Modeling and Executing Heterogeneous Grid Workflows with JOpera for Eclipse

Cesare Pautasso, Thomas Heinis
Department of Computer Science, ETH Zurich, Switzerland
{pautasso, heinis}@inf.ethz.ch – www.jopera.org

Goal: Heterogeneous Grid Service Composition

1. Design a **simple** workflow **language** for rapid composition of Grid **services**
2. Build a user-friendly, efficient and **autonomic system** to support it
3. Ensure their independence from the actual mechanisms and protocols involved

   (there are lots of standards and they change all the time)

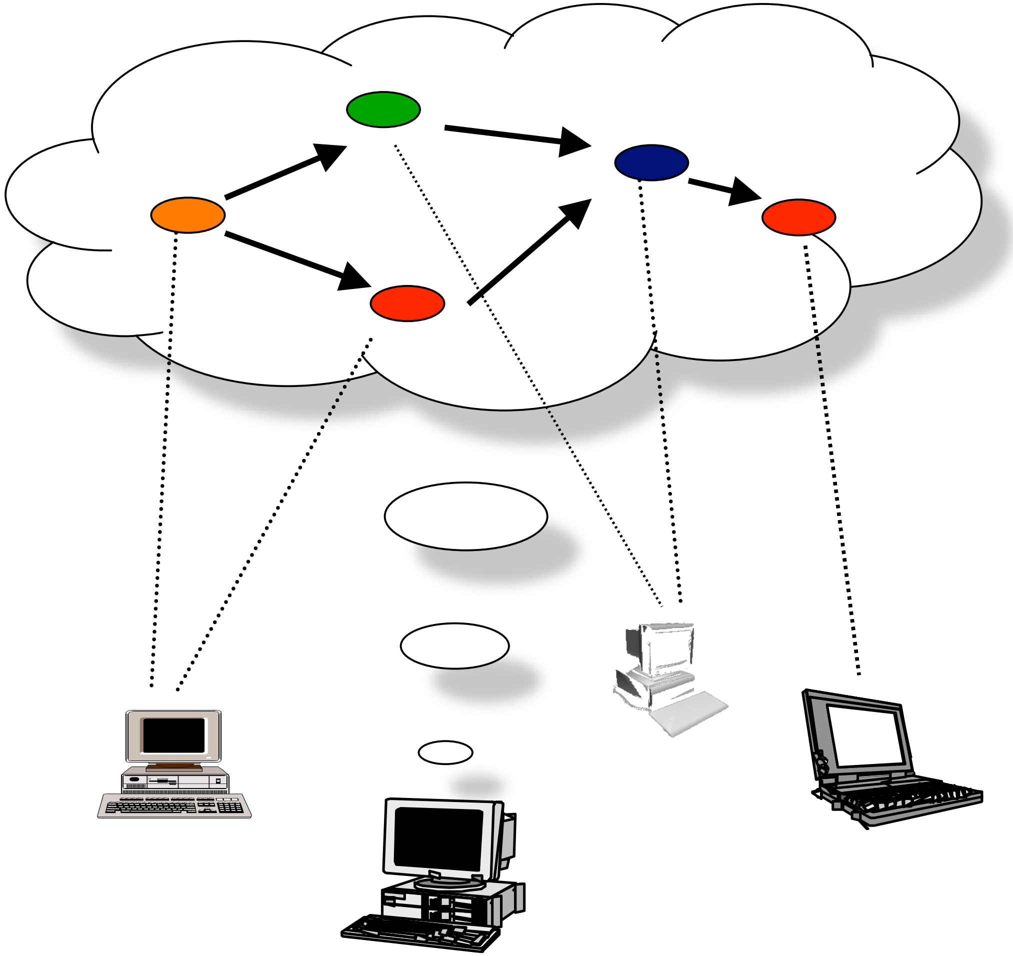
JOpera
Process Support for Web Services

About JOpera for Eclipse

1. **Modeling** service composition as workflow
   - Graph-based, functional workflow modeling language (Visual syntax, XML under the hood)
   - Workflows not limited to Web/Grid services
2. **Execution** of the workflow models
   - Pluggable task scheduling strategies (orthogonal)
   - Extensibility (Eclipse plug-ins to provide custom adapters for service invocation & publishing)
   - Distributed engine (on a cluster of computers)
   - Autonomic engine (self-healing, self-tuning)
   - Efficiency (optimizing compiler to Java bytecode)

[ICAC,ICWS2005]

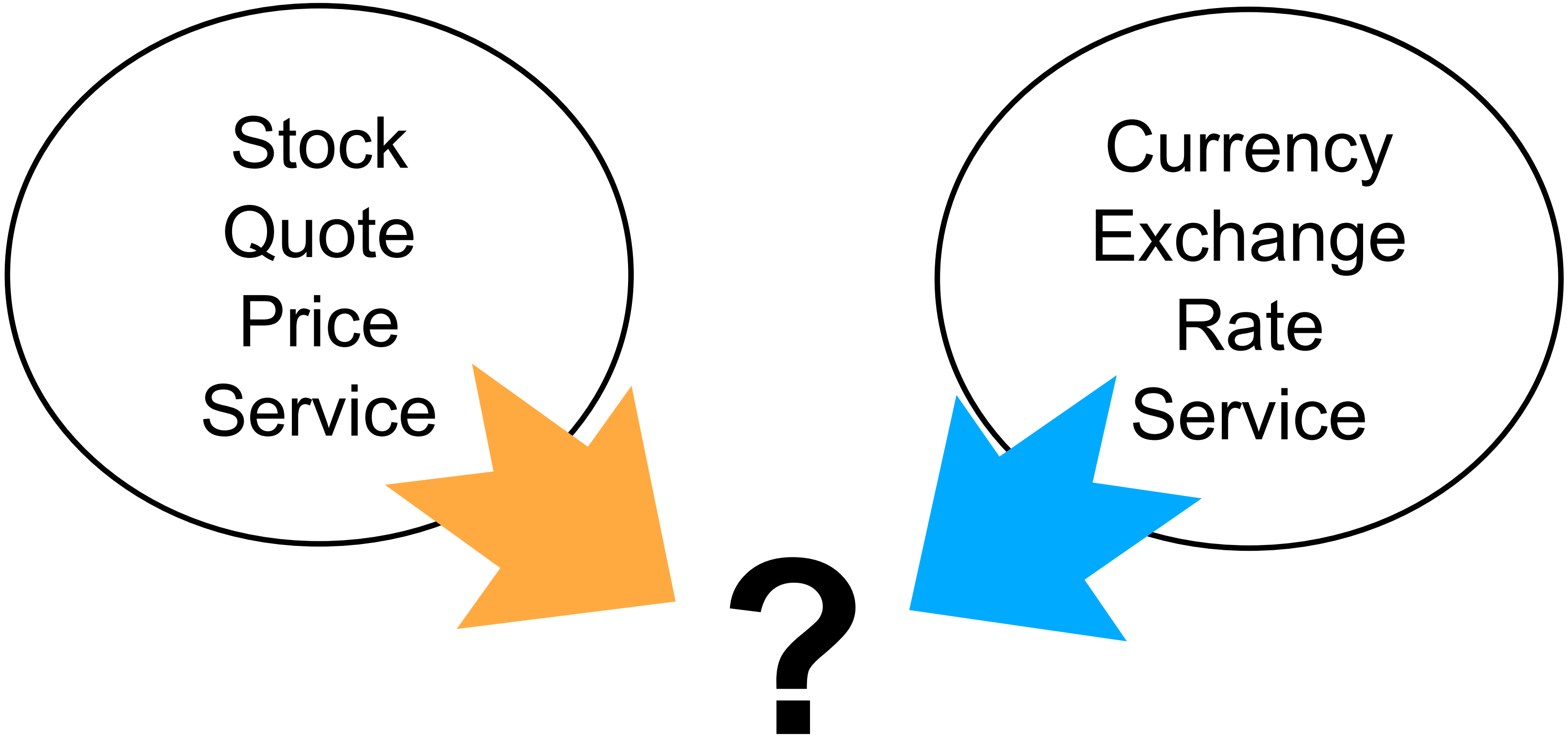
Modeling Grid Workflows with JOpera for Eclipse



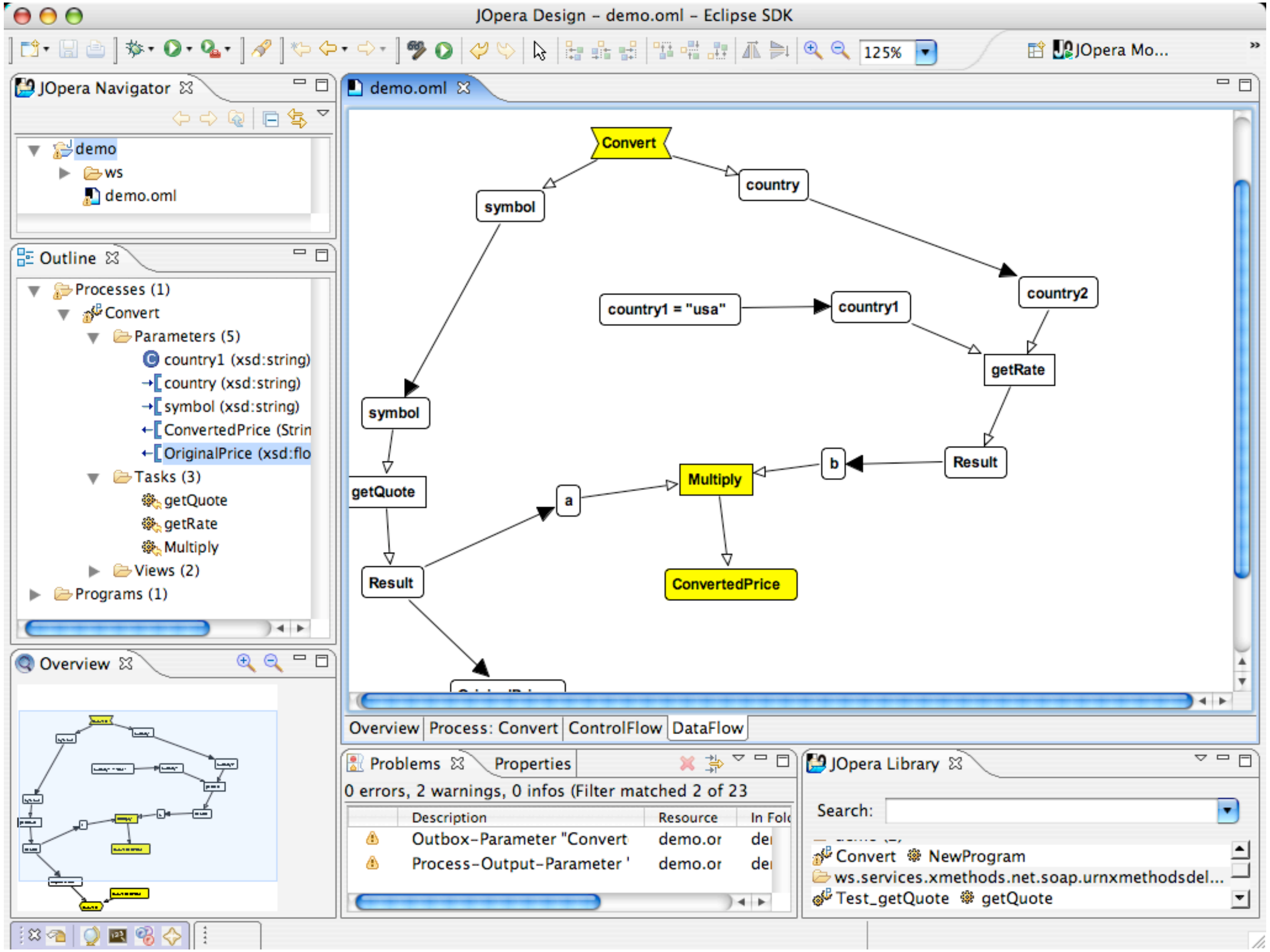
Workflow Lifecycle in JOpera for Eclipse

1. Select component services from a **library**
2. Build a process using a drag, drop and connect **visual** environment
3. Run, Test, and Debug the execution **within the same visual environment**
4. Deploy, Manage, Monitor, and Steer the execution of workflows in production
5. Publish the workflow as Web/Grid Service

- Demo: Stock Quote Currency Conversion



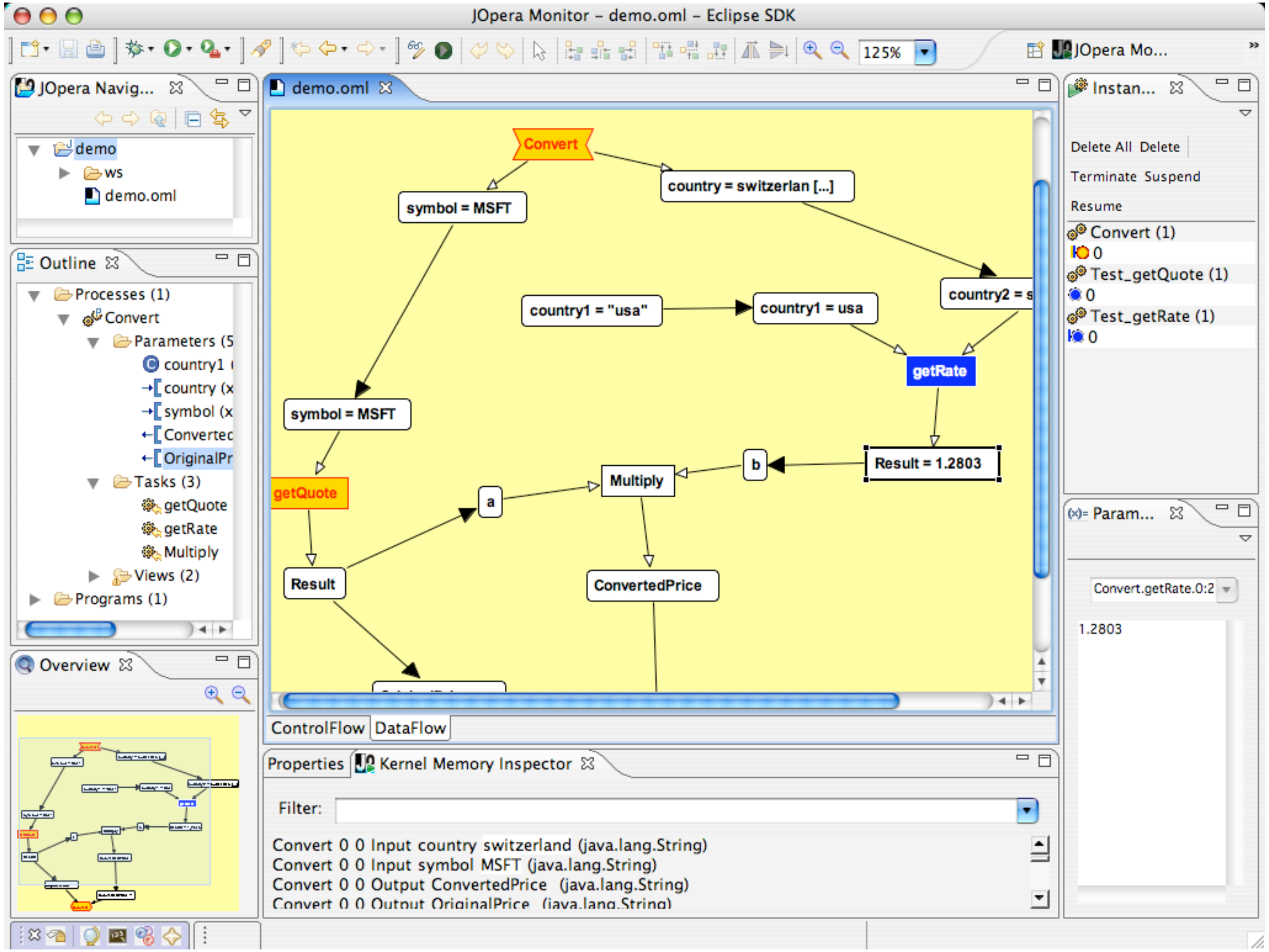
Drag, Drop and Connect



0 errors, 2 warnings, 0 infos (Filter matched 2 of 23

Description	Resource	In Fol
Outbox-Parameter "Convert	demo.or	de
Process-Output-Parameter '	demo.or	de

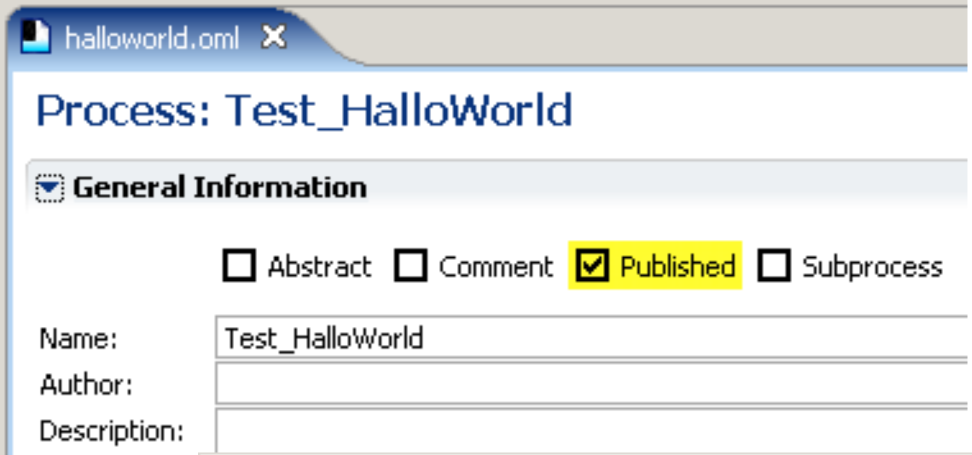
Run, Monitor, Steer and Debug



```
Convert 0 0 Input country switzerland (java.lang.String)
Convert 0 0 Input symbol MSFT (java.lang.String)
Convert 0 0 Output ConvertedPrice  (java.lang.String)
Convert 0 0 Output OriginalPrice  (java.lang.String)
```

Publish as a Web/Grid service

With one mouse click!



Process: Test_HalloWorld

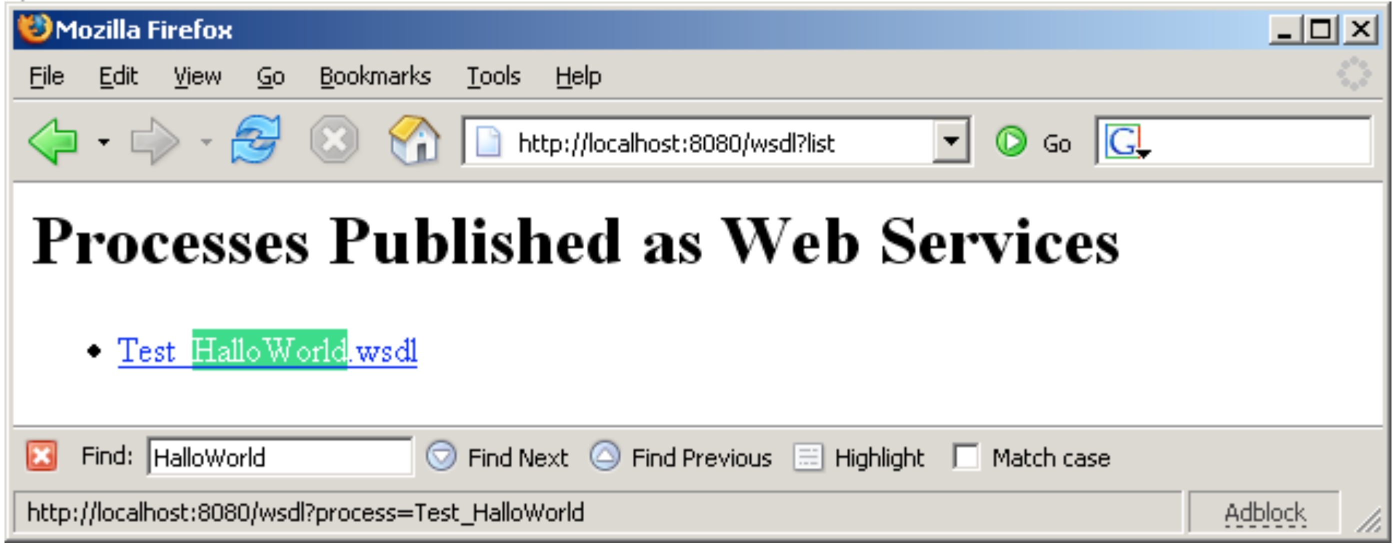
General Information

□ Abstract □ Comment ☑ Published □ Subprocess

Name: Test_HalloWorld

Author:

Description:



http://localhost:8080/wsdl?list

Processes Published as Web Services

- Test_HalloWorld.wsdl

Find: HalloWorld ○ Find Next ○ Find Previous Highlight □ Match case

http://localhost:8080/wsdl?process=Test_HalloWorld

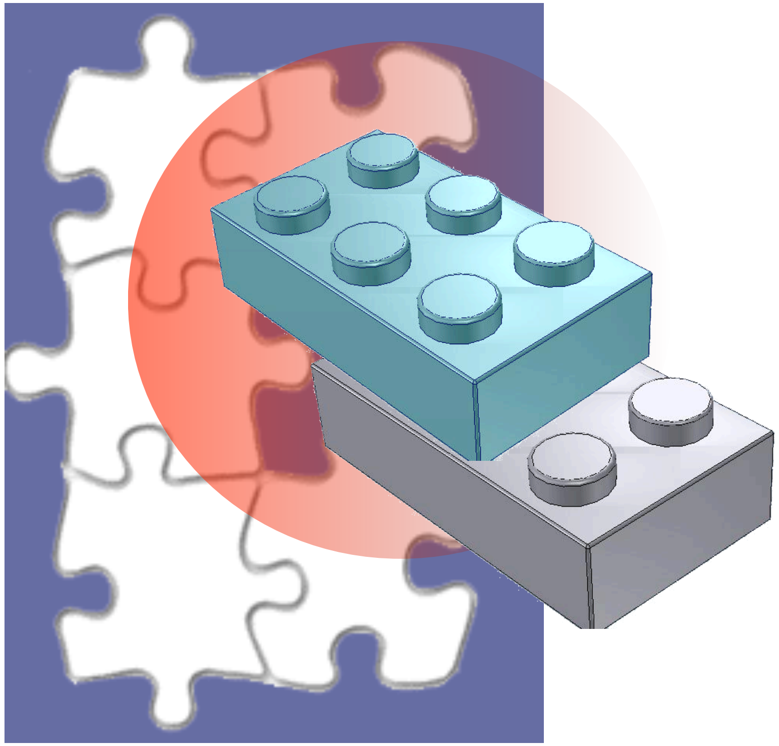
[e-Science2005]

JOpera Workflow Language Features

- Processes model Grid service composition
  - **Data flow** as the primary representation
  - Explicit **control flow** (branch, synchronization, exception handling, loops, pipeline, workflow patterns)
- **SubProcesses**: Modularity, Nesting and Recursion
- **First order functions**
  - Map (parallel/sequential/discriminator) and Reduce
- **Reflection** (introspection)
  - Dynamic late binding
  - Quality of service monitoring

[JVLC2005]

Dealing with Heterogeneity with JOpera for Eclipse



How NOT to deal with heterogeneity

1. Assume that all services to be orchestrated will conform to one standard
2. Force all existing implementations to be wrapped to comply with that standard
3. Modify the workflow language to extend its support to other standards

(See BPEL, BPELJ, BPEL# controversy for an example)

Problems of composing *only* Grid Services

- Grid Services are **coarse-grained**
- All existing heterogeneous tools must be **wrapped** as a Grid Service
  - Wrapping imposes both a performance penalty and additional development & maintenance costs
- The **adapter** between mismatching Grid services must also be a Grid service
- Grid services standards are not stable

Dealing with heterogeneity in JOpera

- How to design a workflow language independent of the kinds of services to be orchestrated?

1. Separate the description of the process from the description of how to invoke each of its tasks
2. A process should make minimal assumptions about its tasks (i.e., data flow signature)
3. Bind tasks to different invocation mechanisms without affecting the process definition

[VLDB/TES2004]

Main advantages

- The developer can choose (or add) the most appropriate type of component:
  - Access Protocols and Mechanisms, Functionality, Performance, Reliability, Security, Convenience, Ease of use, Leverage existing skills
- The workflow language is simpler
  - Many constructs can be shifted from the language definition to the standard library of service types
    - Timeouts
    - Synchronous vs. Asynchronous interaction
    - Adapters and Data transformations

Service Types Supported by JOpera

JOpera provides an *extension-point* for custom **service invocation plugins**

Web servers (HTTP/HTML)

Human activities

SQL Queries (JDBC)

UNIX Commands

Windows

SSH

Web Services (SOAP, WSIF)

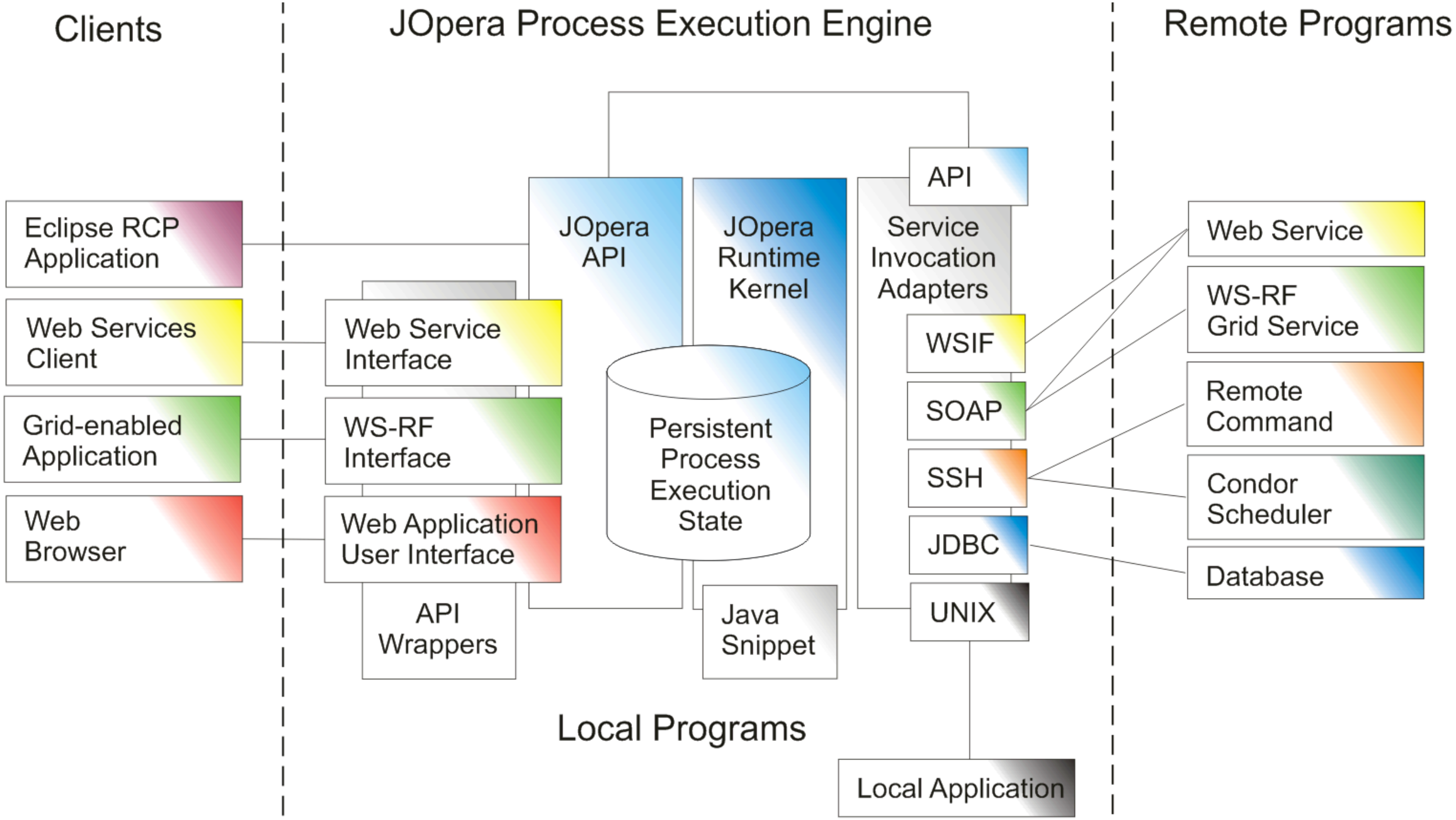
Grid Services (WSRF)

XML Transformations (XSLT, X-Path)

Java snippets

Java methods

Architecture of JOpera for Eclipse



Conclusions

- JOpera is a workflow tool for building distributed applications made out of heterogeneous parts
- Workflows provide high level abstractions for specifying the behavior of such applications
  - Which language constructs are specific to Grid workflows?
- JOpera offers a completely **open**, flexible and extensible service composition platform
  - We have built adapters for invoking Grid services (GT3, GT4), running remote SSH commands, submitting jobs to Condor and will add more as standards evolve (e.g., JSDL)

Outlook

- Working on integrating a “UML2.0/extended Activity Diagram skin”
- Adding support for specialized data viewers
- Developing data mining tools on process execution history space for provenance and lineage tracking analysis
- Is there a benchmark for Grid Workflow Execution engines?

References

[e-SCIENCE2005] Thomas Heinis, Cesare Pautasso, Oliver Deak, Gustavo Alonso, **Publishing Persistent Grid Computations as WS Resources**, accepted to the 1st IEEE International Conference on e-Science and Grid Computing (e-Science 2005), Melbourne, Australia, December 2005.

[ICWS2005] Cesare Pautasso, Thomas Heinis, Gustavo Alonso: **Autonomic Execution of Service Compositions**, In: Proc. of the 3rd International Conference on Web Services (ICWS 2005), Orlando, Florida, July 2005.

[ICAC2005] Thomas Heinis, Cesare Pautasso, Gustavo Alonso: **Design and Evaluation of an Autonomic Workflow Engine**, In: Proc of the 2nd International Conference on Autonomic Computing (ICAC-05), Seattle, Washington, June 2005.

[JVLC2005] Cesare Pautasso, Gustavo Alonso **The JOpera Visual Composition Language** Journal of Visual Languages and Computing (JVLC), 16(1-2):119-152, 2005

[VLDB/TES2004] Cesare Pautasso, Gustavo Alonso: **From Web Service Composition to Megaprogramming** In: Proceedings of the 5th VLDB Workshop on Technologies for E-Services (TES-04), Toronto, Canada, August 29-30, 2004.

Download Plugins & Papers from:
www.jopera.org

Available Today