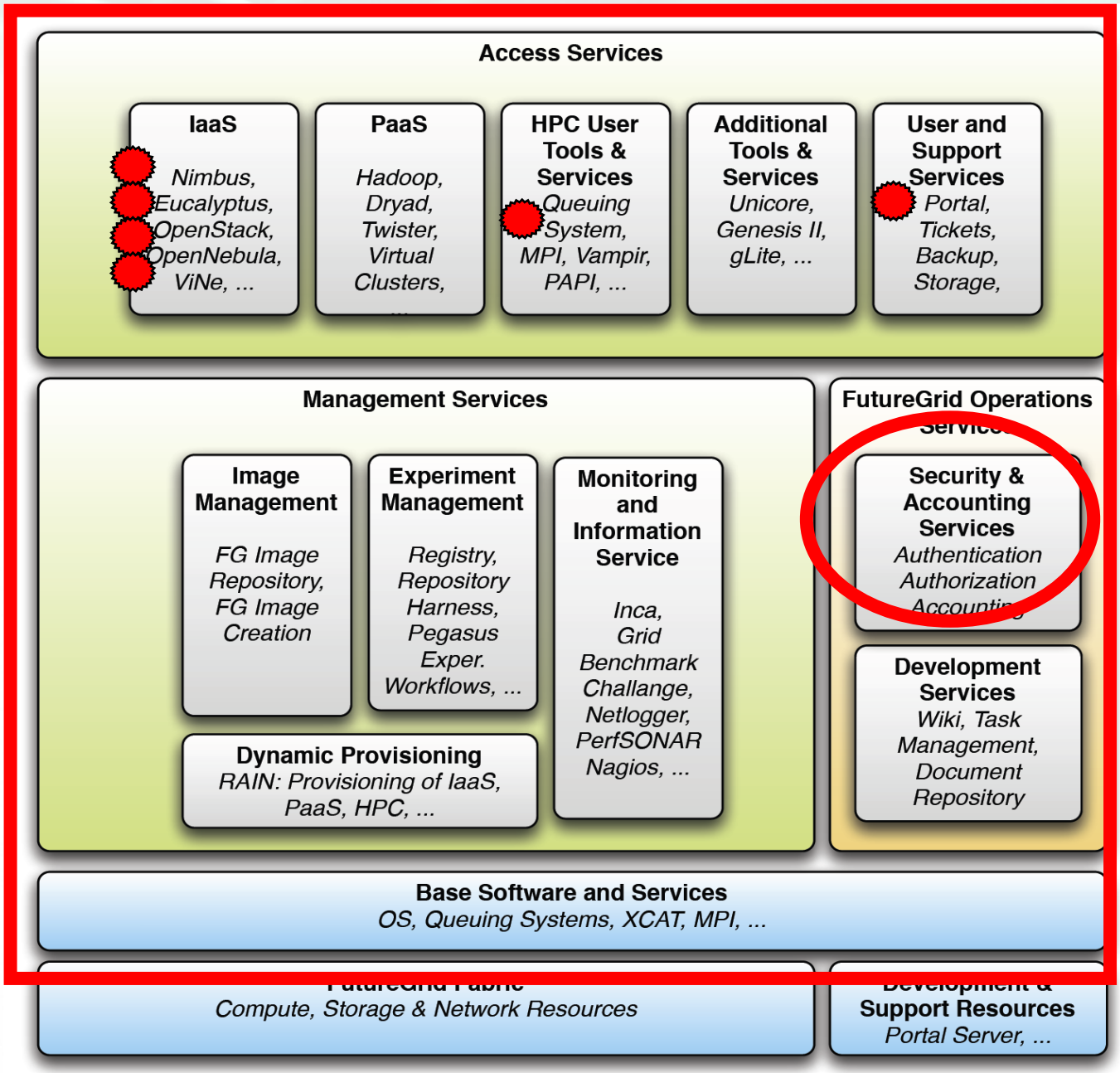
Gregor von Laszewski, Gregory Pike, Archit Kulshrestha, Fugang Wang, David LaBissoniere
Indiana University
University of Chicago

Presenter:

- Gregor von Laszewski
- 6 minutes

Key Points

- Use LDAP to achieve a centralized account management framework, though the deployment could be based on replica.
- Account management through command line and portal.
- Configuration management system like BCFG2 is used to set access control on images/provisioned systems.



Security & Account Management

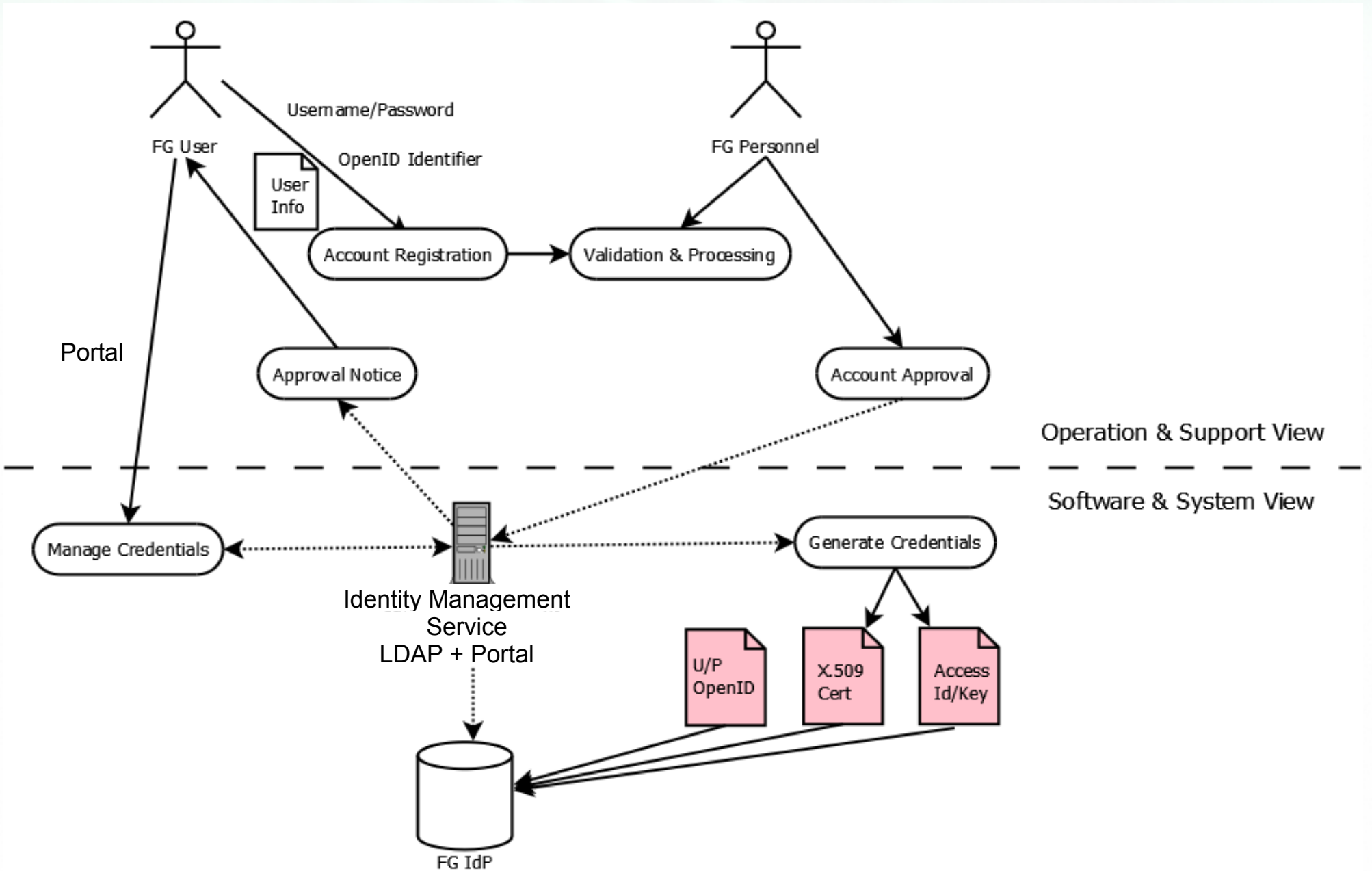
Use Cases

- Immediate acces to HPC, Nimbus, Eucalyptus
  - upon membership of an approved project
- Audit trail in case of security incident
- Introduction of a FG "credit card", e.g. accounting mechanism
- Key Management and Revocation

Requirements

- Single Sign On
  - Except for isolated experimental systems
- OpenID integration
- Accounting (XD, ...)
- Auditing (XD TAS, ...)
- Integration with various Services: HPC, Nimbus, Eucalyptus, Unicore, gLite, Genesis II, ...
- Consider the security issues involved with Image Management
- Integration with XD (work with XD)
- Explore InCommon

Security Architecture



Implementation

- Unified account strategy
  - Initiated from Portal
  - Leverage Drupal security solutions
  - Leverage Web 2.0 security solutions, OpenID, OAuth, CILogon
- Use LDAP replication
  - SSL, PAM, SSH-LPK
- XD Integration
  - X.509 Auth, GSISSH
- Security will be integral part of
  - FG Project & Experiment Management
- Investigate other solutions
  - CROWD, Kerberos realm

Mitigation Strategy

- Consult with former NCSA security expert Von Welch (now at IU) to mitigate architecture level risks
- Interact with XD, once direction is clear
- Sandbox Testing of experimental services and software
  - Friendly user mode to identify issues
- Develop best practices based on experience
  - User input is crucial
- Educate users on security to prevent issues like password less keys
- Team includes systems manager and developers familiar with TeraGrid security

Milestones & Risks

PY1

- Distributed LDAP replicas with SSH key
- PAM integration
- Nimbus integration

PY2

- CROWD
- OpenNebula LDAP integration
- Eucalyptus: we hope for OpenID by Eucalyptus team
- SSO onto other services

PY3

- SSO for development services

Risks

- Software and services deployed on FG have different authentication and authorization mechanisms that complicates our solution.
- Distributed resources and authentication end points
- Hosting experimental services may be a risk
- We are a new environment, best practices are not available for FG like systems
- XD has not yet started, integration may be delayed

Software Roadmap

PY1:

- Enable general services: HPC, Nimbus, Eucalyptus
- Explore dynamic provisioning via queuing system
- Explore raining an environment (Hadoop)

PY2:

- Provide dynamic provisioning via queuing system
- Deploy initial version of fg-rain, fg-hadoop, ...
- Explore replication of experiments
- Allow users to contribute images for "raining"
- Deploy OpenNebula, OpenStack

PY3:

- Deploy reproducibility of experiments
- Deploy reproducibility of comparative studies

PY4:

- Harden software for distribution