Provisioning for Dynamic Instantiation of Community Services

*Summary by:* Ben Rockstroh

August 31, 2009
Provisioning for Dynamic Instantiation of Community Services

Provisioning Approaches

Provenance in Agent-Mediated Healthcare Systems

Provenance Life Cycle

Example Project

What is Gained

Problems to Overcome

Combining the Power of Taverna and caGrid

What is Taverna

What is caGrid

Example Workflow

Provisioning for Dynamic Instantiation of Community Services

Provisioning Approaches
Provisioning for Dynamic Instantiation of Community Services

- **Service-level** - When a service is deployed or undeployed no other service in the container is effected.
- **Container-level** - requires reloading entire container, bring all services down.
- **Node-level** - provides a dependency map for the provisioning center. Thus allowing dynamic allocation of resources according to a propagation policy.
- **VO-level** - Provide infrastructure for service instantiation to other VOs, along with security and consistency.
Provisioning Approaches

▶ Service-level - When a service is deployed or undeployed no other service in the container is effected.
▶ Container-level - requires reloading entire container, bring all services down.
▶ Node-level - provides a dependency map for the provisioning center. Thus allowing dynamic allocation of resources according to a propagation policy.
▶ VO-level - Provide infrastructure for service instantiation to other VOs, along with security and consistency.
System Architecture

(a) VO1: Application 1, Application 2, Application n
(b) VO2: Application 1, Application 2, Application k

QoS-enabled requirement: Node-level, VO-level

Provisioning center
- Dependency analyzer
- Execution planner
- Provisioning repository

Provisioning agent (Workflow engine)

Container-level: Dynamic deploy infrastructure

Service-level: WSRF-enabled Grid Container

Configuration
- Resource properties
- WSDL, WSDD
- Usage metrics
- Local SLA
- Security: Trust users, Trust hosts

Provisioning Targets
1. Web services
2. Legacy programs
3. Operating system
4. Storage
5. Network

Physical resources
- Deploy infrastructure
- GTWS core

planExecution is to analyze and decide the provisioning level according to the specific situation

1. instantiate($S_1$)
2. exec($S_1$)
3. VO monitor lifecycle
4. destroy($S_1$)
5. destroy(ITS$_1$)
6. undeploy
7. construct($S_1$)
8. create
9. repeat
Speedup

(a) HUST-VO only, (b) two VOs
Provenance Life Cycle

**Provenance:** Place of origin or earliest known history. "Lineage"

### 4 Phases of Life Cycle

- Actors create p-assertions that represent their involvement in computation.
- Store the p-assertions in a provenance store.
- Allow users or applications to query results in the provenance store.
- Users manage the provenance store and its contents, subscriptions, and location of content.

**Goal:** To keep an audit trail for records and optimization.
Adapt OTMA for provenance awareness

**OTMA**: Organ Transplant Management Application
Provide assertions to link events carried out by separate agents and services.

- Types of p-assertions used:
  - interaction - Actor asserts the contents of the message.
  - relationship - Actor documents how they obtained the message.
  - actor state - communication between 2 actors (doctors)
Prior treatments are transparent to doctors.
Prior conditions are known so health of organs can be assessed.
Security and Privacy
- Linking 2 doctors without referrals and patient cooperation.
What is Taverna?

- Scientific workflow workbench for designing and executing workflows for my Grid.
- http://taverna.sourceforge.net/
What is caGrid?

- A collection of Services hosted by different universities/organizations.
- Designed to provide different tools for research.
- Message passing via SOAP
- Created by US National Cancer Institute’s Biomedical Information Grid program

*Example Service:* Analyze an array representing gene expression levels. i.e. *microarray*
Example Workflow

(a) VO1
- Application 1
- Application 2
- Application n

(b) VO2
- Application 1
- Application 2
- Application k

QoS-enabled requirement

Node-level
- VO monitor
- Dependency analyzer

VO-level
- Authorization
- Execution planner

Provisioning center
- Provisioning repository

Provisioning agent (Workflow engine)

Container-level
- Dynamic deploy infrastructure

Service-level
- WSRF-enabled Grid Container

Configuration
- Resource properties
- WSDL, WSDD
- Usage metrics
- Local SLA

Security:
- Trust users
- Trust hosts

Provisioning Targets
1. Web services
2. Legacy programs
3. Operating system
4. Storage
5. Network

Physical Layer
- Deploy infrastructure
- GTI compute

Plan Execution is to analyze and decide the provisioning level according to the specific situation.

1. Instantiate(Si)
   - checkAuth
   - checkDependency
   - planExecution
   - provisioning (TISd)

2. exec(Si)
   - provisioning
   - destroy(TISd)
   - repeat
   - deploy
   - create

3. VO monitor lifecycle
   - VO monitor
   - destroy
   - undeploy
Websites

http://www.mygrid.org.uk/
http://taverna.sourceforge.net/
Provisioning Approaches

- Service-level - When a service is deployed or undeployed no other service in the container is effected.
- Container-level - requires reloading entire container, bring all services down.
- Node-level - provides a dependency map for the provisioning center. Thus allowing dynamic allocation of resources according to a propagation policy.
- VO-level - Provide infrastructure for service instantiation to other VOs, along with security and consistency.