Pragmatic SOA Beyond Buzzwords and Flamewars

Stefan Tilkov, innoQ
@stilkov - http://www.innoq.com/blog/st/

Some Claims

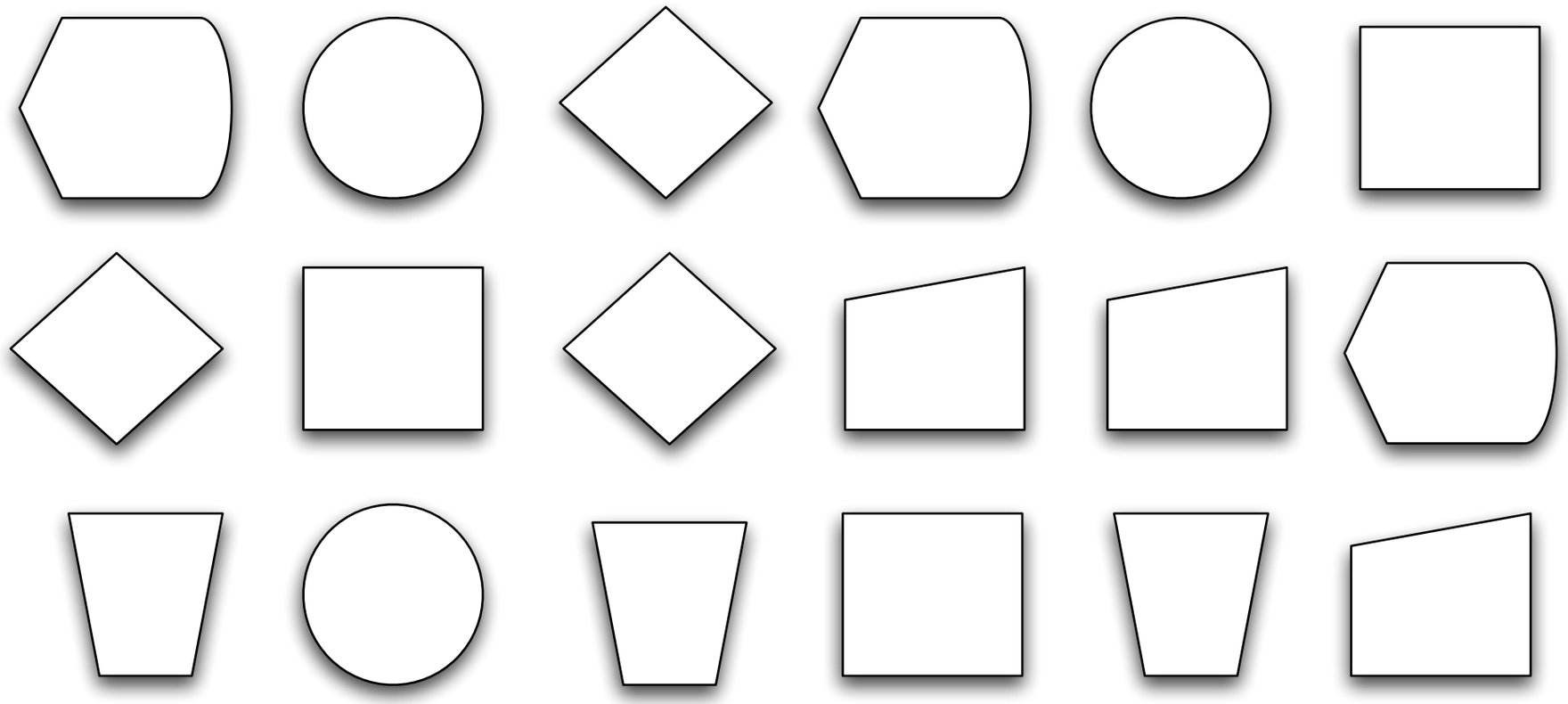
Some Recommendations

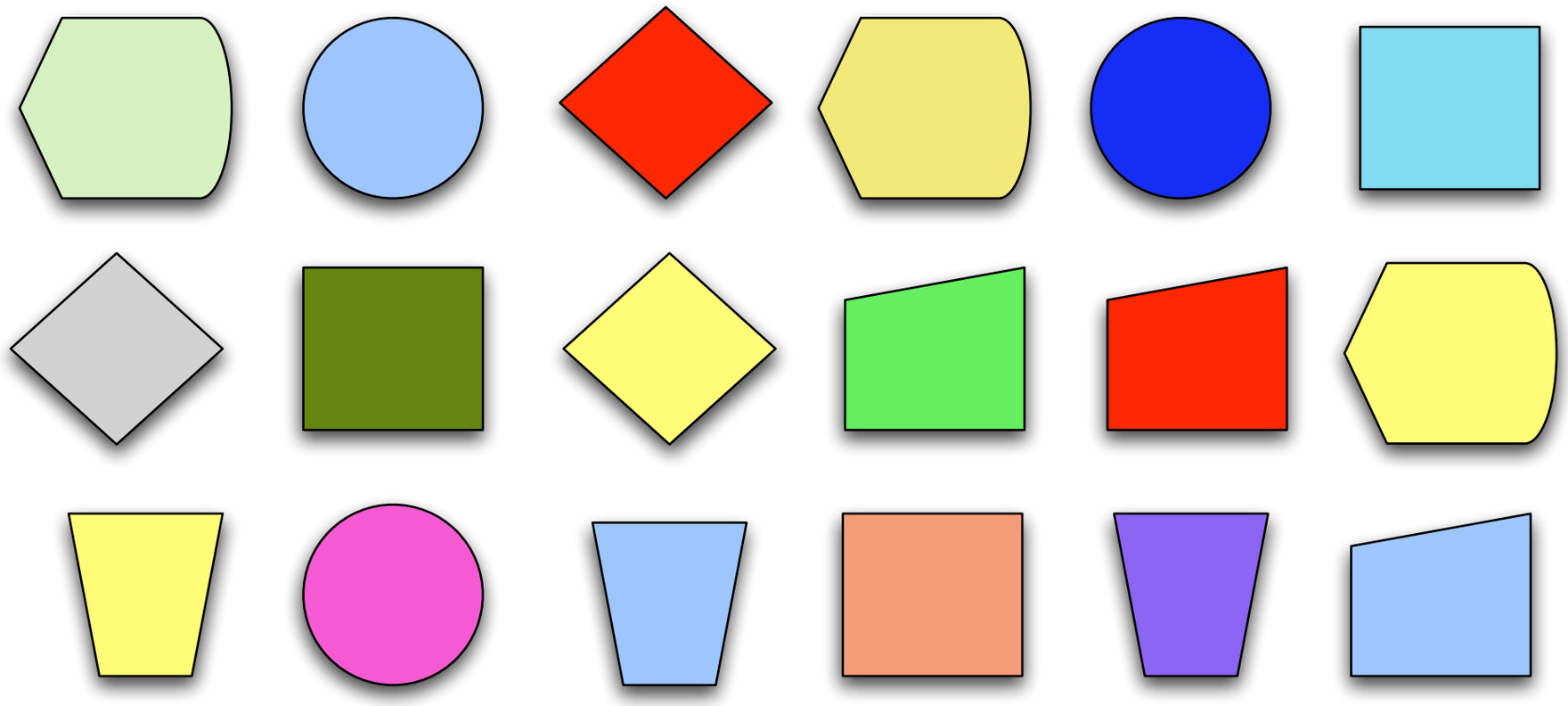
Claim:
Application architecture is irrelevant for your SOA

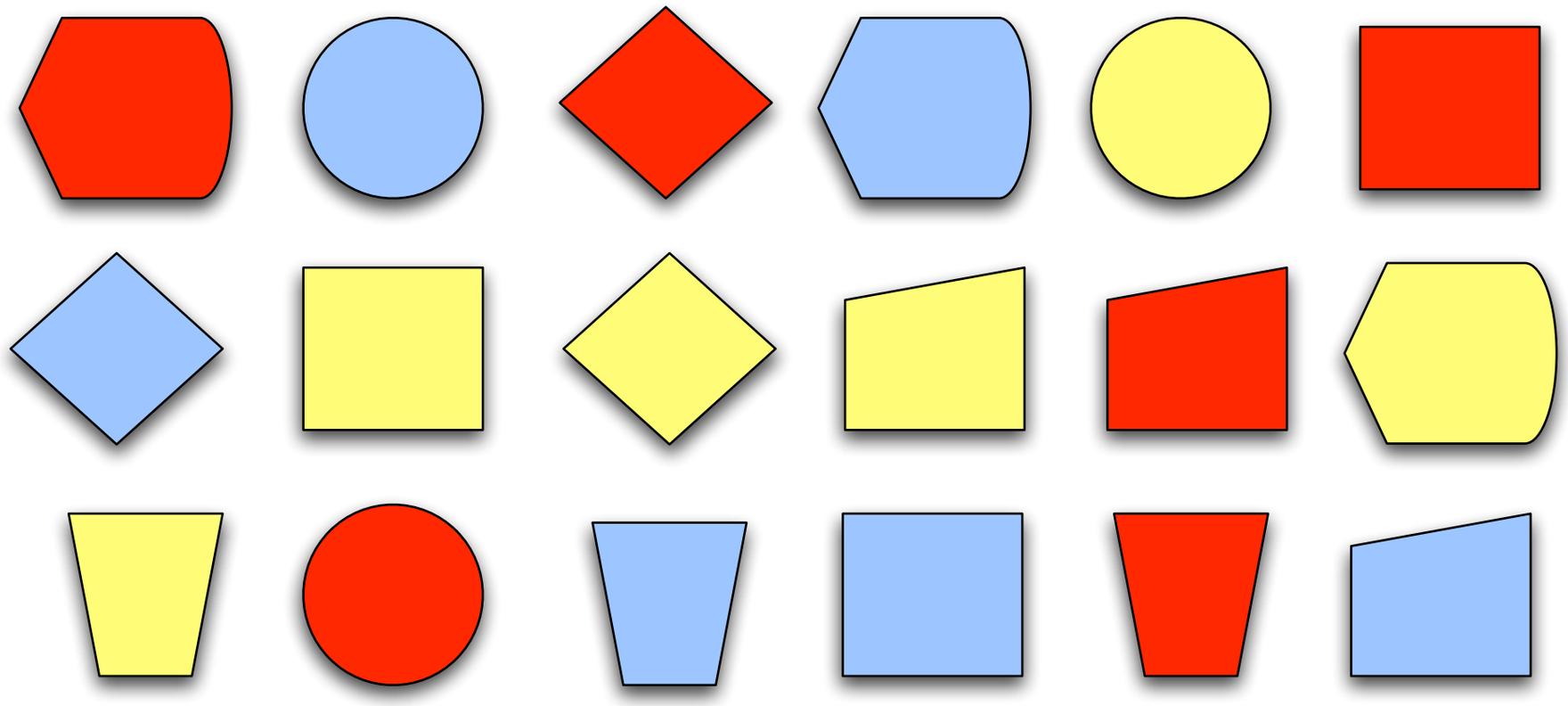
Application Architecture

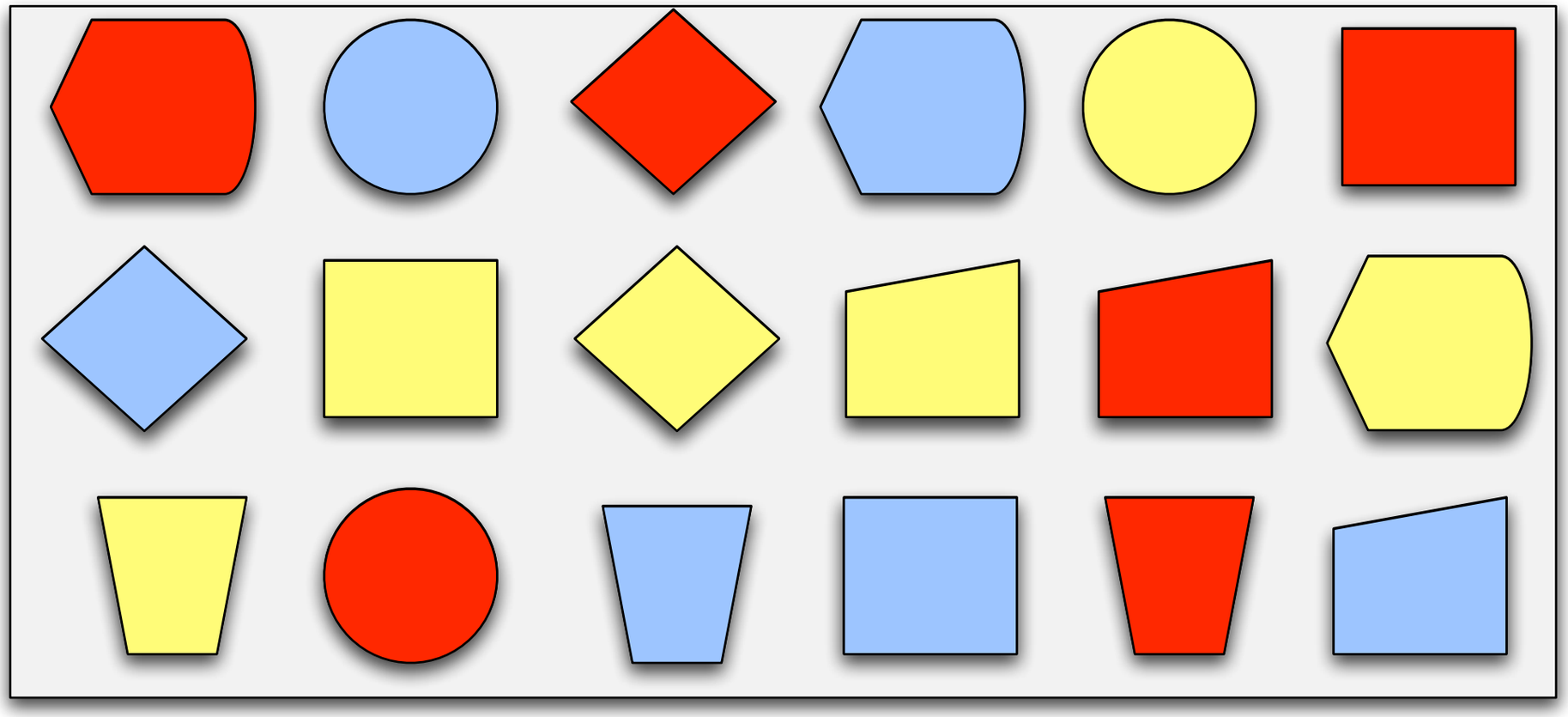
vs.

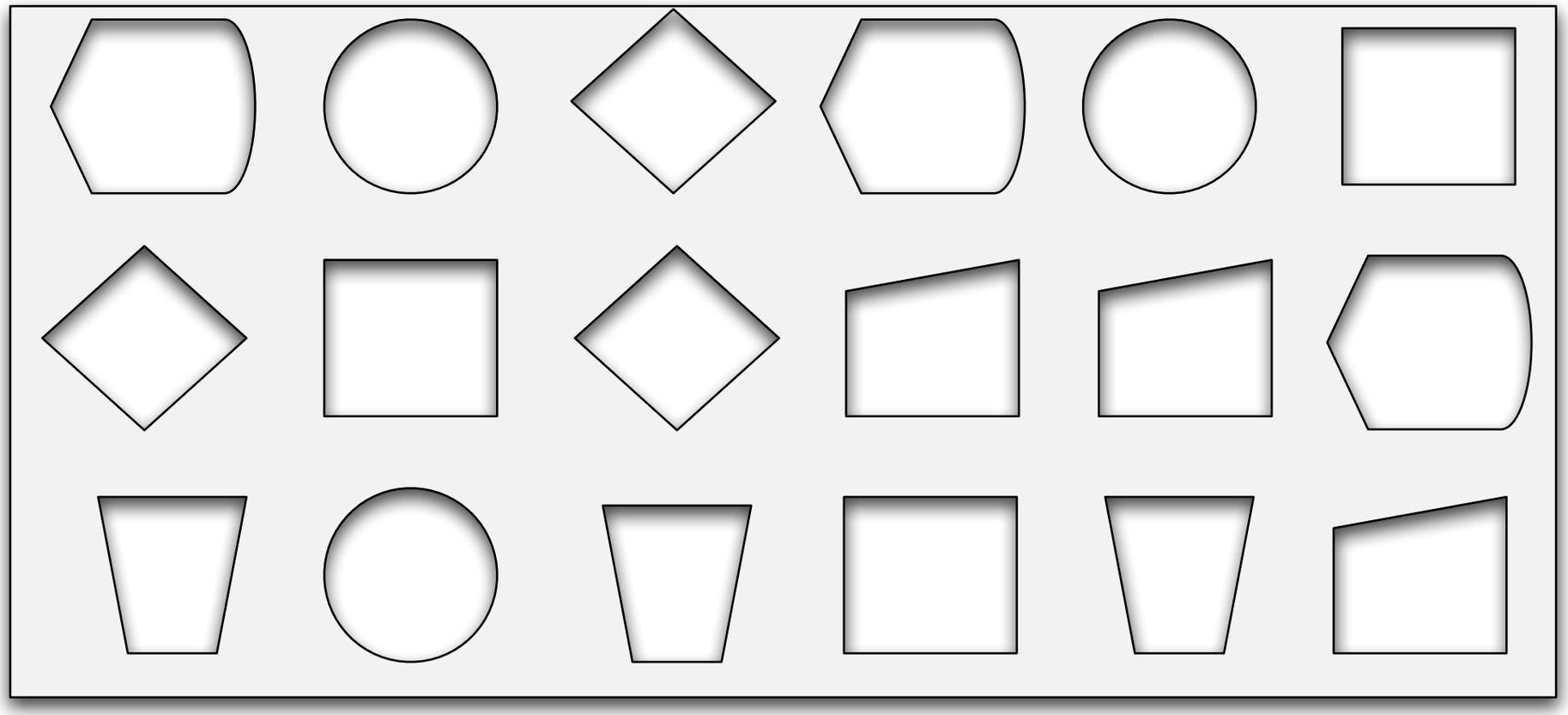
Integration Architecture

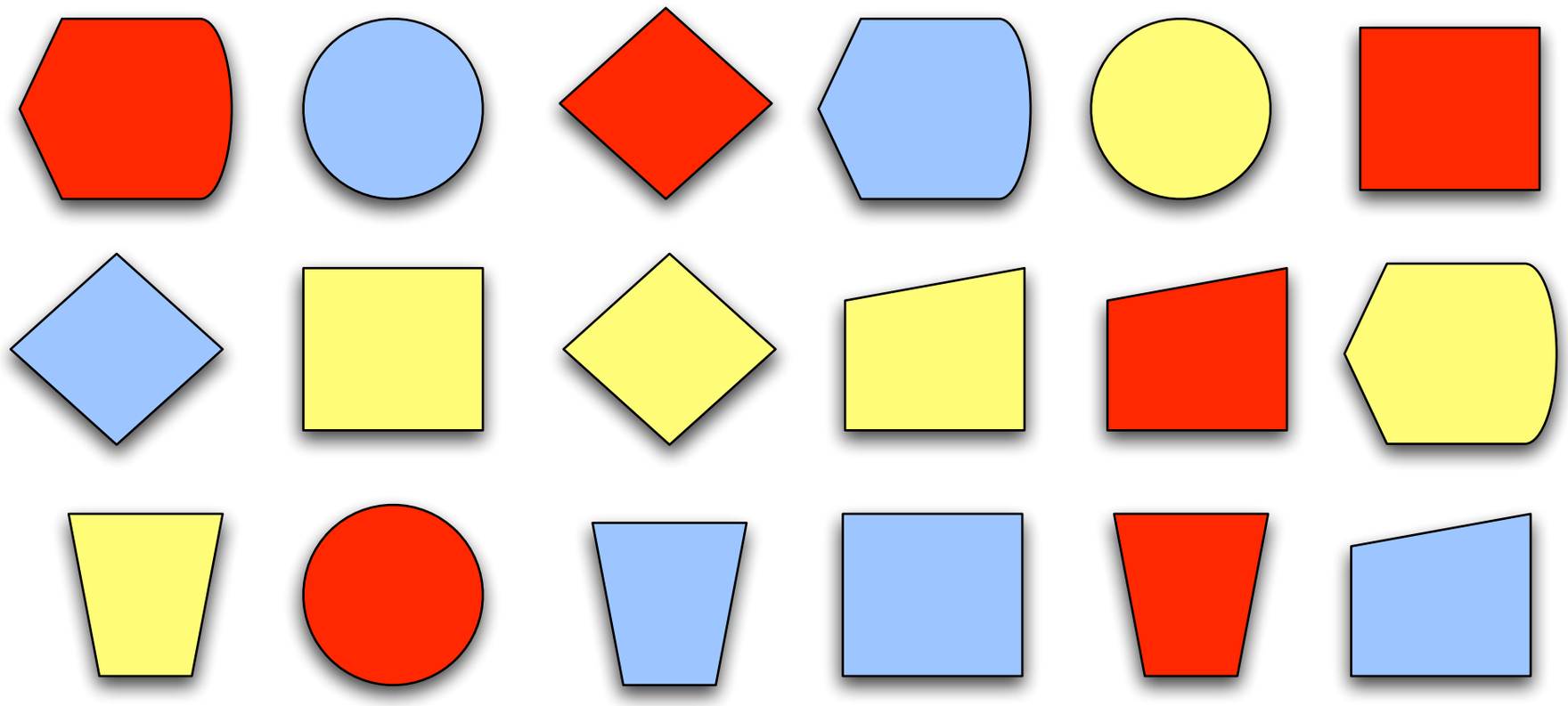


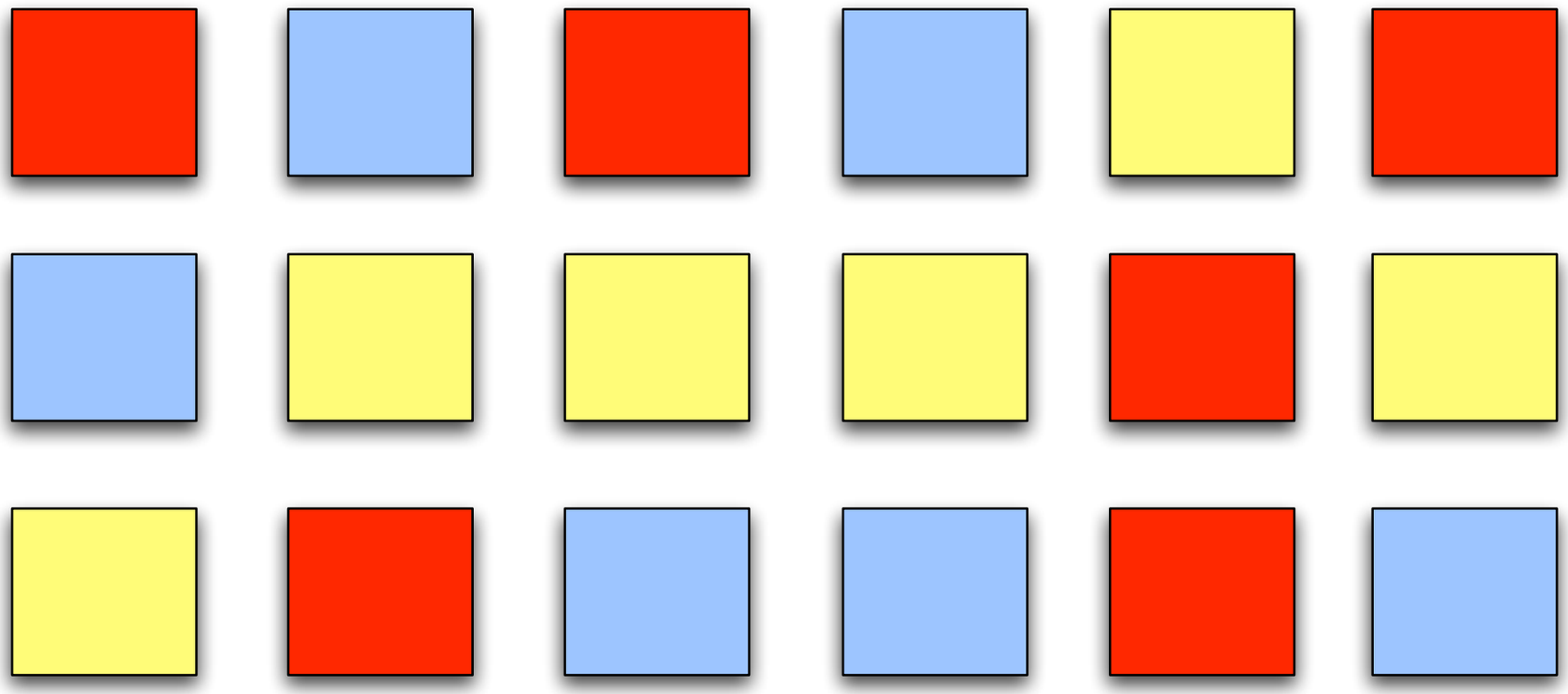


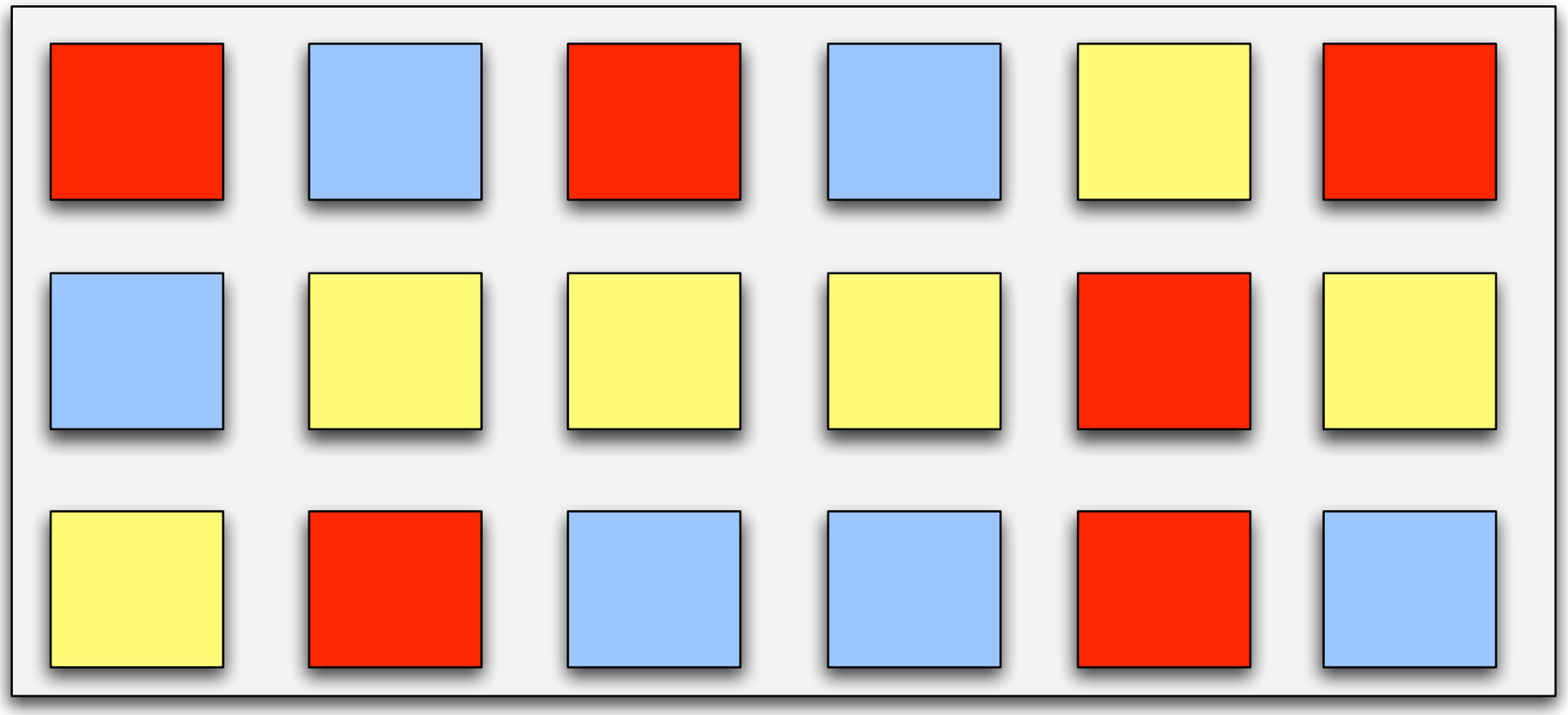


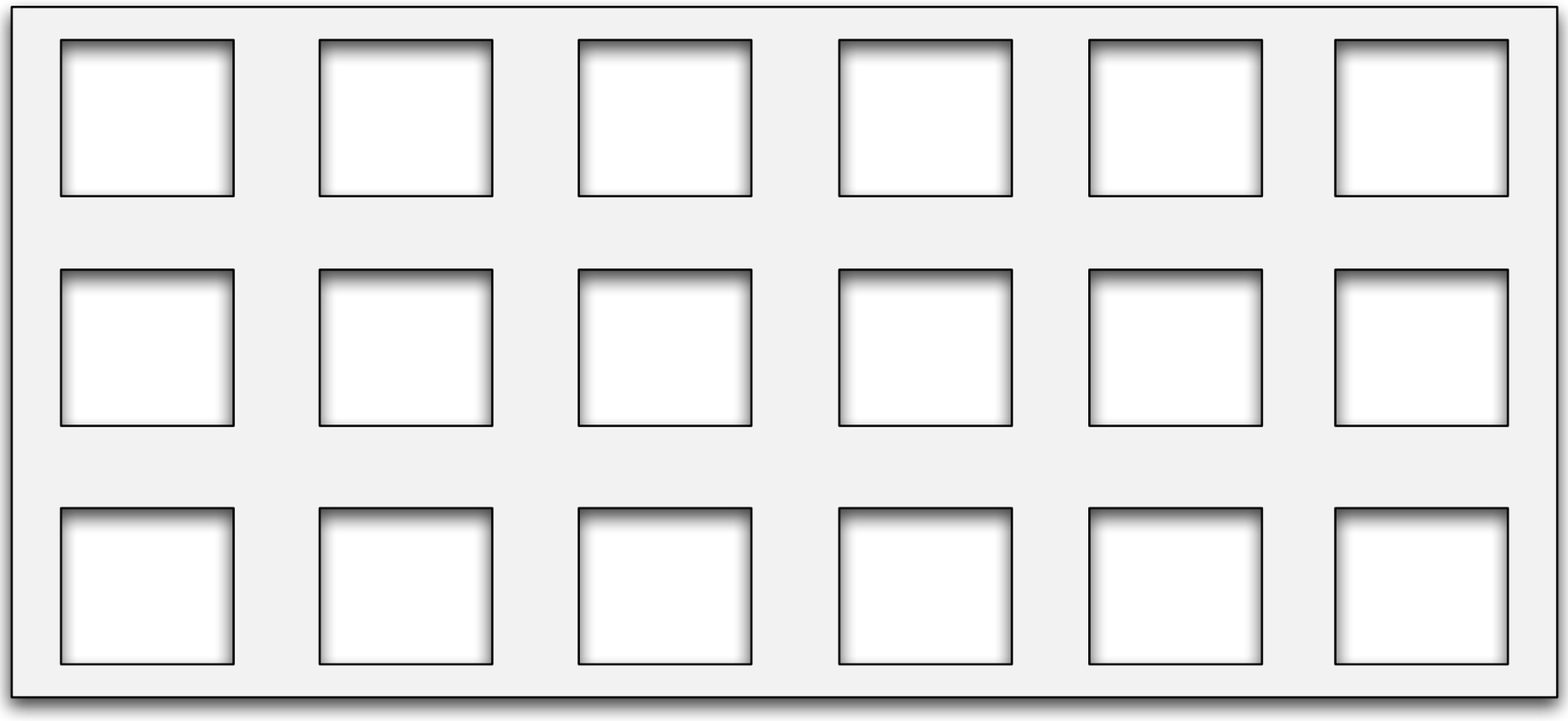




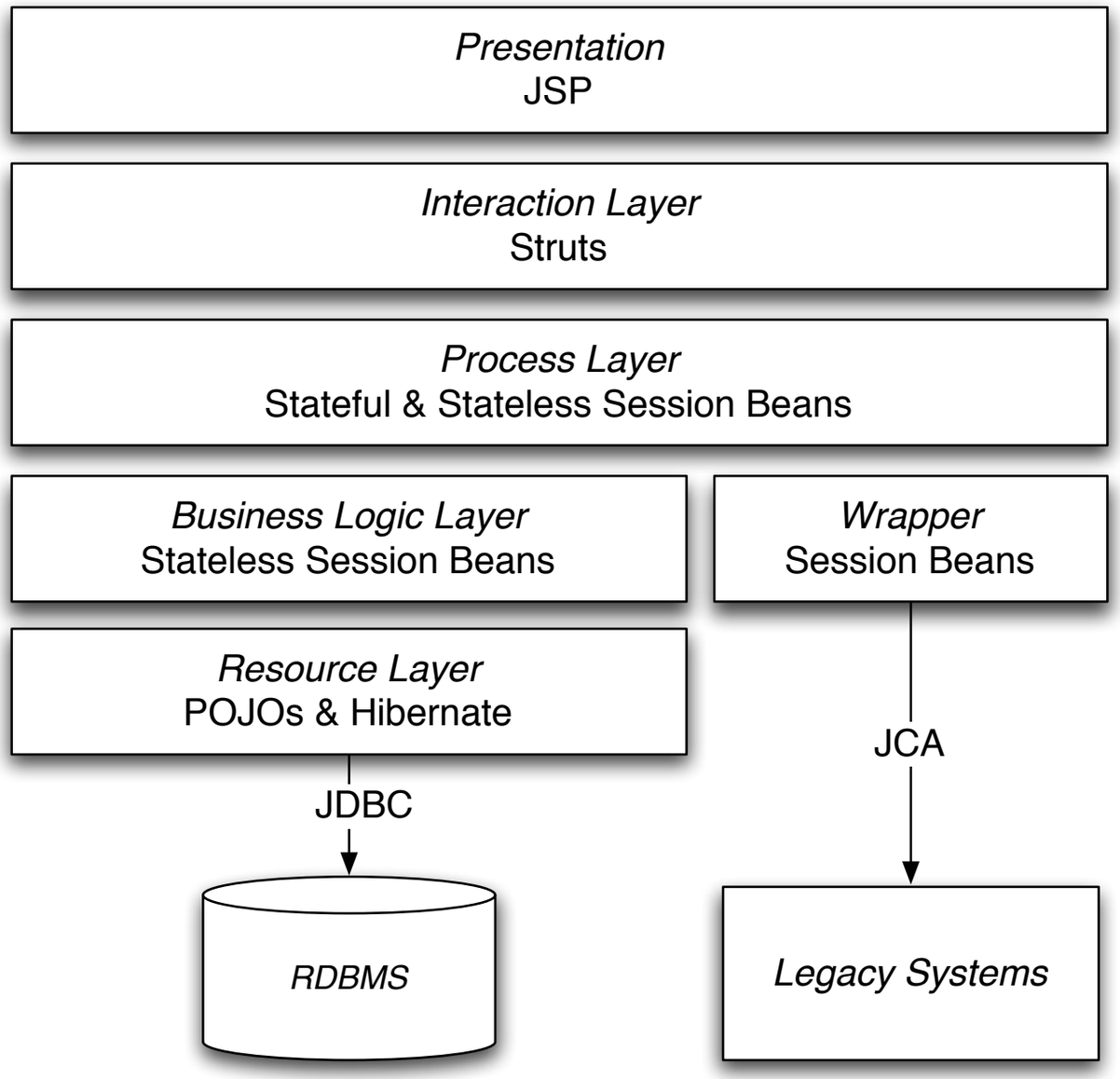






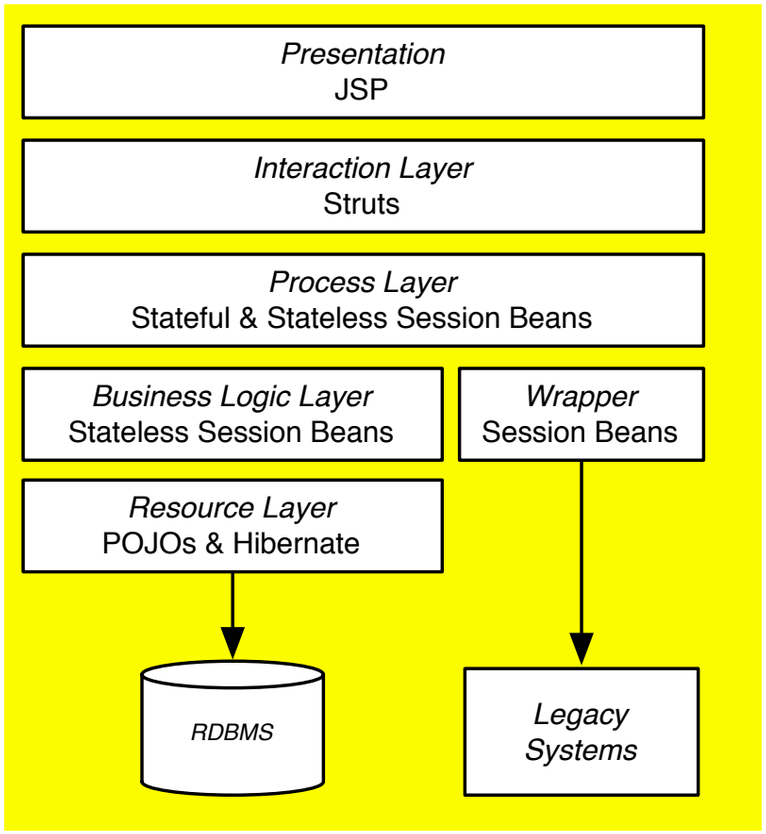


Application Architecture
(Example)

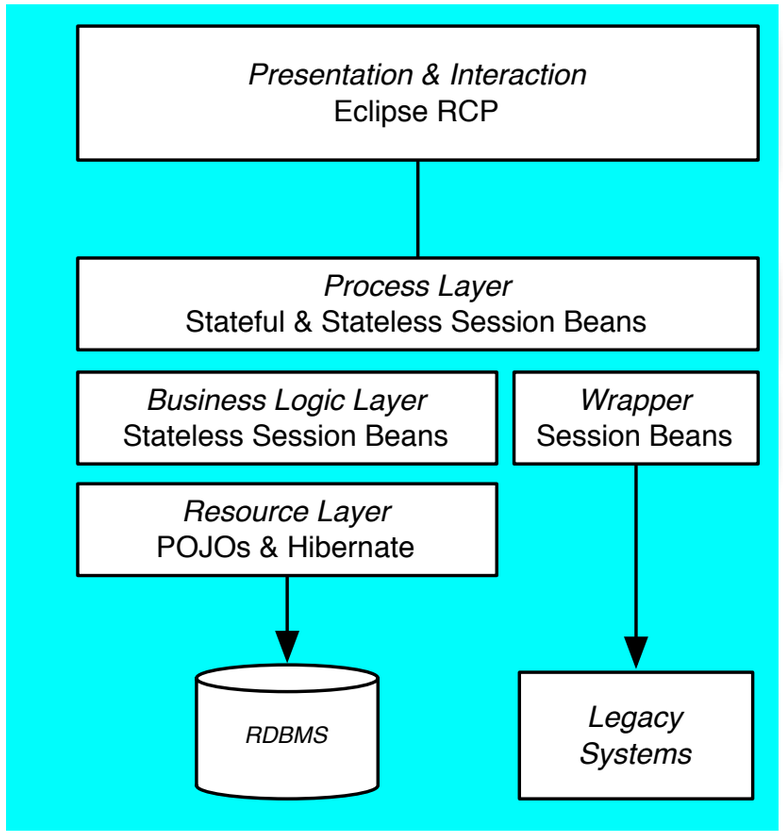


Application Classes

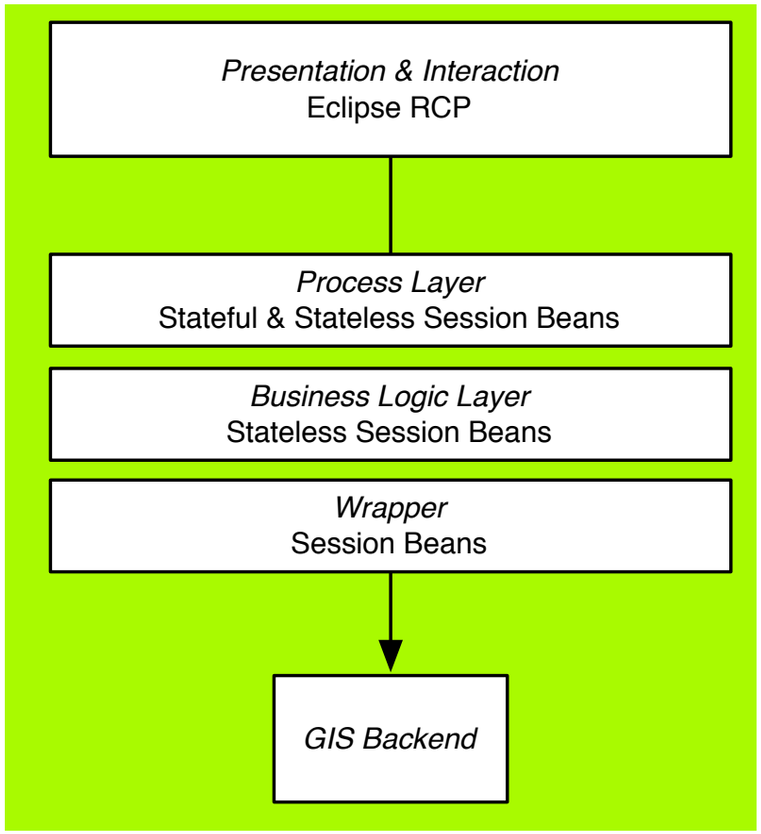
Class A



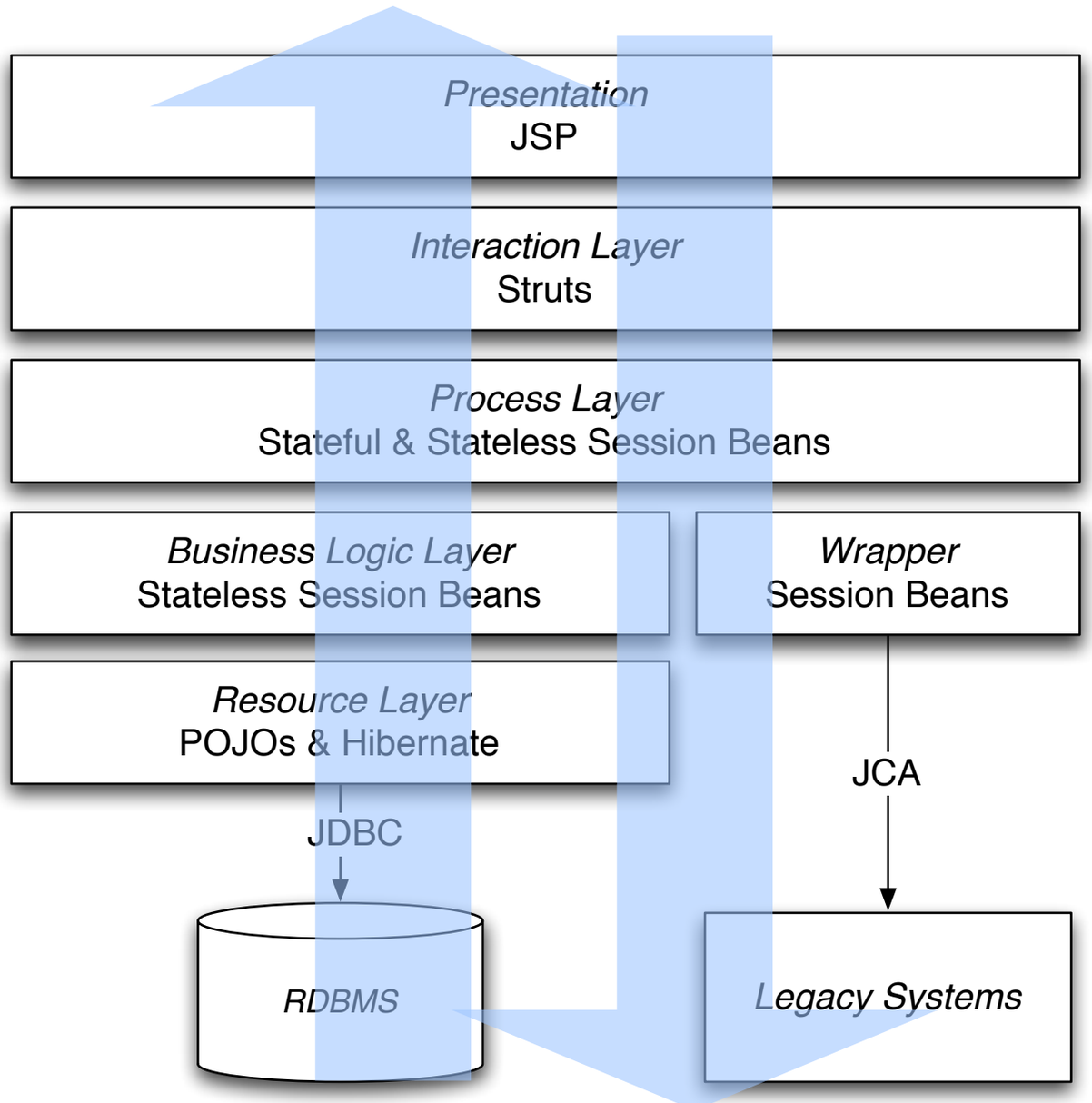
Class B



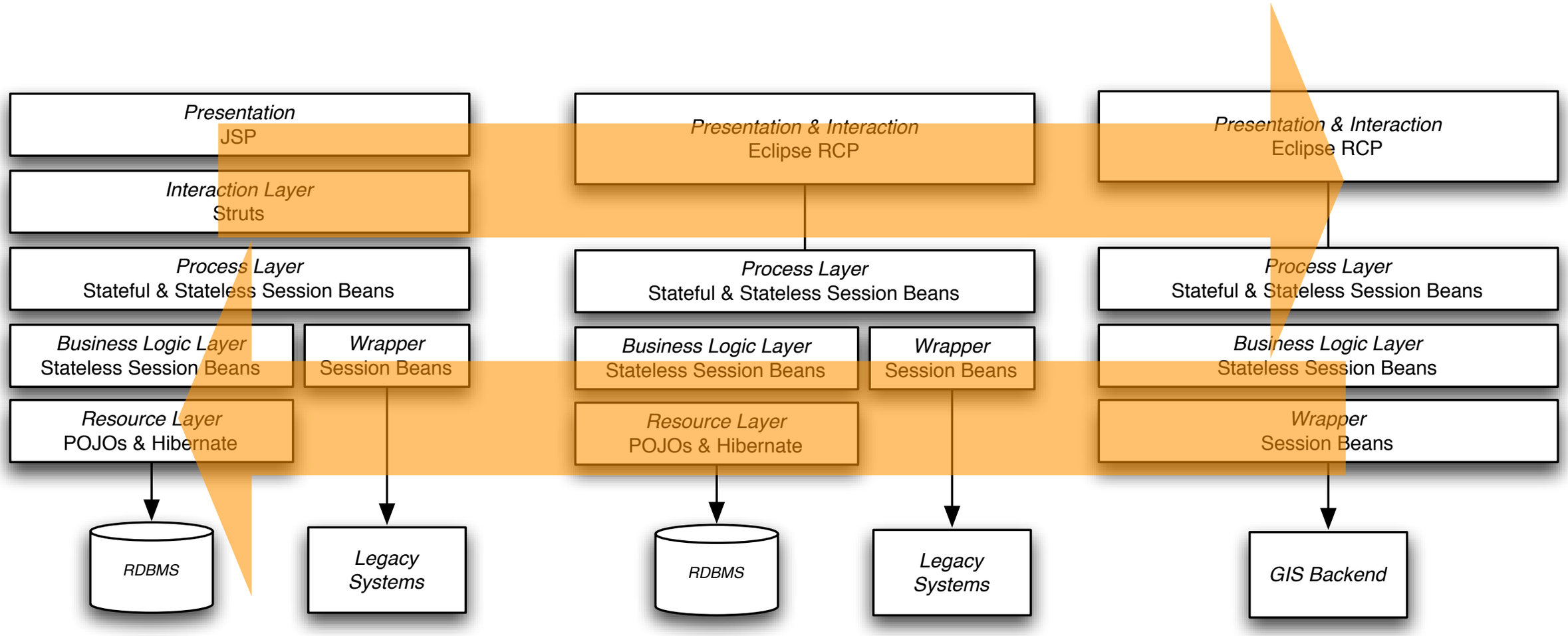
Class C



Focus: *Portability*



Focus: Interoperability



Recommendation: Don’t confuse integration architecture with application architecture – focus on one at a time

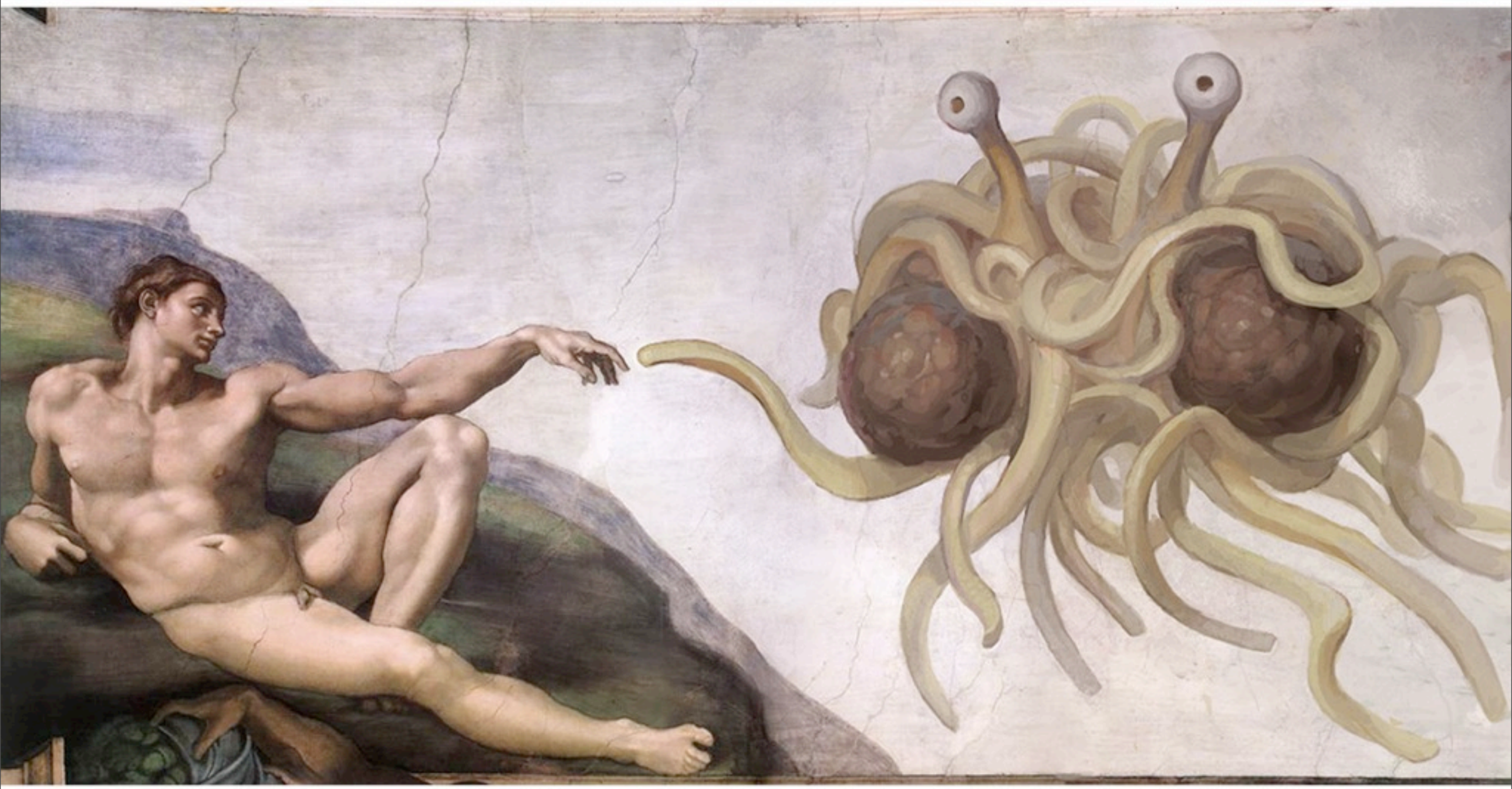
Recommendation: Don’t try to standardize everything at once

Claim:

An ESB should not be at the core of your SOA

ESB Spaghetti Monster

THE FLYING SPAGHETTI MONSTER

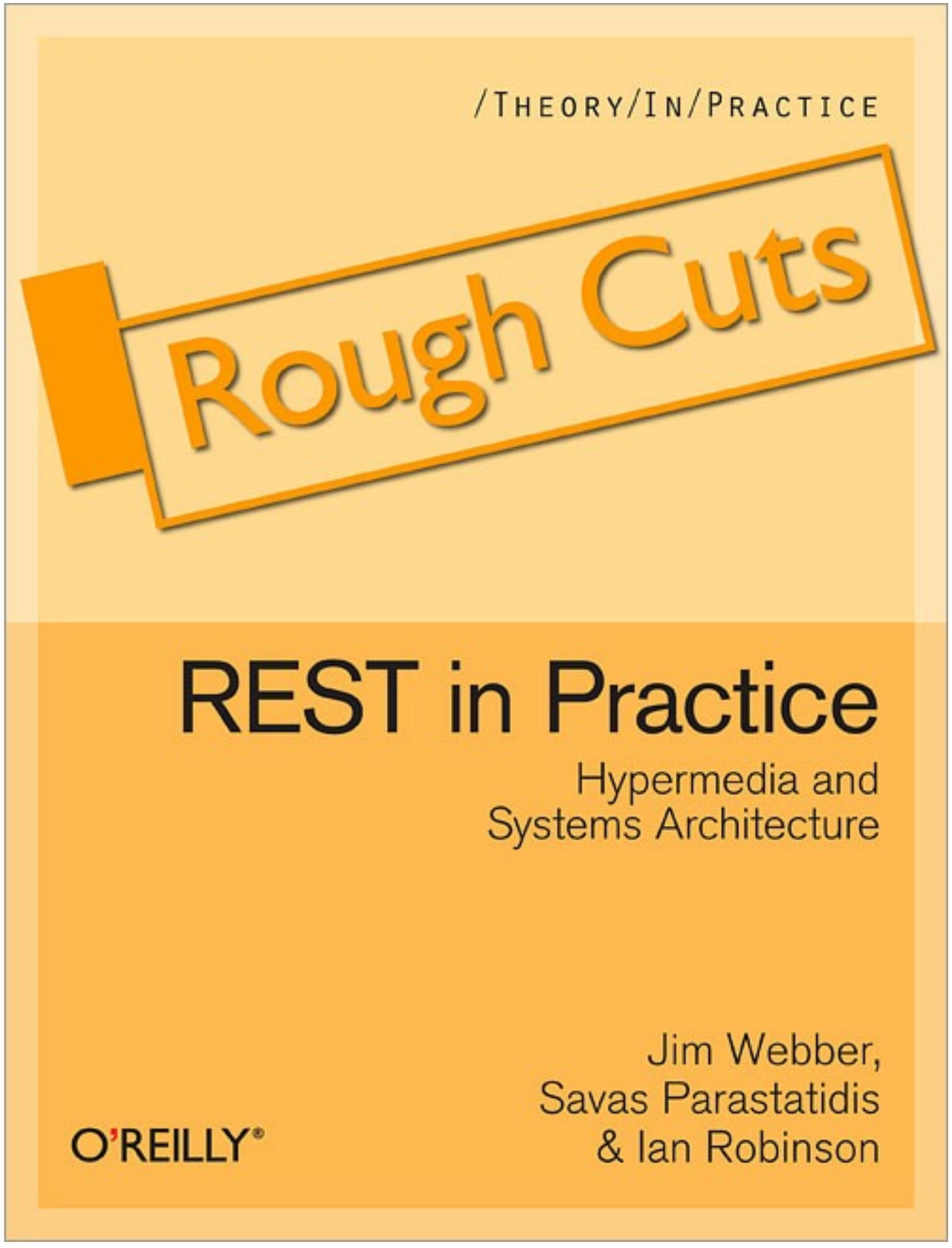


TOUCHED BY HIS NOODLY APPENDAGE

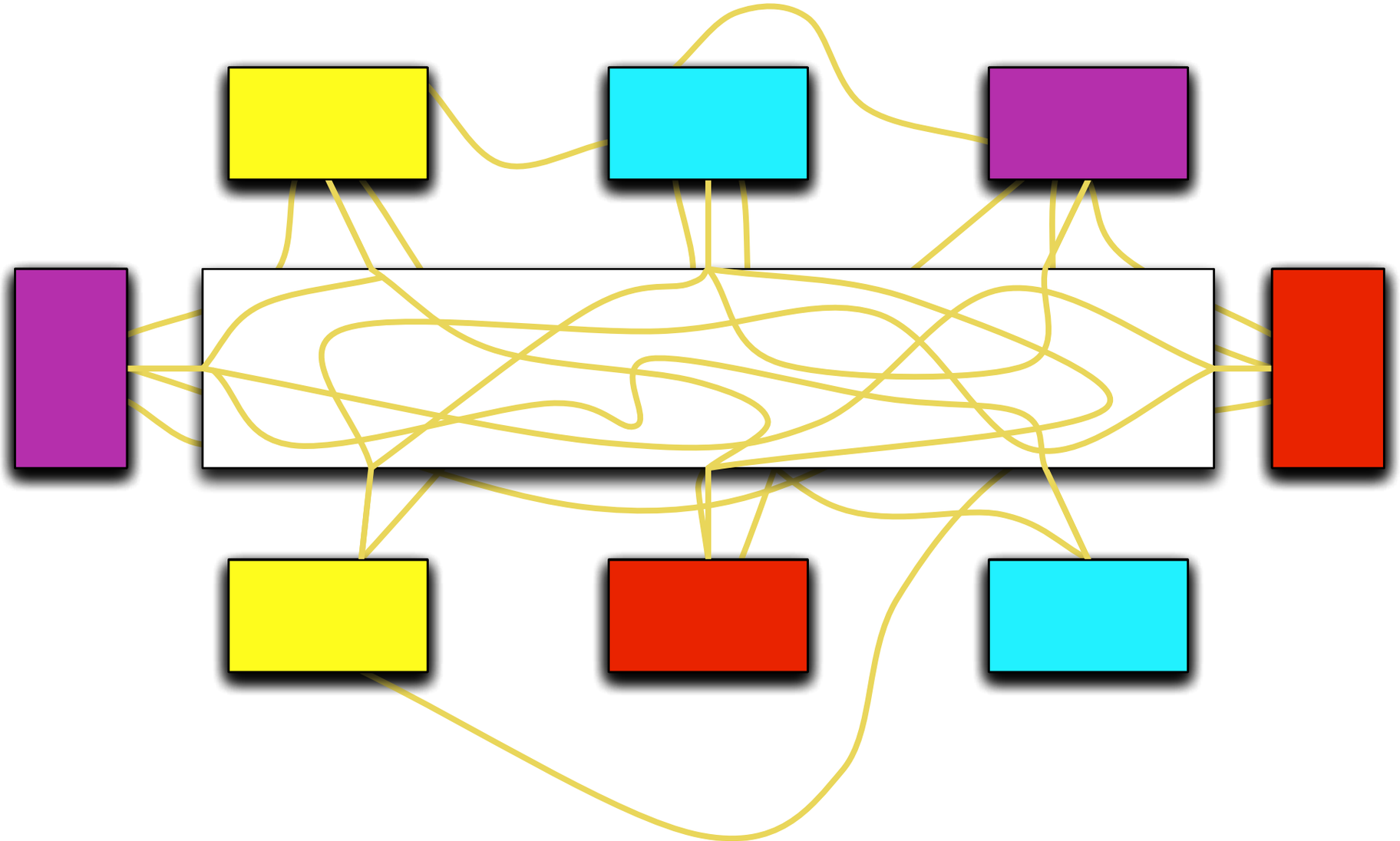
The next slide is
shamelessly stolen
from Jim Webber
(see http://tinyurl.com/ywm5sj)

So I'll make up for it:

Out real soon now



ESB Spaghetti



EOA

ESB-oriented Architecture

Proprietary tooling (“doodleware”)

No (or little) team development

No unit-testing, test-driven development

No (or little) modularization

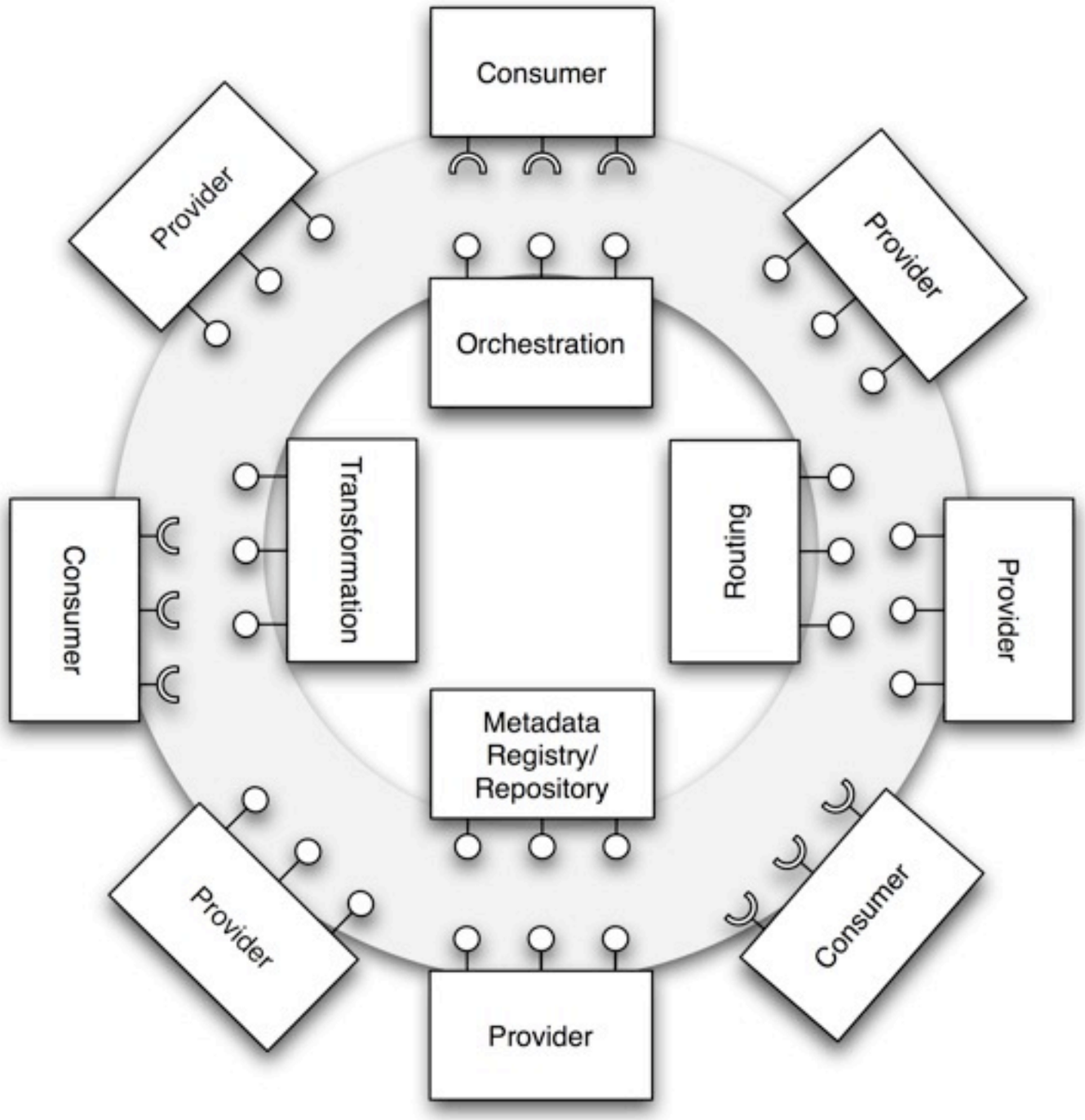
No (or little) versioning

Barrier to entry

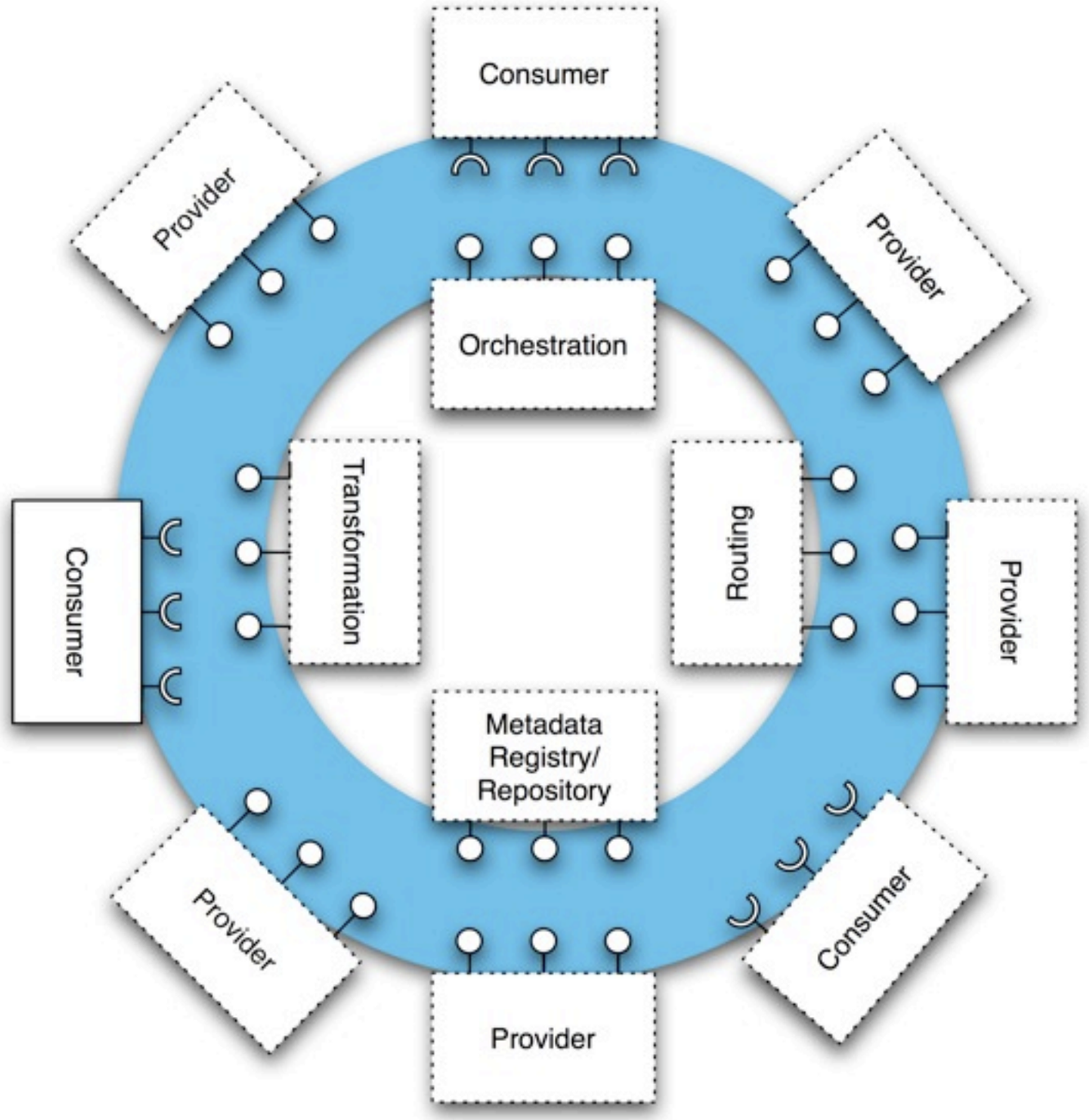
Prolonged existence of proprietary interfaces

Recommendation: Participants in a given SOA should depend on wire formats and protocols only

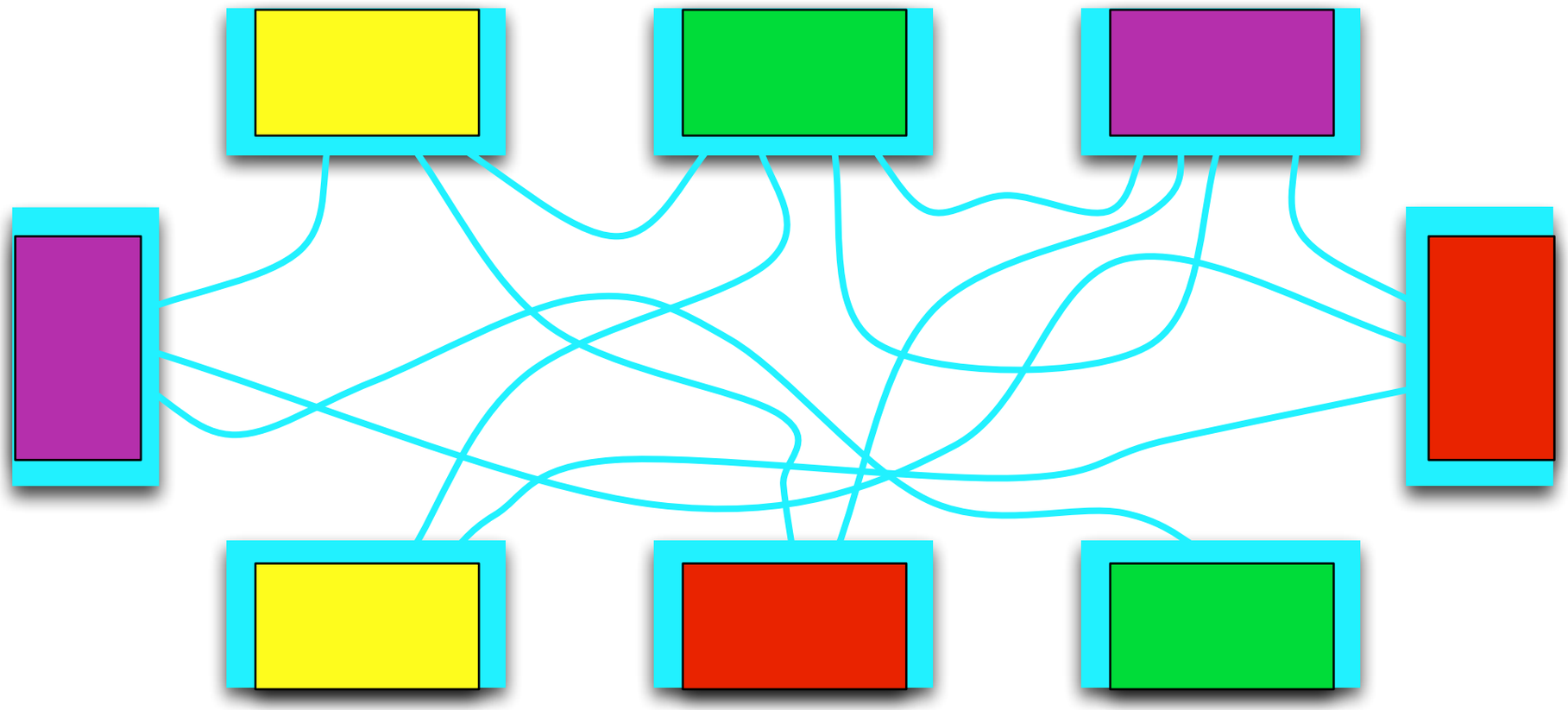
“Virtual” ESB


Consumer
Provider
Provider
Orchestration
Transformation
Routing
Consumer
Provider
Metadata
Registry/
Repository
Provider
Provider
Consumer

“Virtual” ESB


Consumer
Provider
Provider
Orchestration
Transformation
Routing
Consumer
Provider
Metadata
Registry/
Repository
Provider
Provider
Consumer

Virtual ESB



Recommendation:
If you choose SOAP, WSDL, WS-*, go with lightweight/OSS tools

Recommendation: Mainstream + Lightweight Platforms

Pick a decent platform or toolkit for SOAP/WS-* support

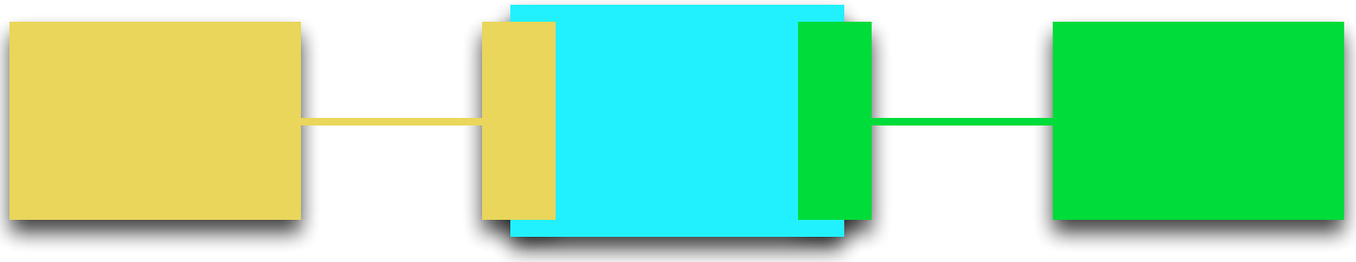
.NET WCF, Spring Web Services, Apache CXF, Apache Axis2, JBossWS, Sun Metro

Standardize on standards, not products, not single points of failure

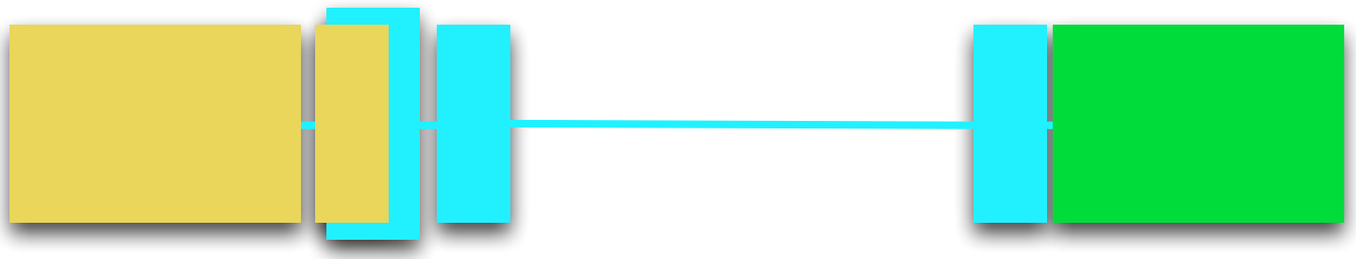
Claim:
Point-to-point communication is perfectly fine

Recommendation:
If you use an ESB …

Central Model



Enabling Model



… don’t put it in the center

Recommendation: Consider Open Source ESBs

Apache Synapse/WSO2 ESB

Mule ESB

JBoss ESB

ServiceMix

Sopera

...

Claim:
Hiding XML is a Bug, not a feature

XML Ecosystem

10 Things to do with XML

View it in tree rendering

Check for wellformedness

Transform it using XSLT

Query with XPath

Process with XQuery

Validate against schema

Encrypt/Decrypt parts

Sign and verify signature

Archive it

Process w/ SAX/DOM

XML & Objects

XML, XML Schema and objects don't match

XML is not an implementation detail

Generating code from XML Schema:

Reap all the disadvantages of XML while ignoring its benefits.

Claim:
Web services (SOAP/WSDL) are not the only way

Claim:
RESTful HTTP moves you closer to SOA goals than WS-*

SOA and RESTful HTTP

Loose Coupling
Low Resistance to Change
Unexpected Re-use
Interoperability
Vendor Independence
Available Skills
Tooling for any Language & Platform

... all served better by RESTful HTTP than WS-*

The REST Uniform Interface

identification of resources

resource manipulation through representations

hypermedia as the engine of application state

self-descriptive messages

The REST Uniform Interface

identification of resources

resource manipulation through representations

hypermedia as the engine of application state

self-descriptive messages

http://example.com/orders?year=2008

http://example.com/customers/1234

http://example.com/orders/2007/10/776654

http://example.com/products/4554

http://example.com/processes/sal-increase-234

The REST Uniform Interface

identification of resources

resource manipulation through representations

hypermedia as the engine of application state

self-descriptive messages

```
GET /customers/1234
Host: example.com
Accept: application/vnd.mycompany.customer+xml
```

```
<customer>...</customer>
```

```
GET /customers/1234
Host: example.com
Accept: text/x-vcard
```

```
begin:vcard
...
end:vcard
```

The REST Uniform Interface

identification of resources

resource manipulation through representations

hypermedia as the engine of application state

self-descriptive messages

```
<order self='http://example.com/orders/3321'>
  <amount>23</amount>
  <product ref='http://example.com/products/4554' />
  <customer ref='http://example.com/customers/1234' />
  <link rel='edit’
        ref='http://example.com/order-edit/ACDB' />
</order>
```

The REST Uniform Interface

identification of resources

resource manipulation through representations

hypermedia as the engine of application state

self-descriptive messages

Standard Method

```
GET /service/customers/1234 HTTP 1.1
Host: www.example.com
User-Agent:     XYZ 1.1
Accept:         text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Keep-Alive:     300
Connection:     keep-alive
If-Modified-Since:      Fri, 02 Oct 2009 16:47:31 GMT
If-None-Match: "600028c-59fb-474f6852c9dab"
Cache-Control: max-age=60

HTTP/1.1 200 OK
Date: Sun, 04 Oct 2009 19:16:23 GMT
Server: Apache/2.1 (Debian)
Last-Modified: Fri, 02 Oct 2009 16:49:31 GMT
Etag:  "600028c-59fb-474f6852c9dce"
Cache-Control:  max-age=300
Accept-Ranges:          bytes
Vary:  Accept-Encoding
Content-Encoding:  gzip
Content-Length:         7160
Keep-Alive:     timeout=15, max=91
Connection:  Keep-Alive
Content-Type:           application/xml

<?xml version='1.0' encoding='utf-8' ?>
...
```

Media Type

Visibility

Control Data

Data

getOrderDetails()

submitApplicationData()

updateQuote()

findMatchingBid()

initiateProcess()

cancelSubscription()

listAuctions()

getUsers()

getOrderDetails()

findMatchingBid()

listAuctions()

getUsers()

GET

initiateProcess()

submitApplicationData()

POST

PUT

updateQuote()

DELETE

cancelSubscription()

```
interface Resource {
      Resource(URI u)
      Response get()
      Response post(Request r)
      Response put(Request r)
      Response delete()
}
```

generic

Any HTTP client
(Firefox, IE, curl, wget)

Any HTTP server

Caches

Proxies

Google, Yahoo!, MSN

```
class CustomerCollection : Resource {
      ...
      Response post(Request r) {
            id = createCustomer(r)
            return new Response(201, r)
      }
      ...
}
```

Anything that knows
your app

specific

Claim:
Advanced WS-* standards are overrated

WS-* Theory & Practice

WS-Addressing, WS-ReliableMessaging are not yet widely interoperable

Nobody uses WS-Coordination, WS-Atomic Transactions, WS-Business Activity

Message-based security is way too expensive

UDDI is a solution looking for a problem

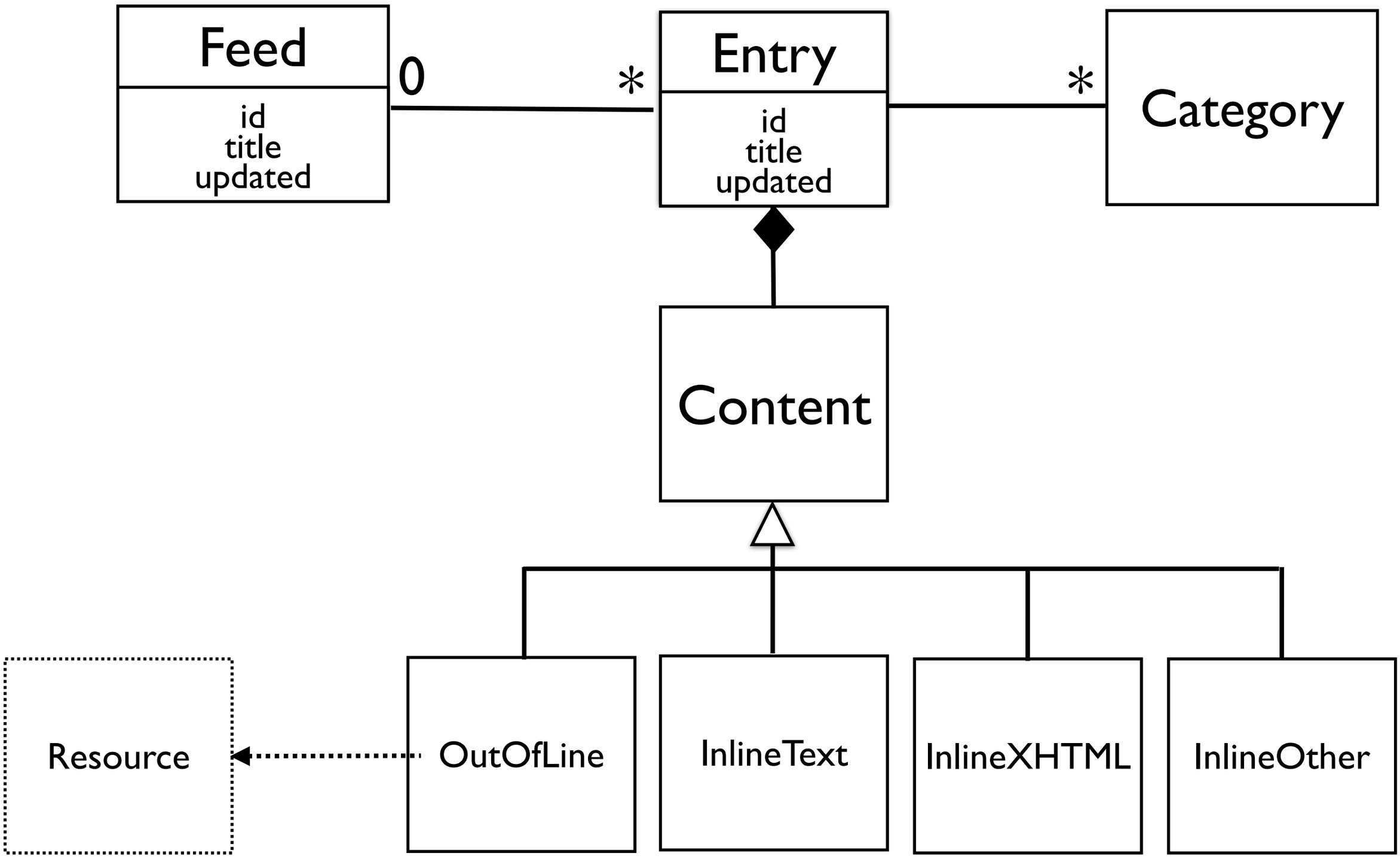
Recommendation: Consider adopting RESTful HTTP instead of WS-*

(Interlude: How to secretly sneak RESTful HTTP into your organization)

Consider WS-* + HTTP GET as the gateway drug

Use feeds to for even notification

Atom Model


Feed
id
title
updated
0
*
Entry
id
title
updated
*
Category
Content
Resource
OutOfLine
InlineText
InlineXHTML
InlineOther

Adopt REST for SOA Governance

UDDI

Inquiry

- find_binding
- find_business
- find_relatedBusinesses
- find_service
- find_tModel
- get_bindingDetail
- get_businessDetail
- get_operationalInfo
- get_serviceDetail
- get_tModelDetail

Publication

- save_binding
- save_business
- save_service
- save_tModel
- delete_binding
- delete_business
- delete_publisherAssertions
- delete_service
- delete_tModel
- add_publisherAssertions
- set_publisherAssertions
- get_assertionStatusReport
- get_publisherAssertions
- get_registeredInfo

420-page specification

Finding and maintaining (meta-)model objects

UDDI *(contd.)*

UDDI could be greatly simplified by using plain HTTP

It would no longer be protocol-independent - but who cares?

Atom (Syndication Format & Protocol) are a great match

See: http://www.xml.com/pub/a/ws/2002/02/06/rest.html?page=2

Lightweight Governance

Simple solution (Webserver, WebDAV, AtomStore) for Document Storage

Atom feeds for update notifications

HTTP GET for lookups

RESTful SOA Governance

WSO2 Registry

Mule Galaxy

HP Systinet

(Final) Claim:
The most important architectural guide is your intelligence

homogeneity
vs.
right tool for the job

integrated solutions
vs.
no dependency on single vendors

central control
vs.
no single point of failure

mainstream
vs.
technical optimum

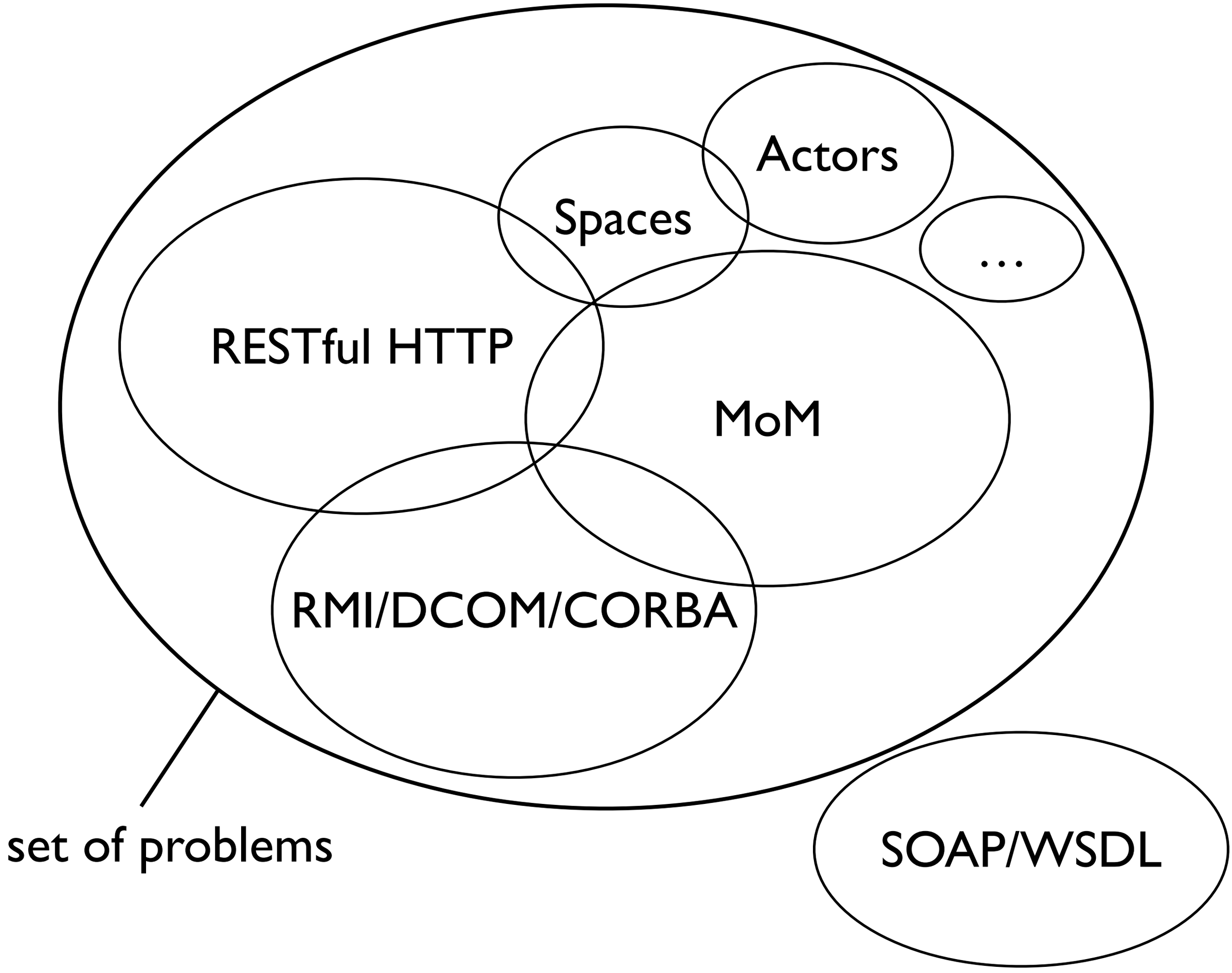
CSOA

Common Sense Oriented Architecture

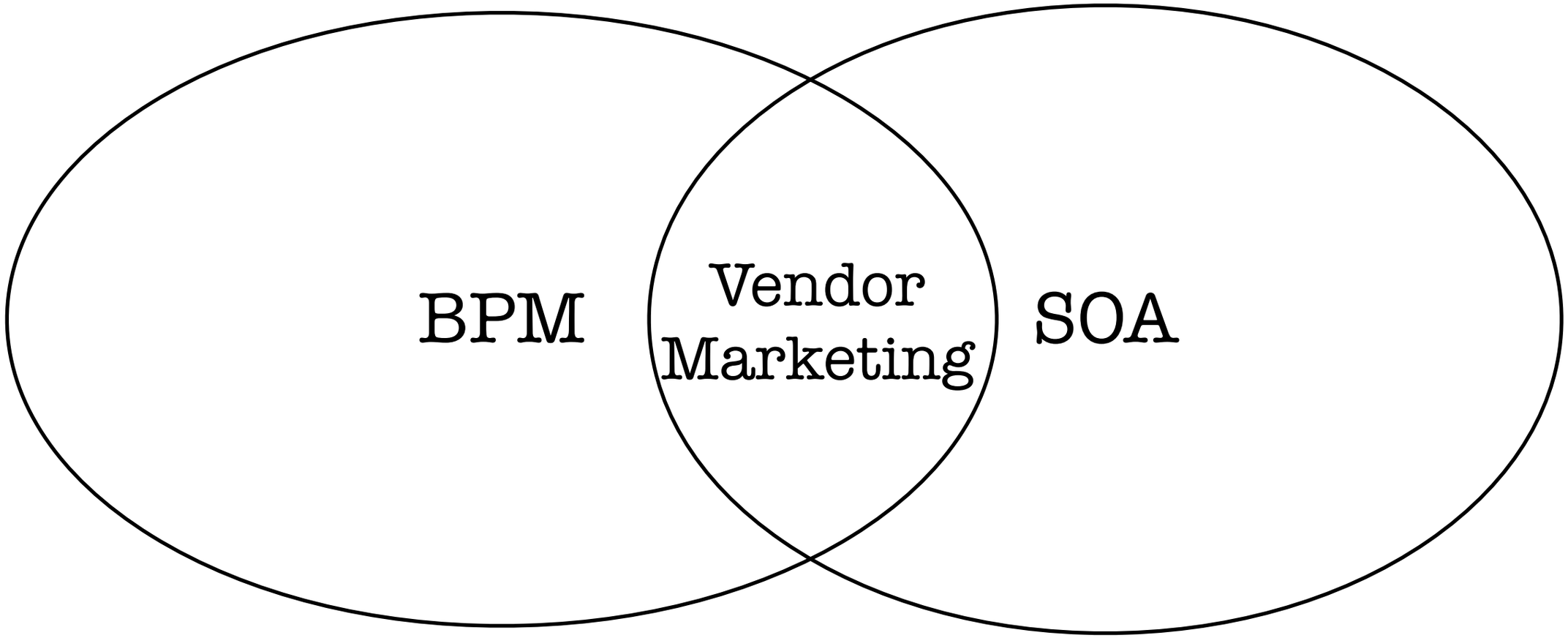
Q&A

stefan.tilkov@innoq.com
http://www.innoq.com/blog/st/
@stilkov

When to use WS-*?


Actors
Spaces
...
RESTful HTTP
MoM
RMI/DCOM/CORBA
set of problems
SOAP/WSDL

What about BPM?

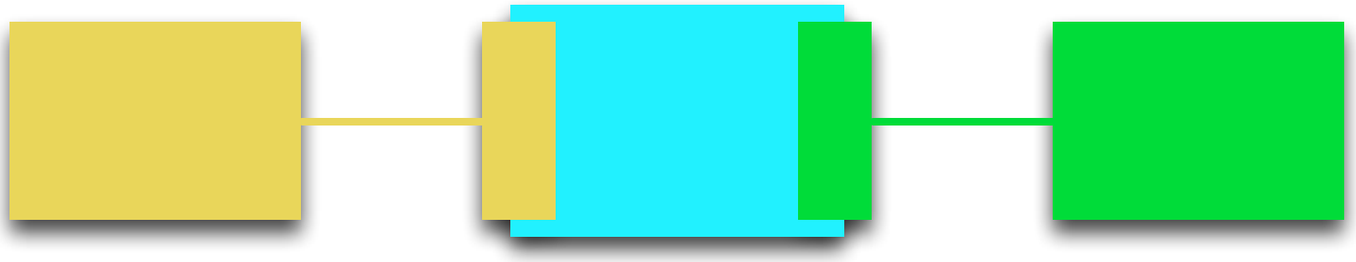


Valid Options:

1. SOA
2. BPM
3. SOA + BPM

Orchestration Options

Central Model



BPM as Impl Model