

Composing REST Services

Open Dagstuhl Session

Cesare Pautasso

Faculty of Informatics

University of Lugano (USI), Switzerland

<http://www.pautasso.info>

Web Service Composition Today

“ The WS-BPEL process model is layered on top of the service model defined by WSDL 1.1. [...]

Both the process and its partners are exposed as WSDL services ”

[BPEL 2.0 Standard, Section 3]

WS-BPEL 2.0

WSDL 1.1

RESTful Web Services APIs...



...do not use
WSDL 1.1

The Goal

- Compose RESTful Web Services
- Compose WSDL Web Services
- Use Business Process Modeling Languages

One Solution: BPEL for REST

- Extend BPEL to support RESTful Web Services

The Goal

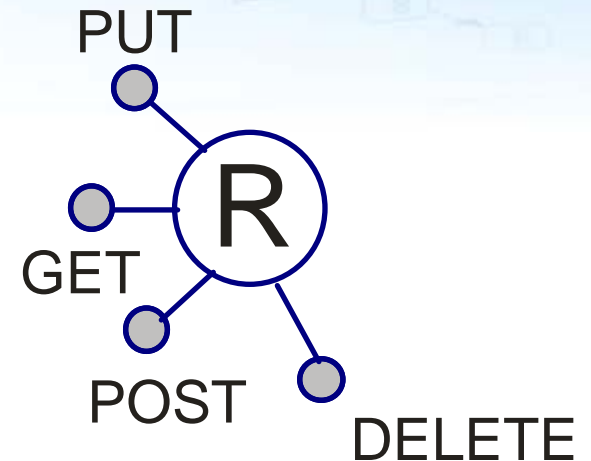
- Compose RESTful Web Services
- Compose WSDL Web Services
- Use Business Process Modeling Languages

Another Solution: JOpera

- Visual Flow Language with Abstract Service Model
- Extensible Autonomic Engine Architecture

REST in one slide

- Web Services expose their data and functionality through **resources** identified by **URI**
- **Uniform Interface Principle**: Clients interact with the state of resources through 4 verbs: GET (read), POST (create), PUT (update), DELETE
- **Multiple representations** for the same resource
- **Hyperlinks** model resource relationships and valid state transitions

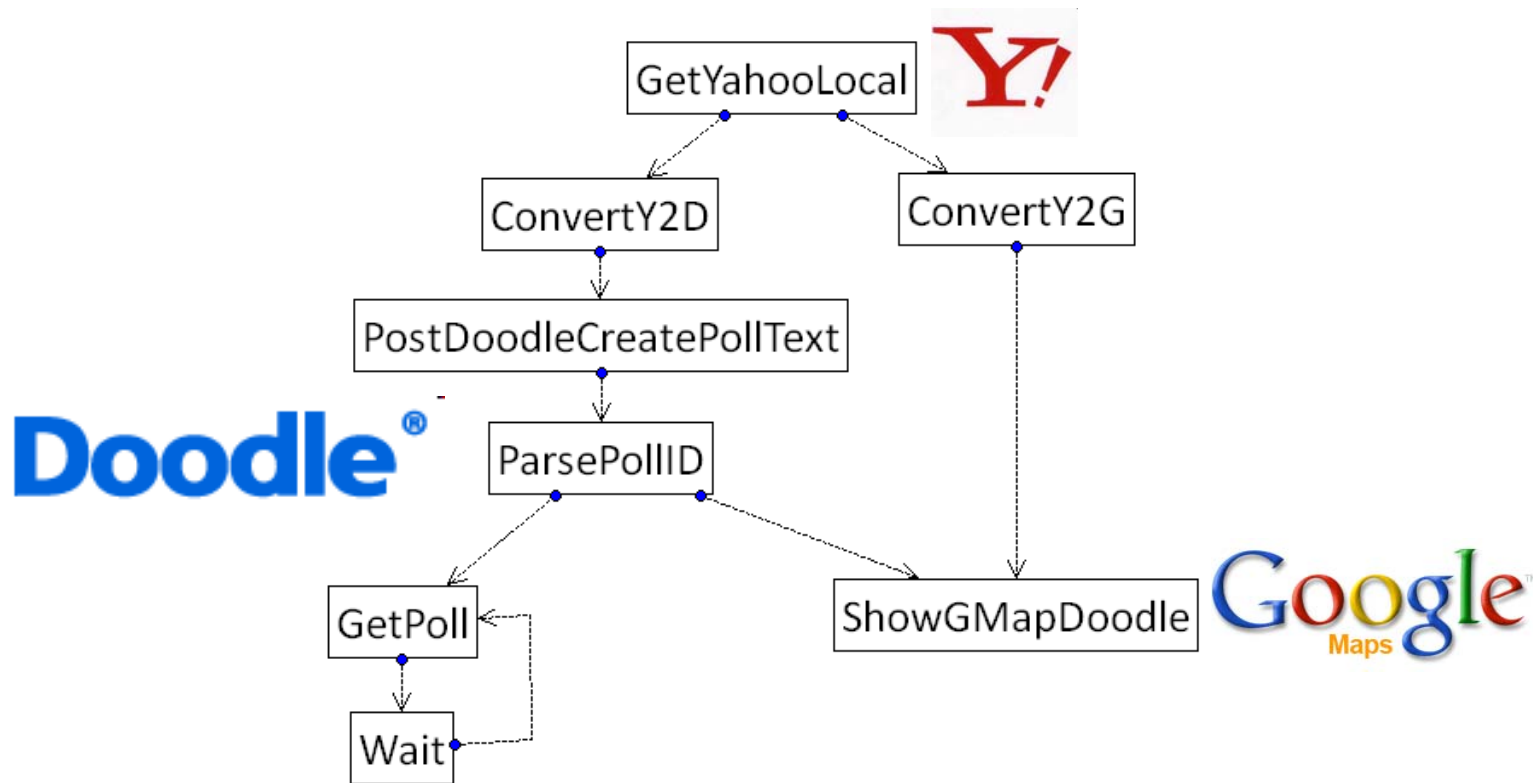


Challenges for Composition Languages

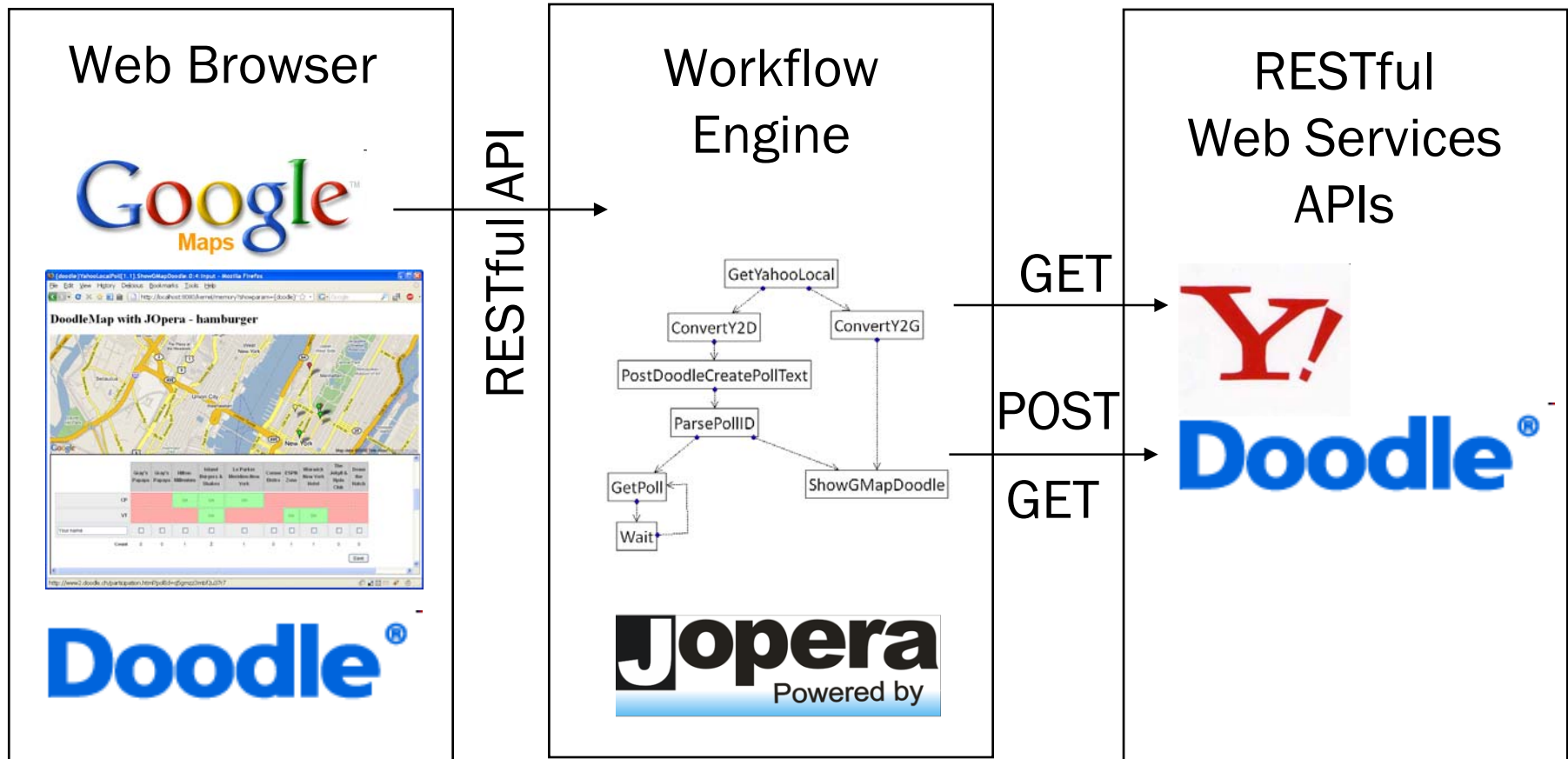
- Resource addressing through URI
 - *How to interact with dynamic, variable set of URI?*
- Uniform Interface (GET, POST, PUT, DELETE)
 - *Does it help to make the verbs explicit in the workflow?*
- Multiple resource representations
 - *How to negotiate the most appropriate representation?*
- Hyperlinks
 - *Can the workflow implement state transition logic of a resource and generate new URIs dynamically as processes run to guide the clients invoking them?*

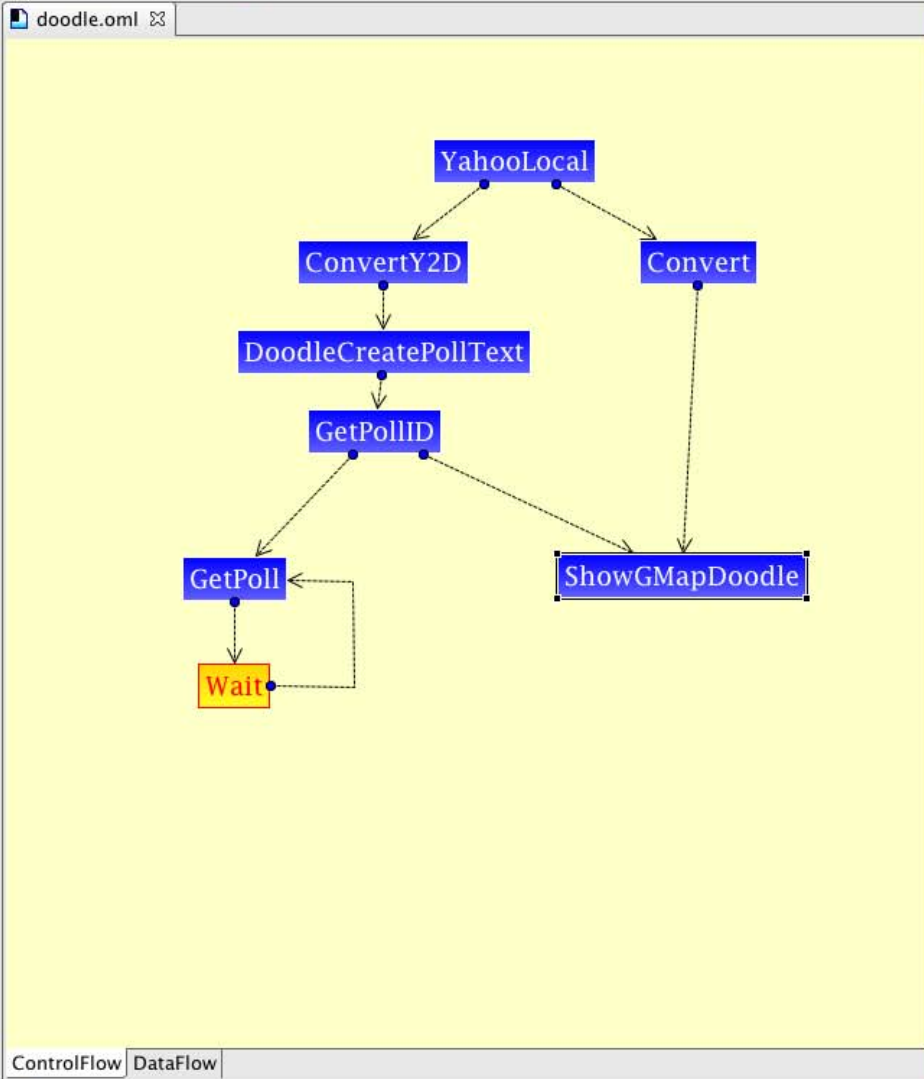
JOpera Example: Doodle Map Mashup

- Setup a Doodle with Yahoo! Local search and visualize the results of the poll on Google Maps



Doodle Map Mashup Architecture





http://localhost:8080/kernel/

DoodleMap with JOpera

Island Burgers & Shakes
Preferences:2

Poll: hamburger
CP has created this poll.
"10001"

	Gray's Papaya	Gray's Papaya	Hilton-Millennium	Island Burgers & Shakes	Le Parker Meridien-New York	Corner Bistro	ESPN Zone	Warwick New York Hotel	The Jekyll & Hyde Club	Dow the Hatc
CP				OK		OK				
PA				OK	OK					

**BPM
Workflow
Languages**

**RESTful
Web Service
Composition**

Solution Space

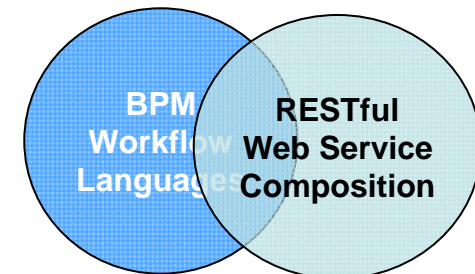
1. Abstract Workflow

- *Service invocation technology does not matter*



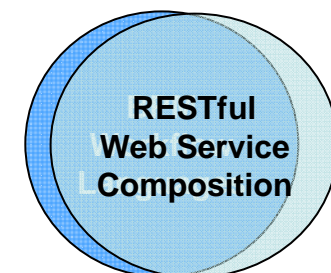
2. Concrete Workflow

- *Expose service invocation technologies as explicit constructs in the workflow language*



3. RESTful Workflow

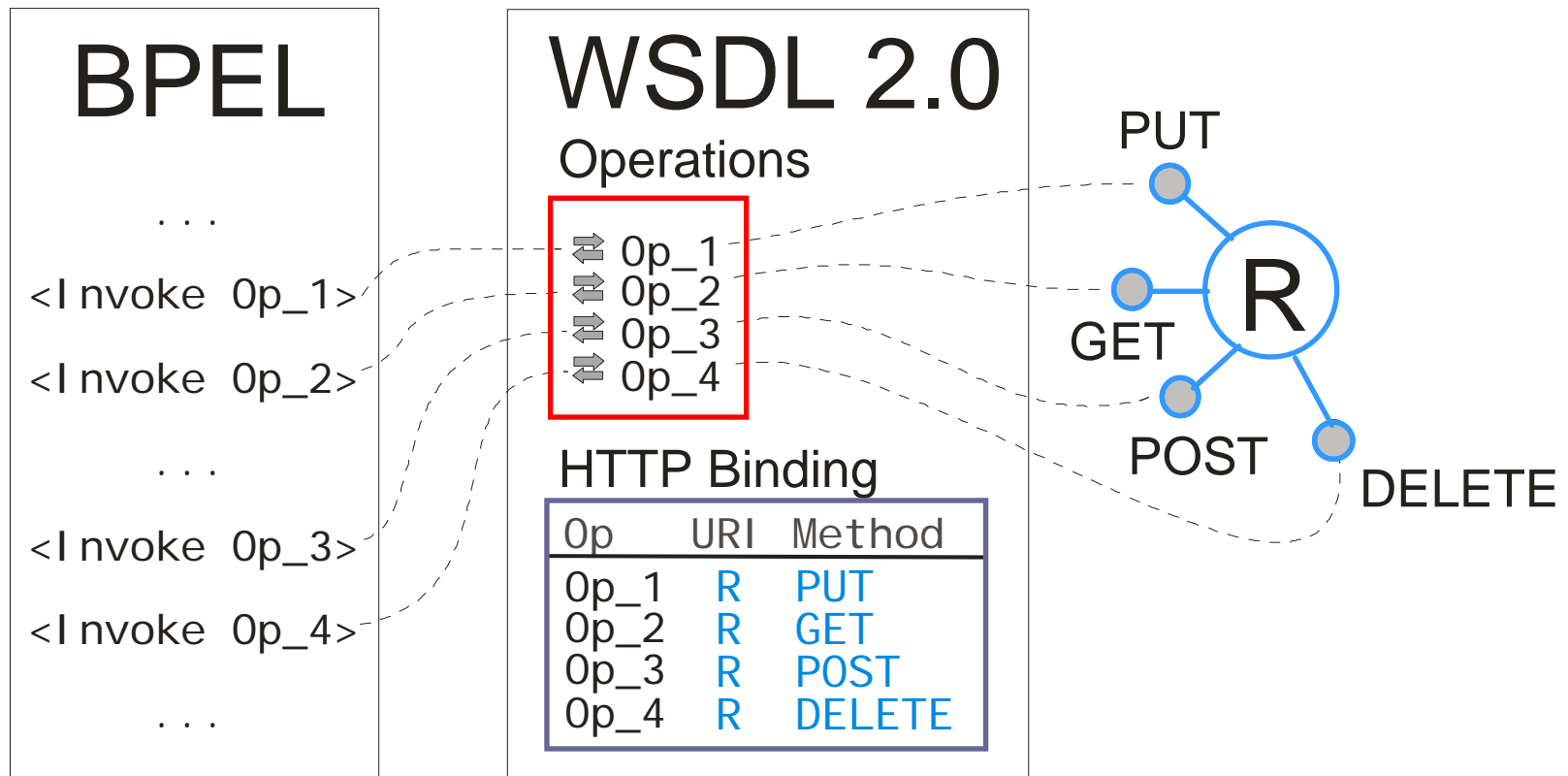
- *Workflow as one kind of resource exposed by a RESTful service*



1. Abstract Workflow Example: BPEL/WSDL

WSDL 2.0 HTTP Binding can wrap RESTful Web Services

(WS-BPEL 2.0 does not support WSDL 2.0)

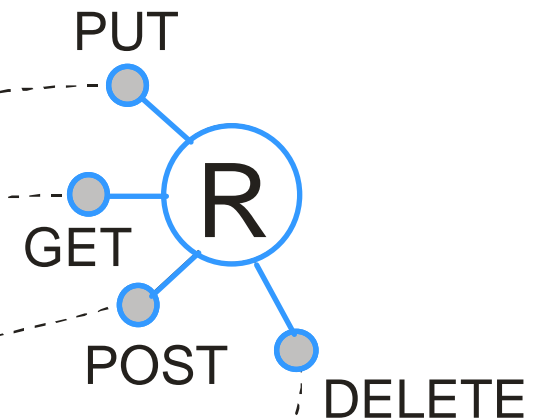


2. Concrete Workflow: BPEL for REST

Idea: Make REST interaction primitives first-class language constructs

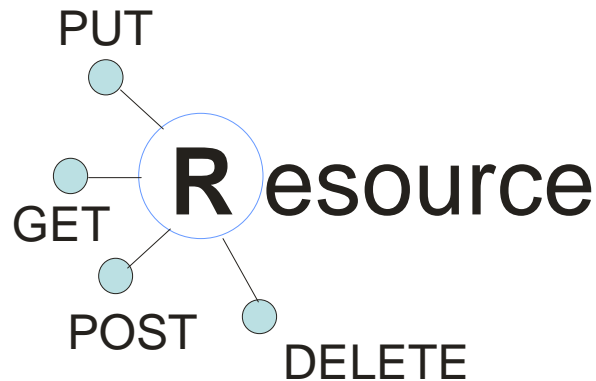
BPEL for REST

```
...  
<Put R>  
<Get R>  
...  
<Post R>  
<Delete R>  
...
```



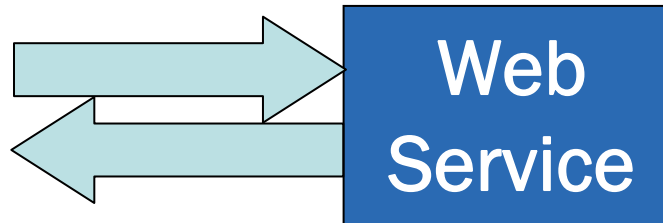
BPEL for REST – New Activities/Handlers

<put>
<get>
<post>
<delete>



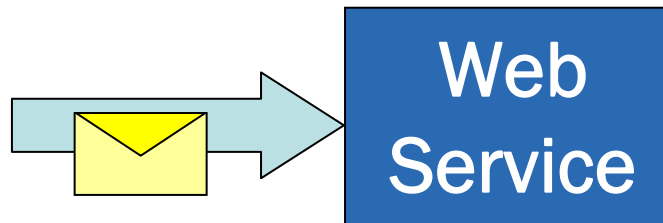
<onPut>
<onGet>
<onPost>
<onDelete>

<invoke>



<receive>
<reply>

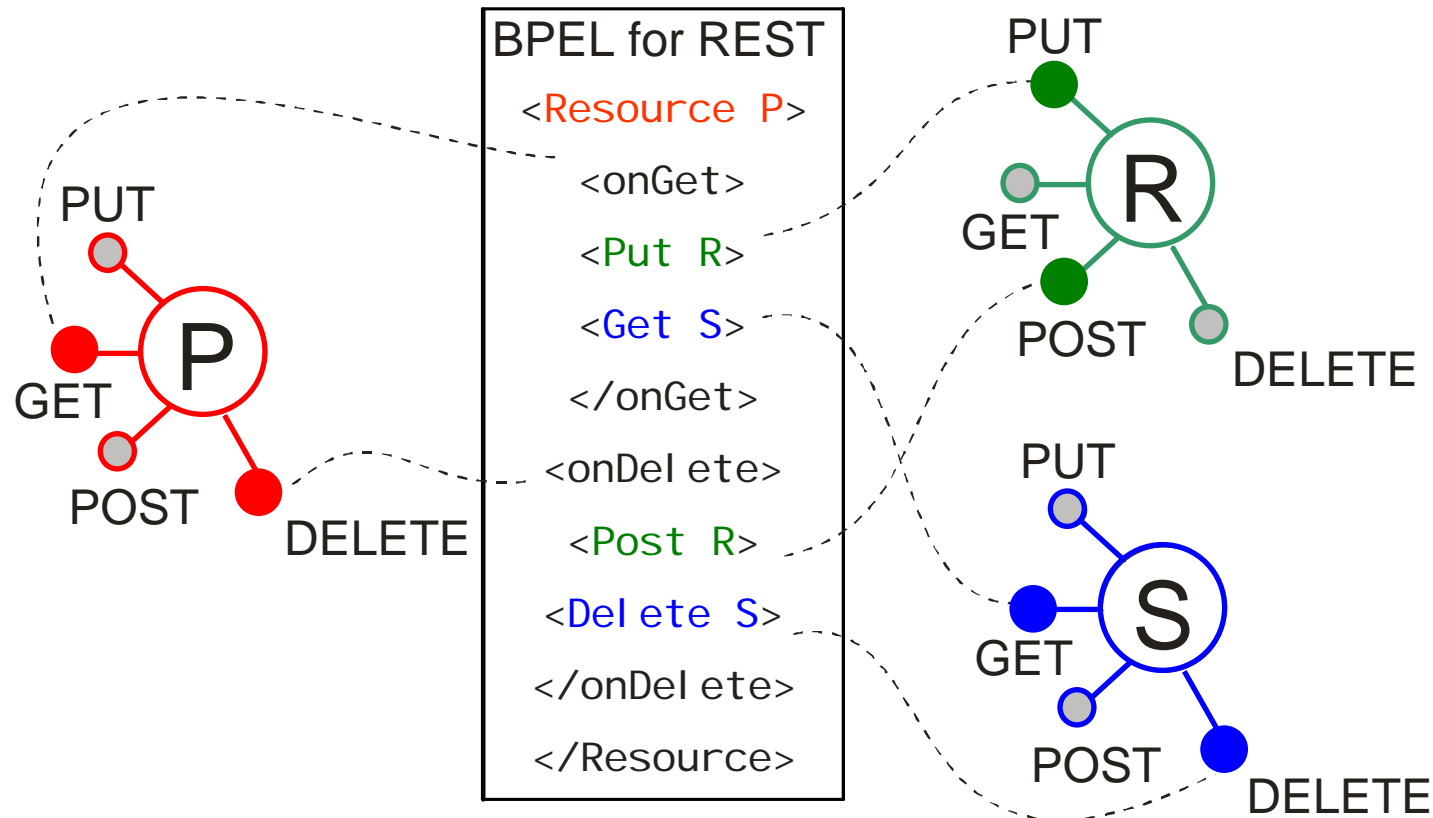
<invoke>



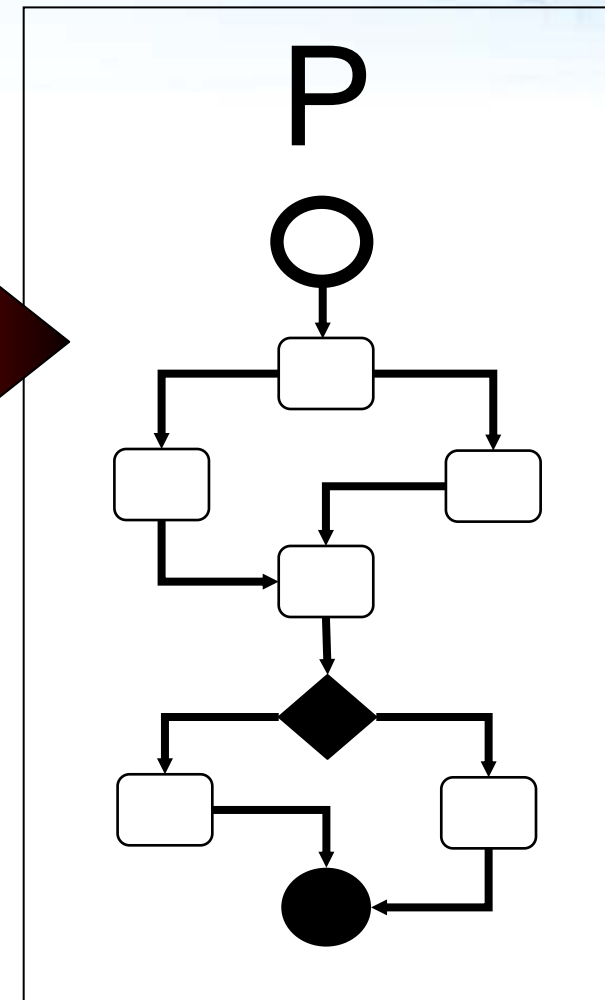
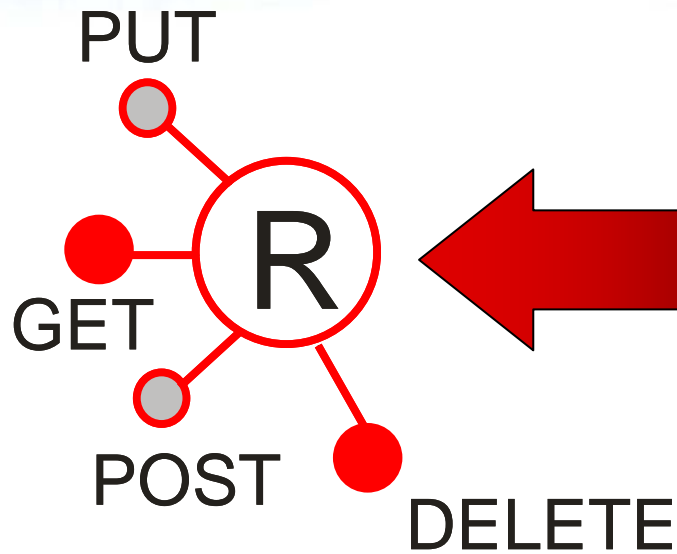
<receive>

BPEL for REST – Resource Block

- Dynamically publish resources from BPEL processes and handle client requests



3. RESTful Workflows



Use the resource interface abstraction to publish the state of the workflow

Workflows as Resources – URI

- Publish workflows as resources identified by the URIs:

/package/process

/package/process/version

/package/process/version/instance

/package/process/version/instance/task

/package/process/version/instance/task/parameter

Reading the state of the workflow resources

- GET /package/process
 - Enumerate deployed process versions
- GET /package/process/version
 - Enumerate active instances of a given process version
- GET /package/process/version/instance
 - Read the current state of a workflow instance
- GET /package/process/version/instance/task
 - Read the current state of a workflow instance task
- GET /package/process/version/instance/task/param
 - Read the current value of a workflow instance parameter

Creating new workflow resources

- POST /package
 - Deploy new process template into package
- POST /package/process
 - Deploy new version of a process
- POST /package/process/version
 - Create new process instance

Updating the workflow resource

- PUT /package/process/version/instance/task
 - Update the state of a workflow task (e.g., finished, failed)
- PUT /package/process/version/instance/task/param
 - Write into task parameters some values

Deleting workflow resources

- DELETE /package/process
 - Undeploy all versions of a process
(and all the corresponding process instances)
- DELETE /package/process/version
 - Undeploy a version of a process (and all of its instances)
- DELETE /package/process/version/instance
 - Remove the state of a specific process instance only

Discussion

- Why should a workflow engine care about REST?
 - Use workflows to compose RESTful Web services
 - Implement RESTful services with a workflow
- Should a process explicitly include RESTful activities?
 - Or it is better/enough to model REST implicitly?
- How much of the state of a process instance should be exposed as a resource?
 - How to control which “parts” are visible?

Conclusion

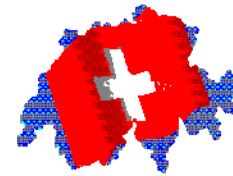
- Business Process Modeling Languages have been applied with success to compose “traditional” WS-* Web Services (BPM = SOA + BPEL)
- Business Process Modeling Languages should also be applied to compose RESTful Web Services
- BPEL for REST is a lightweight BPEL extension for REST and WS-* service composition
- JOpera for Eclipse is a visual process modeling tool with an extensible engine for composing both kinds of services (and many more)

Some References

- R. Fielding, [Architectural Styles and the Design of Network-based Software Architectures](#), PhD Thesis, University of California, Irvine, 2000
- C. Pautasso, O. Zimmermann, F. Leymann, [RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision](#), Proc. of the 17th International World Wide Web Conference ([WWW2008](#)), Beijing, China, April 2008
- C. Pautasso, [BPEL for REST](#), Proc. of the 7th International Conference on Business Process Management (BPM 2008), Milano, Italy, September 2008
- Xiwei (Sherry) Xu, Liming Zhu, Yan Liu, Mark Staples, **Resource-Oriented Architecture for Business Processes**, APSEC'08

Università
della
Svizzera
italiana

Facoltà
di scienze
informatiche



PhD positions available!

Università
della Svizzera
italiana

Faculty of
Informatics

Prof. Cesare Pautasso
University of Lugano, Switzerland
c.pautasso@ieee.org
<http://www.pautasso.info>