Science gateways made easy: the In-VIGO approach

A. Matsunaga, M. Tsugawa, S. Adabala, R. Figueiredo, H. Lam and J. Fortes

Advanced Computing and Information Systems Laboratory
University of Florida

June 28, 2005 – GGF 14 – Chicago, IL, USA

Workshop: Science Gateways: Common Community Interfaces to Grid Resources
What is In-VIGO?

- Enables computational engineering and science In-Virtual Information Grid Organizations
Motivations and goals for science gateways

Motivations:
- Need middleware to hide complexity of dealing with cross-domain issues
  - From application developers
  - From end users
- While preserving security and privacy of data, codes and other users’ information

Goals:
- Application-centric:
  - Support unmodified applications
    - Sequential, parallel
    - Batch, interactive
    - Open-source, commercial
  - Support automated Grid-enabling of applications
- User-centric: support Grid-unaware users
What does In-VIGO bring to science gateways?

- Enables **simple, flexible and sustainable** addition of computational applications and computing power
  - nanoHUB: ~20 tools in 3 months
  - GUI and batched applications
In-VIGO Approach
In-VIGO Approach

- Extensive use of virtualization
  - Security
  - Flexibility/customization
- Decouples grid users from resources
  - Users do not have to manage several credentials
  - In-VIGO proxies on behalf of user(s) simplify resource administration for providers
- Provides execution environments transparently
- Turns tools into Grid-services accessible via a user-friendly Web-interface efficiently
Grid-enabling unmodified applications

- **Enabler provides**
  - Command-line syntax
  - Application-related labels
    - Parameter(s), type-set values, entire applications
  - Resource and execution environment metadata
    - Architecture, OS libraries, environment variables

- **Grid-services are created, deployed and possibly customized using**
  - Generic Application Service (GAP)
  - Virtual Application Service (VAS)

- **Grid-user interacts with the virtual application through a Web-portal to execute applications on virtualized resources**

---

<table>
<thead>
<tr>
<th>Portal Tier</th>
<th>Virtual Application Tier</th>
<th>Virtual Grid Tier</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>VA 1 Service</td>
<td>VM Service</td>
</tr>
<tr>
<td>Administrator</td>
<td>VA 2 Service</td>
<td>VFS Service</td>
</tr>
<tr>
<td>Enabler</td>
<td>VA 3 Service</td>
<td>VN Service</td>
</tr>
<tr>
<td></td>
<td>Customization Service</td>
<td>IS Service</td>
</tr>
<tr>
<td></td>
<td>GAP/VAS Generator Service</td>
<td></td>
</tr>
</tbody>
</table>

**Legend**
- VA Framework
- Other Frameworks
- Enabling
- Customization and Generation
- Utilization
Virtual Machine System

- Provides means to efficiently create/configure/destroy VMs, that is generic across VM technologies [SC 2004]

- Directed Acyclic Graph (DAG) model for defining application-centric VMs

- Cost-bidding model for choosing compute servers for VM instantiation
Logical User Accounts

- **Traditional user account**
  - User is assigned fixed identifier (e.g. Unix UID)

- **Logical user account**
  - Physical (shadow) account temporarily assigned to a user by resource management in In-VIGO

- [HCW01, HPDC01]
Grid Virtual File System (GVFS)

  - Virtualization on Network File System (NFS)
  - User-level proxy based implementation

Support for

- on-demand, cross-domain data access
- unmodified binary applications
- unmodified NFS clients/servers
Putting it all together: GUI Application example

1: user request
2: query (user data, compute server)
3: get VM or shadow account
4: copy/access user data
5: return handler to user (URL)
6: VNC X-window

Role-Based Access Control enables Single Sign-On to VNC X-window

Isolation via shadow accounts or virtual machines
In-VIGO Status

- Portals
  - SCOOP, HPC, Netcare, nanoHUB
- Publications
  - IEEE Proc., FGCS, DCS, HPDC, SC, IPDPS, ICAC, Europar, GGF, …
- Classes
  - Distributed Computing
  - Virtual Computers
- Team: 10 Ph.D. students, 4 Faculty