SOA, REST & JSON

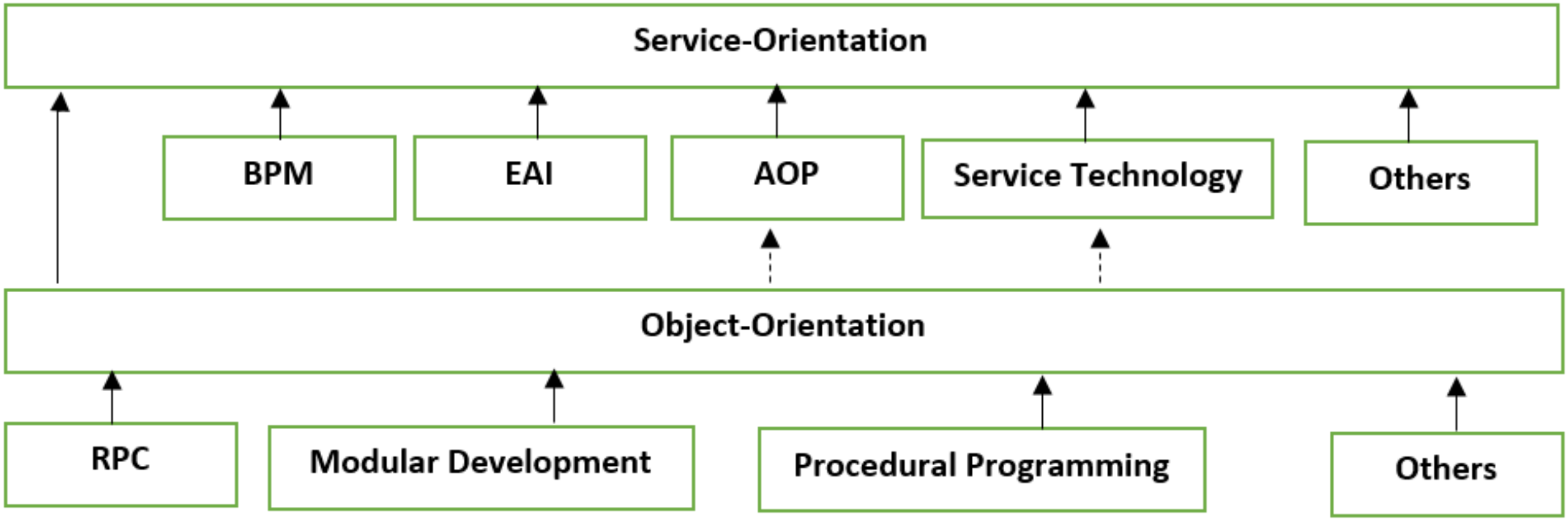
Hannah Siegel & Andreas Vogt

2015-03-20

Service Oriented Architecture

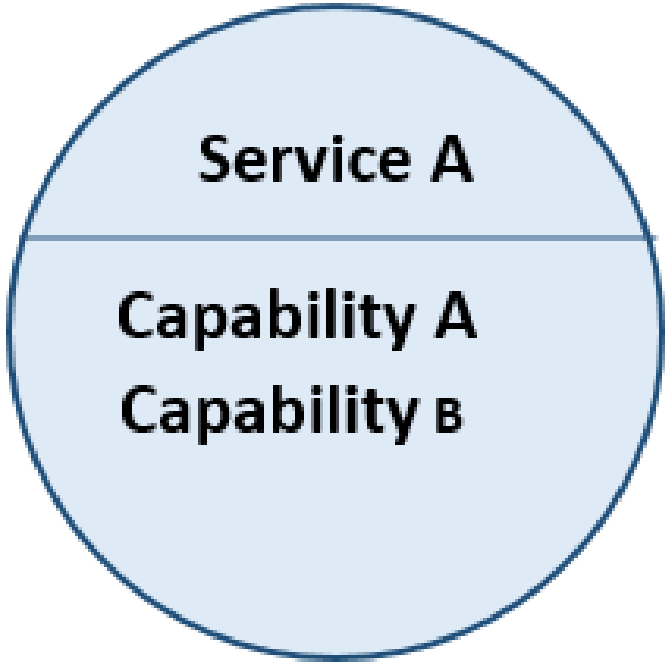
- Service – Orientation
- Service Oriented Computing
- Actors
- Platforms
- Success Formula
- Contributors

Service Oriented Architecture



Service

- Unit of logic
- SOA Design Principles Applied
- Clearly defined function
- Description of the functionality



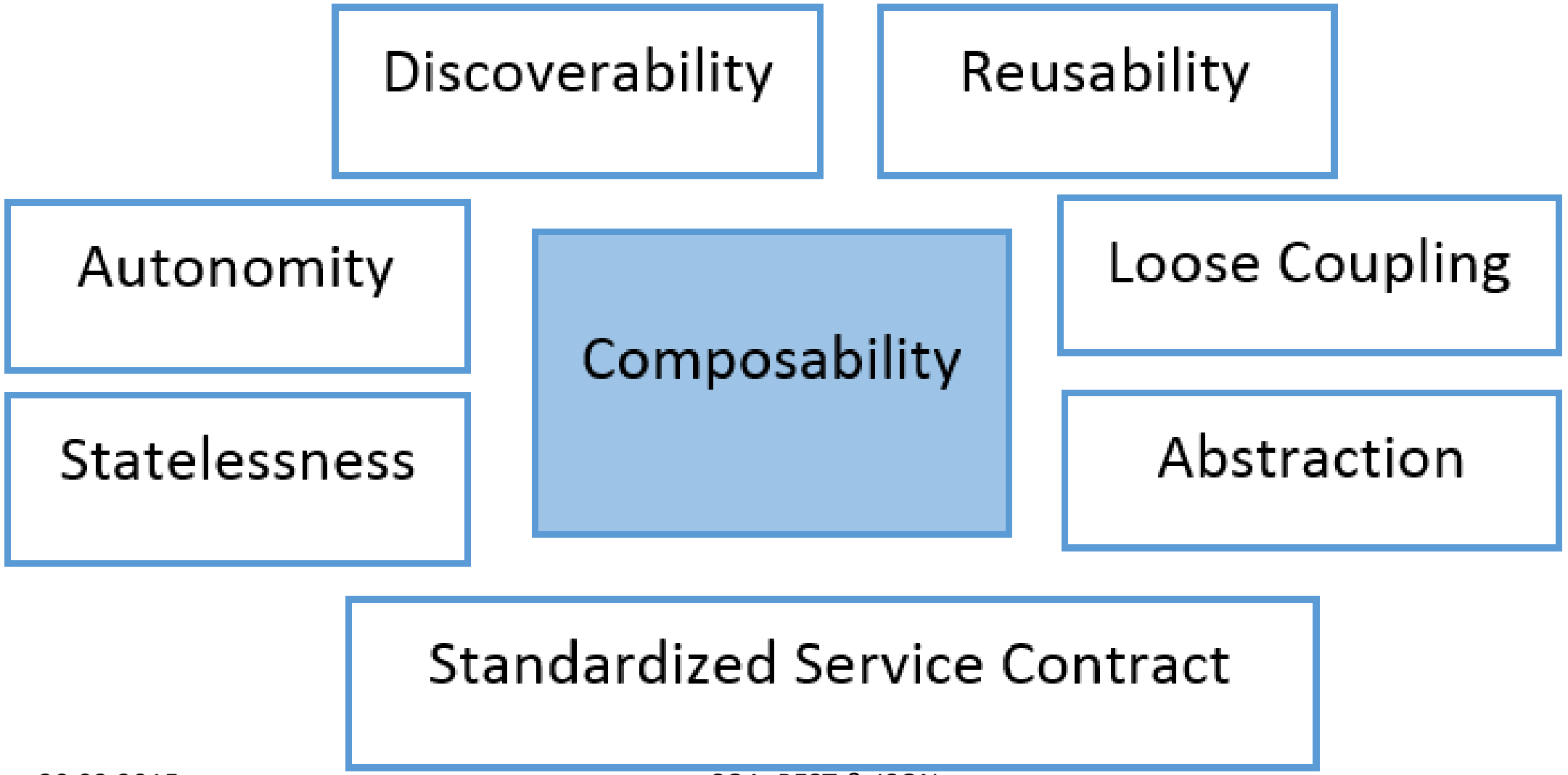
Problems?

- Resistant to change
- Communication and data transmission
- Vendor dependency
- Not enough support to BPM

SOA Manifesto

- Business value over technical strategy
- Strategic goals over project-specific benefits
- Intrinsic interoperability over custom integration
- Shared services over specific-purpose implementations
- Flexibility over optimization
- Evolutionary refinement over pursuit of initial perfection

Design Principles

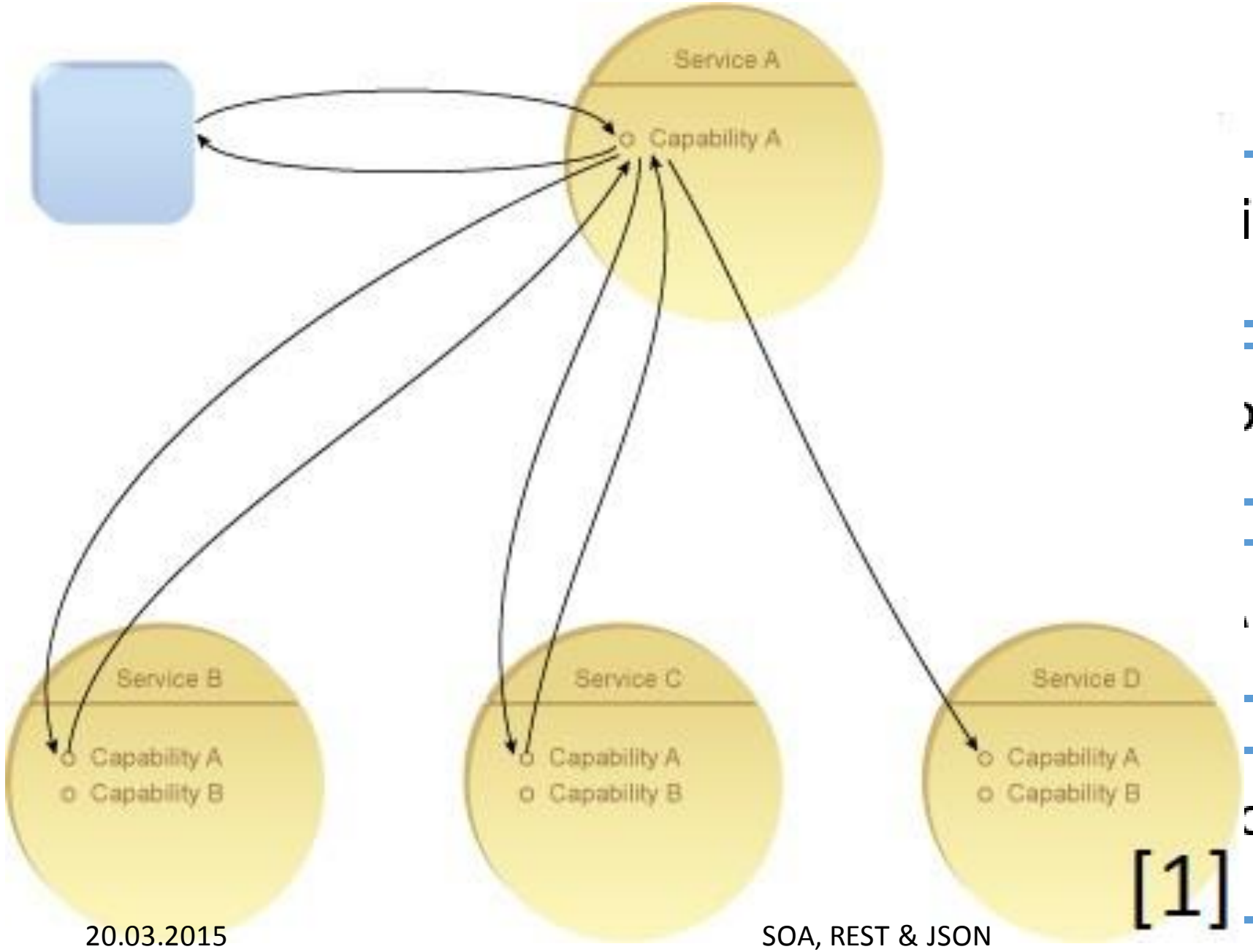


Design Principles

ling

on

[1]



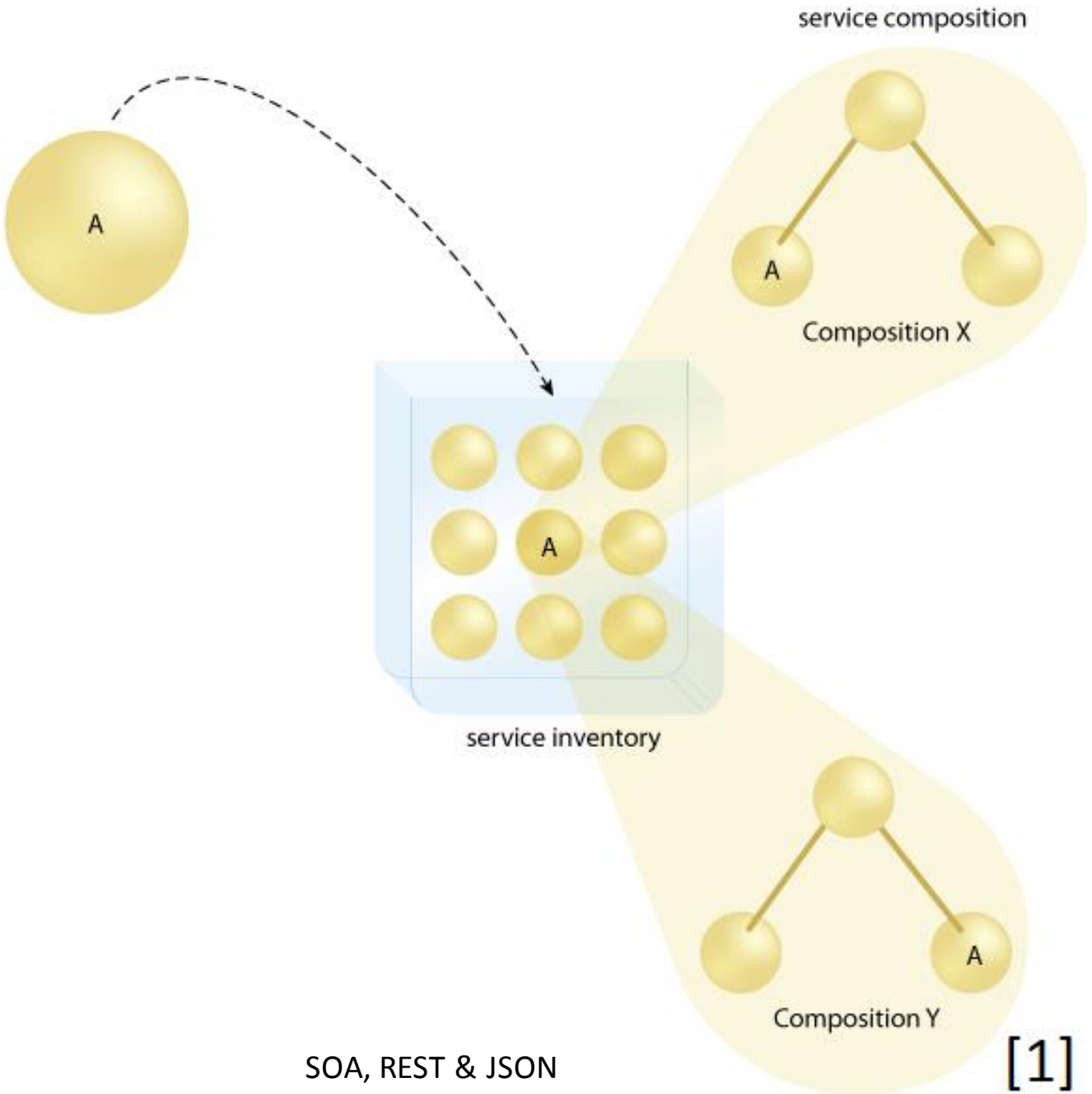
ity

se Coupling

bstraction

ct

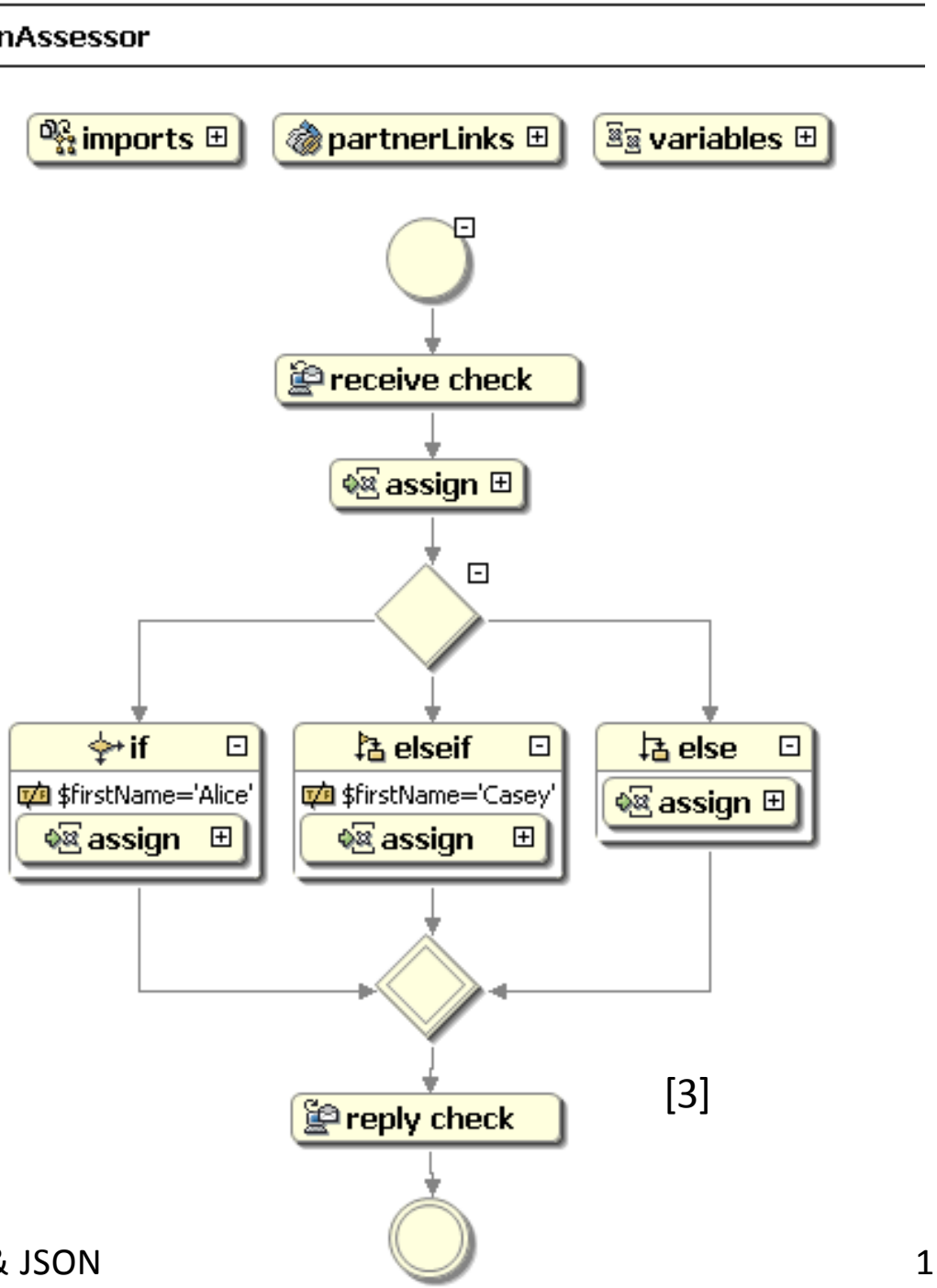
[1]


service composition
A
A
Composition X
A
service inventory
A
Composition Y

Orchestration

- BPEL
- Graphically composing services from a very business process view
- Generating new processes without coding
- Generating compositions
- Power of SOA

BPEL

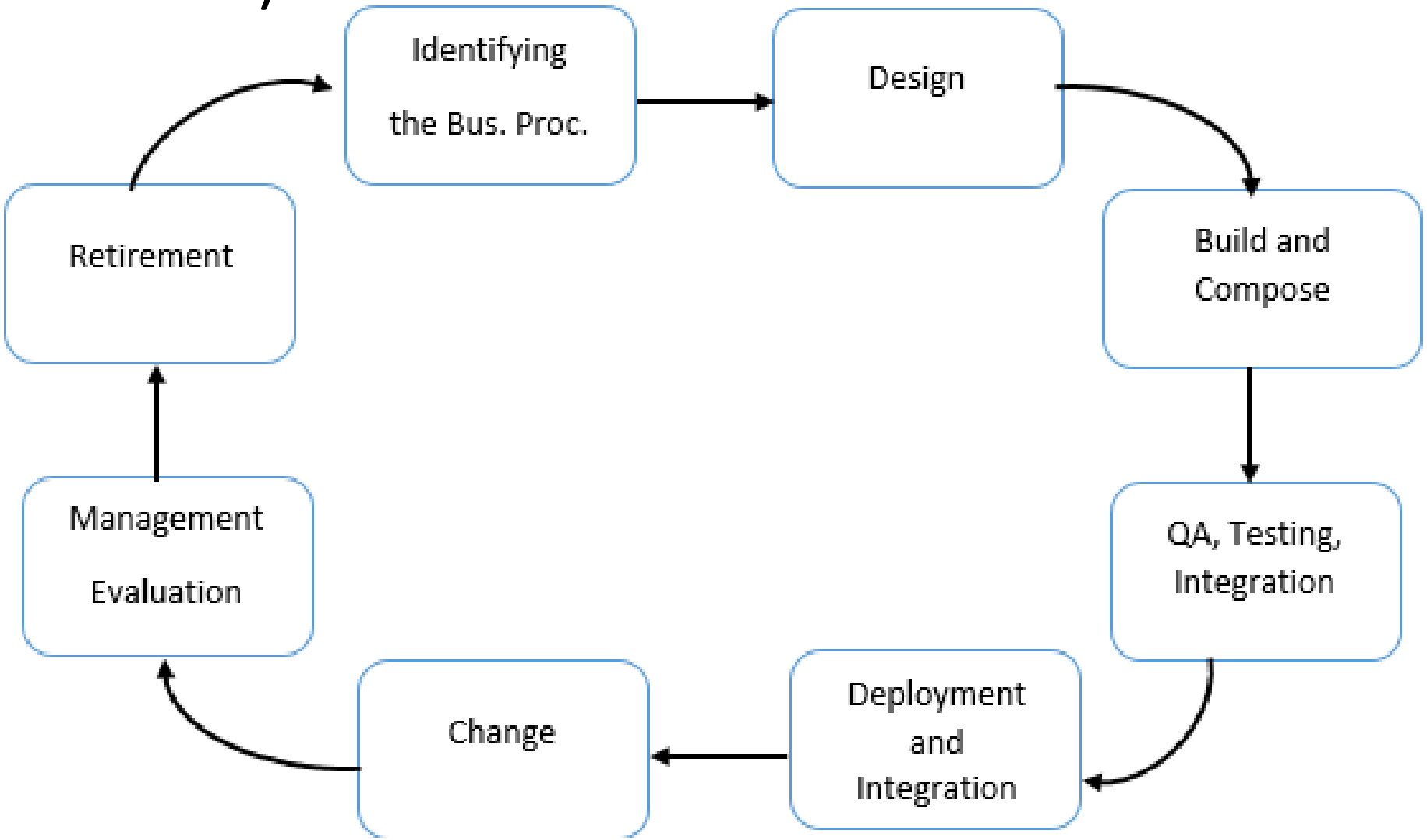

process loanAssessor
imports
partnerLinks
variables
receive check
assign
if
$firstName='Alice'
assign
elseif
$firstName='Casey'
assign
else
assign
reply check

[3]

Lifecycle

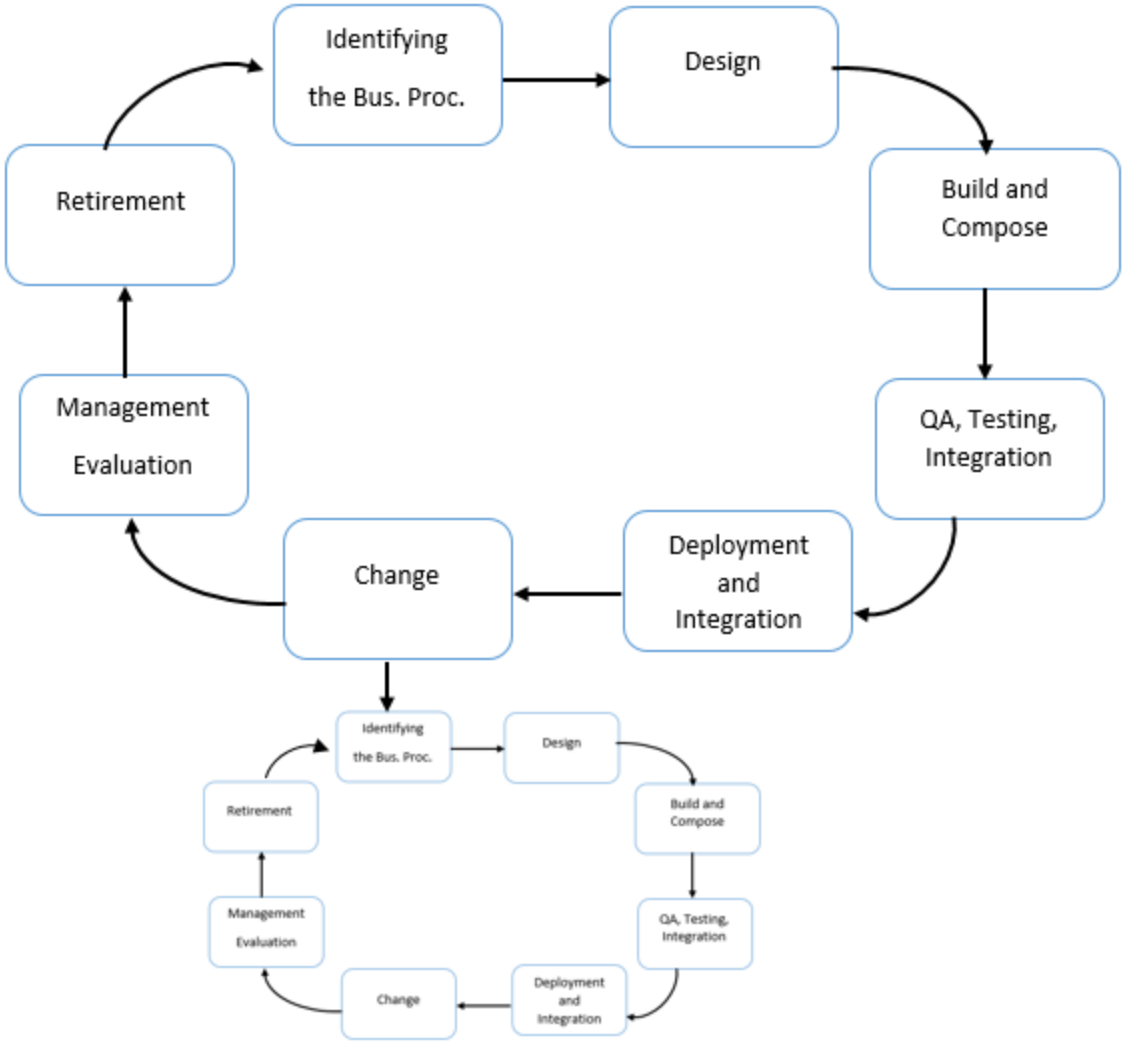
- Normal Lifcycle + Service Lifecycle
- Differences from normal software development:
  - Identifying the Business Process
  - Build and compose
  - Change
  - Retirement

Lifecycle

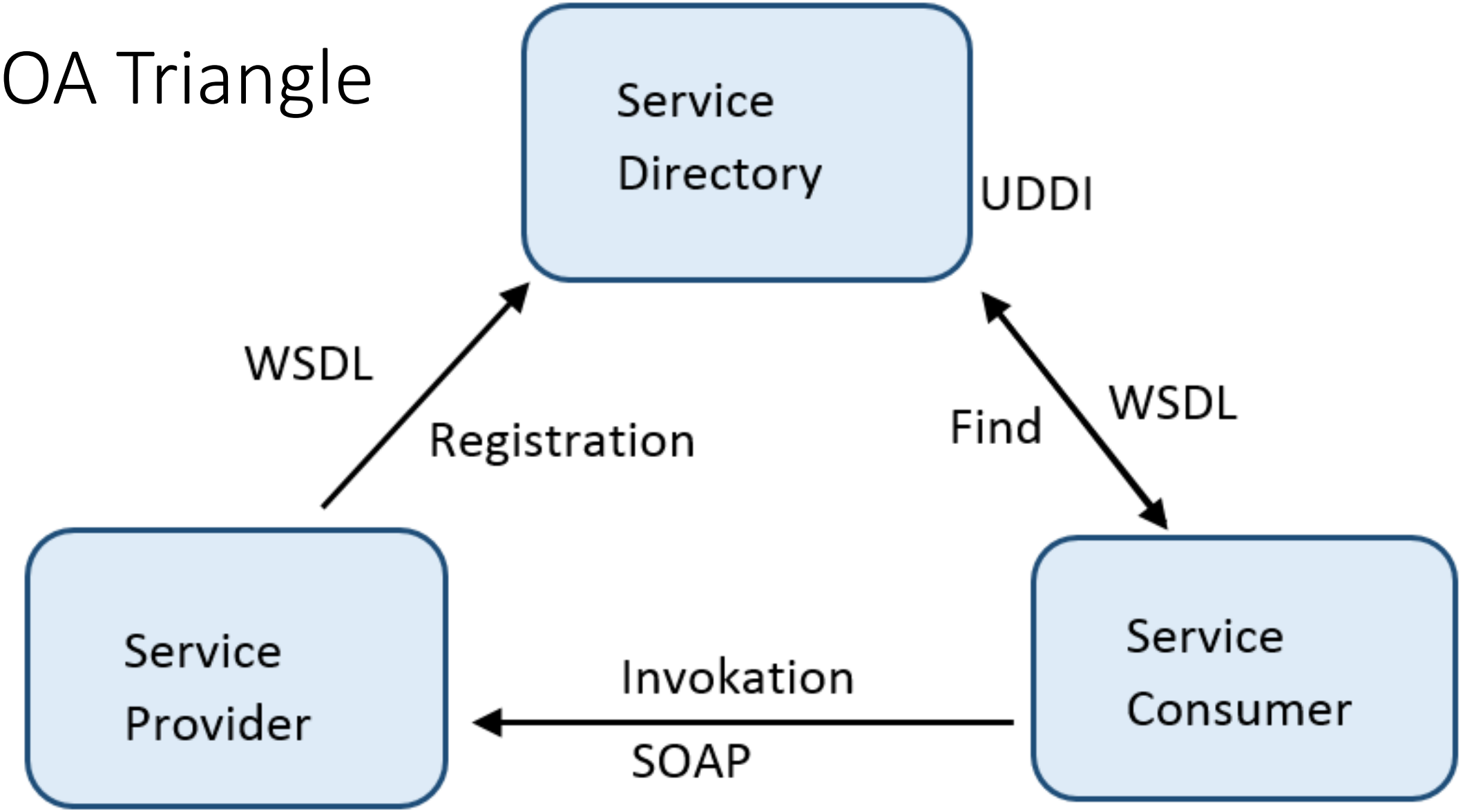

Identifying
the Bus. Proc.
Design
Build and
Compose
QA, Testing,
Integration
Deployment
and
Integration
Change
Management
Evaluation
Retirement

Lifecycle

- Normal Lifcycle + Service Lifecycle
- Differences from normal software
  - Identifying the Business Process
  - Build and compose
  - Change
  - Retirement



SOA Triangle

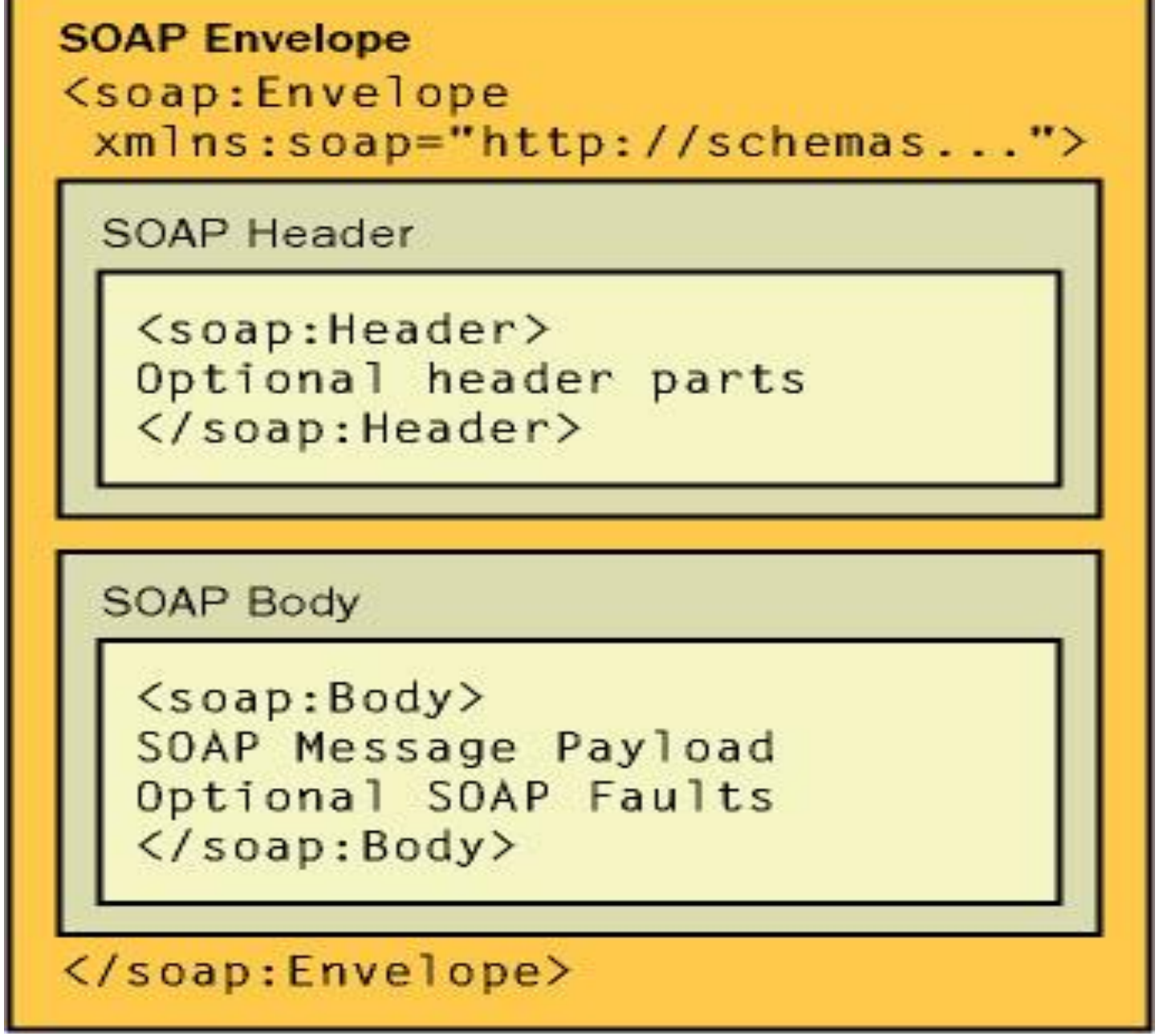


SOAP

- SOAP is an XML based protocol for accessing Web Services
- Originally shortcut for "Simple Object Access Protocol“
- Stand for itself now
- Ordinary XML document

SOAP Message Information

- An Envelope element that identifies the XML document as a SOAP message
- A Header element that contains header information
- A Body element that contains call and response information
- A Fault element containing errors and status information"


SOAP Envelope
<soap:Envelope
xmlns:soap="http://schemas...">
SOAP Header
<soap:Header>
Optional header parts
</soap:Header>
SOAP Body
<soap:Body>
SOAP Message Payload
Optional SOAP Faults
</soap:Body>
</soap:Envelope>

Request

```
<soap:Envelopexmlns:soap="http://www.w3.org/2001/12/soap-
envelope"soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
 <m:GetStockPrice>
  <m:StockName>IBM</m:StockName>
  </m:GetStockPrice>
</soap:Body>

</soap:Envelope>
```

Response

```
<soap:Envelopexmlns:soap="http://www.w3.org/2001/12/soap-
envelope"soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
<m:GetStockPriceResponse>
  <m:Price>34.5</m:Price>
</m:GetStockPriceResponse>
</soap:Body>

</soap:Envelope>
```

WSDL

- Web Service Description Language
- XML based Interface description language
- Used to describe web services which are called by SOAP-Messages
- Specifies location and operations the service exposes

WSDL File Contents

- **Definition :** Defines web-service name
- **Message** : Describes data being exchanges (in-ouput)
- **Type** : Define data types
- **Port Type** : Combines message elements to one operation
- **Binding** : How Port Type will be transmitted
- **Port** : Where the service can be accessed
- **Service:** What Ports support the web service

UDDI

- Universal Description, Discovery and Integration
- Businesses can register and search for Web services
- Communicates via SOAP
- Uses WSDL to describe interfaces to web services
- Tools : Apache: jUDDI, INFRAVIO X-Registry Platform

Migration of Legacy Systems

- 50% applications designed SOA
- Rarely start from scratch
- Existing legacy systems -> Services
- Already achieved different domains

REST

- REpresentational State Transfer
- Guidelines and best practices For Creating Web-Services
- Work with HTTP-Protocol
- Rest is basically using HTTP Requests

HTTP Requests

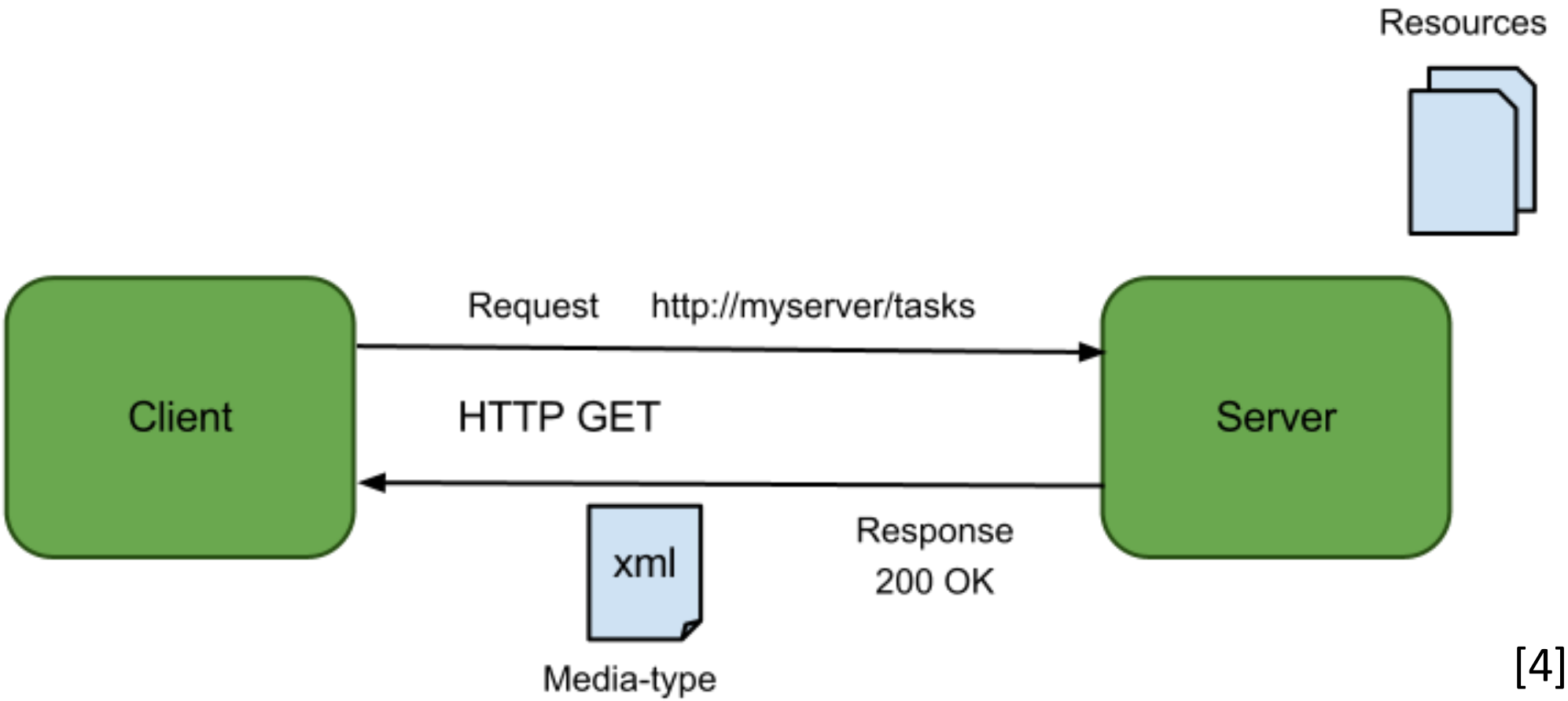
- GET : used to get a representation of a resource (READ)
- PUT : is most-often utilized for update capabilities(UPDATE)
- POST :most-often utilized for creation of new resources(CREATE)
- DELETE : It is used to delete a resource identified by a URI(DELETE)

RESTful Web-Services

- Restful applications use HTTP requests
- HTTP Requests GET,PUT,POST,DELETE
- To access Recoures use URI

(Uniform Resource Identifiers)

- Exchange of representations of resources


Resources
Client
Request http://myserver/tasks
HTTP GET
Server
xml
Response
200 OK
Media-type

[4]

XML

- EXtensible Markup Language
- Designed to describe data
- Software- and hardware-independent tool for carrying information
- Nearly all other standards originate from XML

XML Example

```
<employees>
        <employee>
                <firstName>John</firstName>
                <lastName>Doe</lastName>
        </employee>
        <employee>
                <firstName>Anna</firstName>
                <lastName>Smith</lastName>
        </employee>
        <employee>
                <firstName>Peter</firstName>
                <lastName>Jones</lastName>
        </employee>
</employees>
```

JSON

- JavaScript Object Notation
- Lightweight data-interchange format
- does not support name spaces and schemed based validation
- Easier to understand than XML

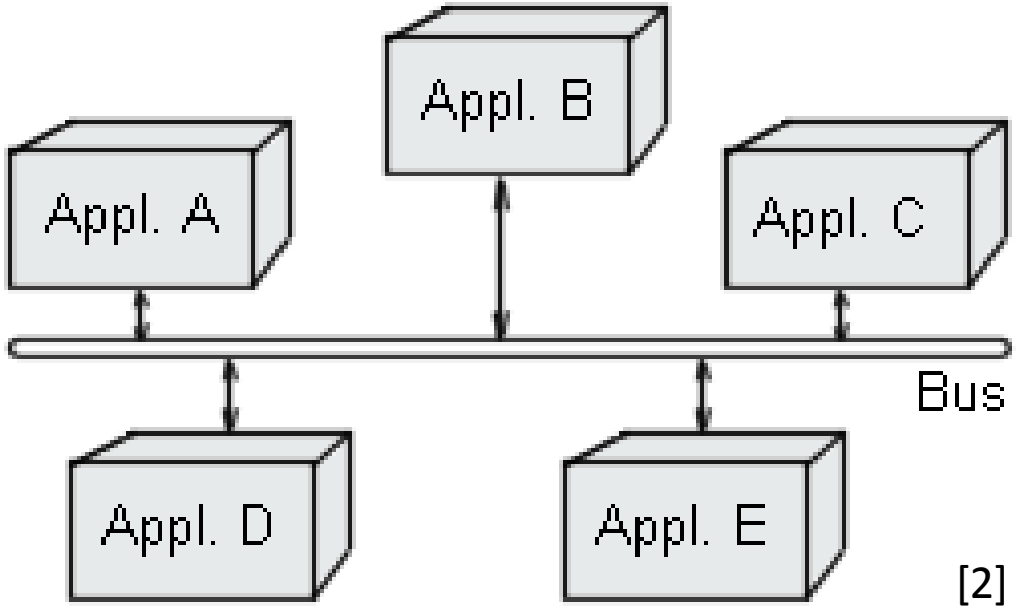
JSON Example

```
{{"employees:[
{"firstName":"John", "lastName":"Doe"},
{"firstName":"Anna", "lastName":"Smith"},
{"firstName":"Peter", "lastName":"Jones"}]}
```

REST EXAMPLE

[6]

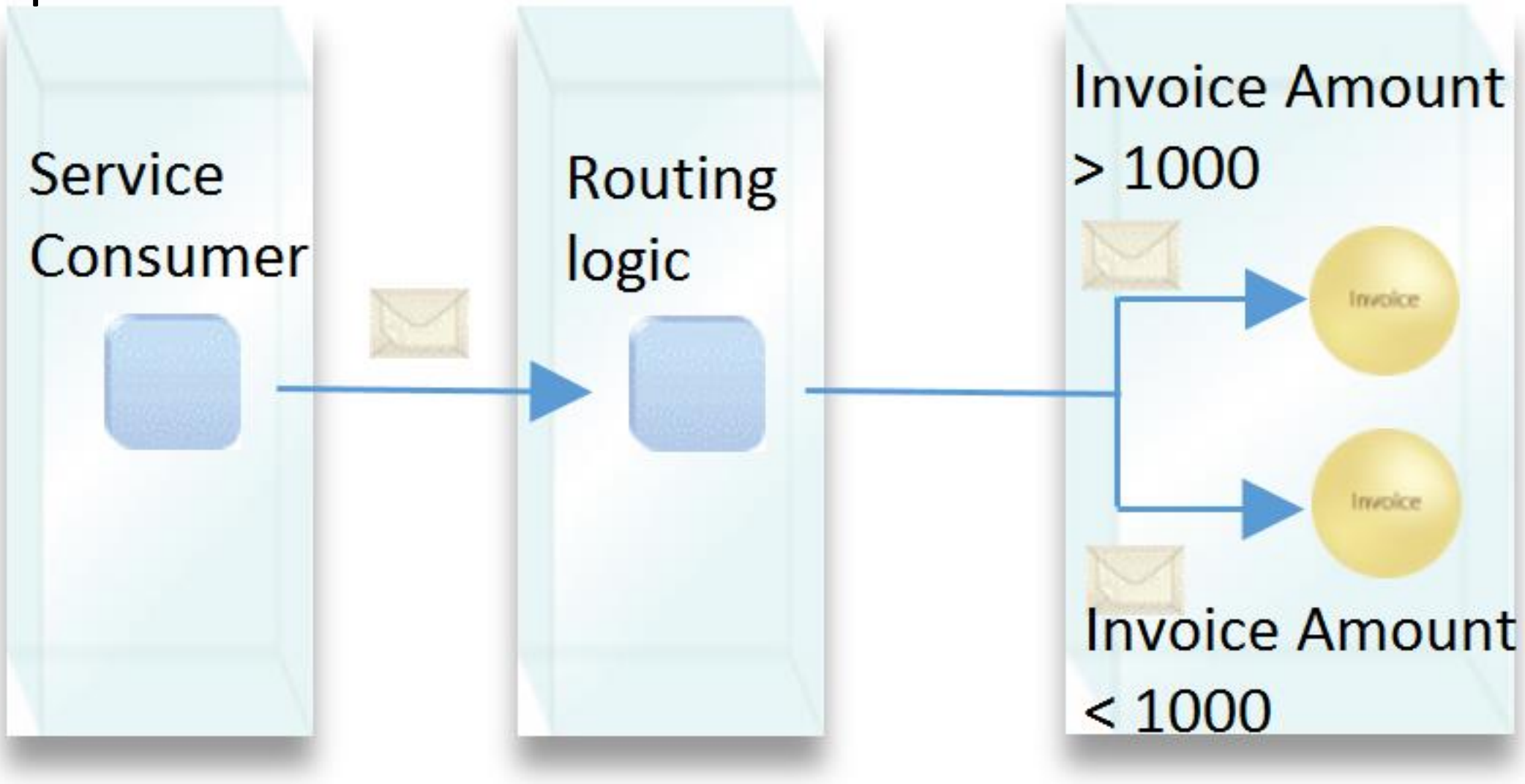
Enterprise Service Bus



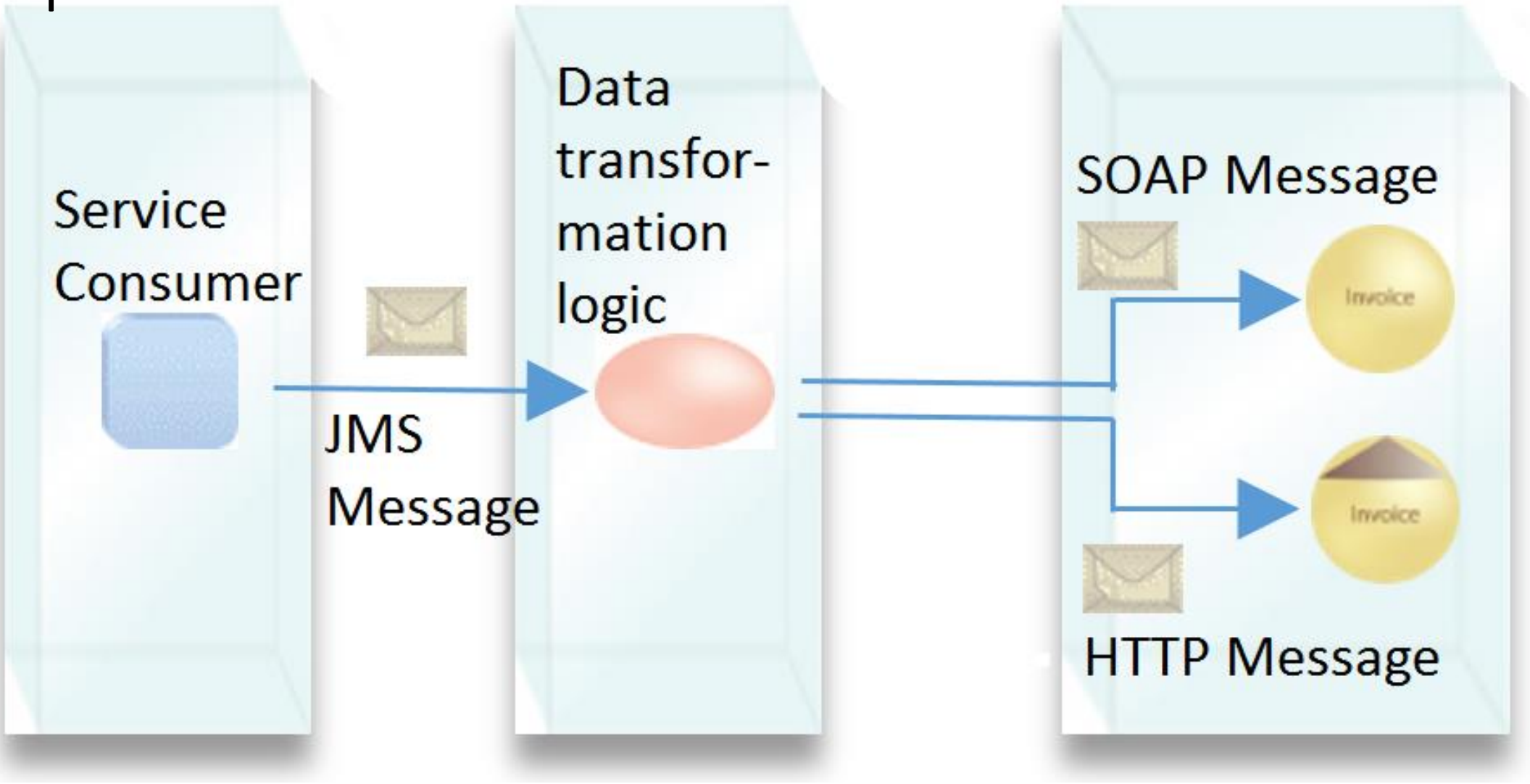
[2]

- Distribute information
- Routing
- Mask differences among underlying platforms
- Ensure information delivery
- Re-route, log, and enrich information

Enterprise Service Bus



Enterprise Service Bus



Summary & Conclusion

Thank you for your attention!
Any Questions?

Sources

[1] SOA Principles of Service Design, (c) Prentice Hall/PearsonPTR - http://serviceorientation.com

[2] EAI Enterprise Application Integration,Thorsten Horn
http://www.torsten-horn.de/techdocs/eai.htm
last used : 28.02.2014, 20:25

[3] http://documentation.progress.com/infocenter/sonic/8.5/index.jsp

[4] http://prideparrot.com/blog/archive/2012/3/creating_a_rest_service_using_asp_net_web_api

[5] http://flylib.com/books/en/2.439.1.22/1/

[6] https://spring.io/guides/gs/rest-service/