Università della Svizzera italiana

Faculty of Informatics

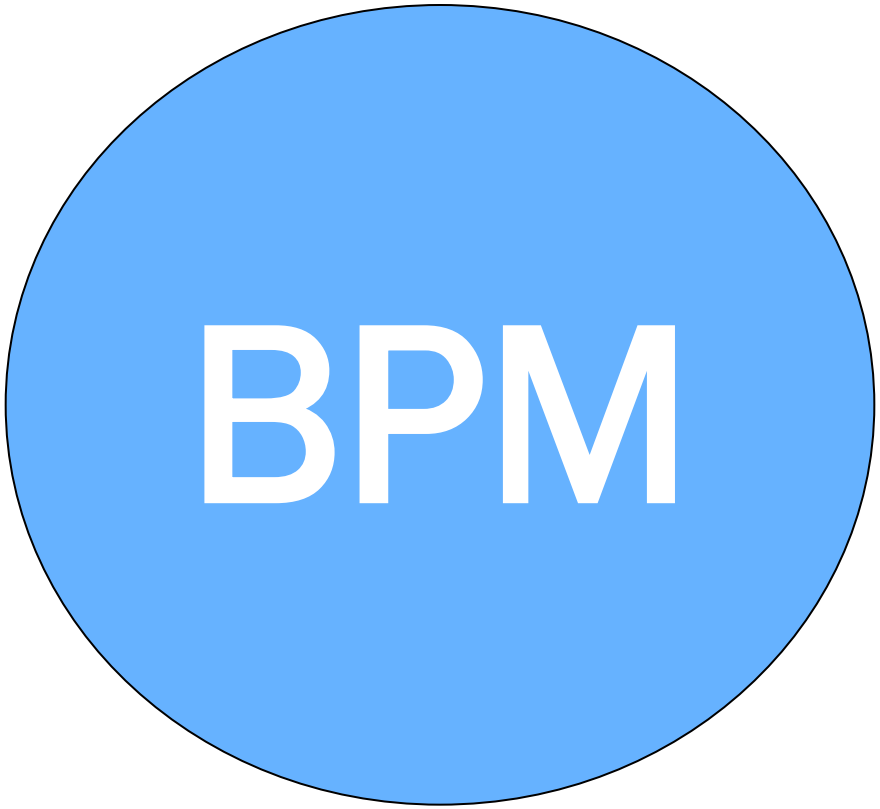
RESTful Business Process Management

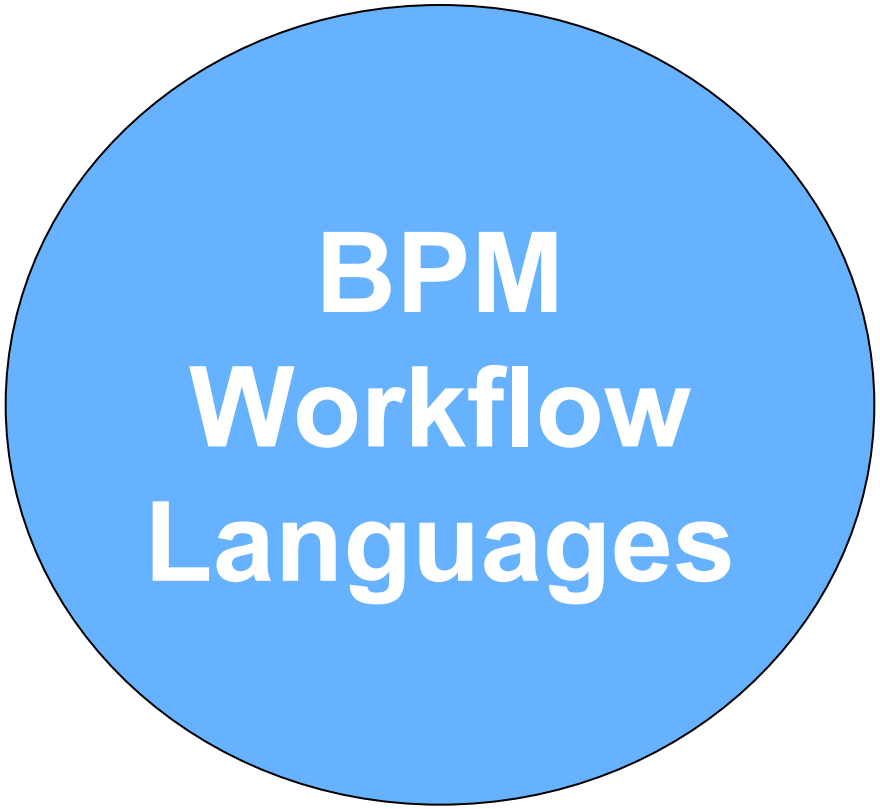
Cesare Pautasso
Faculty of Informatics
University of Lugano, Switzerland

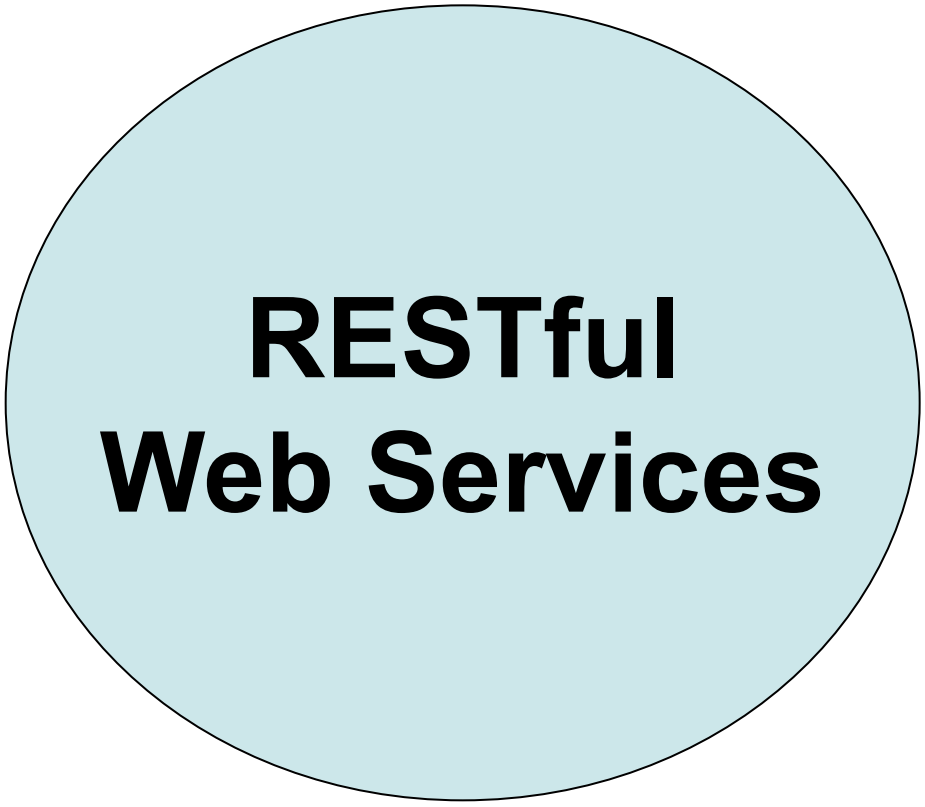
c.pautasso@ieee.org
http://www.pautasso.info

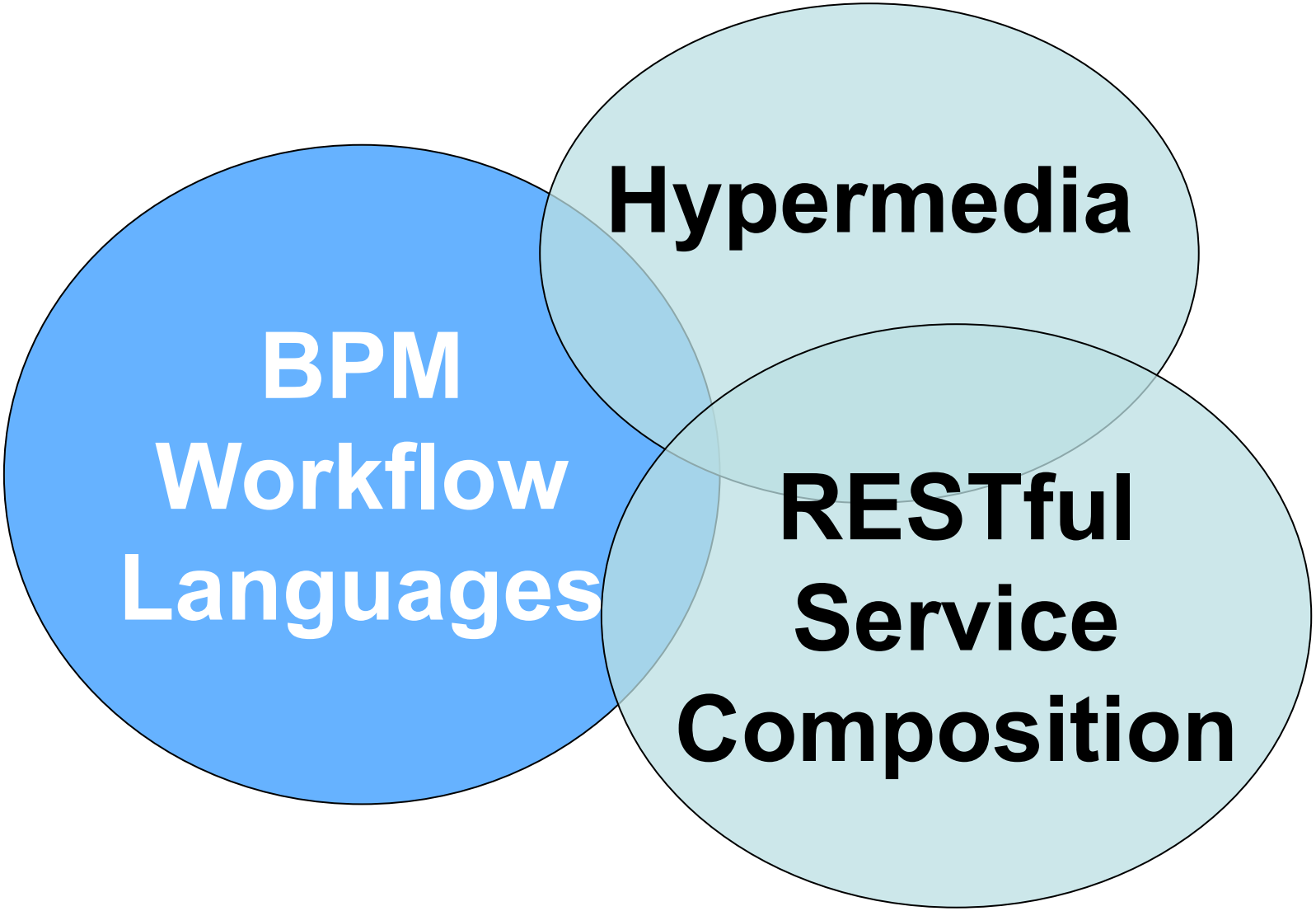
Abstract

- Next generation Web services technologies challenge the assumptions made by current standards for process-based service composition. For example, most existing RESTful Web service APIs cannot natively be composed using the WS-BPEL standard.
- In this talk we discuss the conceptual relationship between business processes and stateful resources with the goal of enabling lightweight access to service compositions published with a RESTful API. We show that the uniform interface and the hyper-linking capabilities of RESTful services provide an excellent abstraction for publishing as a resource and exposing in a controlled way the execution state of business processes.









About Cesare Pautasso

QCon

Università della Svizzera italiana

- Assistant Professor at the Faculty of Informatics, University of Lugano, Switzerland (since Sept 2007)
- Research Projects:
  - SOSOA – Self-Organizing Service Oriented Architectures
  - CLAVOS – Continuous Lifelong Analysis and Verification of Open Services
  - BPEL for REST
- Researcher at IBM Zurich Research Lab (2007)
- Post-Doc at ETH Zürich
  - Software: JOpera: Process Support for more than Web services http://www.jopera.org/
- Ph.D. at ETH Zürich, Switzerland (2004)

- Representations: http://www.pautasso.info/ (Web) http://twitter.com/pautasso/ (Twitter Feed)

ws://rest.2010

First International Workshop on RESTful Design

- 26 April 2010, Raleigh NC, USA
- World Wide Web 2010 Conference
- Keynote by **Sam Ruby**, Apache & IBM
- http://ws-rest.org/

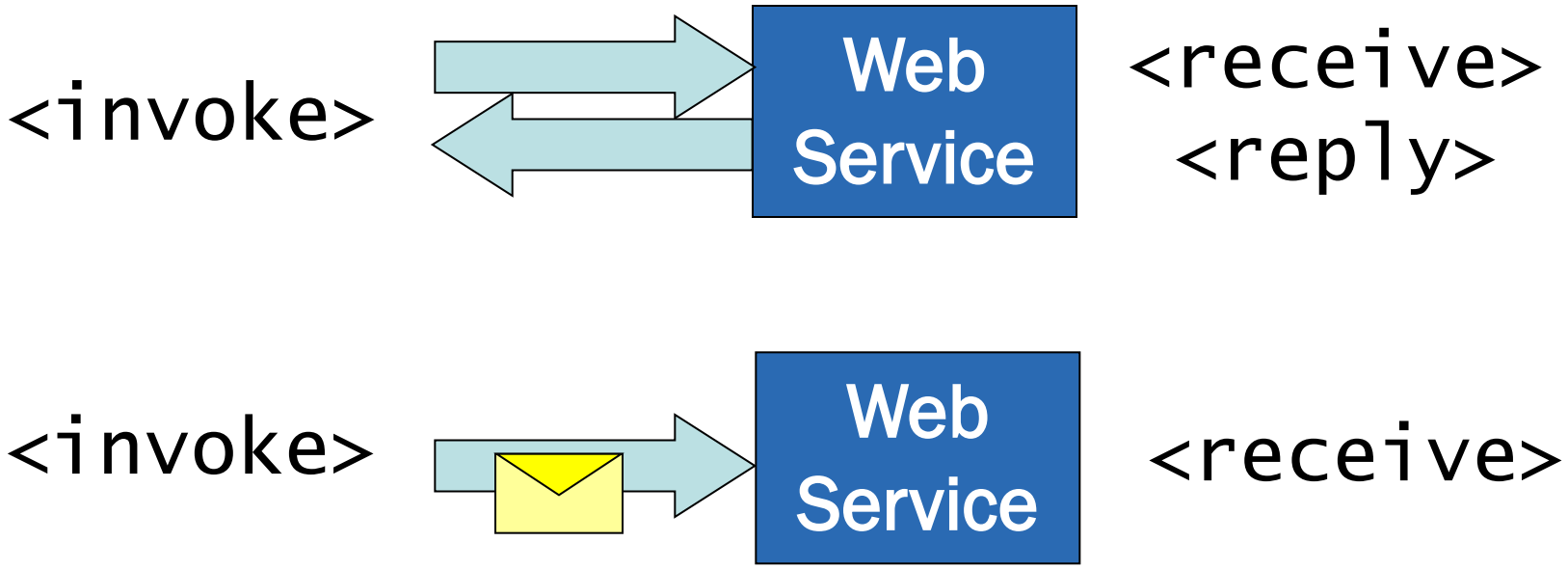
Raj Balasubramanian,
Benjamin Carlyle,
Thomas Erl,
Cesare Pautasso,
SOA with REST,
Prentice Hall,
to appear in 2010

From REST-*

> We believe there is huge potential to marrying REST with workflow and BPM.
>
> [...]
>
> Combined with the architecture of the Web, a workflow service can provide both a truly **simple, portable, and flexible** way to build workflow driven integrations and applications.

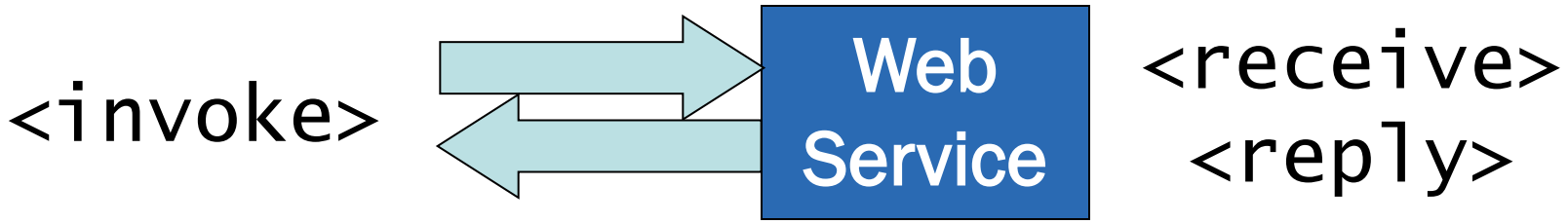
http://www.jboss.org/reststar/specifications/workflow.html

WS-BPEL Primitives



- The workflow language natively supports the RPC or message-based connectors

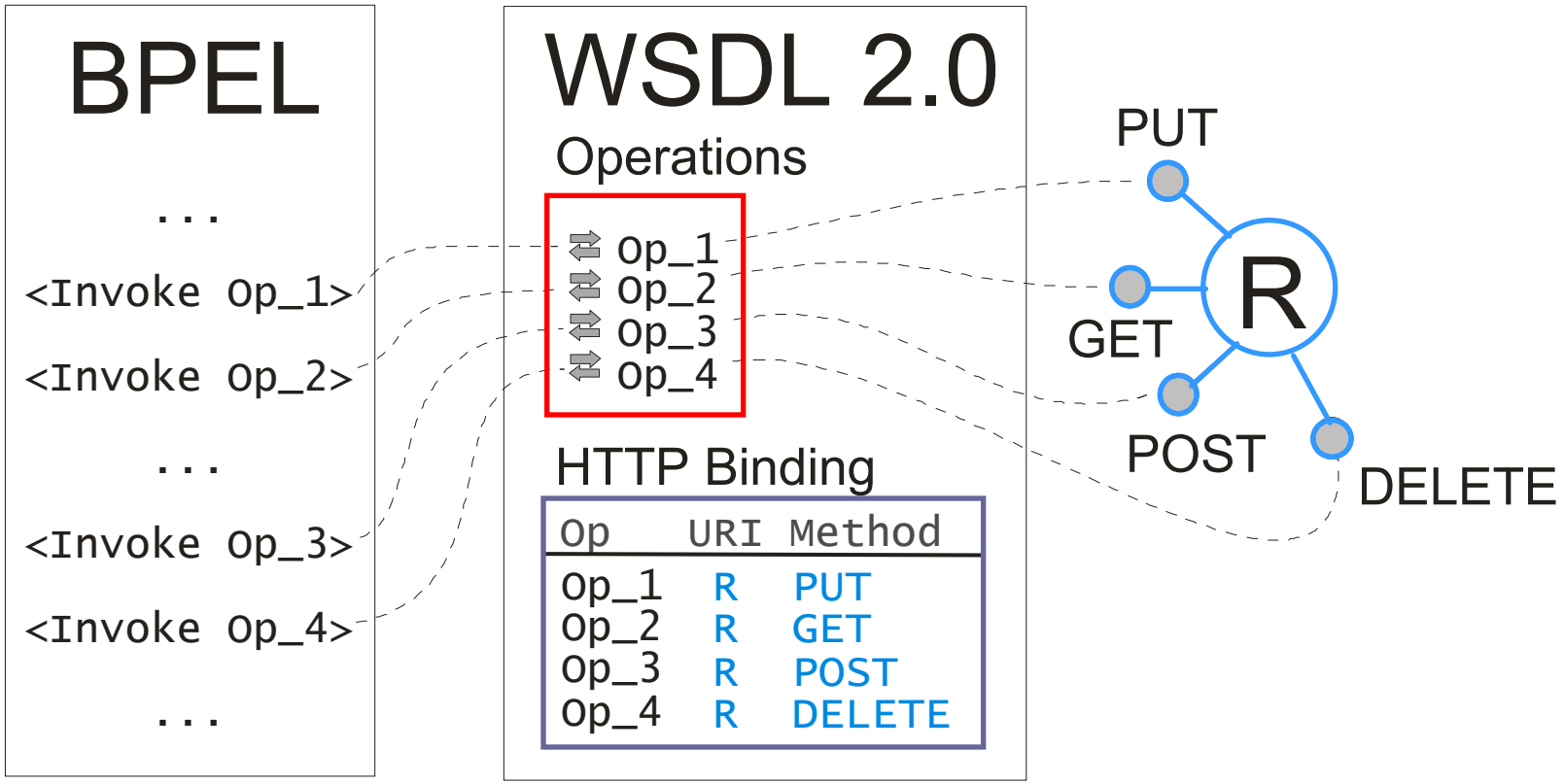
WS-BPEL Primitives



- Easy to map this to HTTP!

WSDL 2.0 HTTP Binding can wrap RESTful Web Services

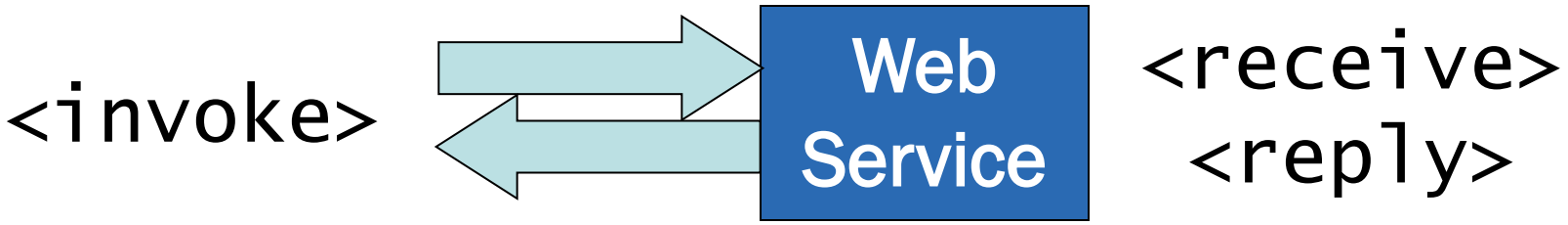
(WS-BPEL 2.0 does not support WSDL 2.0)



RESTful APIs...

...do not use WSDL

WS-BPEL Primitives



- Easy to map this to HTTP?
- We need something else

From REST-*

> We believe there is huge potential to marrying REST with workflow and BPM.
>
> - The HATEOAS (hypermedia and linking) principal of REST is logically a dynamic state machine and **fits very well** with how workflow and BPM systems are designed.
> - Combined with the architecture of the Web, a workflow service can provide both a truly simple, portable, and flexible way to build workflow driven integrations and applications.

http://www.jboss.org/reststar/specifications/workflow.html

RESTful BPM Challenges

QCon Università della Svizzera italiana

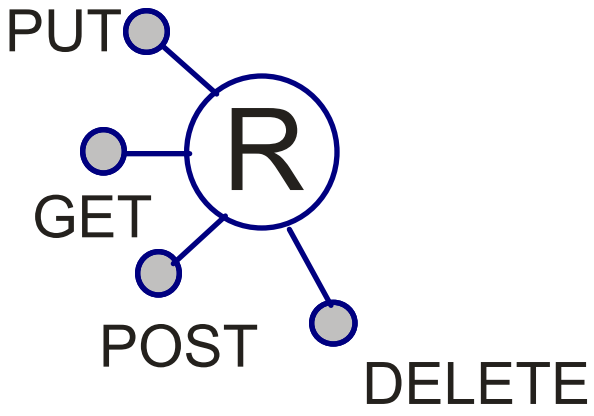
- Can you drive the execution of tasks with PUT/POST/DELETE requests?
- Can you monitor your processes with an RSS/ATOM feed?
- Can you bookmark a process instance?
- Can you send an email to your colleague with a link to a task from your worklist?
- Can you ask a process to give you links to its tasks left to be done?
- Can you publish your process as a resource?
- Can you publish resources from your process?
- Can you call RESTful APIs directly?

Outline

- RESTful BPM
- Challenges and Opportunities
- RESTful Service Composition
- Example: DoodleMap
- Mashups with BPM
- Hypermedia with BPM
- Example: TinyRESTBucks
- Outlook

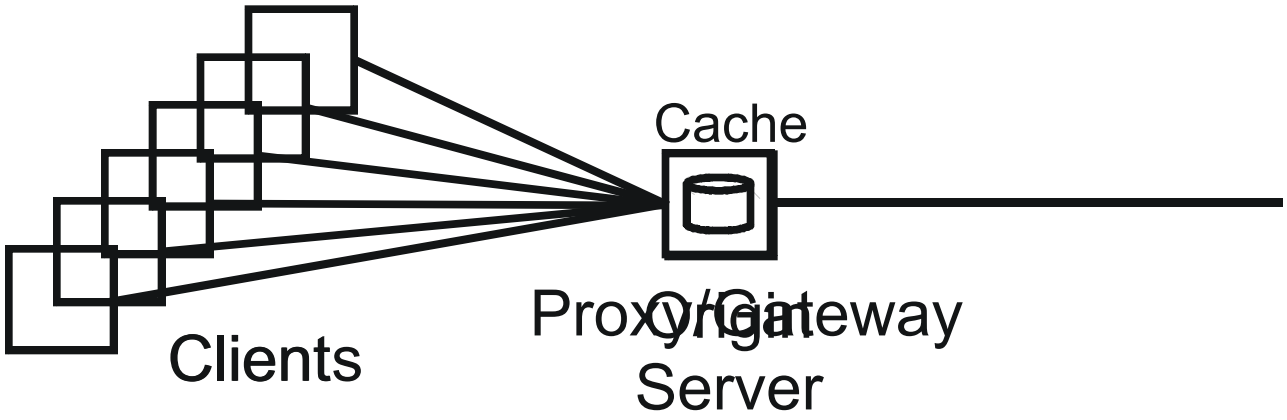
REST in one slide

- Web Services expose their data and functionality trough **resources** identified by **URI**
- **Uniform Interface** Principle: Clients interact with resources through a fix set of verbs.
  Example HTTP:
  GET (read), POST (create), PUT (update), DELETE



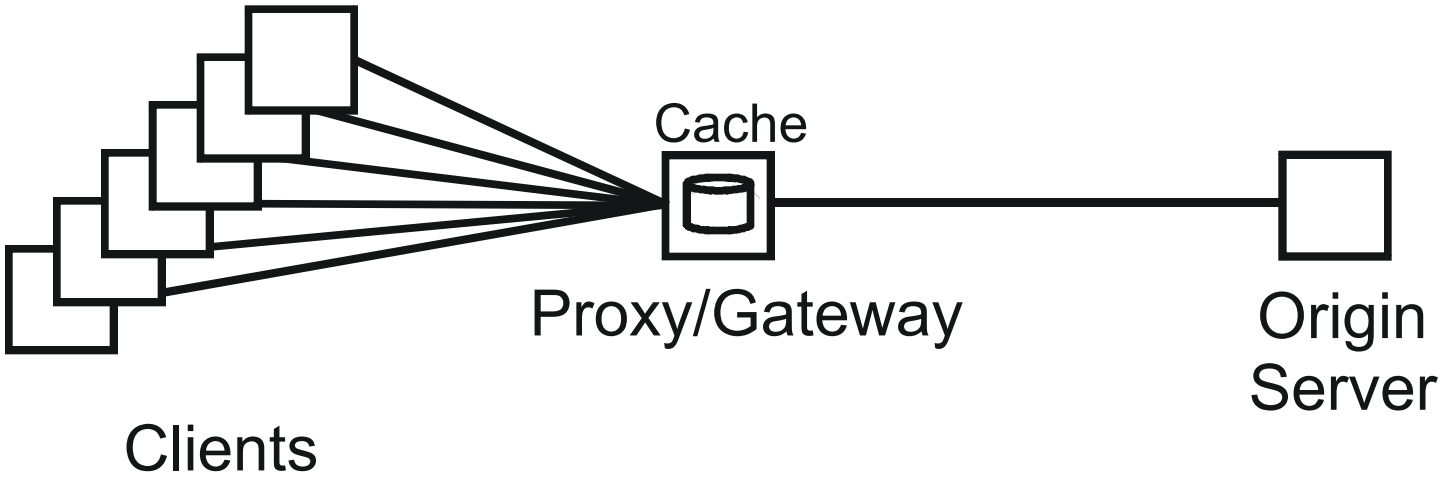
- **Multiple representations** for the same resource
- **Hyperlinks** model resource relationships and valid state transitions for dynamic protocol description and discovery

REST Scalability



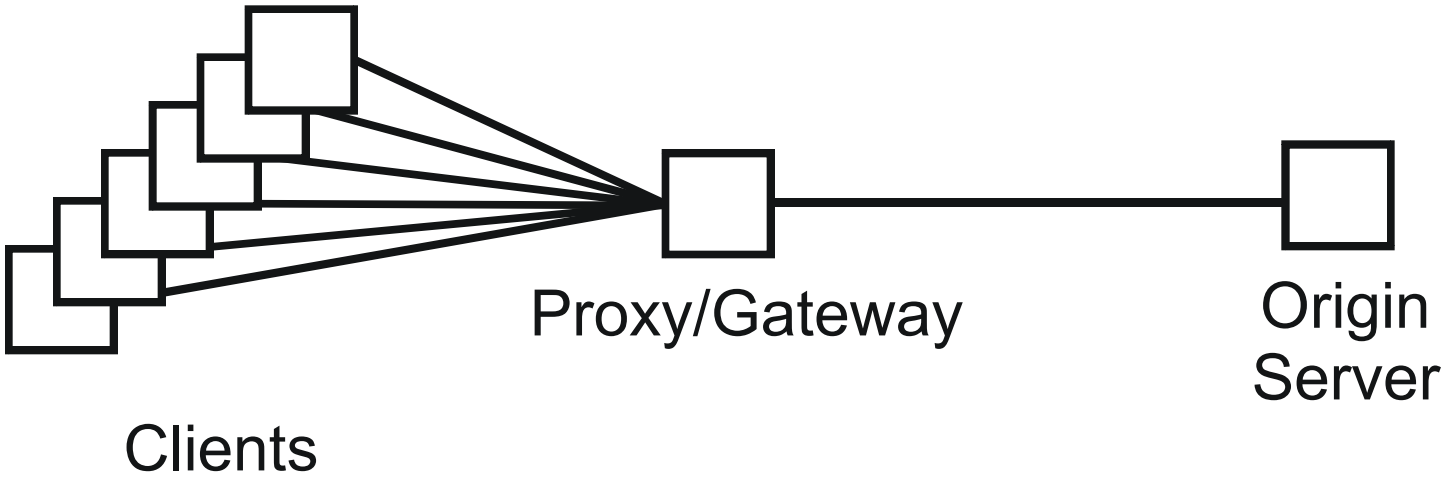
- One example of REST middleware is to help with the scalability of a server, which may need to service a very large number of clients

REST Scalability



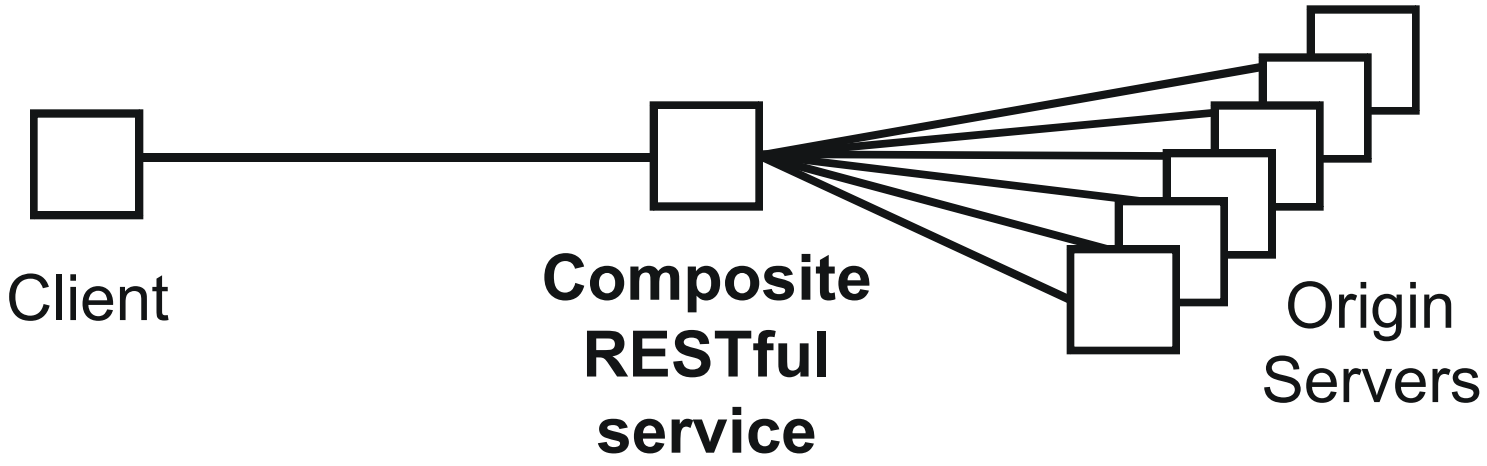
- One example of REST middleware is to help with the scalability of a server, which may need to service a very large number of clients

REST Composition



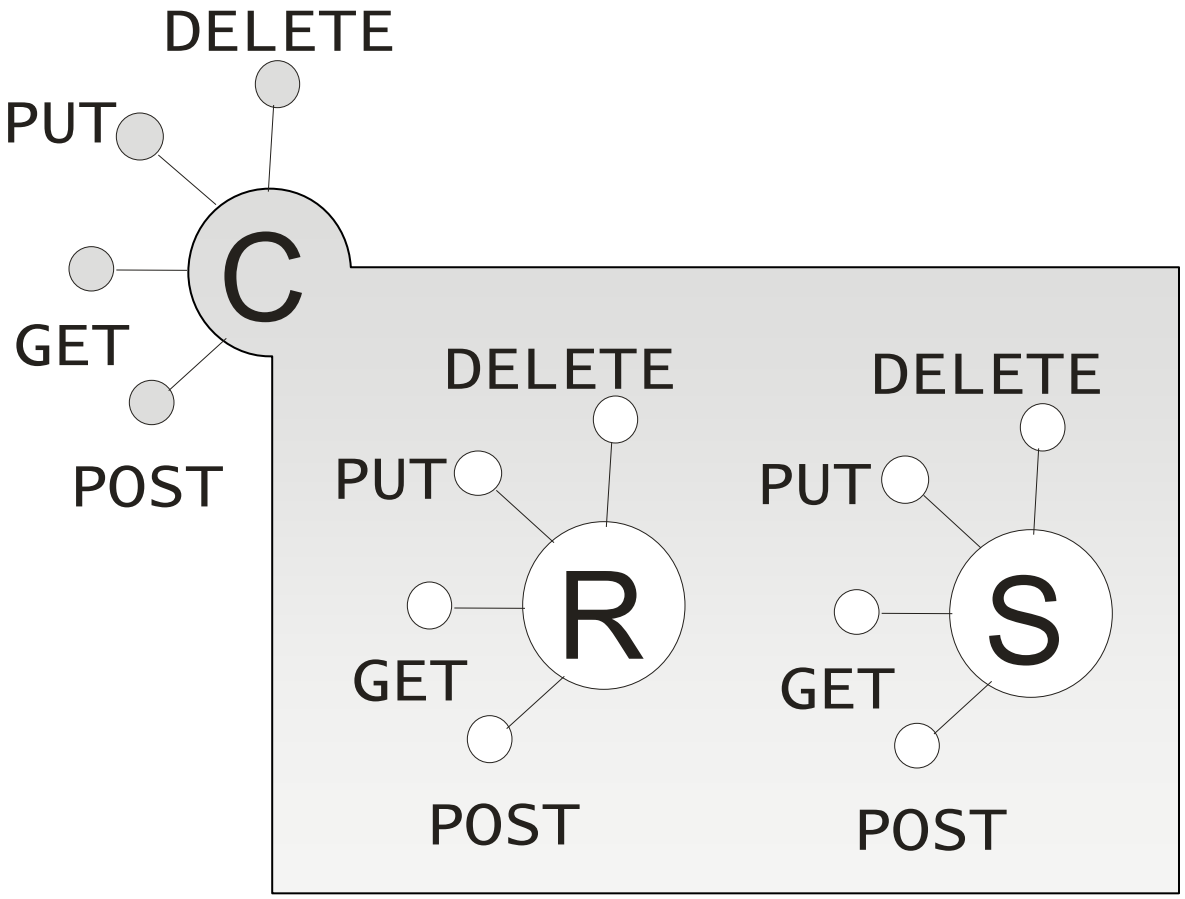
- Composition shifts the attention to the client which should consume and aggregate from many servers

REST Composition



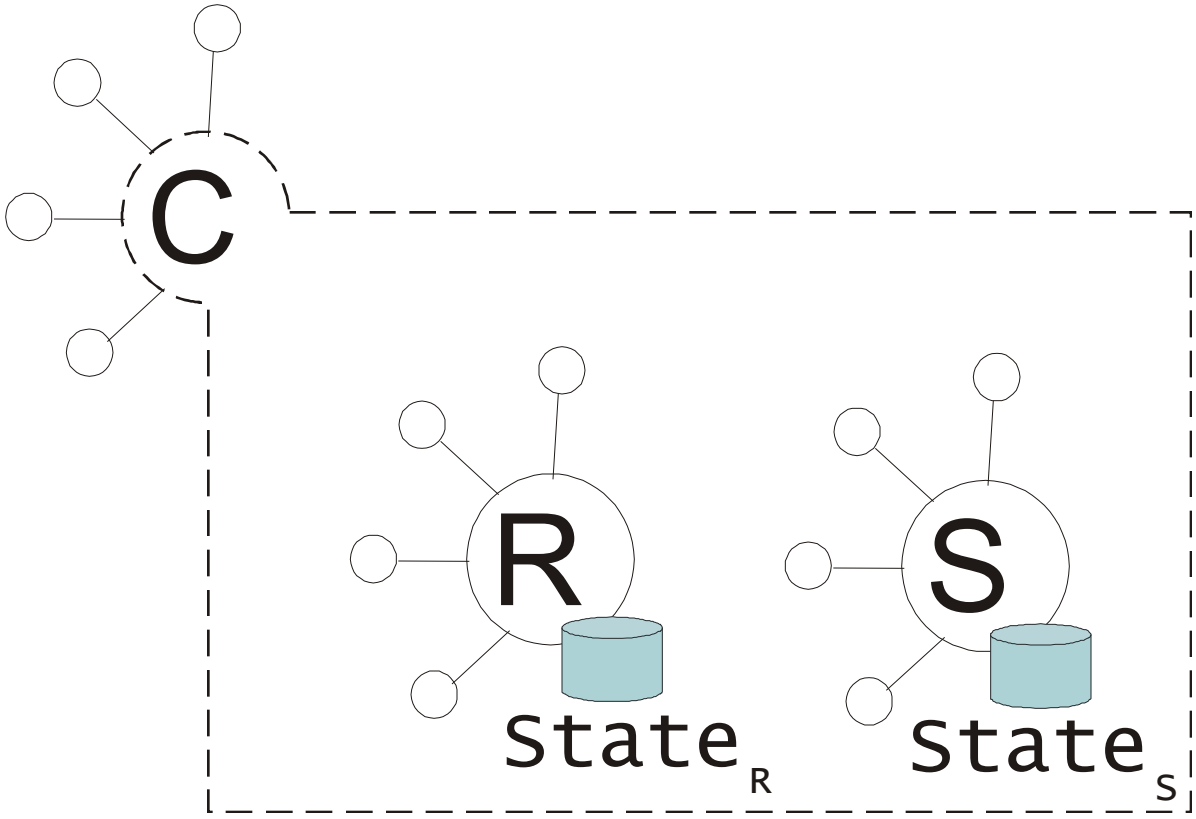
- The “proxy” intermediate element which aggregates the resources provided by multiple servers plays the role of a composite RESTful service
- Can/Should we implement it with BPM?

Composite Resources



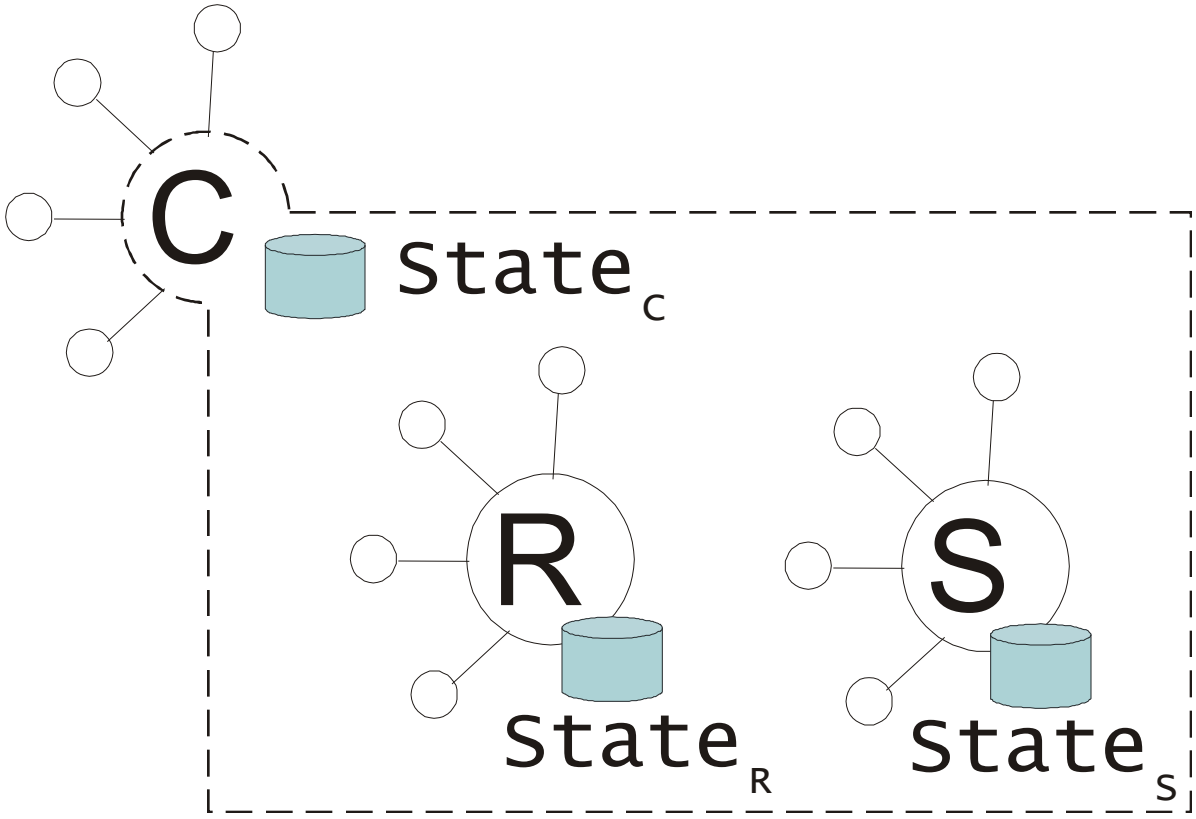
Composite Resources

- The composite resource only aggregates the state of its component resources



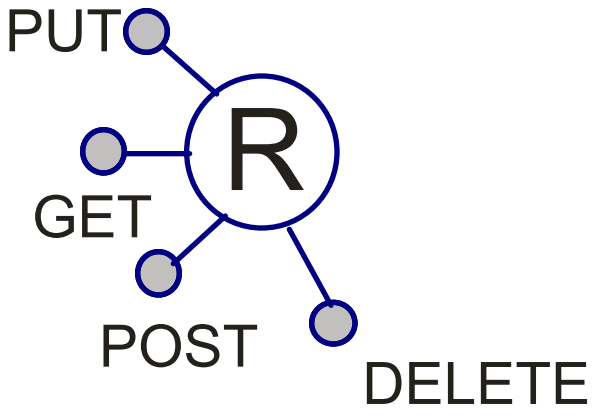
Composite Resources

- The composite resource augments (or caches) the state of its component resources



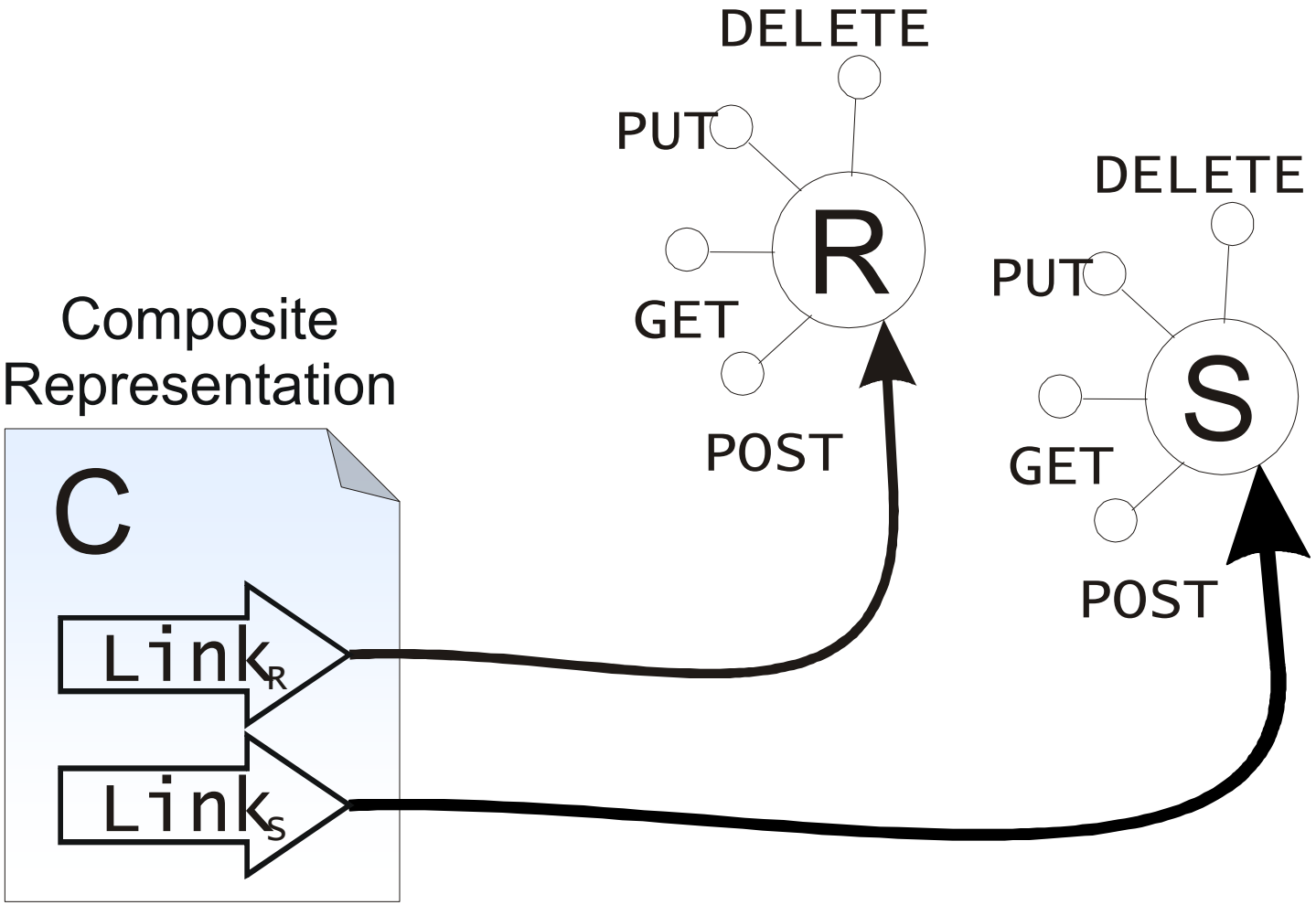
Enter HATEOAS

- Web Services expose their data and functionality trough **resources** identified by **URI**
- **Uniform Interface** Principle: Clients interact with resources through a fix set of verbs.
  Example HTTP:
  GET (read), POST (create), PUT (update), DELETE

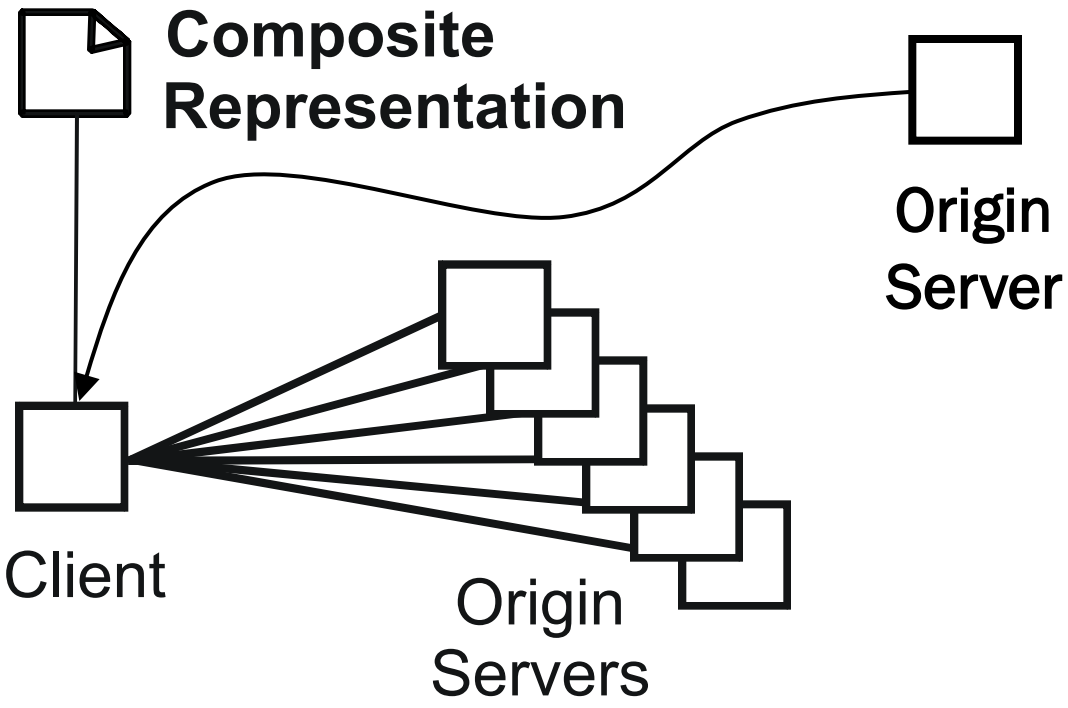


- **Multiple representations** for the same resource
- **Hyperlinks** model resource relationships and valid state transitions for dynamic protocol description and discovery

Composite Representations

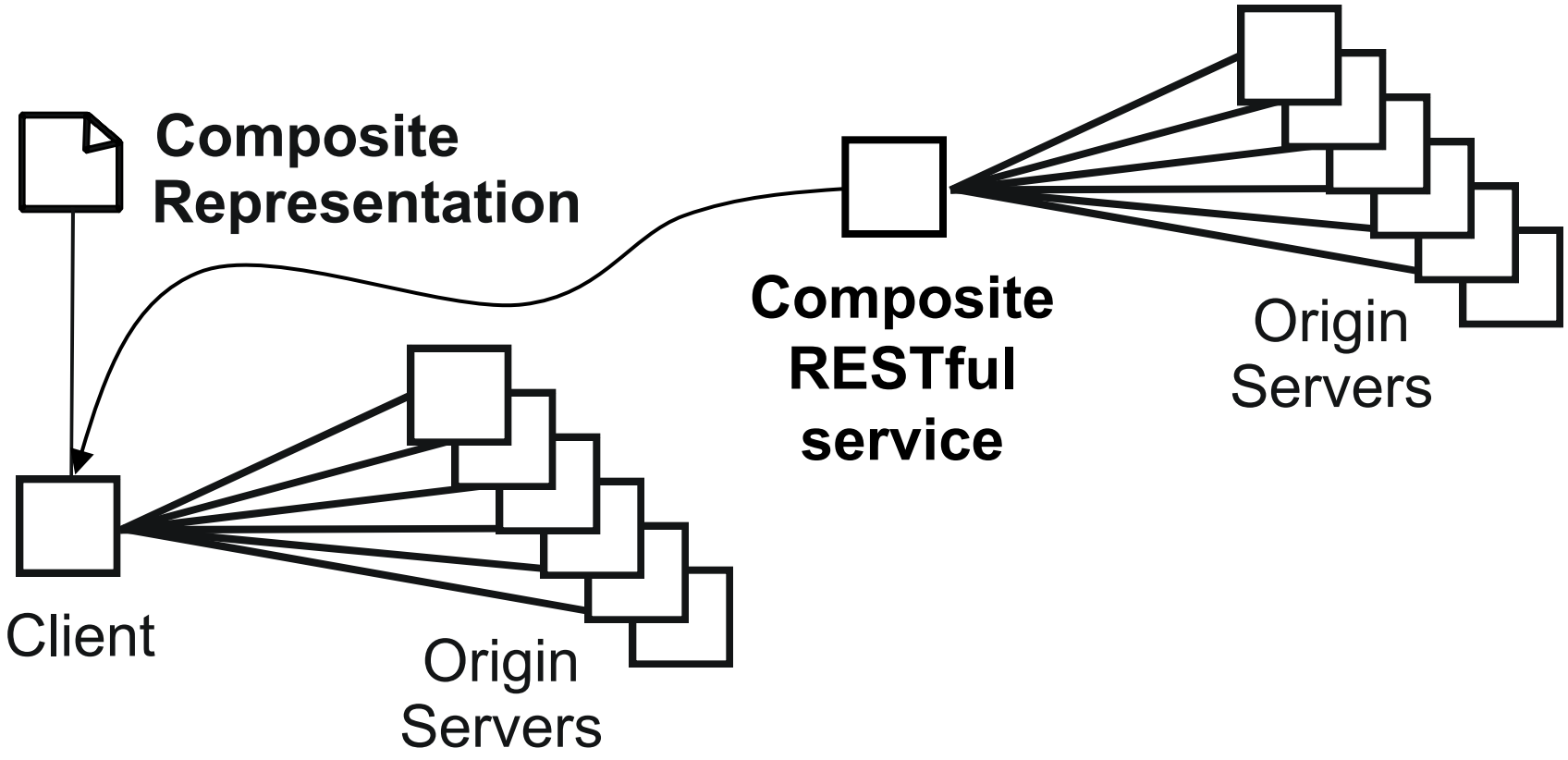


Composite Representation



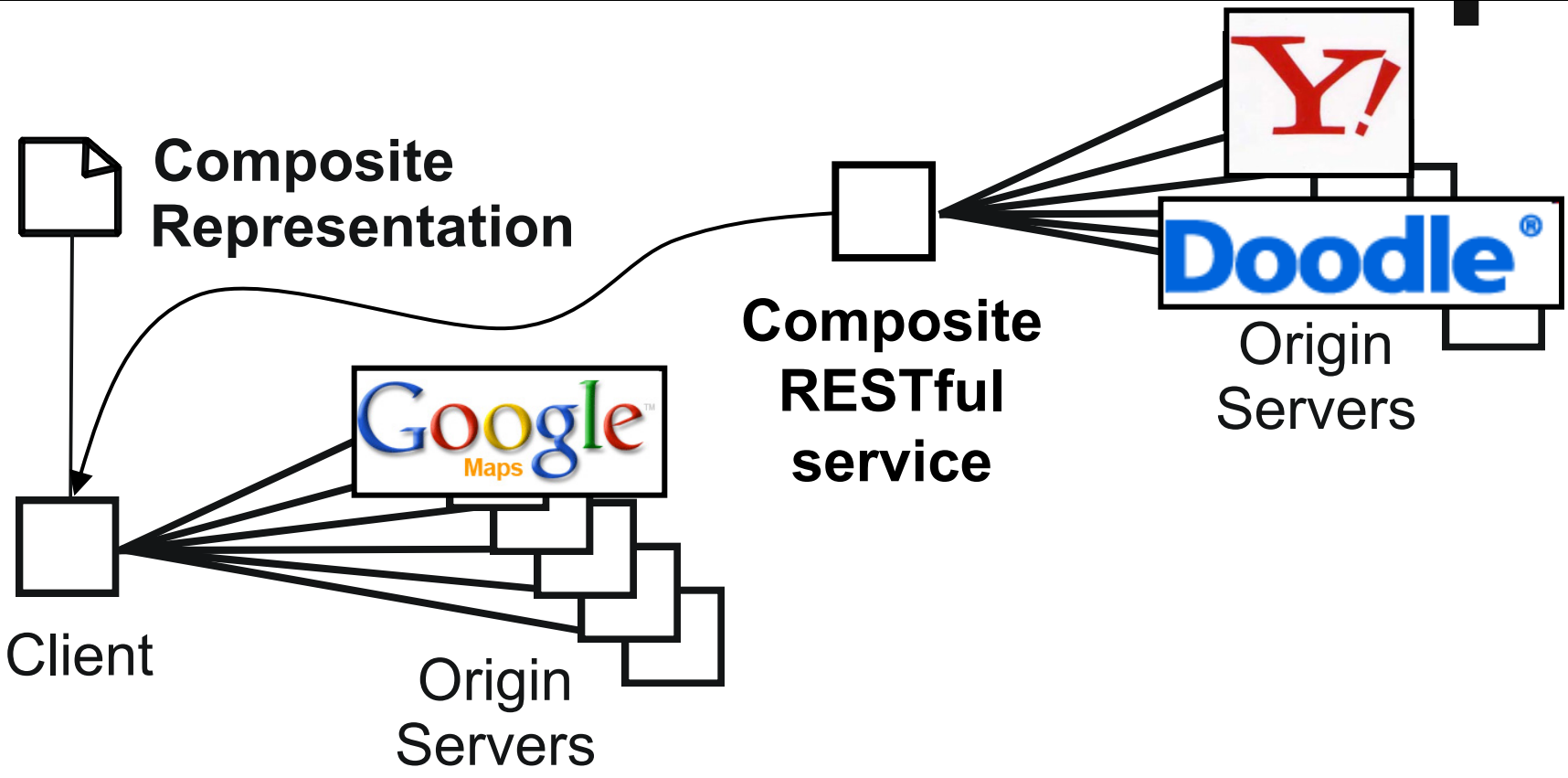
- A composite representation is interpreted by the client that follows its hyperlinks and aggregates the state of the referenced component resources

Bringing it all together



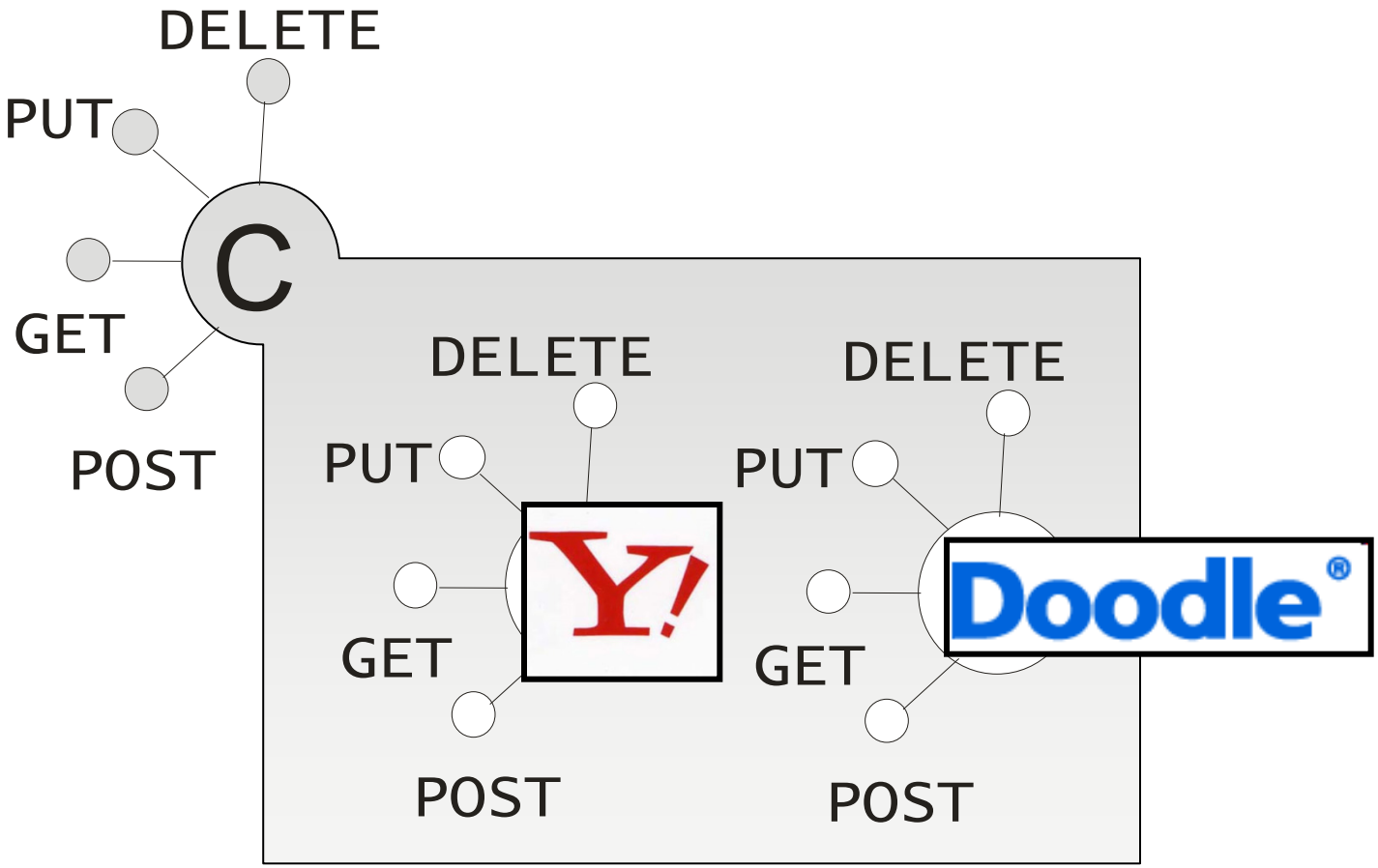
- A composite representation can be produced by a composite service too

Doodle Map Example

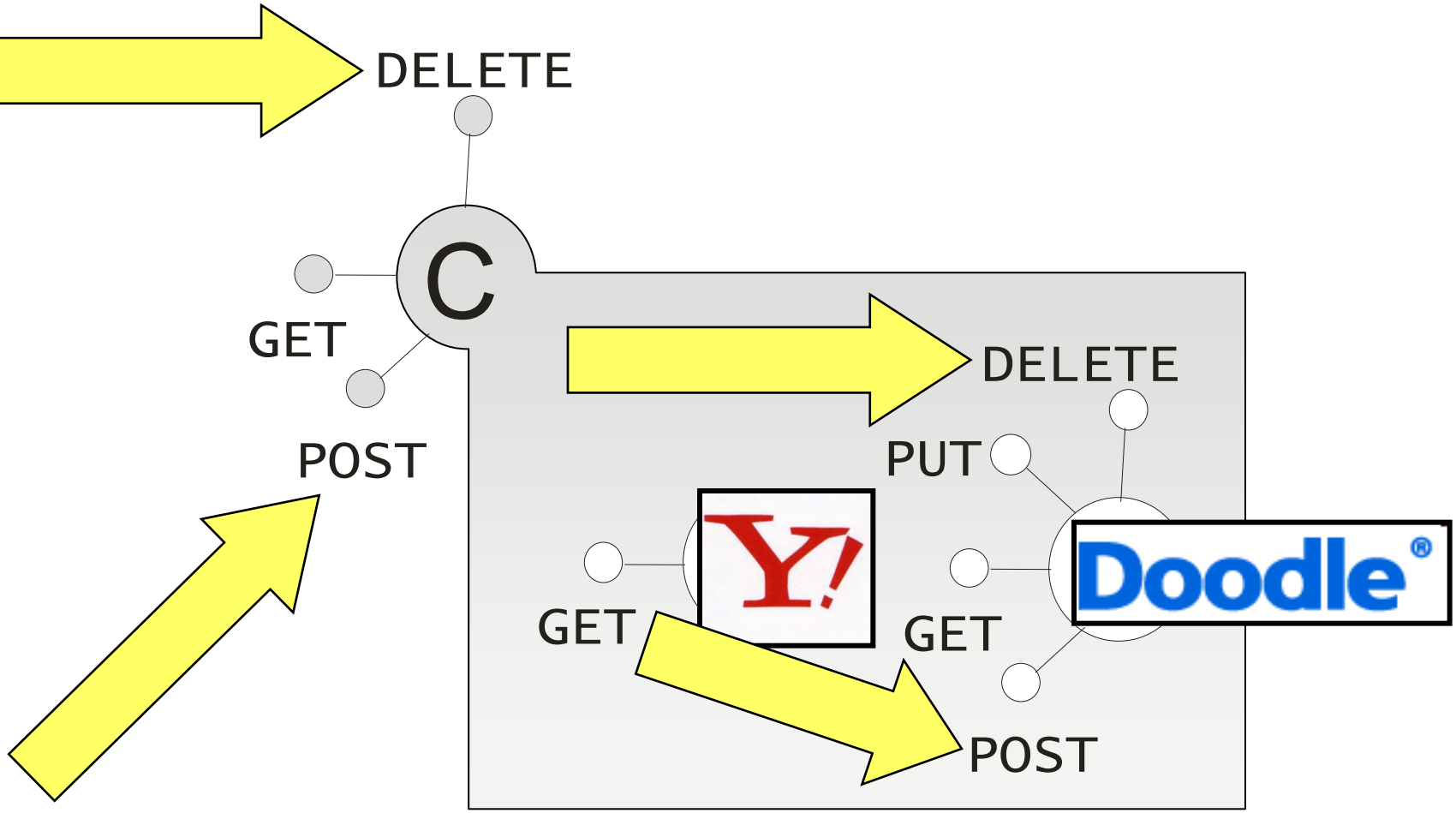


- Vote on a meeting place based on its geographic location

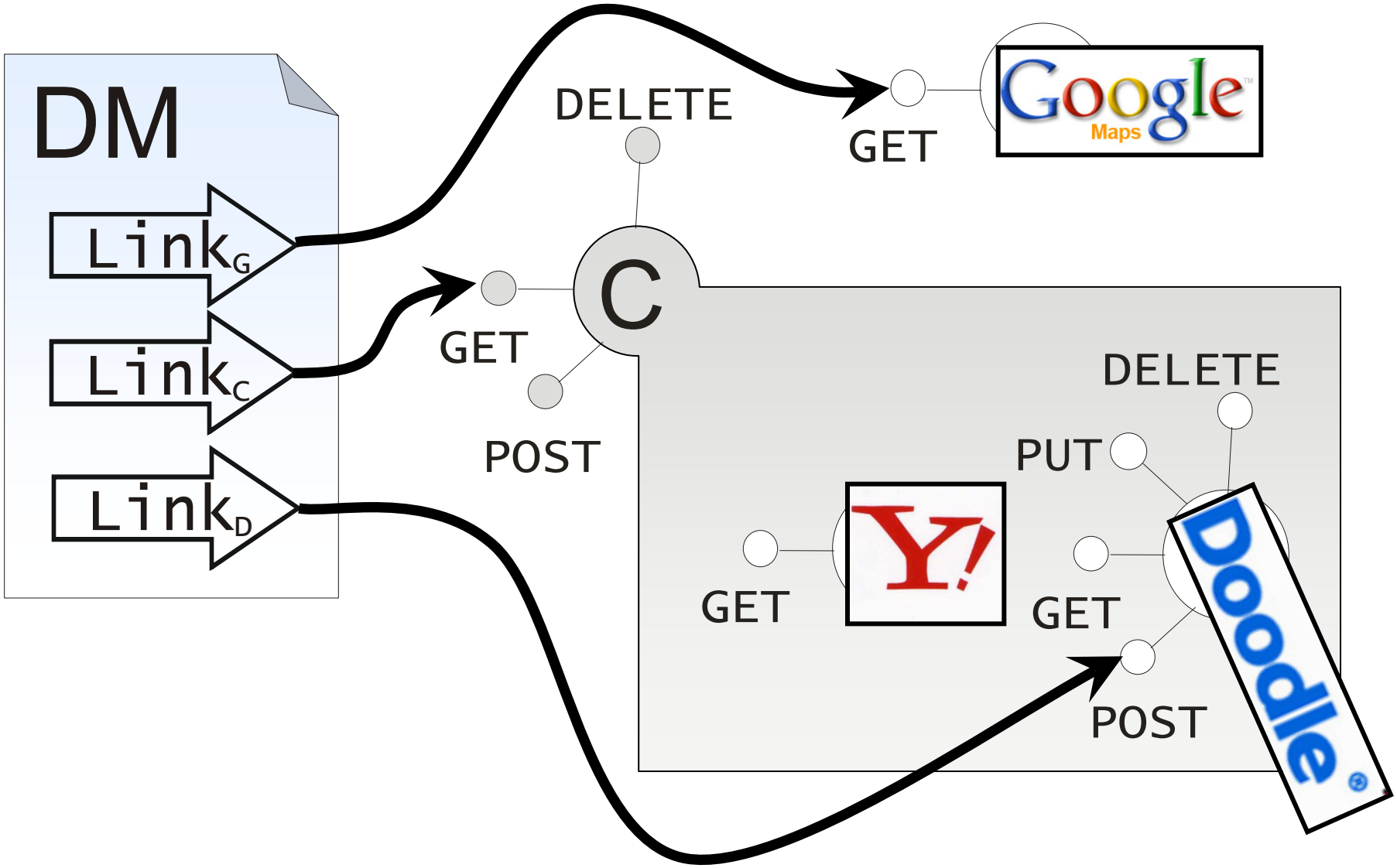
1. Composite Resource



1. Composite Resource



2. Composite Representation

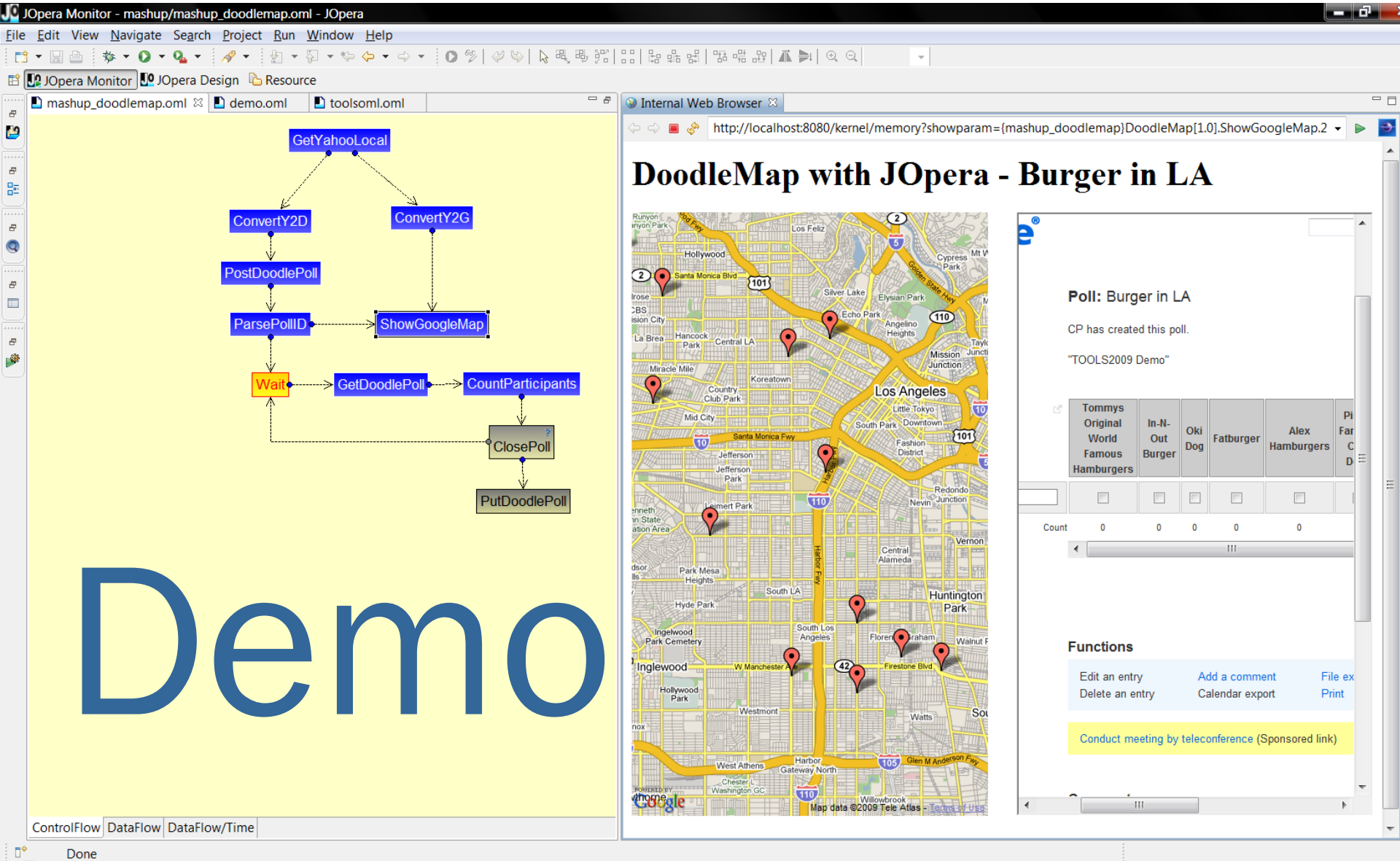


DoodleMap with JOpera

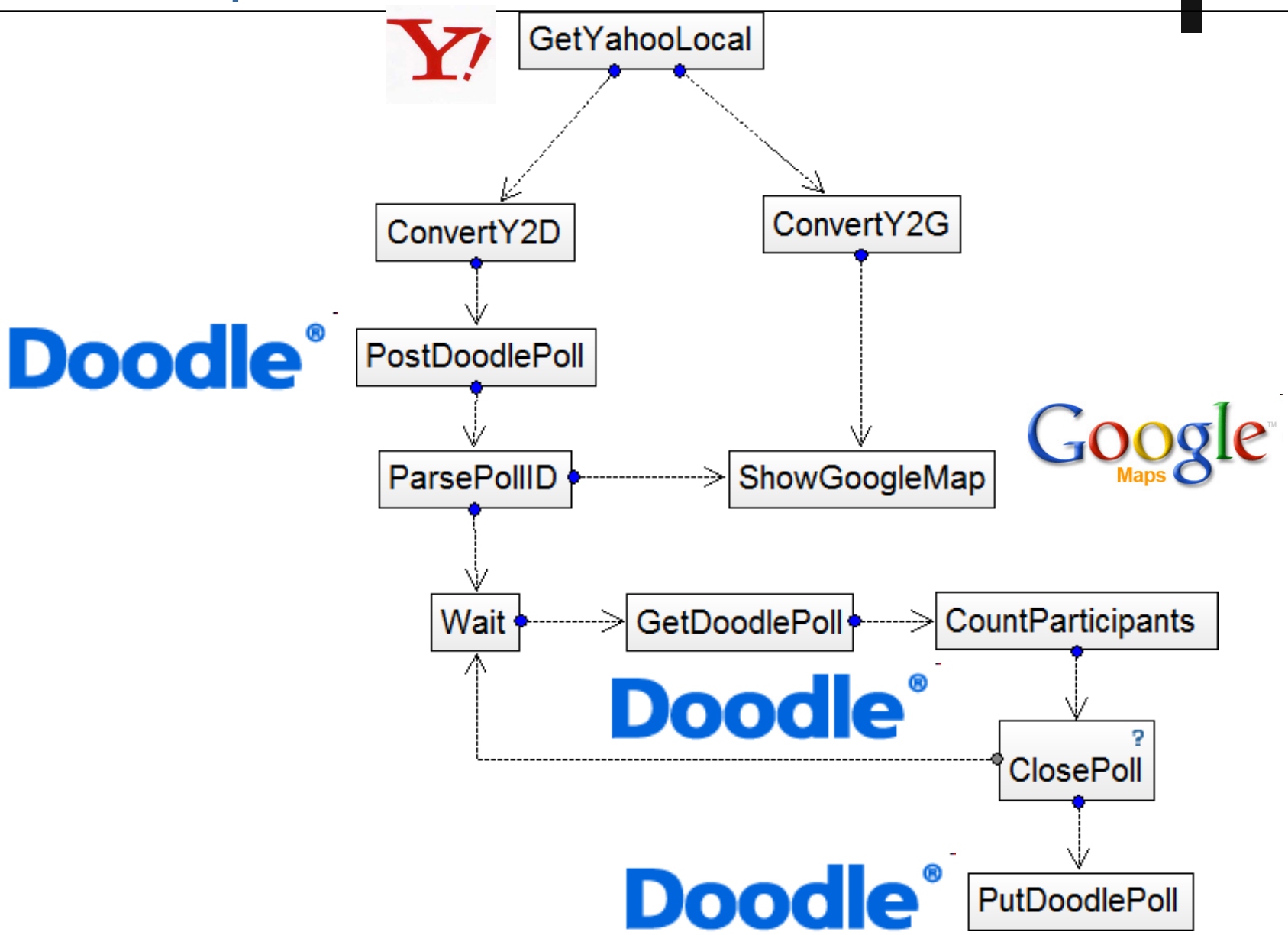
Process Support for Web Services

QCon

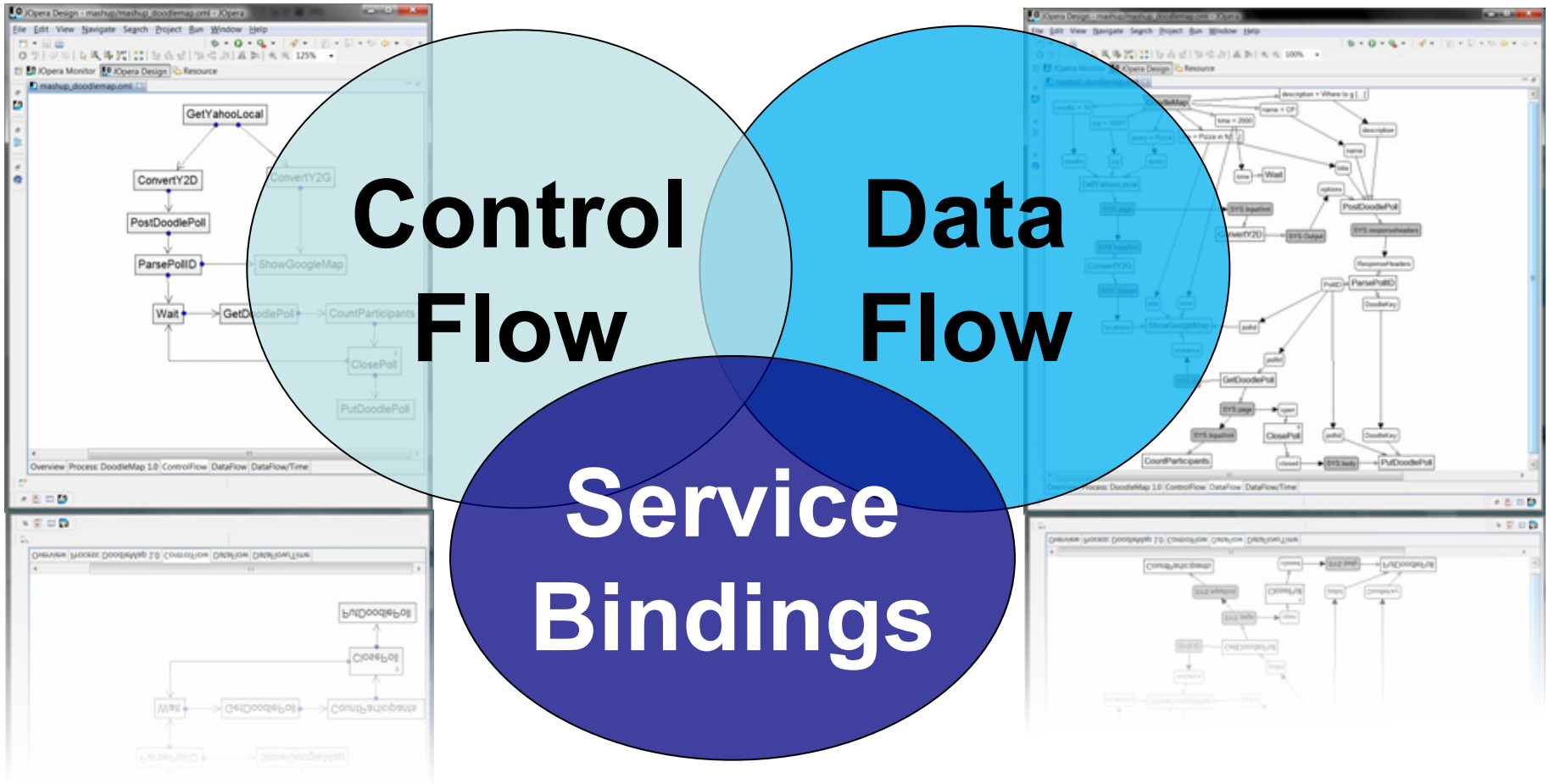
Università della Svizzera italiana



DoodleMap Workflow

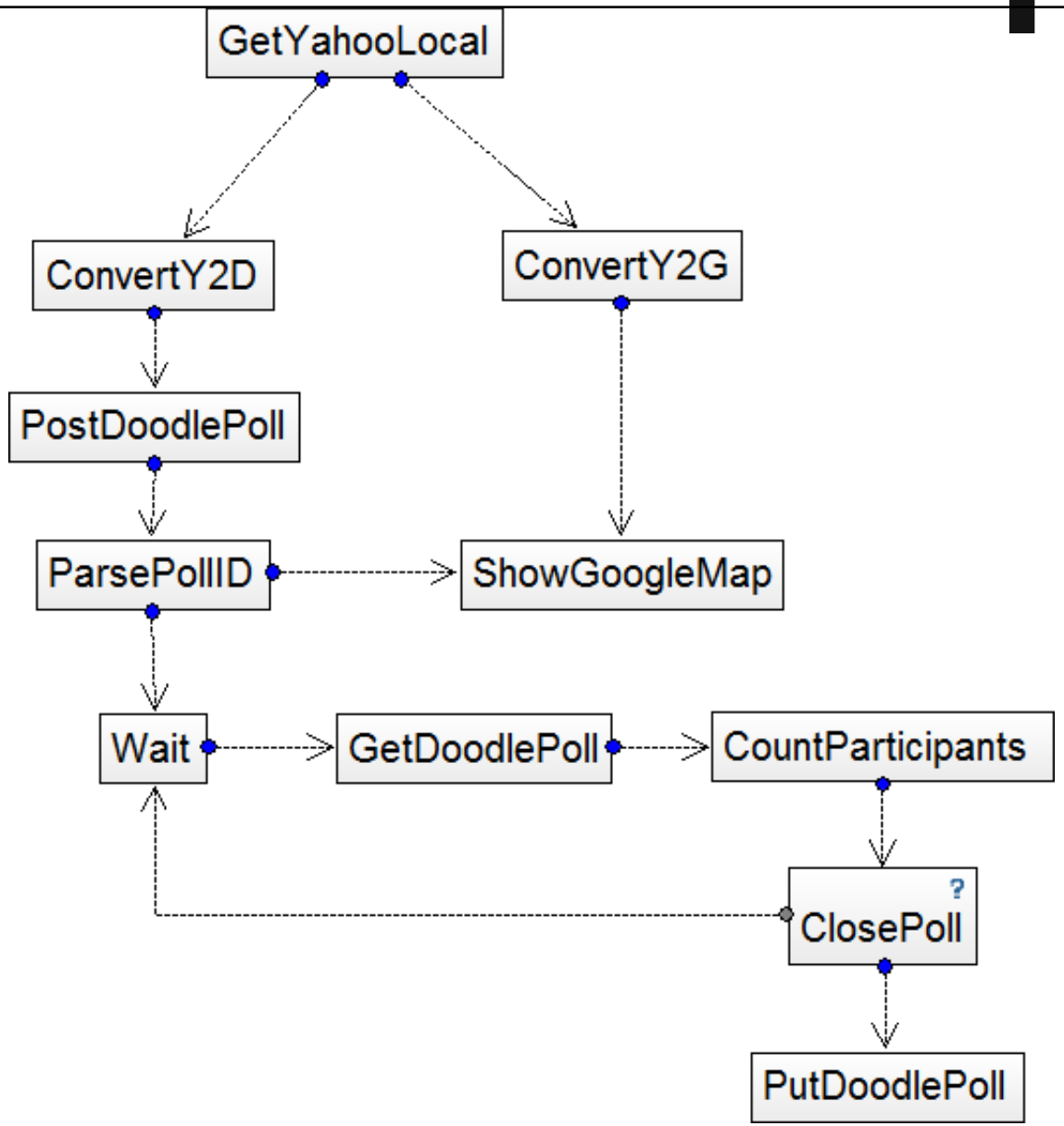


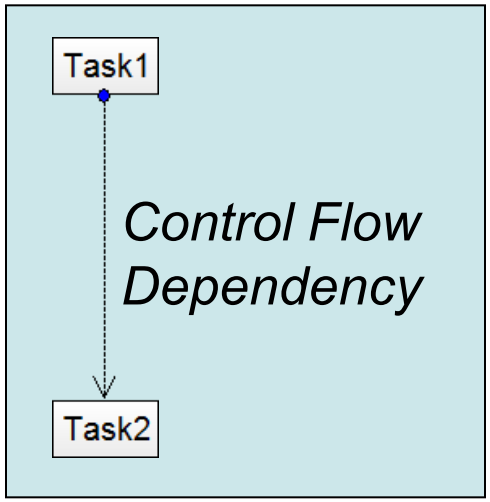
Viewpoints



JAVA | XPATH | XSLT | HTML | HTTP | ...

Control Flow





Service Bindings

HTTP

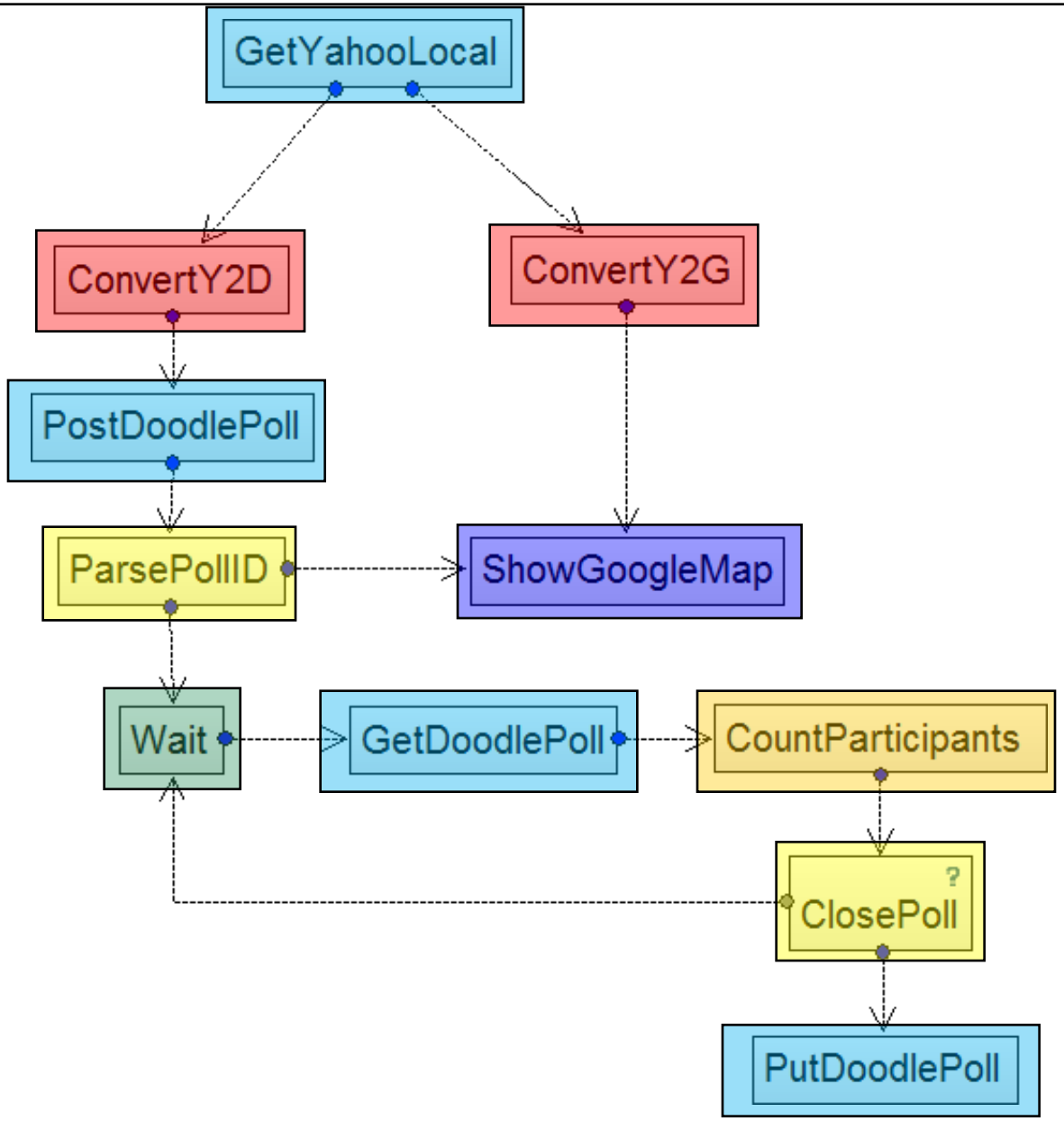
HTML

XSLT

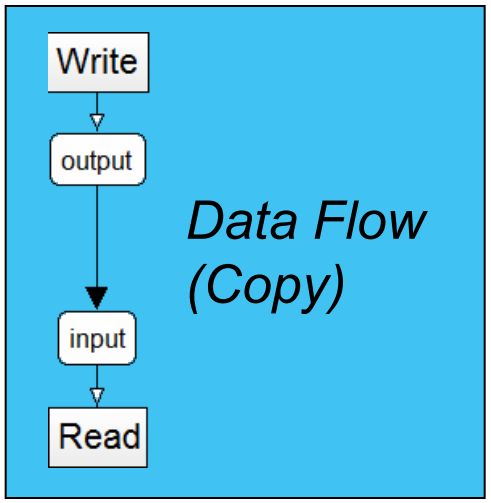
XPATH

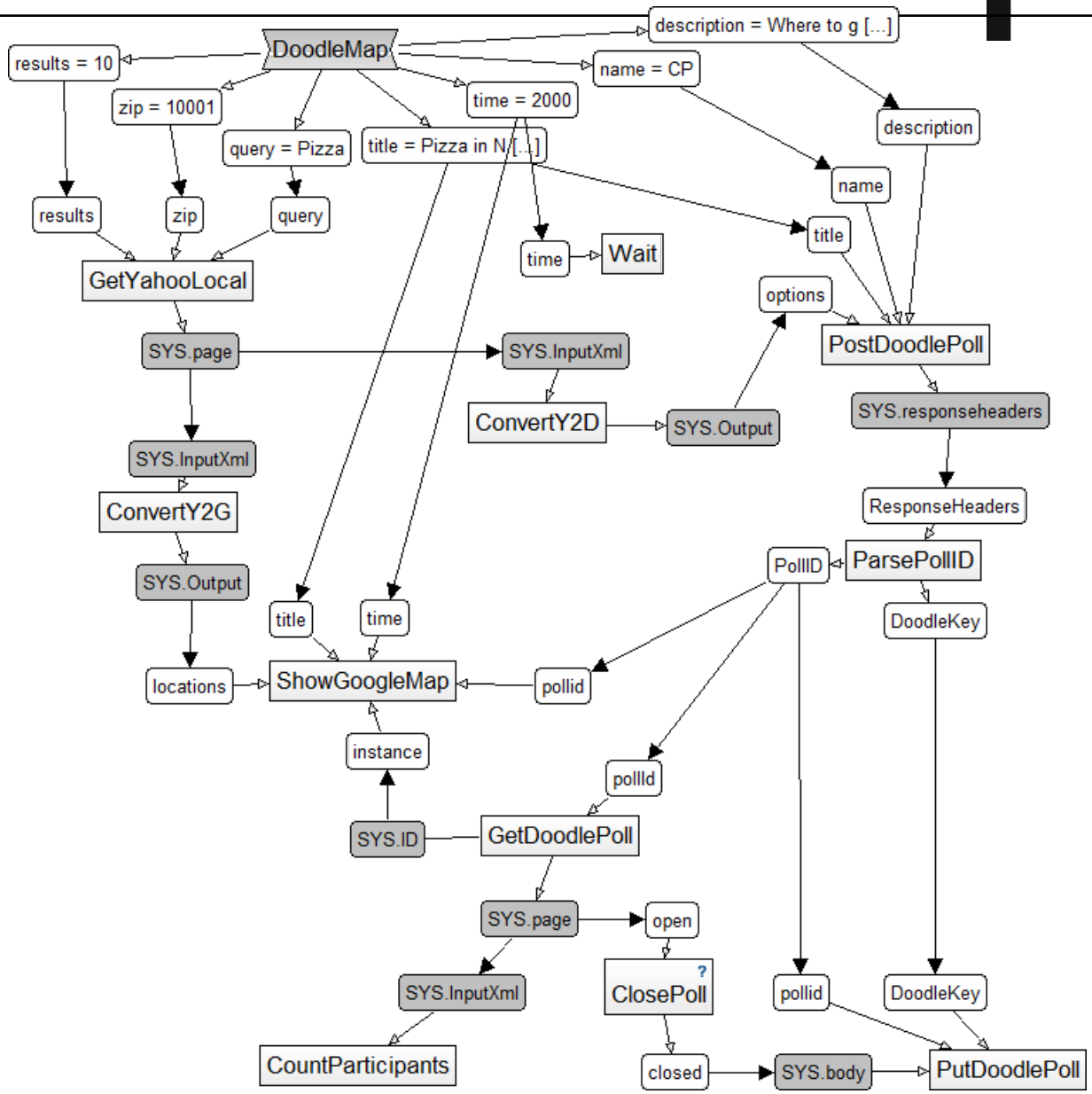
JAVA

...

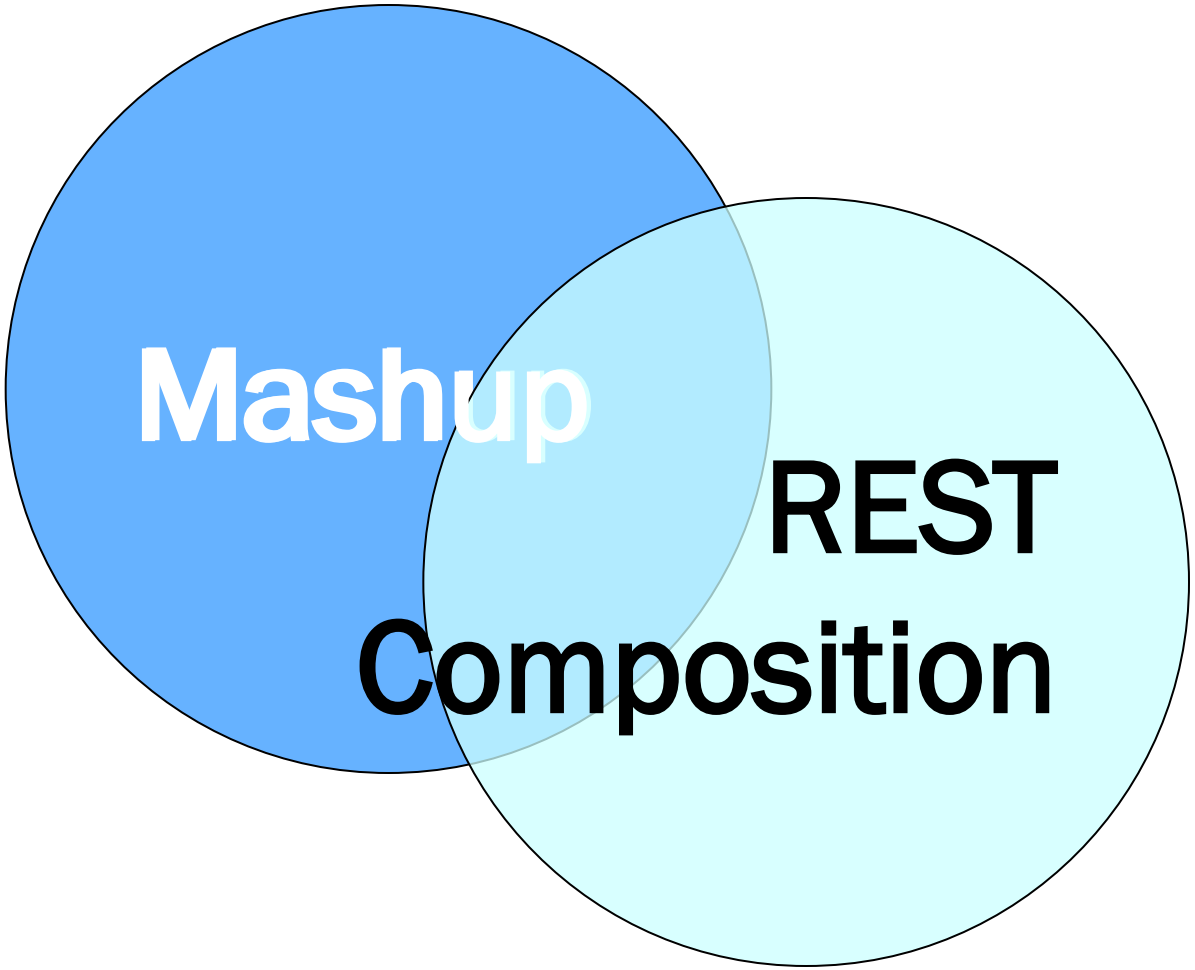


Data Flow


Write
output
input
Read
Data Flow (Copy)


DoodleMap
description = Where to g [...]
results = 10
name = CP
zip = 10001
time = 2000
query = Pizza
title = Pizza in N [...]
description
name
results
zip
query
title
time
Wait
GetYahooLocal
options
SYS.page
SYS.InputXml
PostDoodlePoll
ConvertY2D
SYS.Output
SYS.responseheaders
SYS.InputXml
ResponseHeaders
ConvertY2G
PollID
ParsePollID
SYS.Output
title
time
DoodleKey
locations
ShowGoogleMap
pollid
instance
pollId
SYS.ID
GetDoodlePoll
SYS.page
open
SYS.InputXml
ClosePoll
pollid
DoodleKey
CountParticipants
closed
SYS.body
PutDoodlePoll

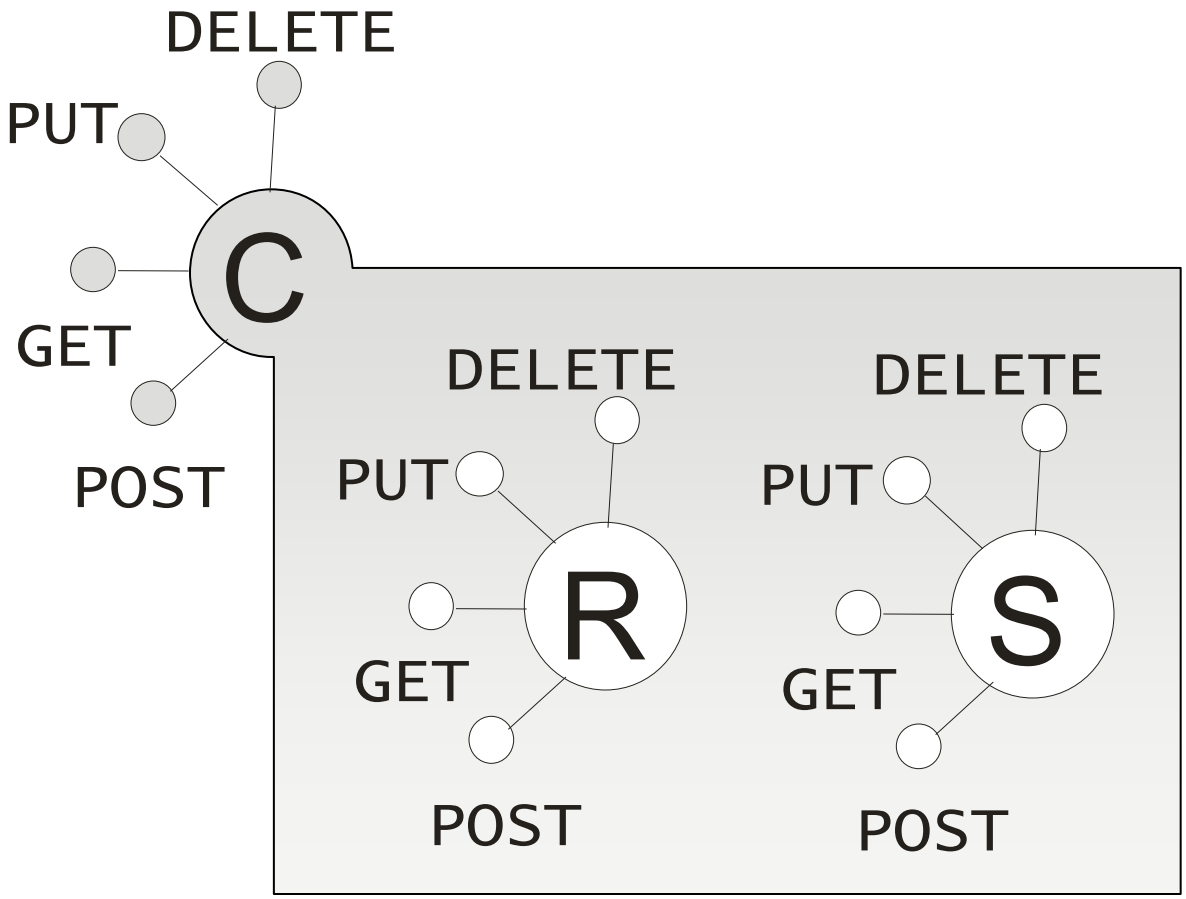
BPM for Mashups?



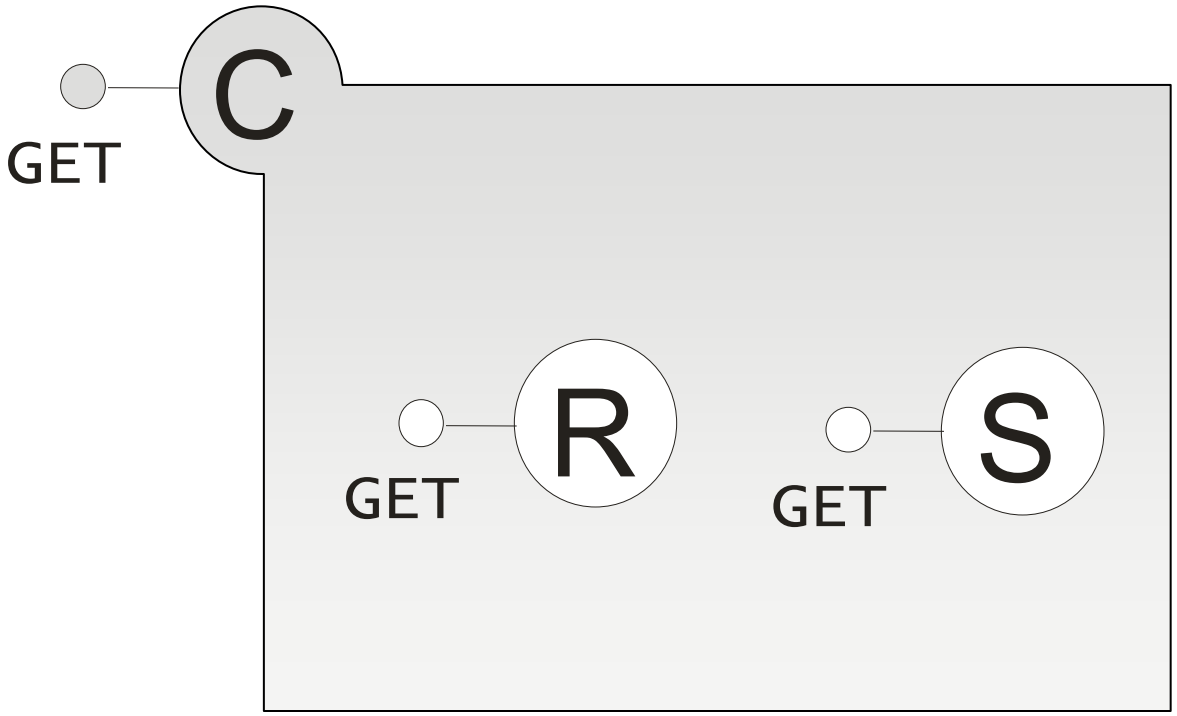
(It depends on the definition of Mashup)

Moving state around

- Read-only vs. Read/Write

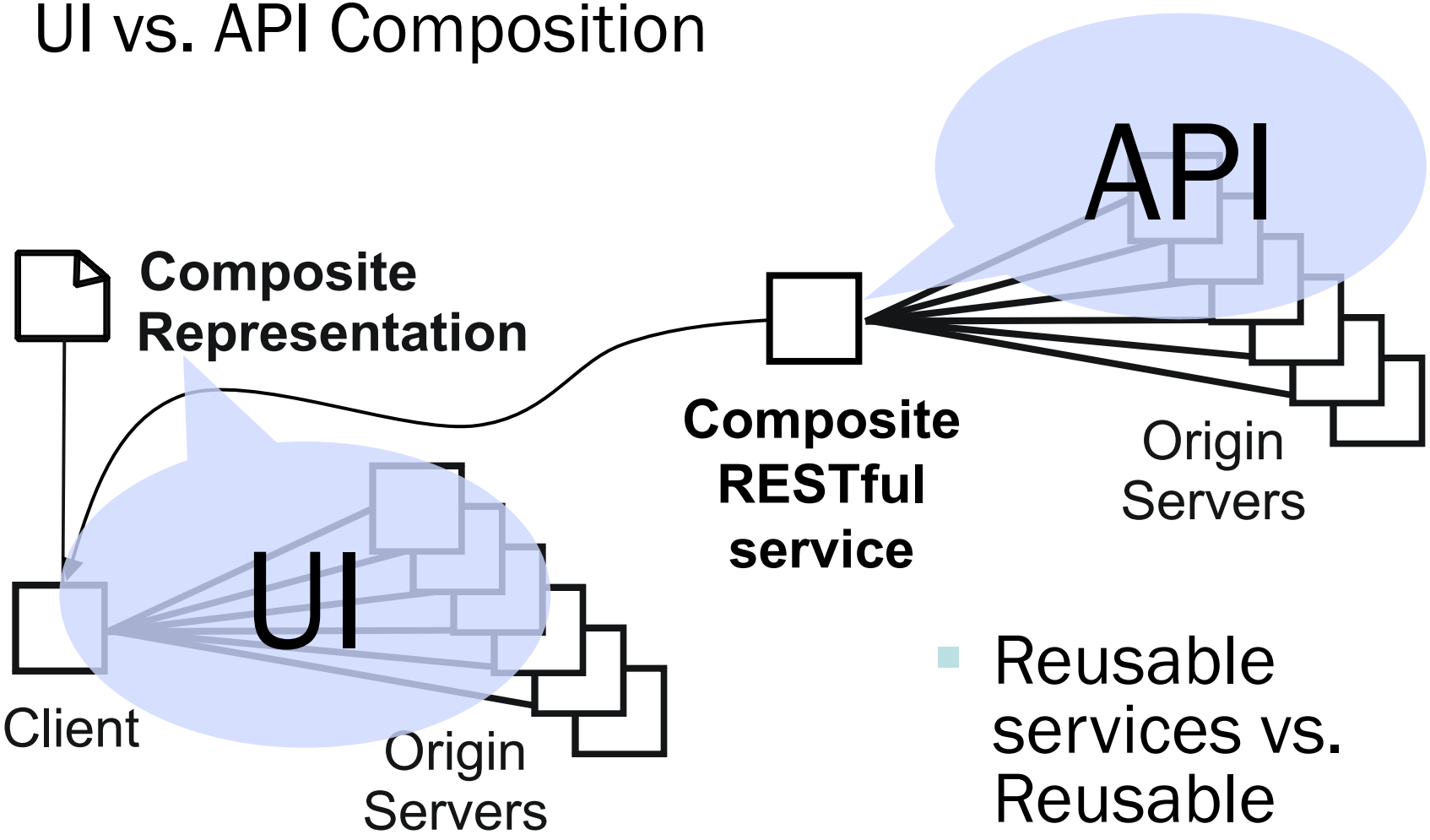


- Read-only vs. Read/write



Is your composition reusable?

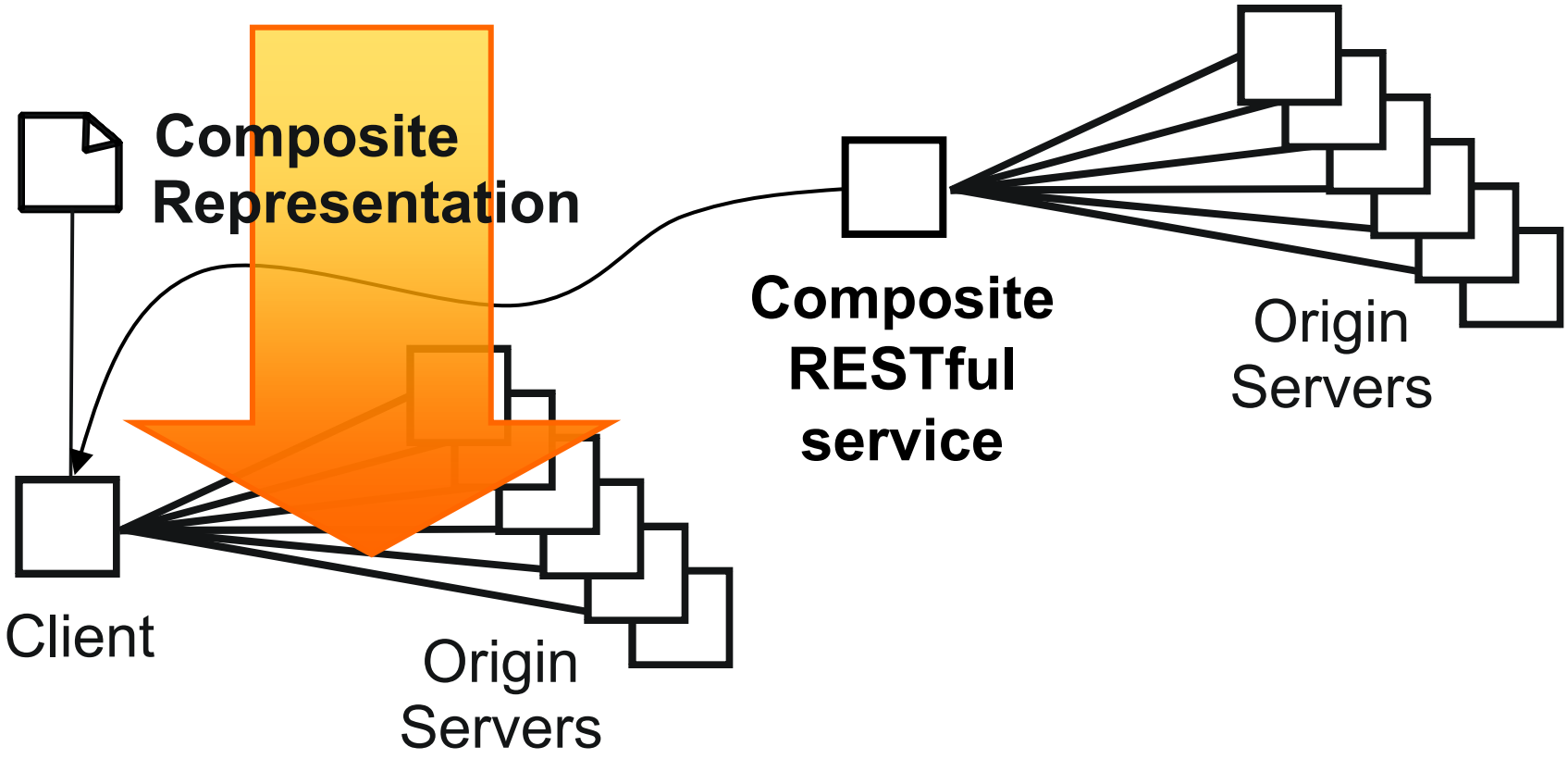
- UI vs. API Composition



- Reusable services vs. Reusable Widgets

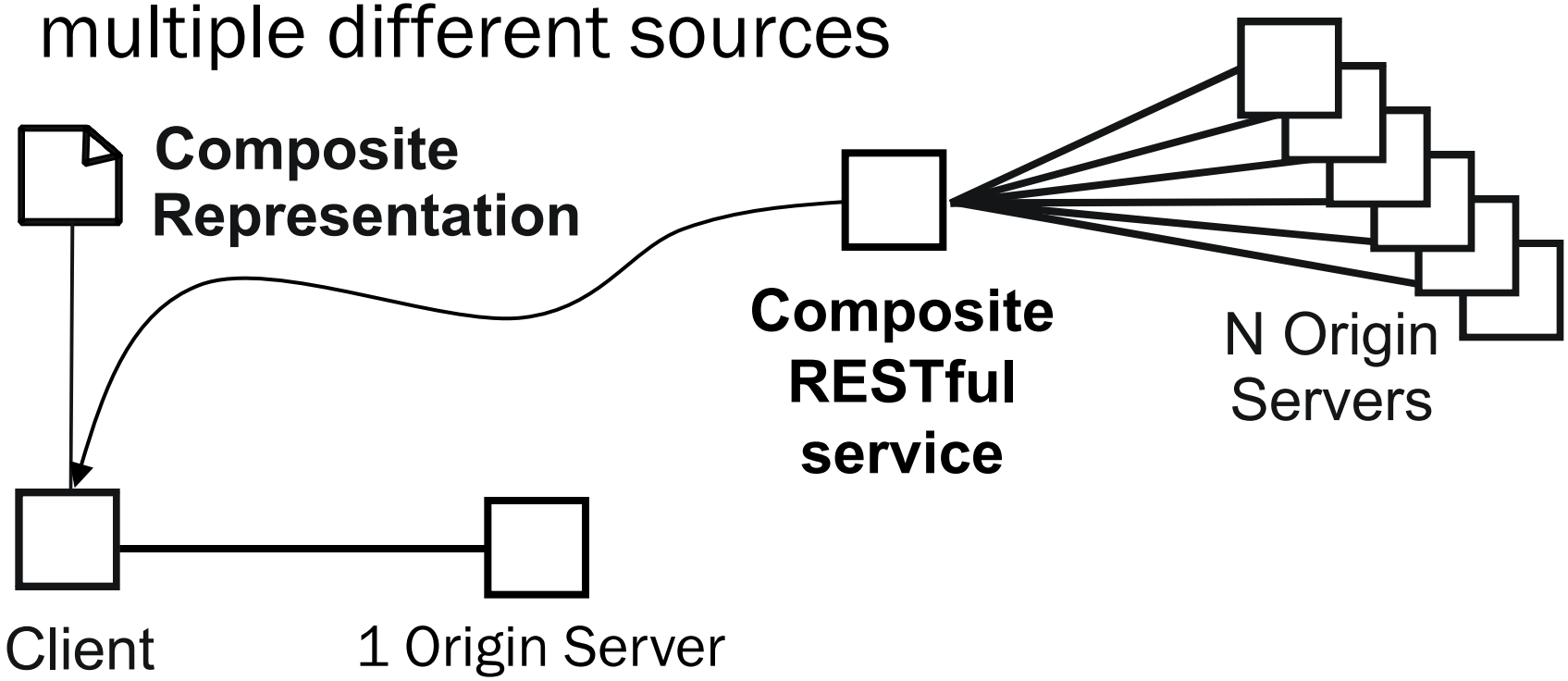
Single-Origin Sandbox

- Can you always do this from a web browser?

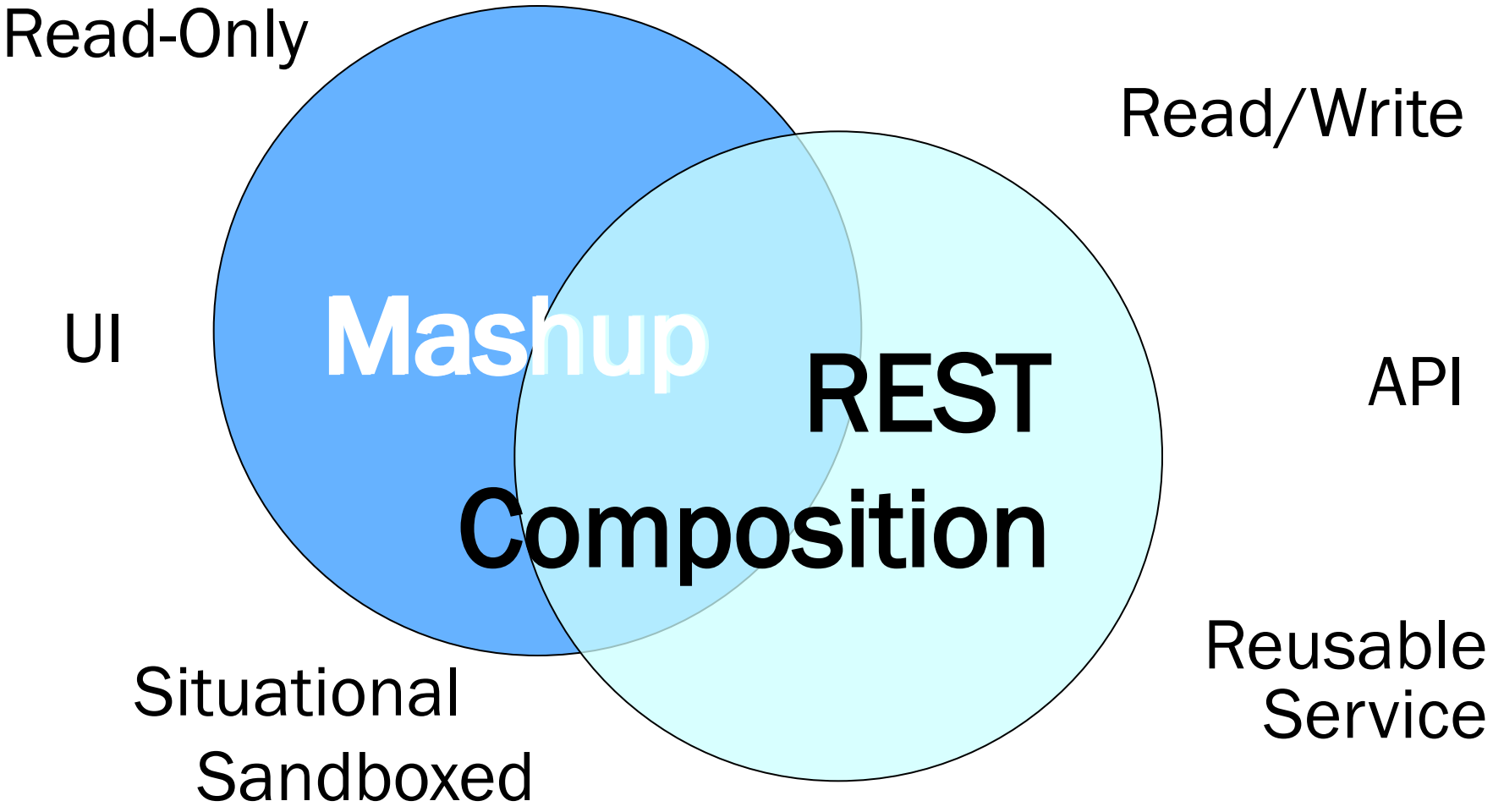


Single-Origin Sandbox

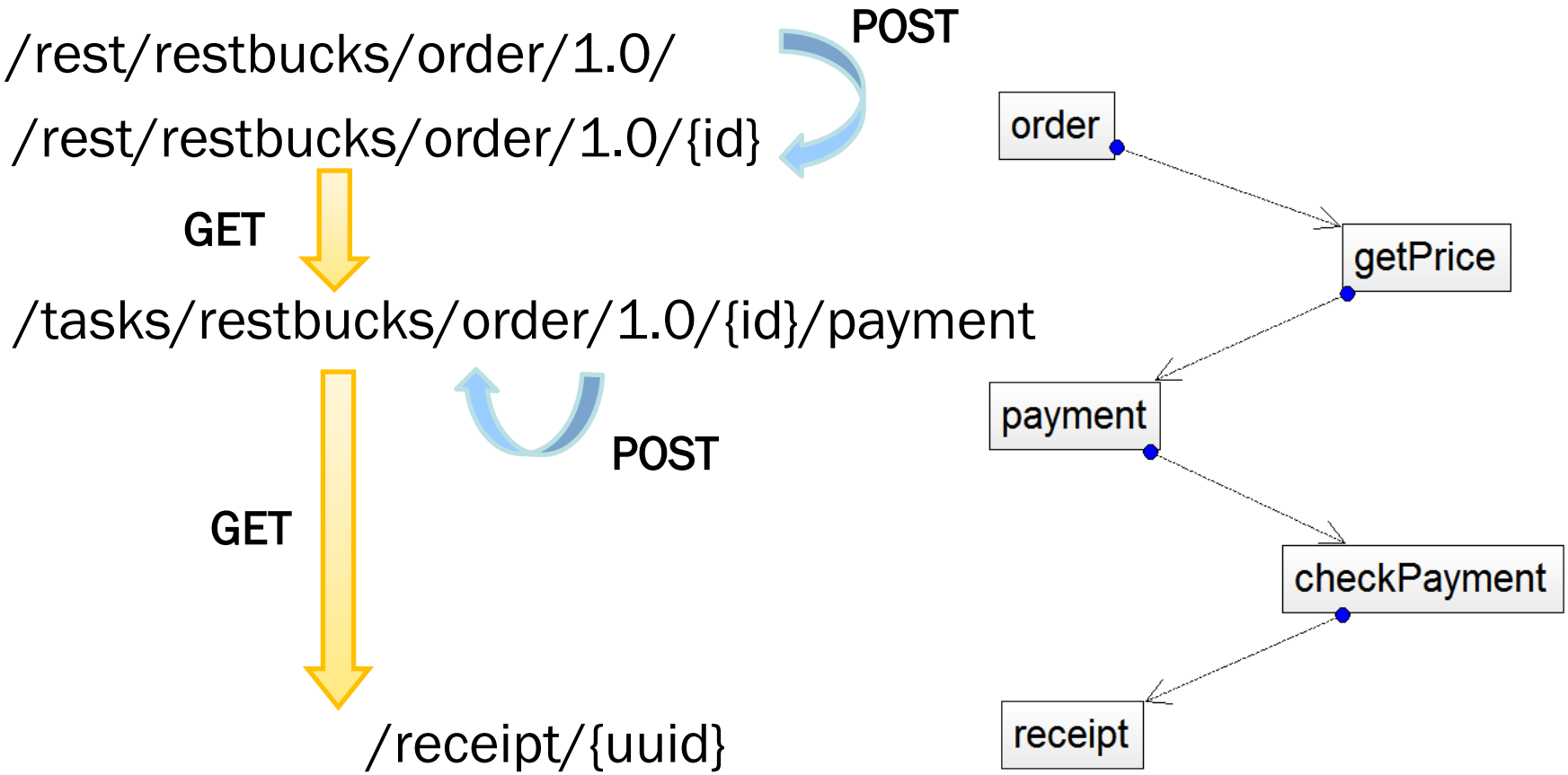
- Security Policies on the client may not always allow it to aggregate data from multiple different sources



Complementary

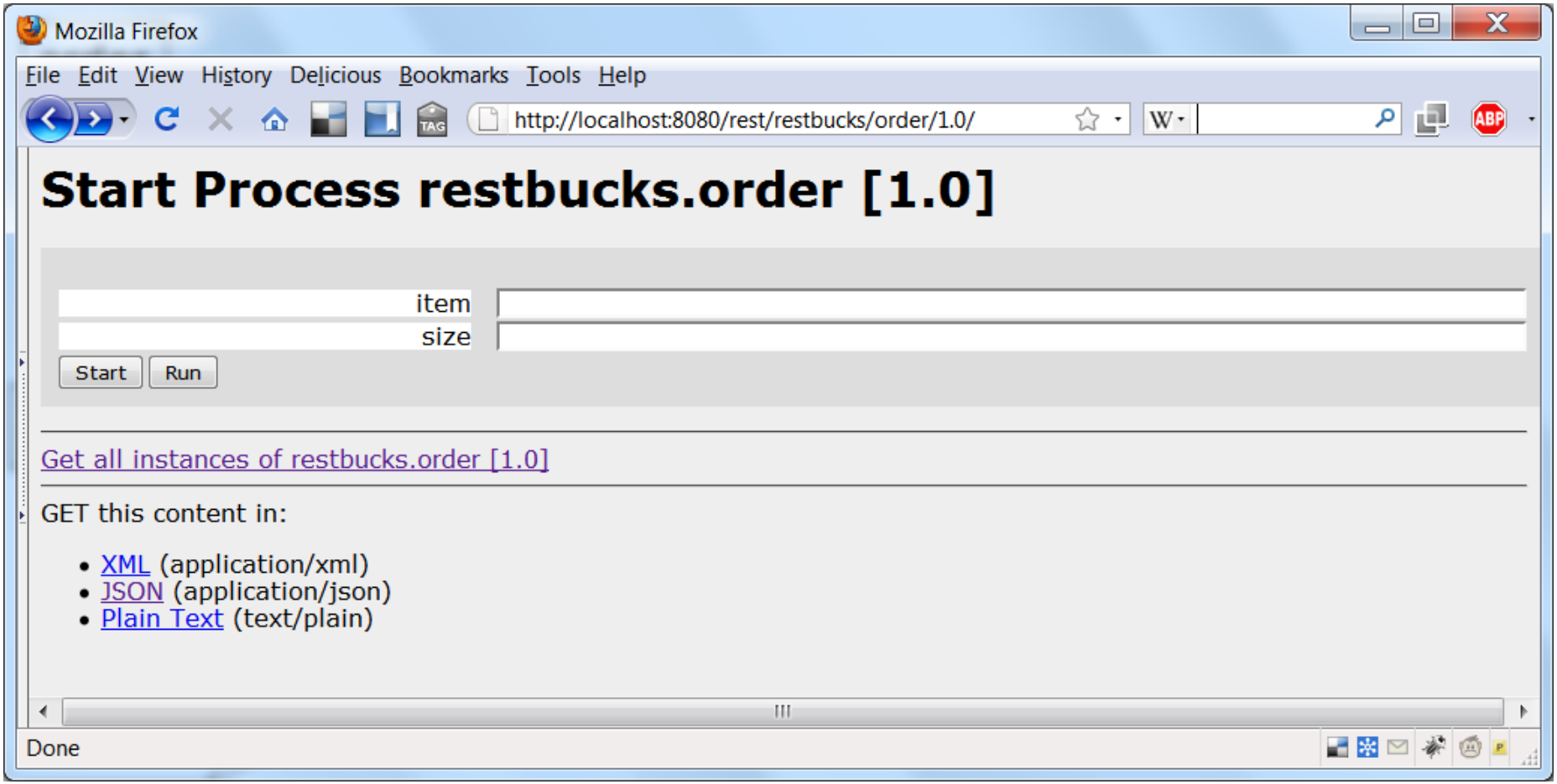


TinyRESTBucks Example



Instantiating a process

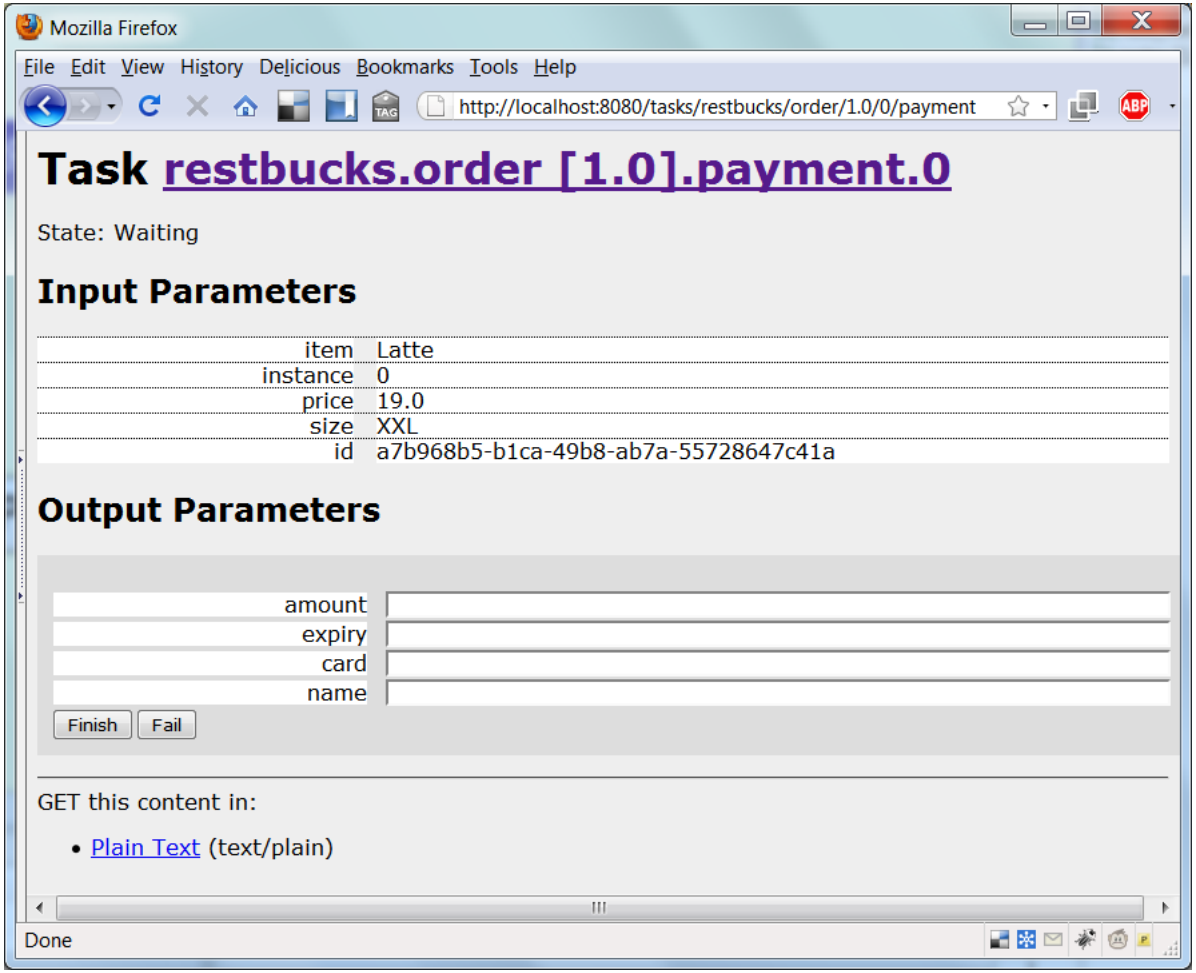
GET /rest/restbucks/order/1.0/

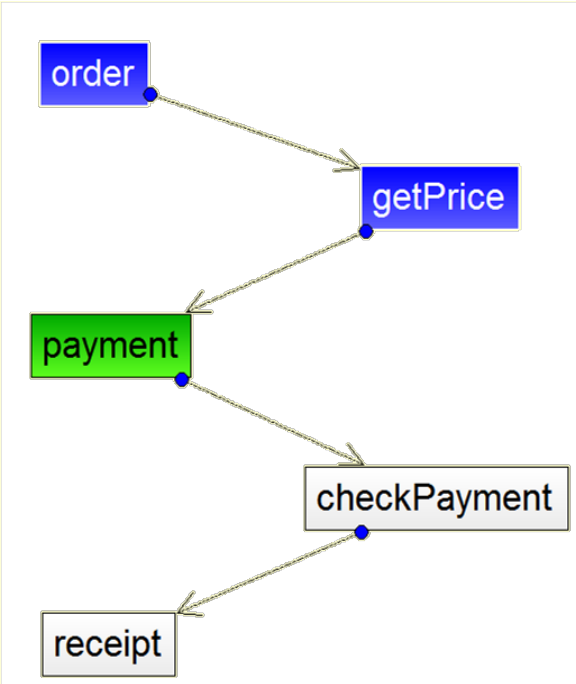


Interacting with a task

QCon Università della Svizzera italiana

GET /rest/restbucks/order/1.0/0/payment

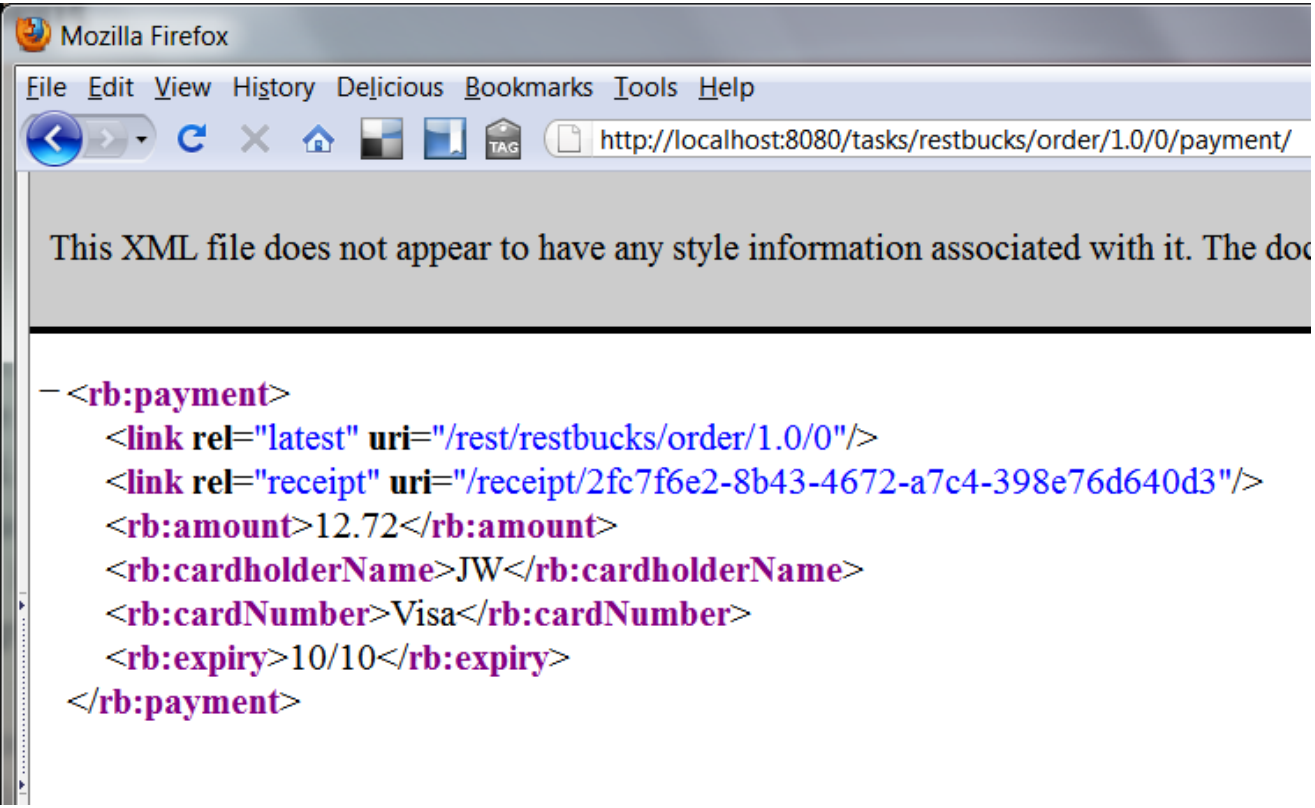


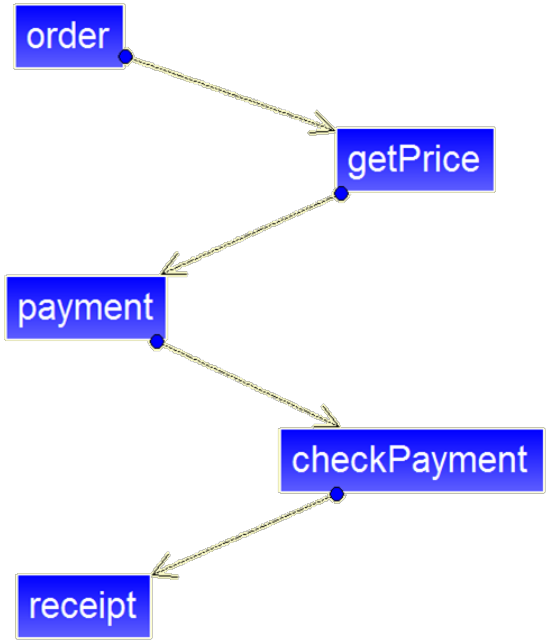


Interacting with a task

QCon Università della Svizzera italiana

POST /rest/restbucks/order/1.0/0/payment





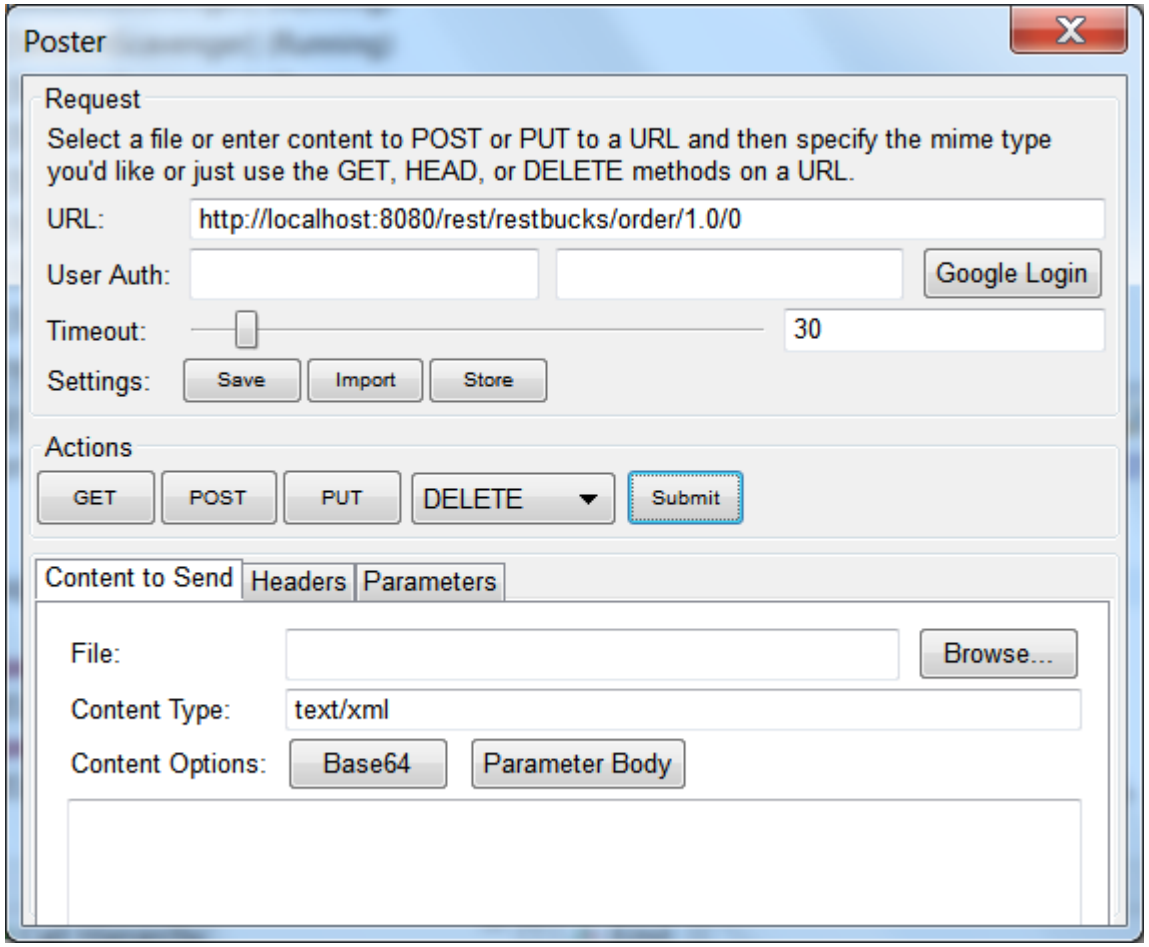
Interacting with a resource

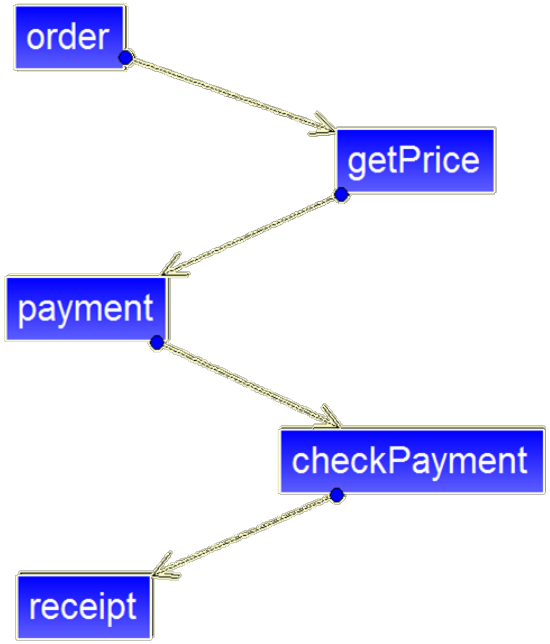
GET /receipt/2fc7f6e2-8b43-4672-a7c4...

Interacting with a resource

QCon Università della Svizzera italiana

DELETE /rest/restbucks/order/1.0/0



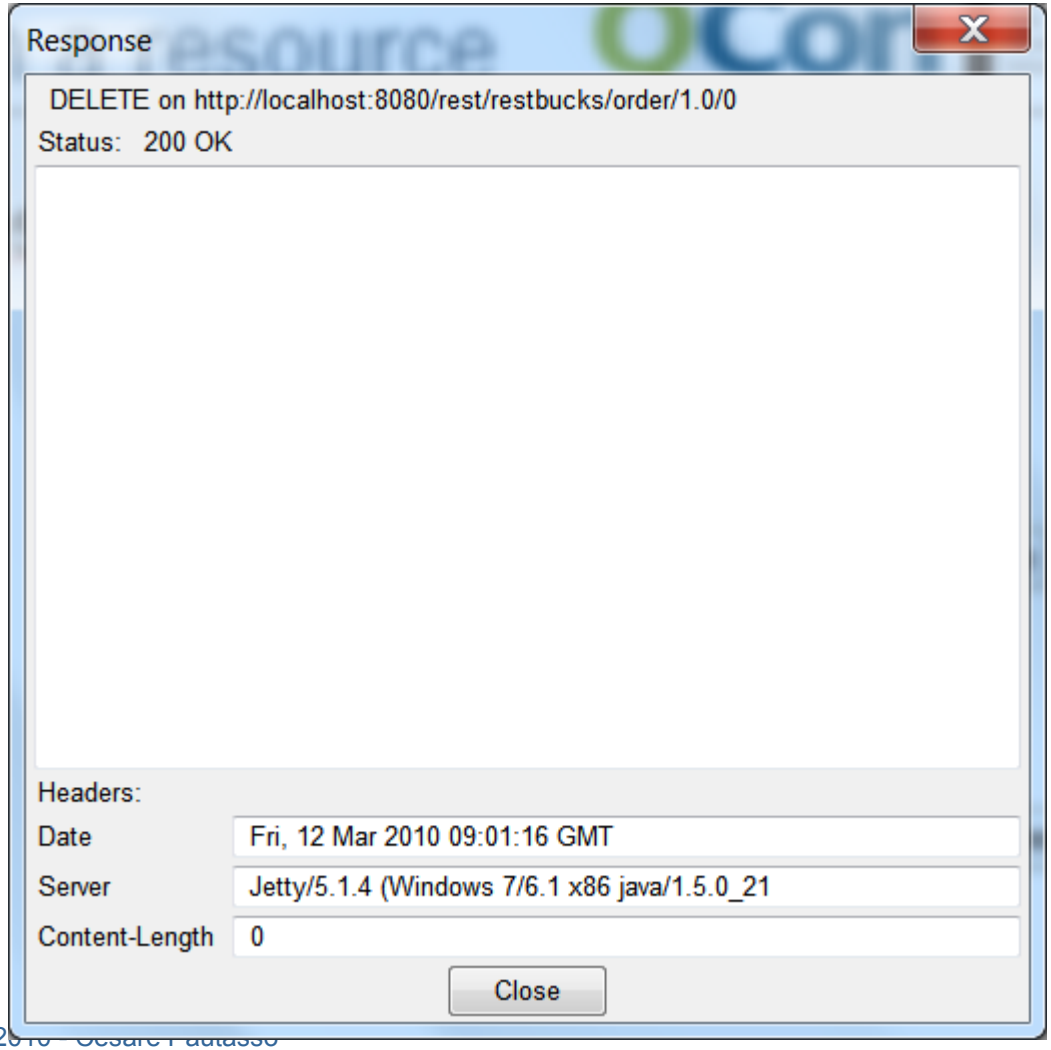


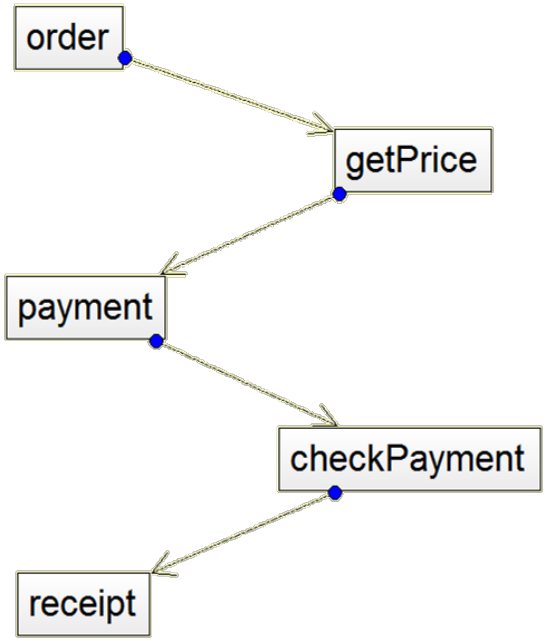
Deleting a process resource

QCon

Università della Svizzera italiana

DELETE /rest/restbucks/order/1.0/0





Service Bindings

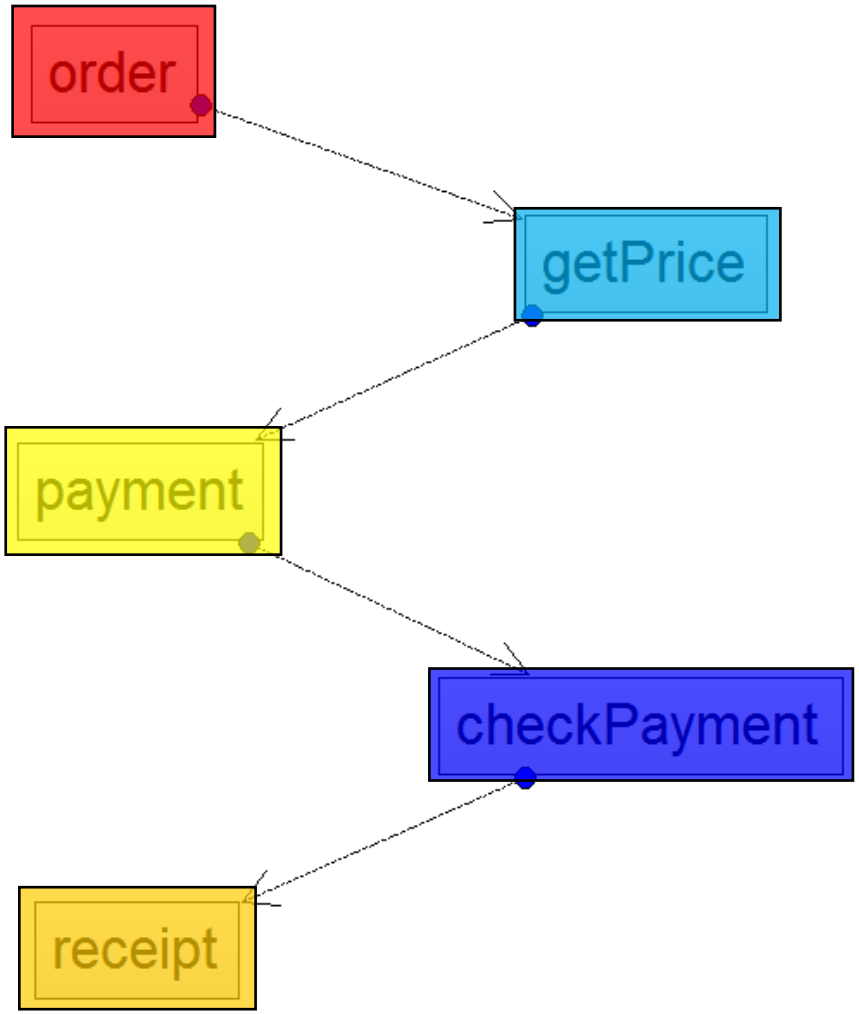
SQL

WS-*

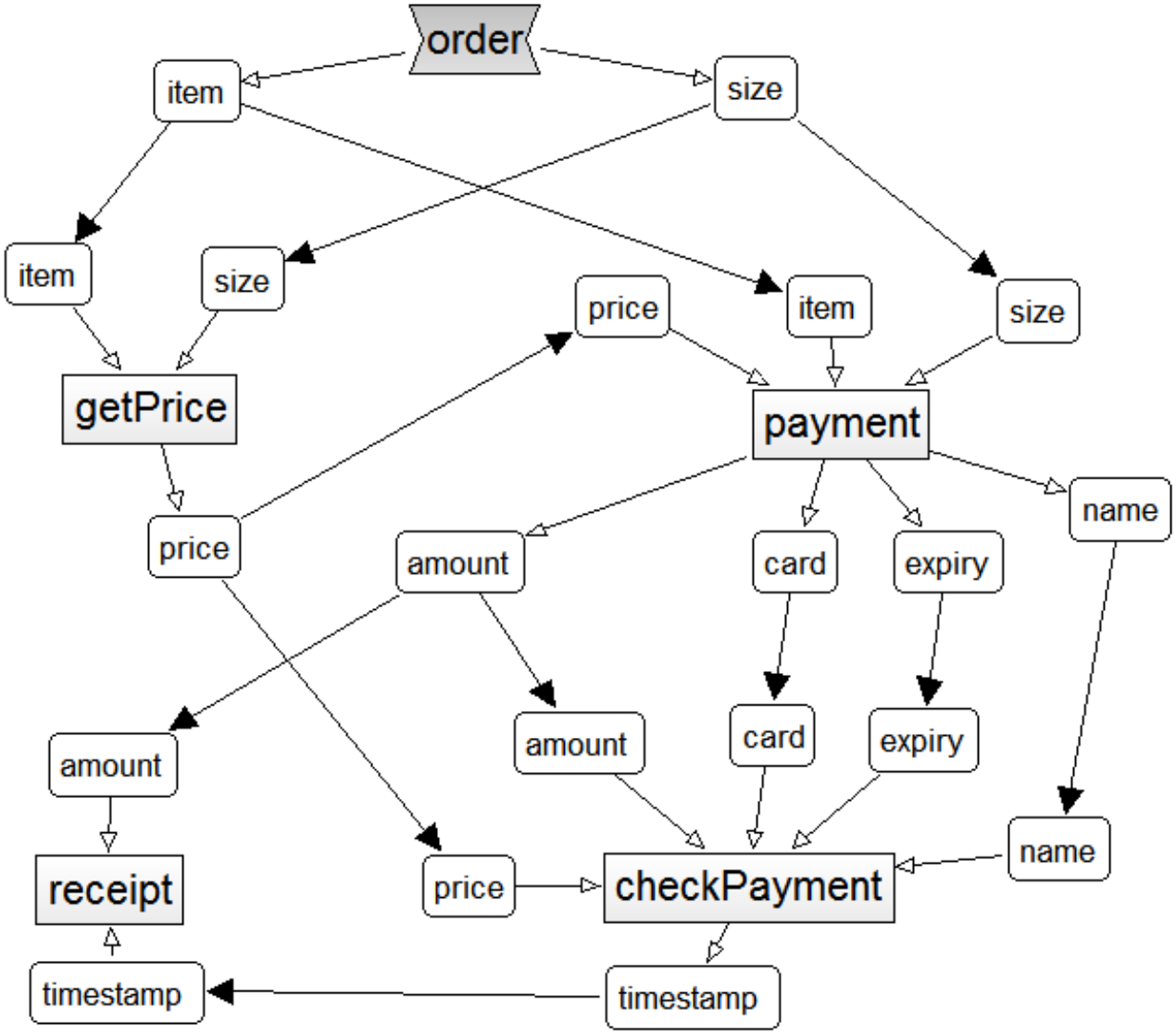
REST

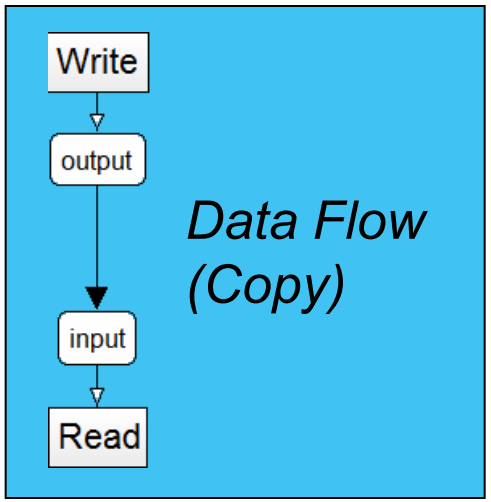
REST.URI

REST.TASK

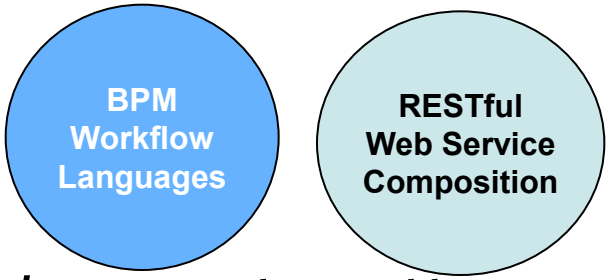


Data Flow

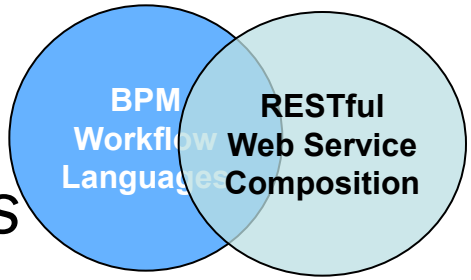




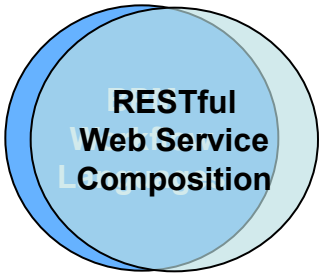
Solution Space



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*



Conclusions

- RESTful HTTP is good enough to interact **without any extension** with process execution engines and their processes and tasks published as resources
- RESTful Web service composition is different than mashups, but both can be built using BPM
- If done right, BPM can be a great modeling tool for Hypermedia-centric service design **(and implementation!)**
- GET http://www.jopera.org/

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), Bejing, China, April 2008
- C. Pautasso, BPEL for REST, Proc. of the 7th International Conference on Business Process Management (BPM 2008), Milano, Italy, September 2008
- C. Pautasso, Composing RESTful Services with JOpera, In: Proc. of the International Conference on Software Composition (SC2009), July 2009, Zurich, Switzerland.