NBCR NATIONAL BIOMEDICAL COMPUTATION RESOURCE
Conduct, catalyze and enable multiscale biomedical research

Opal: Simple Web Services Wrappers for Scientific Applications

Sriram Krishnan*, Brent Stearn, Karan Bhatia, Kim K. Baldridge, Wilfred W. Li, Peter Arzberger

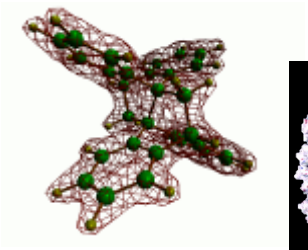
*sriram@sdsc.edu

ICWS 2006 - Sept 21, 2006

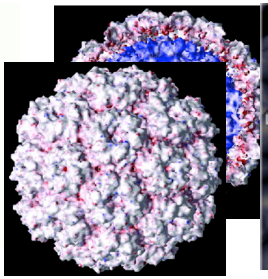
CRBS SDSC it² UCSD NATIONAL INSTITUTES OF HEALTH

NBCR Computational Infrastructure

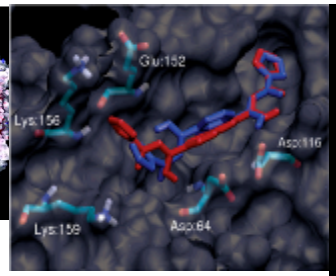
Set of Biomedical Applications



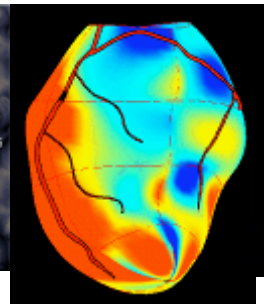
QMView
GAMESS



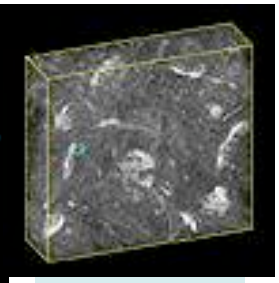
APBS



Autodock



Continuity

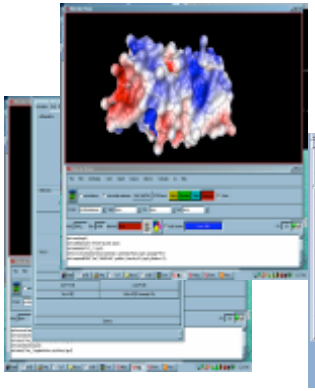


Gtomo2
TxBR

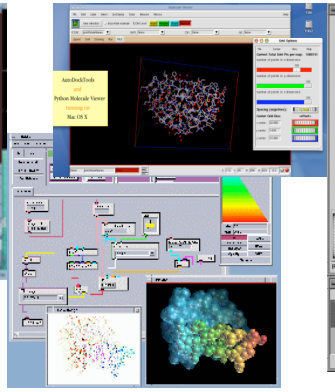
Resources

Computational Grid

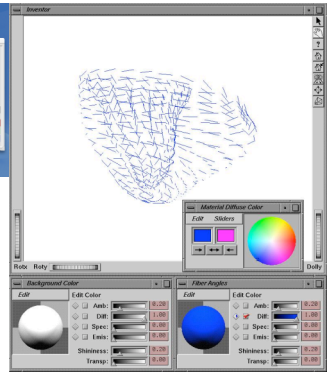
Rich Clients



APBSCommand

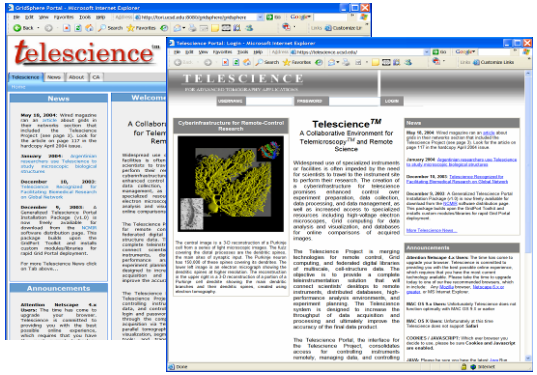


PMV ADT
Vision



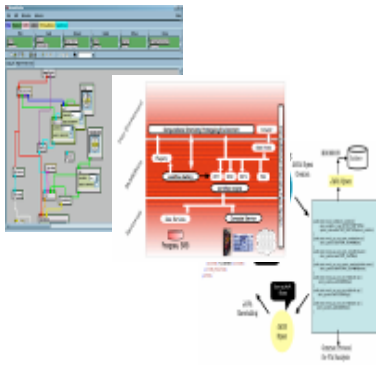
Continuity

Web Portals



Telescience Portal

Web Services



Workflow
Middleware

Requirements

- Enable access to scientific applications on Grid resources
  - Seamlessly via a number of user interfaces
  - Easily from the perspective of a scientific user
- Enable the creation of scientific workflows
  - Possibly with the use of commodity workflow toolkits

Challenges for the Scientific User

- Access to Grid resources is still very complicated
  - User account creation
  - Management of credentials
  - Installation and deployment of scientific software
  - Interaction with Grid schedulers
  - Data management

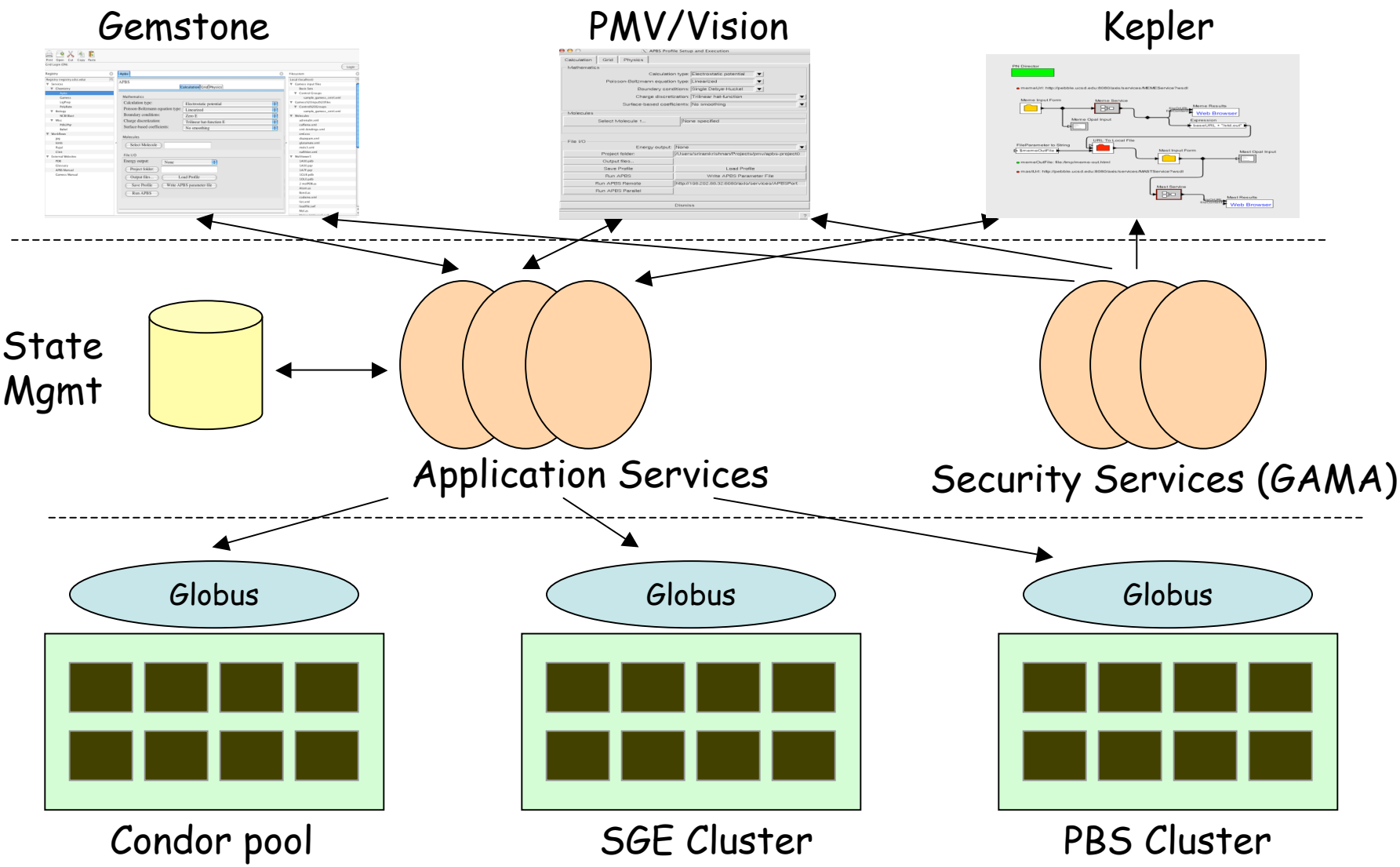
Towards Services Oriented Architectures (SOA)

- Scientific applications wrapped as Web services
  - Provision of a SOAP API for programmatic access
- Clients interact with application Web services, instead of Grid resources
  - Used in practice by several scientific communities
    - NBCR: http://nbcr.net
    - GEON: http://geongrid.org
    - GLEON: http://gleon.org/
    - CAMERA: http://camera.calit2.net

Talk Outline

- Motivation for a Services Oriented Architecture (SOA)
- Overall end-to-end architecture
- The Opal Toolkit: Introduction & technical details
- Some use cases
- Concluding remarks

Architecture Overview



Scientific SOA: Benefits

- Applications are installed once, and used by all authorized users
  - No need to create accounts for all Grid users
  - Use of standards-based Grid security mechanisms
- Users are shielded from the complexities of Grid schedulers
- Data management for multiple concurrent job runs performed automatically by the Web service
- State management and persistence for long running jobs
- Accessibility via a multitude of clients

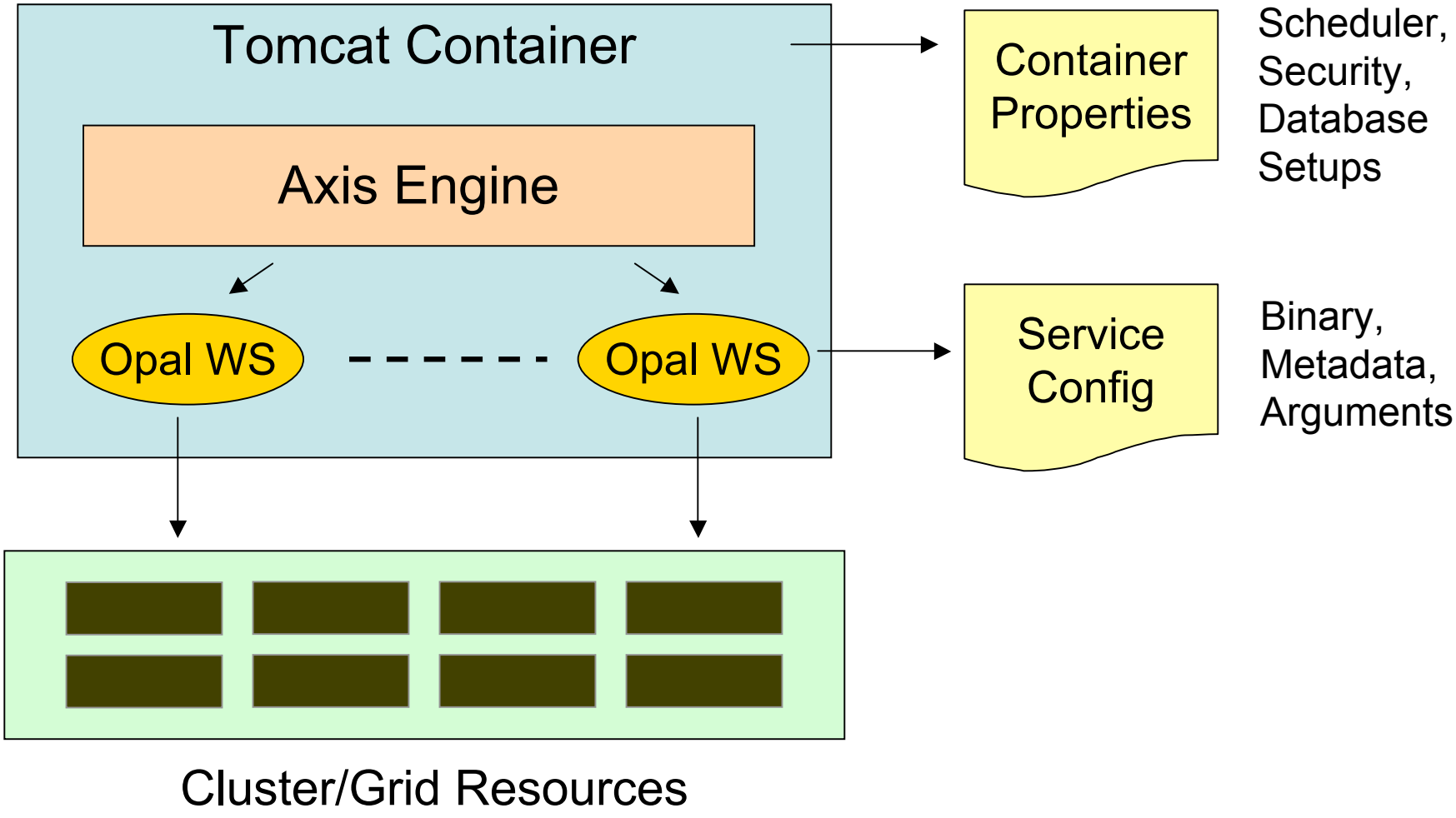
Possible Approaches

- Write application services by hand
  - Pros: More flexible implementations, stronger data typing via custom XML schemas
  - Cons: Not generic, need to write one wrapper per application
- Use a Web services wrapper toolkit, such as Opal
  - Pros: Generic, rapid deployment of new services
  - Cons: Less flexible implementation, weak data typing due to use of generic XML schemas

The Opal Toolkit: Overview

- Enables rapid deployment of scientific applications as Web services (< 2 hours)
- Steps
  - Application writers create configuration file(s) for a scientific application
  - Deploy the application as a Web service using Opal's simple deployment mechanism (via Apache Ant)
  - Users can now access this application as a Web service via a unique URL

Opal Architecture



Implementation Details

- Implemented as a regular Axis service
  - Application behavior specified by the service configuration
  - Configuration passed as a parameter via the auto-generated deployment descriptor (WSDD)
- Possible to have multiple instances of the same class for different applications
  - Distinguished by a unique URL for every application
- No need to generate sources or WSDL prior to deployment

Sample Container Properties

```
# the base URL for the tomcat installation
# this is required since Java can't figure out the IP
# address if there are multiple network interfaces
tomcat.url=http://ws.nbcr.net:8080

# database information
database.use=false
database.url=jdbc:postgresql://localhost/app_db
database.user=<app_user>
database.passwd=<app_passwd>

# globus information
globus.use=true
globus.gatekeeper=ws.nbcr.net:2119/jobmanager-sge
globus.service_cert=/home/apbs_user/certs/apbs_service.cert.pem
globus.service_privkey=/home/apbs_user/certs/apbs_service.privkey

# parallel parameters
num.procs=16
mpi.run=/opt/mpich/gnu/bin/mpirun
```

Sample Application Configuration

```
<appConfig xmlns="http://nbcr.sdsc.edu/opal/types"
           xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <metadata>
    <usage><![CDATA[psize.py [opts] <filename>]]></usage>
    <info xsd:type="xsd:string">
       <![CDATA[
         --help              : Display this text
         --CFAC=<value>      : Factor by which to expand mol dims to
                               get coarse grid dims
                               [default = 1.7]
       ...
       ]]>
    </info>
  </metadata>
  <binaryLocation>/homes/apbs_user/bin/psize.py</binaryLocation>
  <defaultArgs>--GMEMCEIL=1000</defaultArgs>
  <parallel>false</parallel>
</appConfig>
```

Application Deployment & Undeployment

- To deploy onto a local Tomcat container:

```
ant -f build-opal.xml deploy -DserviceName=<serviceName>
-DappConfig=<appConfig.xml>
```

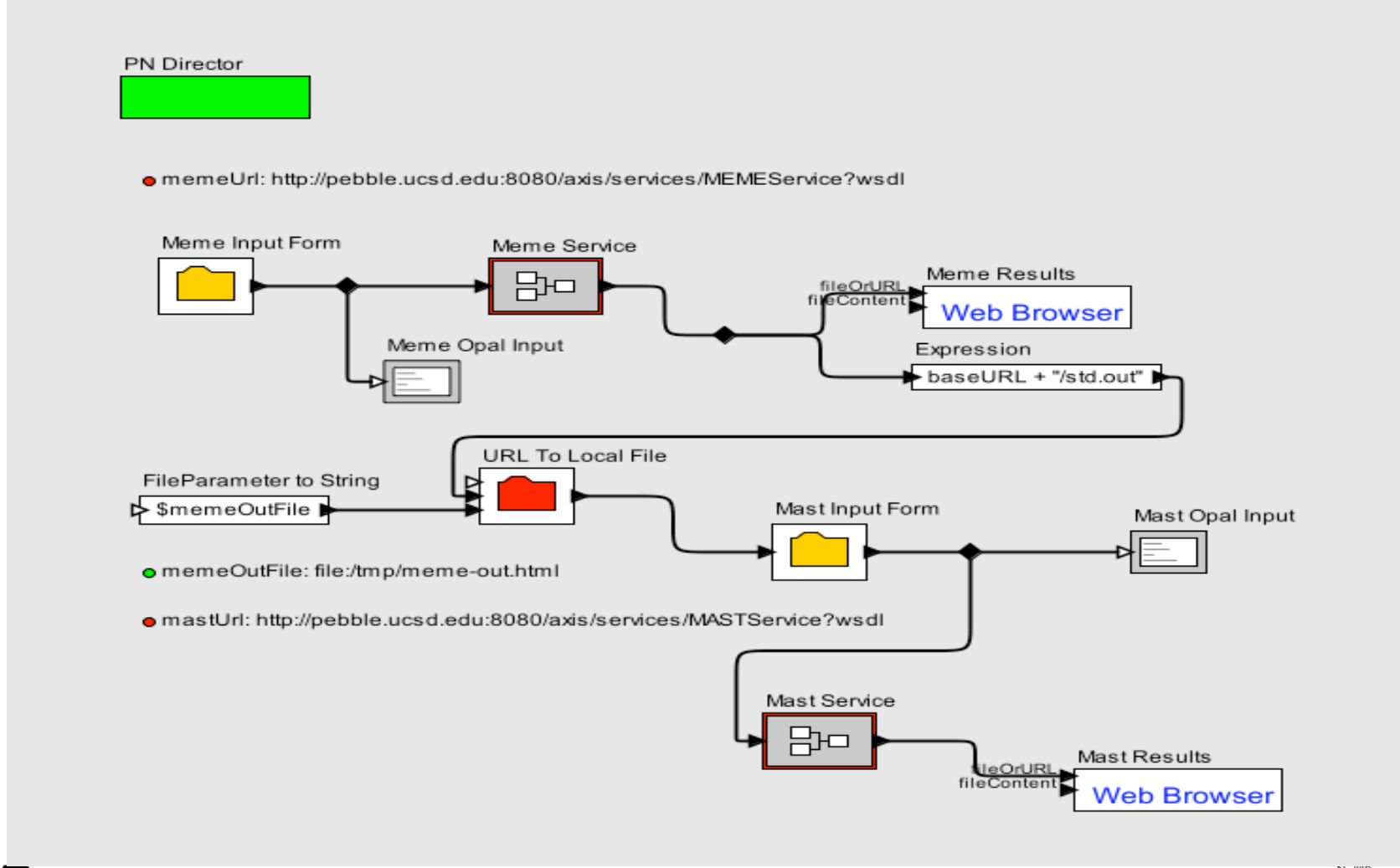
- To undeploy a service:

```
ant -f build-opal.xml undeploy -DserviceName=<serviceName>
```

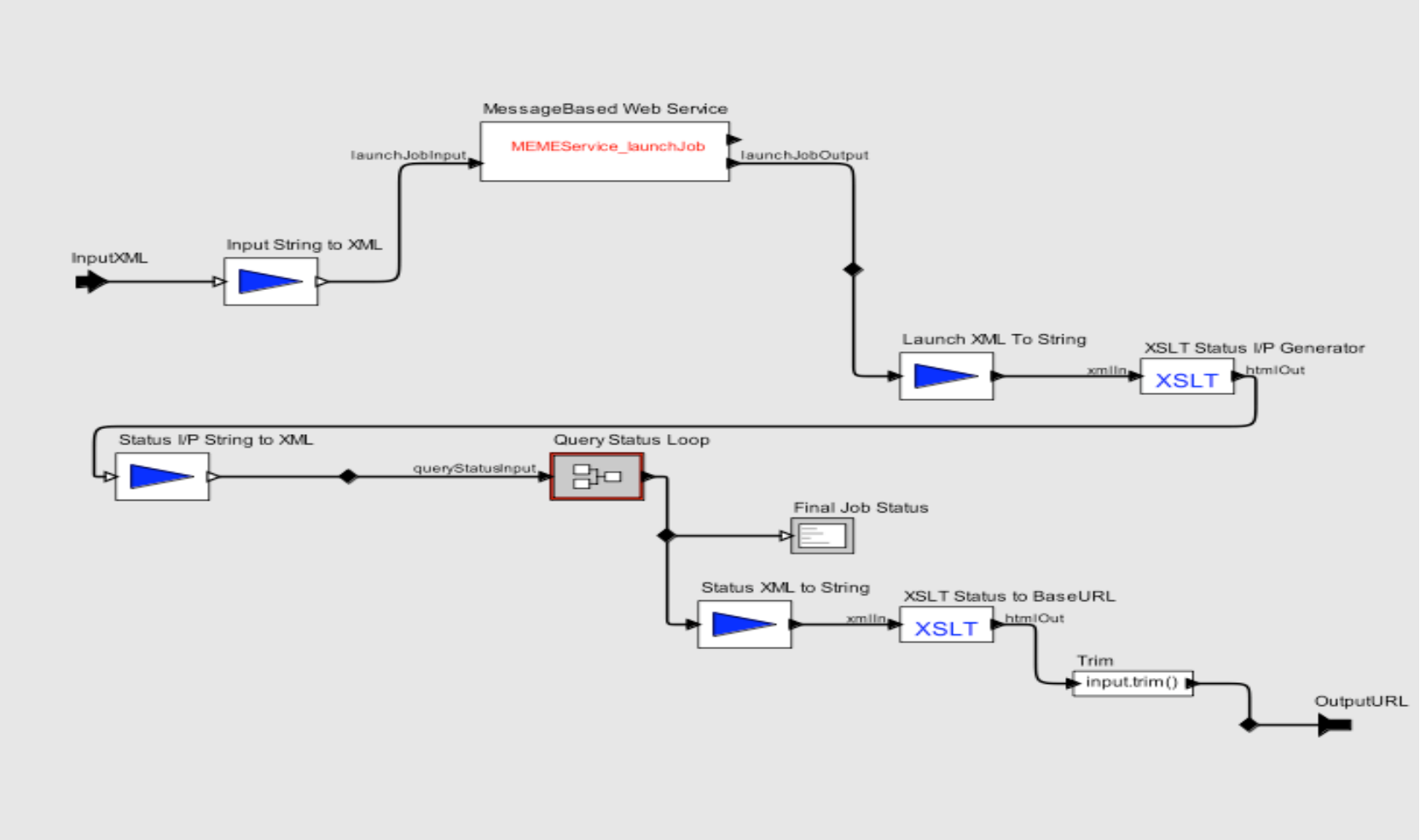
Service Operations

- Get application metadata: Returns metadata specified inside the application configuration
- Launch job: Accepts list of arguments and input files (Base64 encoded), launches the job, and returns a jobID
- Query job status: Returns status of running job using the jobID
- Get job outputs: Returns the locations of job outputs using the jobID
- Get output as Base64: Returns an output file in Base64 encoded form
- Destroy job: Uses the jobID to destroy a running job

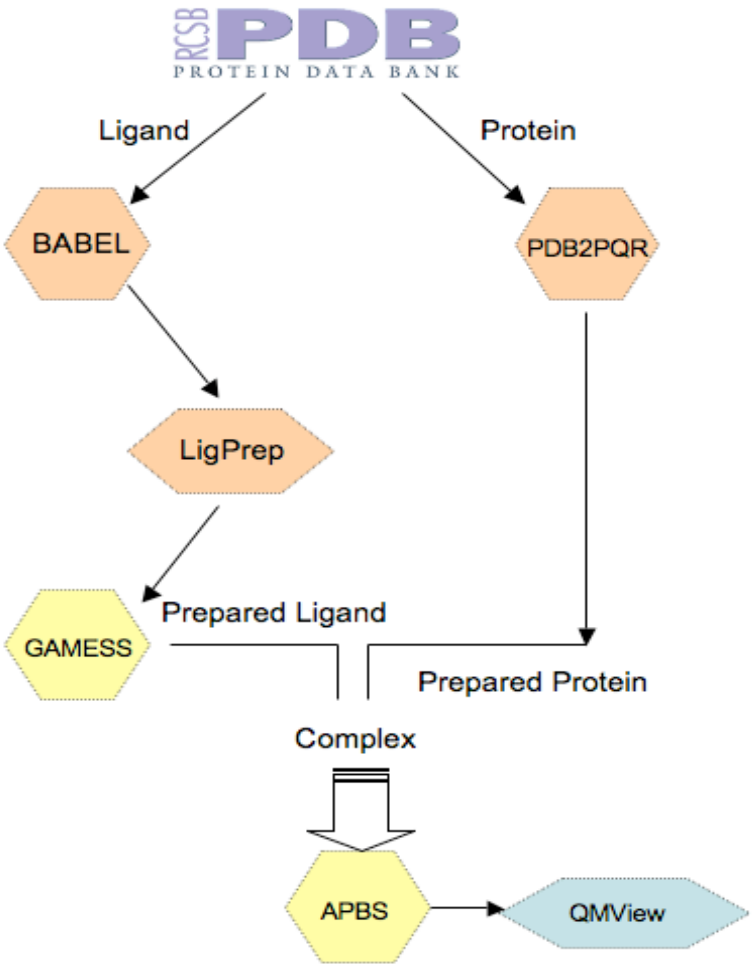
MEME+MAST Workflow using Kepler



Kepler Opal Web Services Actor



Ligand Protein Interaction Using Web Services



- Baldridge, Greenberg, Amoreira, Kondric
- GAMESS Service
  - More accurate Ligand Information
- LigPrep Service
  - Generation of Conformational Spaces
- PDB2PQR Service
  - Protein preparation
- APBS Service
  - Generation of electrostatic information
- QMView Service
  - Visualization of electrostatic potential file
- Applications:
  - Electrostatics and docking
  - High-throughput processing of ligand-protein interaction studies
  - Use of small molecules (ligands) to turn on or off a protein function

Conclusions

- We presented Opal, a toolkit for rapidly exposing legacy scientific applications as Web services
  - Provides features like Job management, Scheduling, Security, and Persistence, in an easy to use and configurable manner
  - Lowers inertia in the scientific community against the adoption of Web services and SOAs
  - Currently being used by the NBCR community for Grid-enabling several scientific applications, via a multitude of interfaces

Future Work

- WSRF Integration
  - State management using standard Grid mechanisms
  - Asynchronous status notifications via WS-Notification
- Meta-scheduling and resource monitoring
  - Use of CSF4 and GFarm
- Alternate mechanisms for I/O staging
  - GridFTP, RFT
- Addition of strong data typing for I/O
  - Use of XML schemas, DFDL

Thanks for listening!

- More information, downloads, documentation:
  - http://nbcr.net/services/
- Questions and comments welcome!
  - sriram@sdsc.edu

Appendix

Grid Computing is …

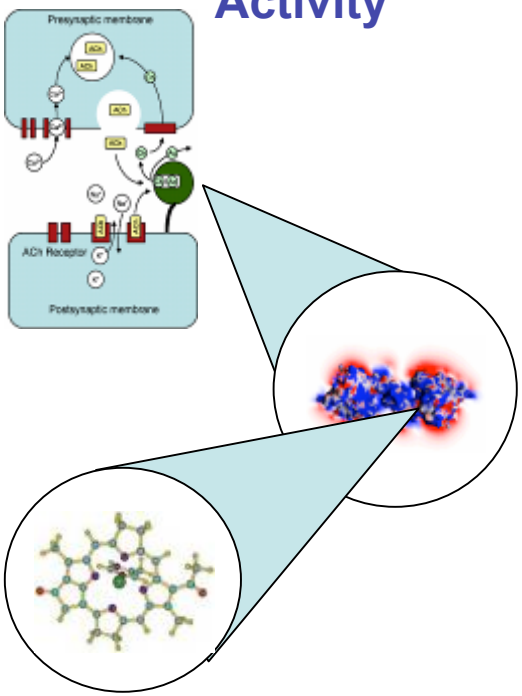
- "Co-ordinated resource sharing and problem solving in dynamic multi-institutional virtual organizations." [*Foster, Kesselman, Tuecke*]
  - Co-ordinated - multiple resources working in concert, eg. Disk & CPU, or instruments & database, etc.
  - Resources - compute cycles, databases, files, application services, instruments.
  - Problem solving - focus on solving scientific problems
  - Dynamic - environments that are changing in unpredictable ways
  - Virtual Organization - resources spanning multiple organizations and administrative domains, security domains, and technical domains

Grid Computing is … (Industry)

- "About finding distributed, underutilized compute resources (systems, desktops, storage) and provisioning those resources to users or applications requiring them." [*The Grid Report, Clabby Analytics*]
  - Distributed - all the resources laying around in departments or server rooms.
  - Underutilized - organizations save money by increasing utilization versus purchasing new resources.
  - Resources - servers and server cycles, applications, data resources
  - Provisioning - predict and schedule resource use depending on load.

Scientific Objective: Modeling and Analysis Across Scales

Modeling Synaptic Activity



Organisms

Organ

Tissue

Cell

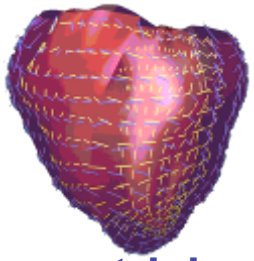
Subcellular

Macromolecular

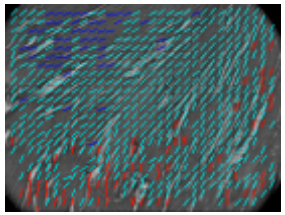
Molecule

Atom

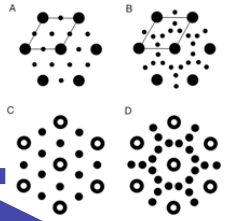
Modeling the Heart



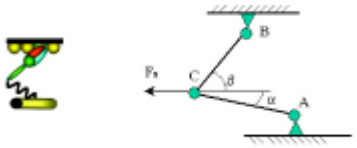
ventricles



multicellular



lattice



crossbridge

filament

Tools that Integrate Data, Construct Models and Perform Analysis across Scales