Event Driven Architectures

Java Usergroup (TU Wien)
2. Feb 2009

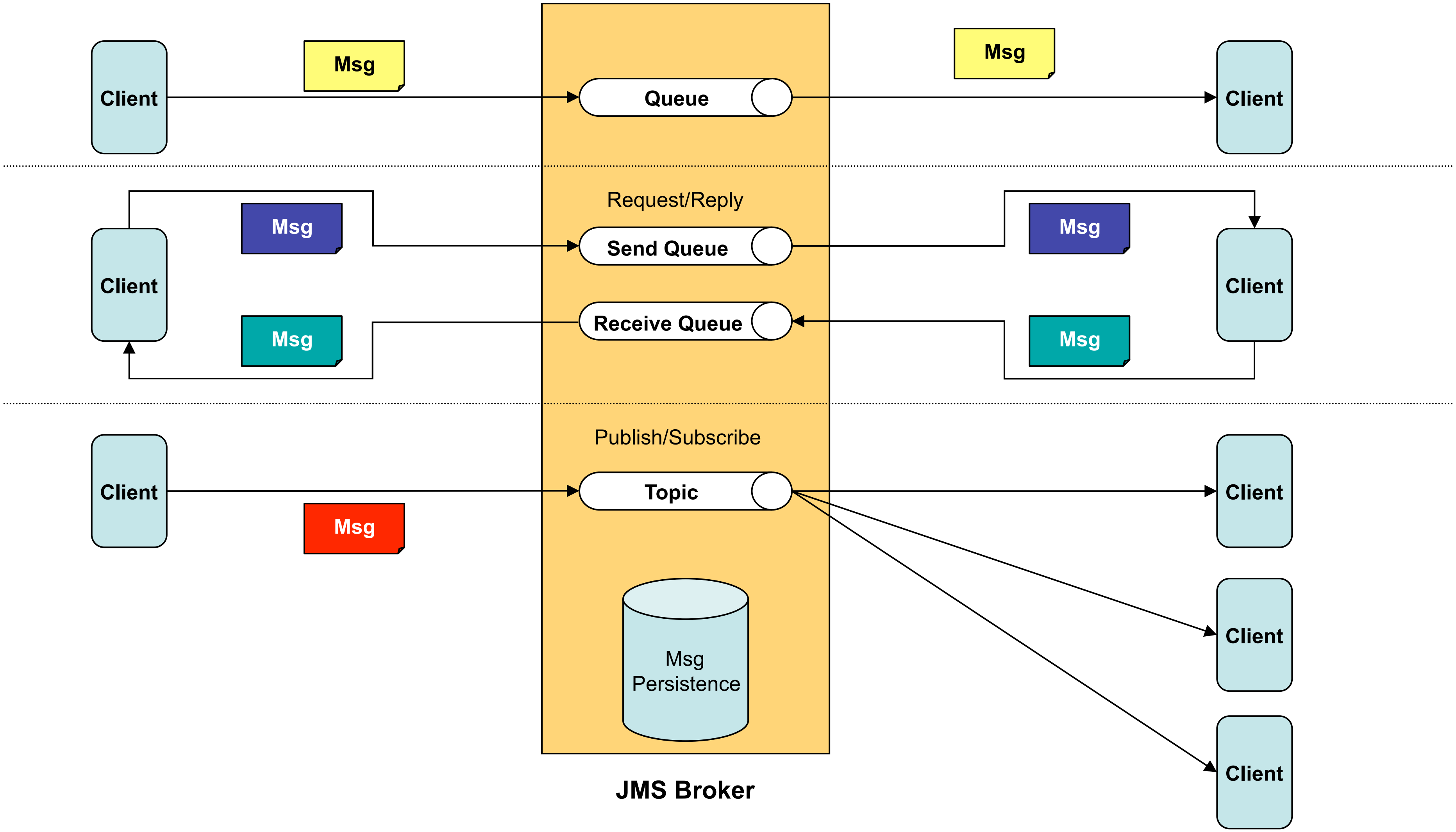
Alexander Schatten

Agenda

- Basics: Message Integration Patterns
- Enterprise Service Bus
- Enterprise Integration Patterns
- Event Driven Architecture
- Command vs. Event Messages
- Event-Driven Business
- Showcase/Example illustrating EDA
- EDA vs. (?) SOA
- Java Frameworks

Basics:
Messaging Integration Patterns

Example: Java Messaging Service Broker



Enterprise Integration Patterns

- See Website and Book from Gregor Hohpe: http://www.integrationpatterns.com/
- Patterns to provide guidance in many “typical” enterprise integration scenarios
- Organisation of Patterns
  - Integration Style
  - Channel Patterns
  - Message Construction Patterns
  - Routing Patterns
  - Transformation Patterns
  - Endpoint Patterns
  - System Management Patterns

Enterprise Service Bus?
Example: Apache Service Mix

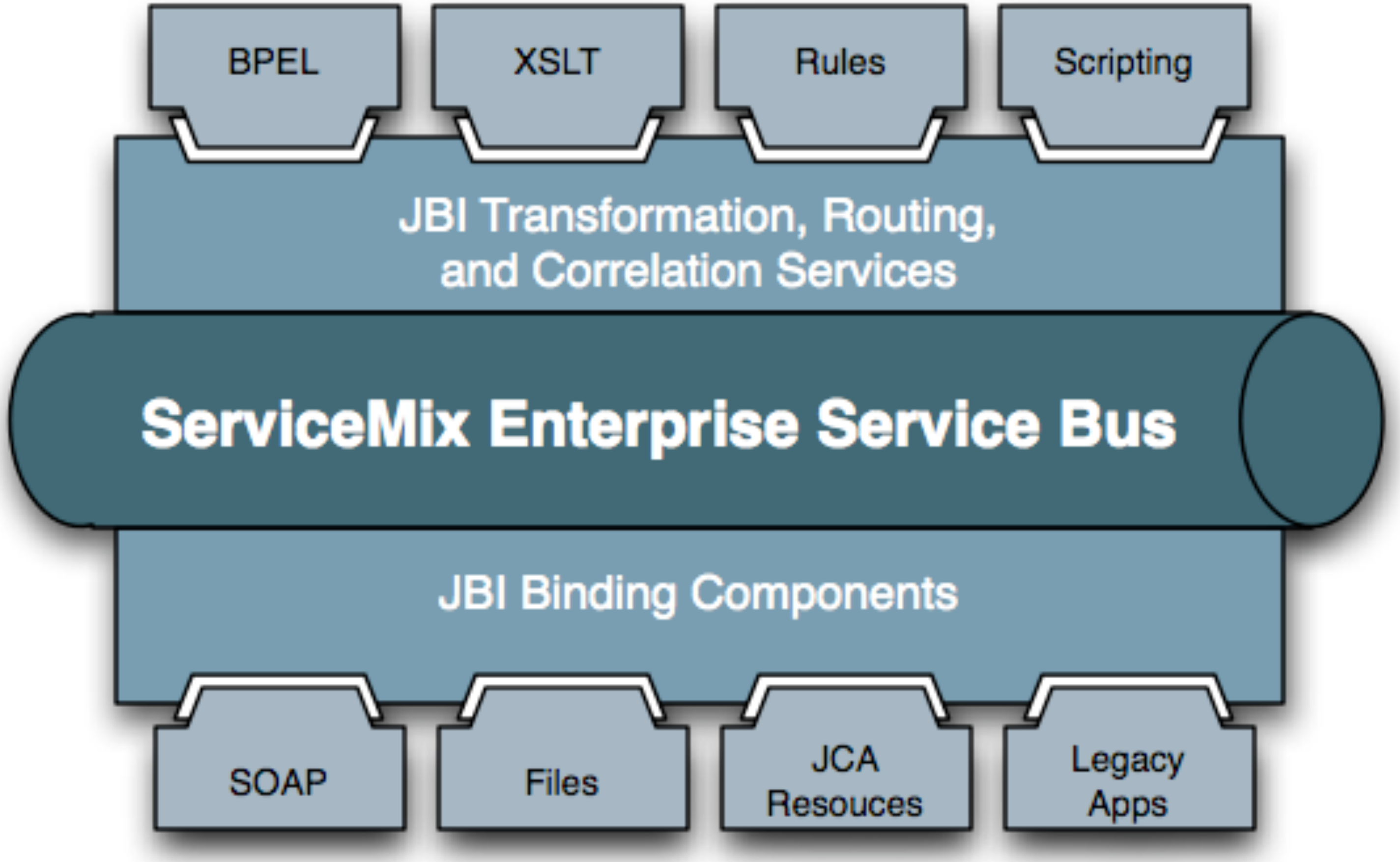

BPEL
XSLT
Rules
Scripting
JBI Transformation, Routing, and Correlation Services
ServiceMix Enterprise Service Bus
JBI Binding Components
SOAP
Files
JCA Resouces
Legacy Apps

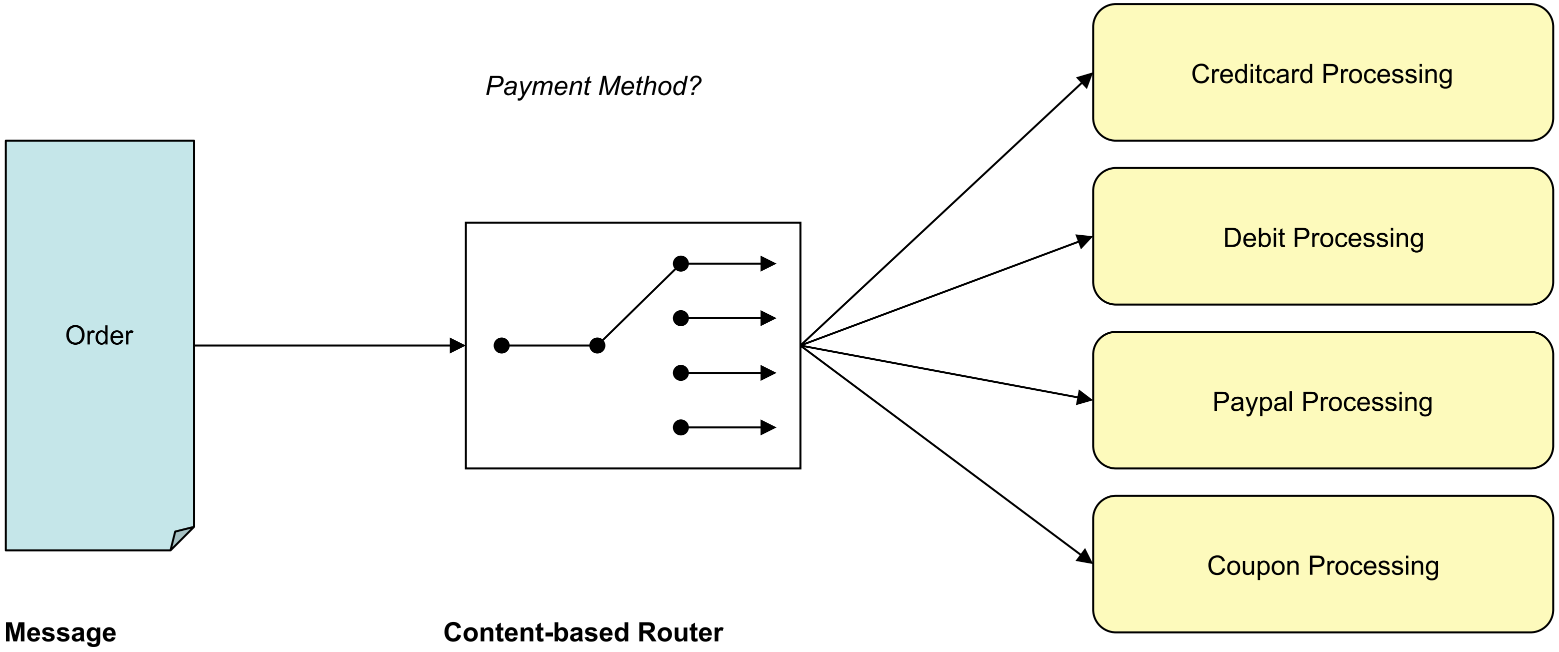
Figure taken from Apache Service Mix documentation

JBI Standard

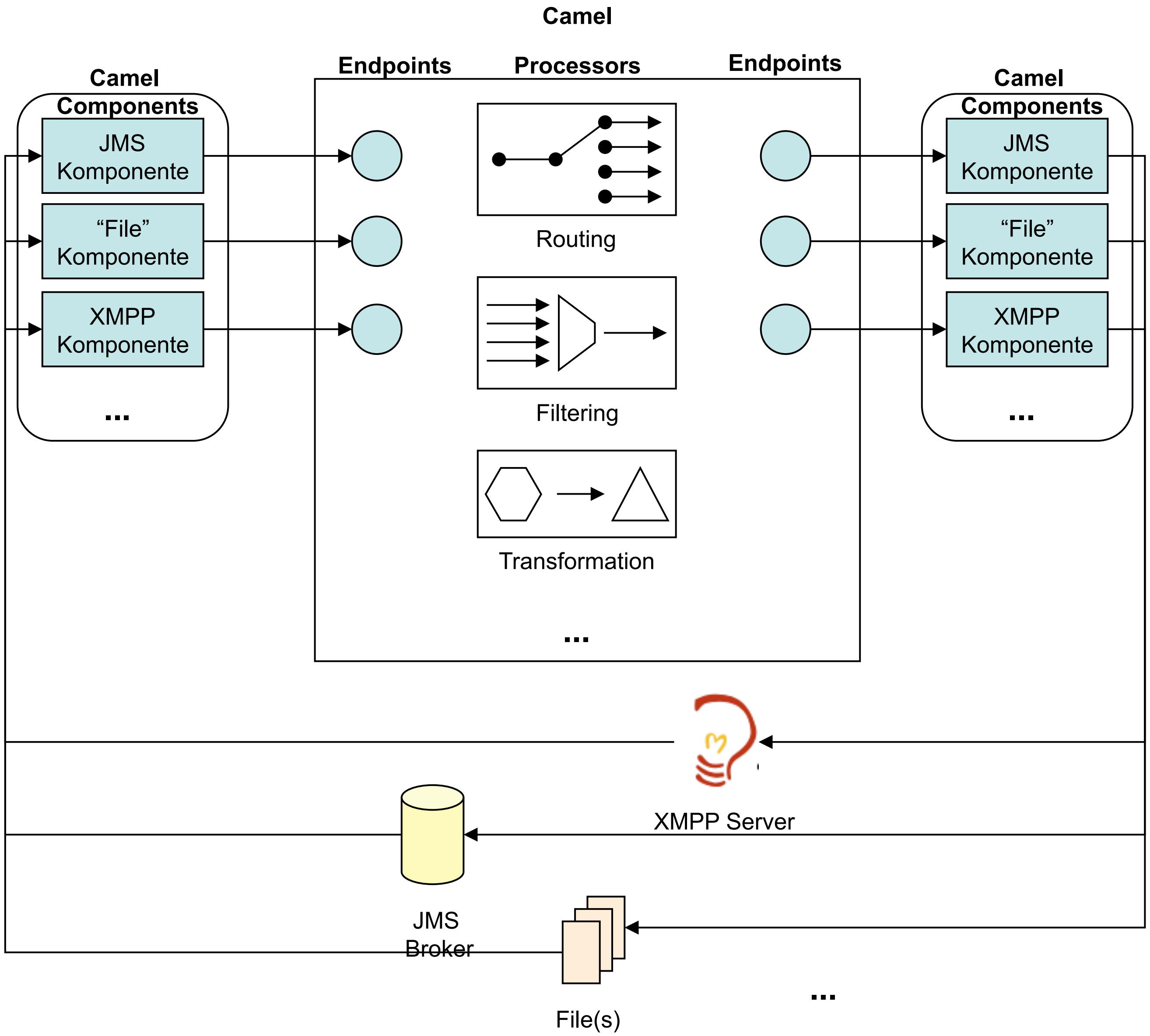
- Java Business Integration
- Container for integration components and services
- WSDL-Based Messaging Model
- Normalized Message Service and Router
- Components
  - Binding Components (e.g., JMS, Jabber, REST, ...)
  - Service Engines (BPEL, Camel, Quartz, Drools, ...)
- JMX Based Administration (Admin Tools)

- **JBI Components independent of ESB Implementation**

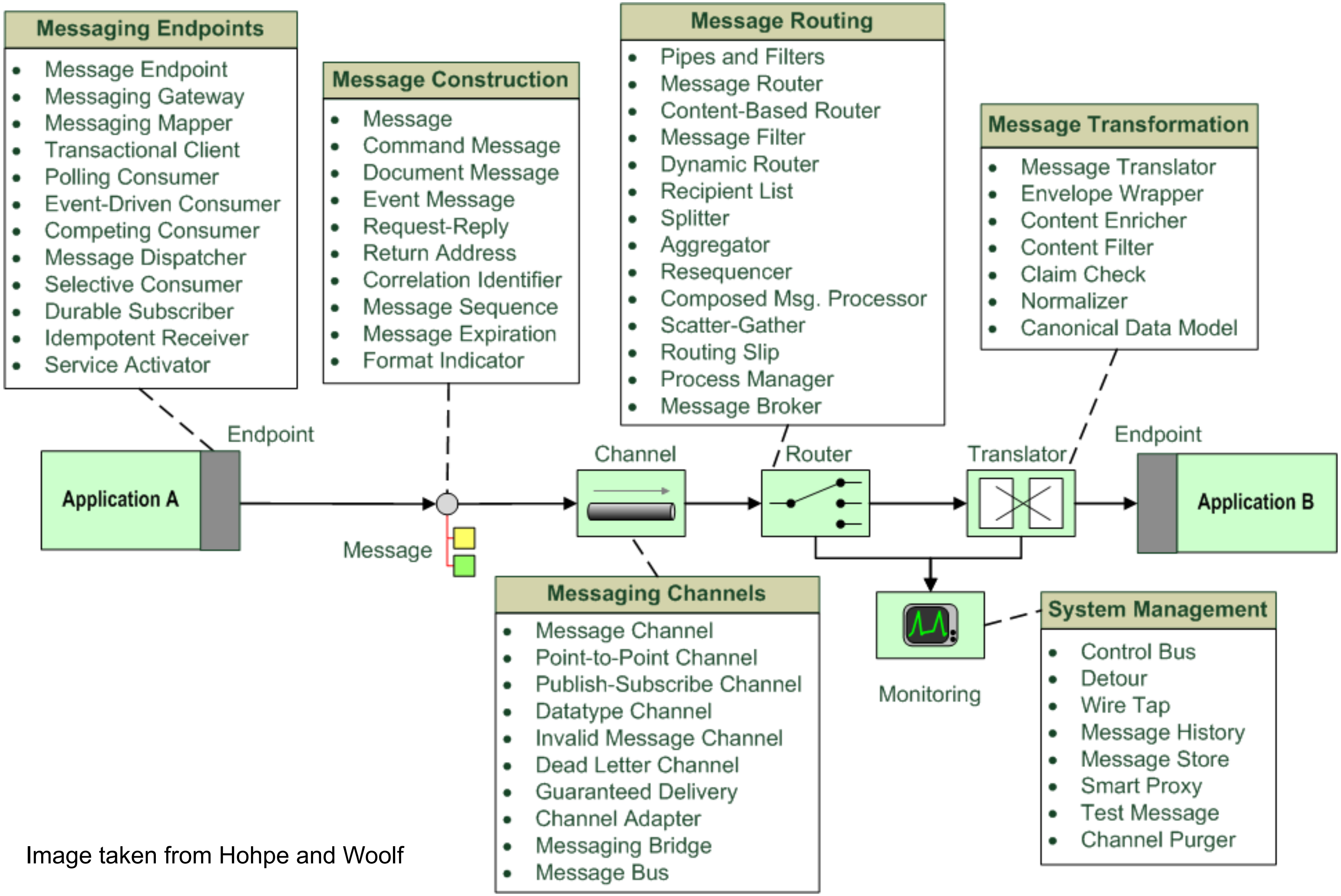
Example: Content-based Router



Example: MOM, ESB and Apache Camel



Enterprise Integration Patterns



Messaging Endpoints

- Message Endpoint
- Messaging Gateway
- Messaging Mapper
- Transactional Client
- Polling Consumer
- Event-Driven Consumer
- Competing Consumer
- Message Dispatcher
- Selective Consumer
- Durable Subscriber
- Idempotent Receiver
- Service Activator

Message Construction

- Message
- Command Message
- Document Message
- Event Message
- Request-Reply
- Return Address
- Correlation Identifier
- Message Sequence
- Message Expiration
- Format Indicator

Message Routing

- Pipes and Filters
- Message Router
- Content-Based Router
- Message Filter
- Dynamic Router
- Recipient List
- Splitter
- Aggregator
- Resequencer
- Composed Msg. Processor
- Scatter-Gather
- Routing Slip
- Process Manager
- Message Broker

Message Transformation

- Message Translator
- Envelope Wrapper
- Content Enricher
- Content Filter
- Claim Check
- Normalizer
- Canonical Data Model

Messaging Channels

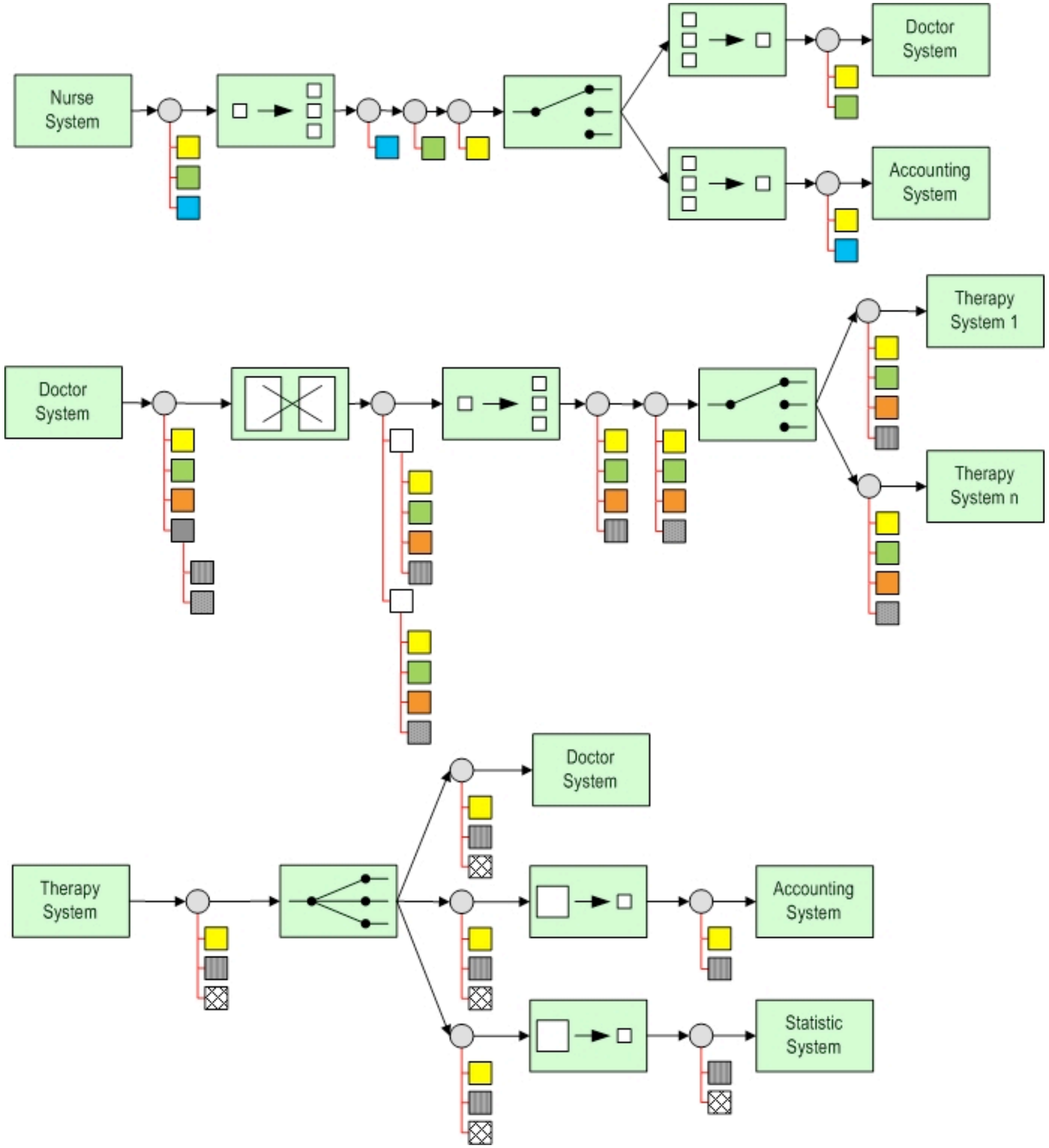
- Message Channel
- Point-to-Point Channel
- Publish-Subscribe Channel
- Datatype Channel
- Invalid Message Channel
- Dead Letter Channel
- Guaranteed Delivery
- Channel Adapter
- Messaging Bridge
- Message Bus

System Management

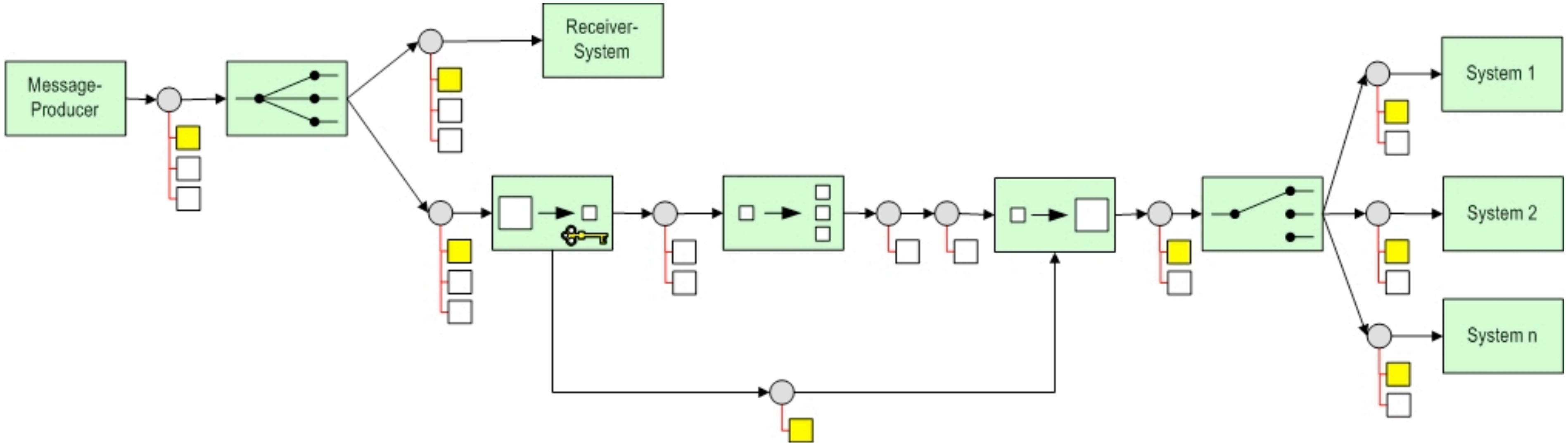
- Control Bus
- Detour
- Wire Tap
- Message History
- Message Store
- Smart Proxy
- Test Message
- Channel Purger

Image taken from Hohpe and Woolf

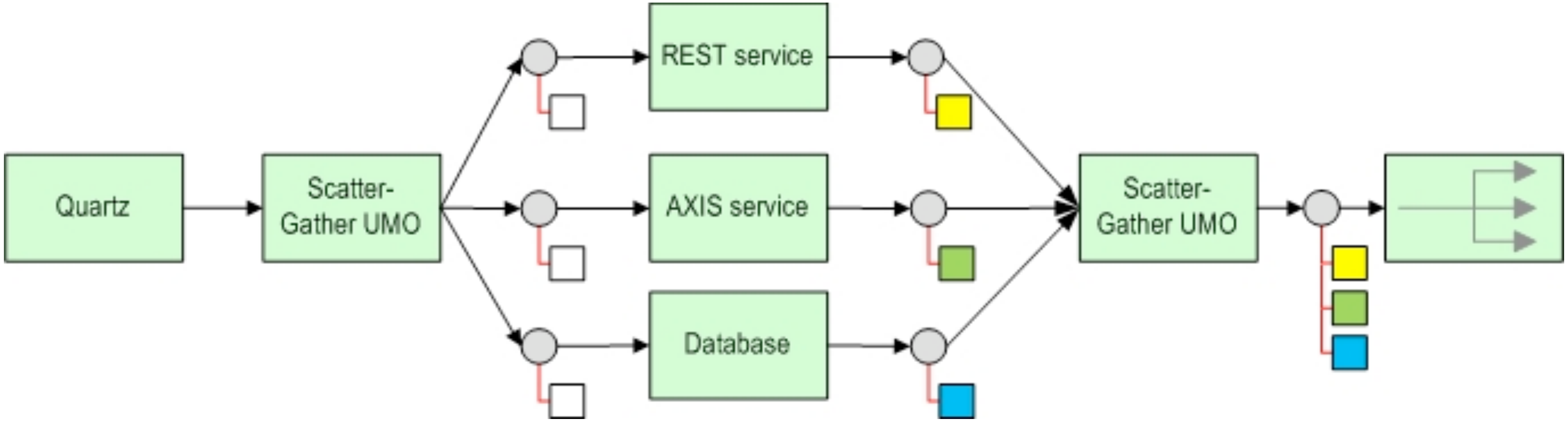
Scenario: Hospital



Scenario: Railway Splitter



Weather Information System



Event-Driven Architecture

Command vs. Event

Command

- Calculate Exchange from X $ → €
- Get Temperature at Sensor X
- Store new Address in Database
- Send Email to …
- Print Document …
- Lock account of customer

Event

- Concentration Level of Substance X on Sensor Y exceeded threshold
- Airplane X landed
- Customer X orders book Y
- Bell rings
- Credit Check of order X failed
- Container with ID X loaded

Command → Command-Message

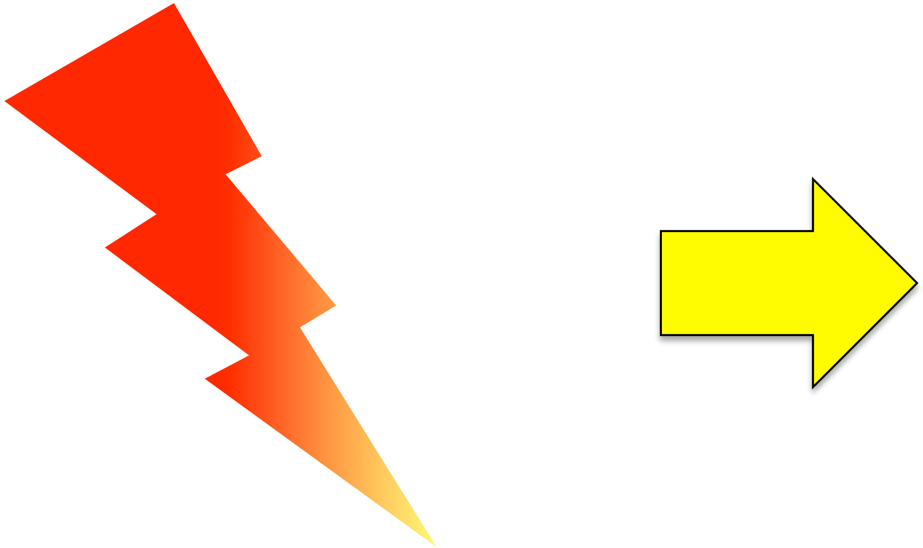
Command:

Store new Customer “Alexander Schatten” to Database

```
<xml …

<rpc:storeCustomer …>
  <customer>
    <firstname>
      Alexander
    </firstname>
    <lastname>
      Schatten
    </lastname>
    …
  </customer>
</rpc:storeCustomer>
```

Message:
Representation of Command

Event → Event-Message



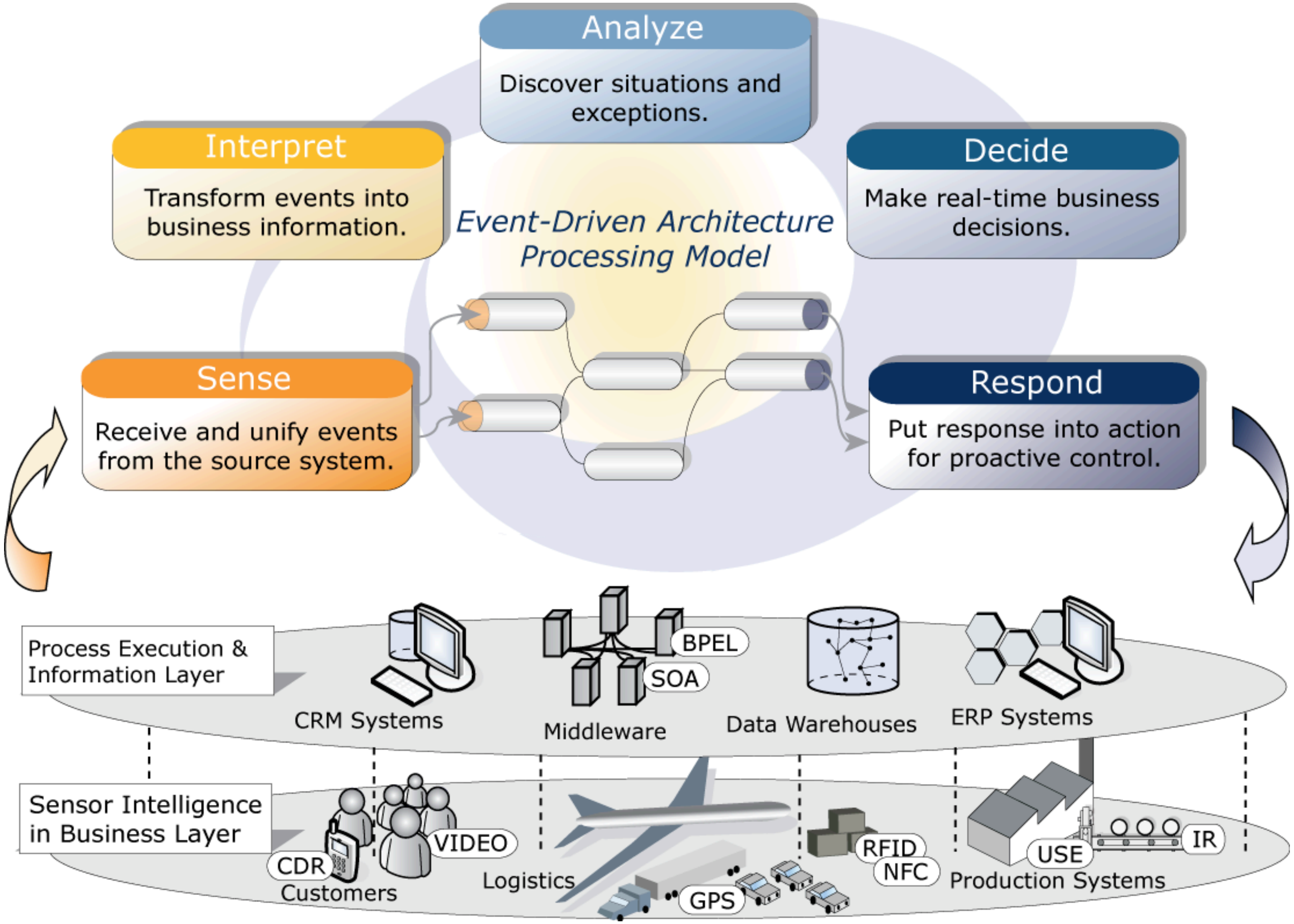
```
<xml …

<landing>
   <flight nr="1234" />
   <passengers>
      …
   </passengers>
   …
</landing>
```

Event:
Airplane Flight Nr 1234 lands

Message:
Representation of Event

Event Driven Business

- Real World Systems get more and more enriched with "sensory components" like RFID
- Every process step in the real world (particularly when multiple companies are involved) has an according step in the IT System (traditionally paper-flow)
- Towards Event-Driven Architectures:
  - Events in "real world" trigger IT Systems


Analyze
Discover situations and exceptions.
Interpret
Transform events into business information.
Decide
Make real-time business decisions.
Event-Driven Architecture Processing Model
Sense
Receive and unify events from the source system.
Respond
Put response into action for proactive control.
Process Execution & Information Layer
CRM Systems
BPEL
SOA
Middleware
Data Warehouses
ERP Systems
Sensor Intelligence in Business Layer
CDR
VIDEO
Customers
Logistics
GPS
RFID
NFC
USE
IR
Production Systems

Event-Driven Architecture

- Event (Type) not commands and Receiver in the centre of attention
- Decoupling of Sender/Receiver
- Publish/Subscribe or Routing
- Sender does not necessarily know Receiver
- No hard top-down modeling but “emergent” behaviour
- Very high flexibility of architecture
- Potential issue: Keeping overview on running processes
- Alternative or addition to close coupled architectures
- Addition to SOA

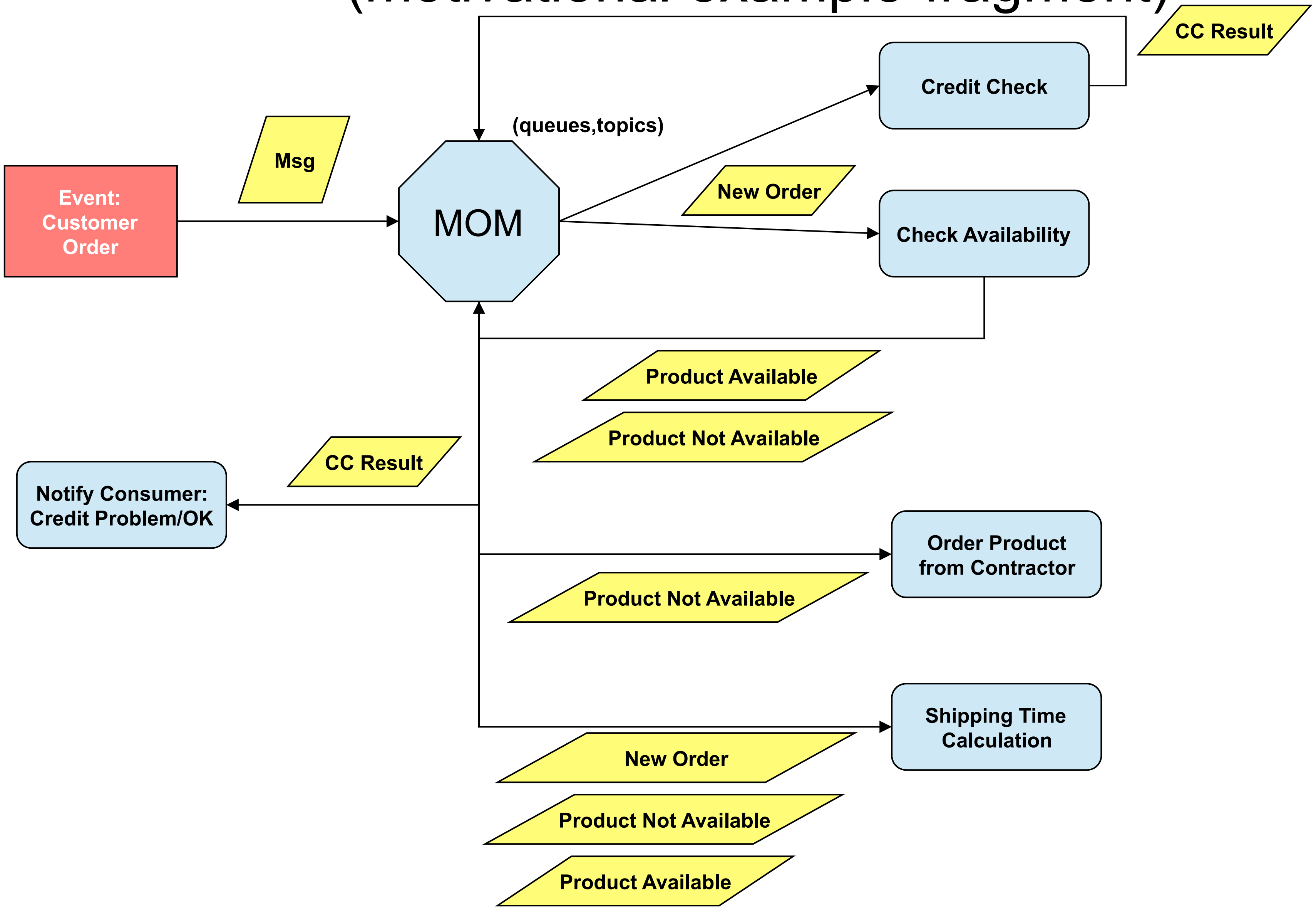
Showcase Requirements

- Webshop
- Customer orders products
- Check customer credit
- Check availability of product(s), order products from contractor(s) if required
- Calculate shipping time
- Start delivery procedure, invoke other systems like ERP, CRM, logistics systems
- Notify Customer about progress
- Perform delivery
- Charge Customer

Traditional Architecture

- Central controler component
- Interacts with several services to perform tasks
- ...
- ...
- ...

Event-Driven Architecture (motivational example-fragment)



SOA and EDA

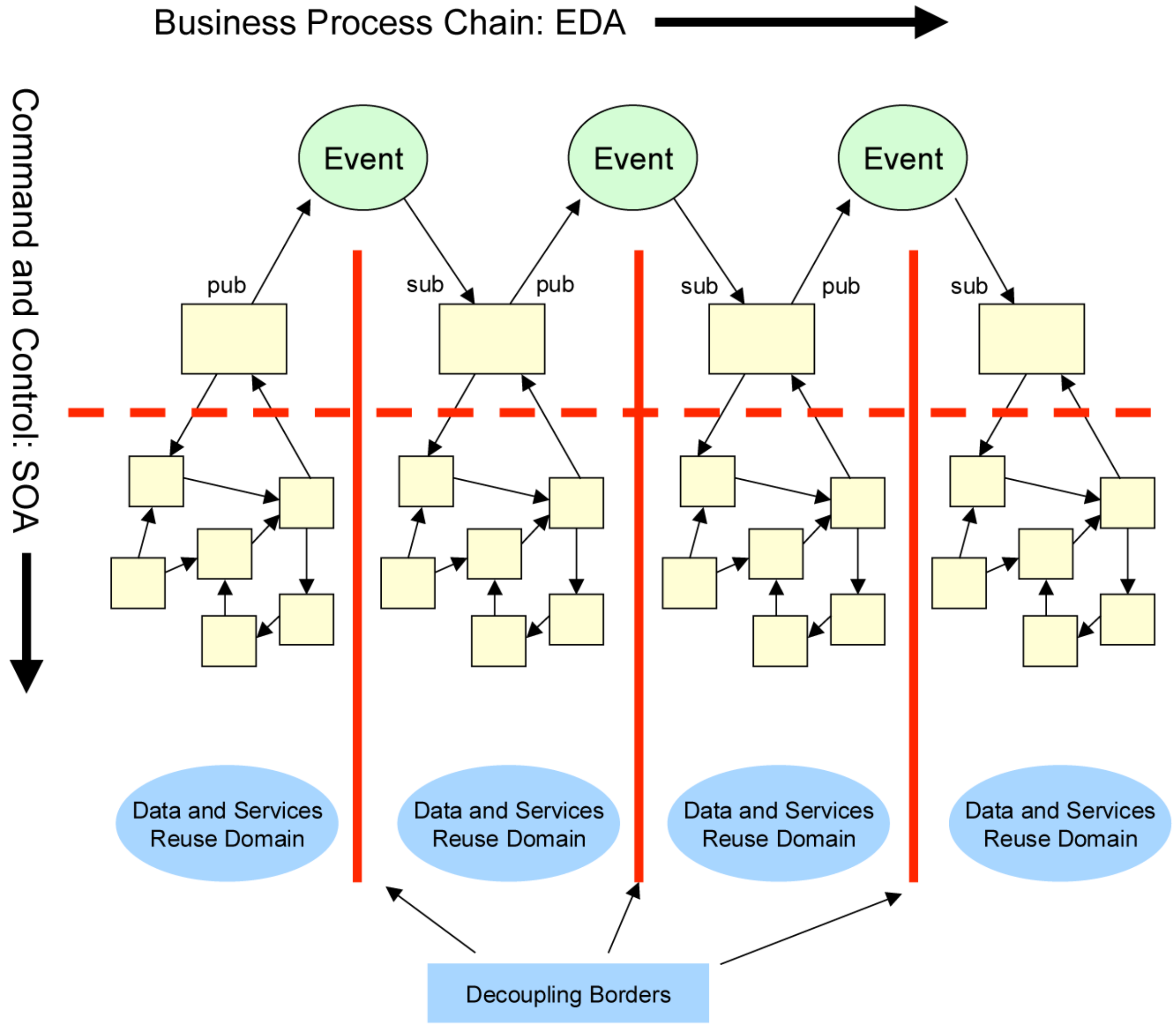

Business Process Chain: EDA
Command and Control: SOA
Event
Event
Event
pub
sub
pub
sub
pub
sub
Data and Services Reuse Domain
Data and Services Reuse Domain
Data and Services Reuse Domain
Data and Services Reuse Domain
Decoupling Borders

Figure taken from Jack van Hoof, “How EDA extends SOA and why it is important”

Complex Event Processing (CEP) ?

New Year in Vienna!

Photo by bithead (flickr)

Java Frameworks

- Message Broker
  - Apache Active MQ
  - Glassfish Open MQ
- Enterprise Service Bus
  - Apache Service Mix (JBI)
  - Mule
  - Glassfish Open ESB
- Enterprise Integration Patterns Support
  - ESB Level: e.g. Mule
  - Apache Camel

Contact

Dr. Alexander Schatten

Institute for Software Technology and Interactive Systems

Vienna University of Technology

http://www.schatten.info