Enterprise Integration Patterns

Enterprise Application Integration with Java EE

Alexander Heusingfeld, @goldstift, #j1eai

We take care of it - personally!

innoQ

EAI Pattern in 2013?

innoQ

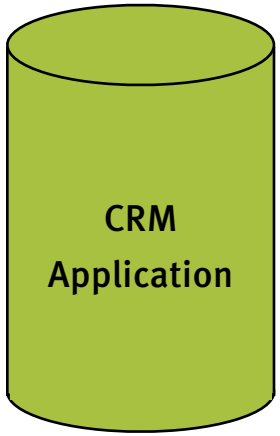
EAI Pattern in 2013?

Nobody does that anymore!

innoQ

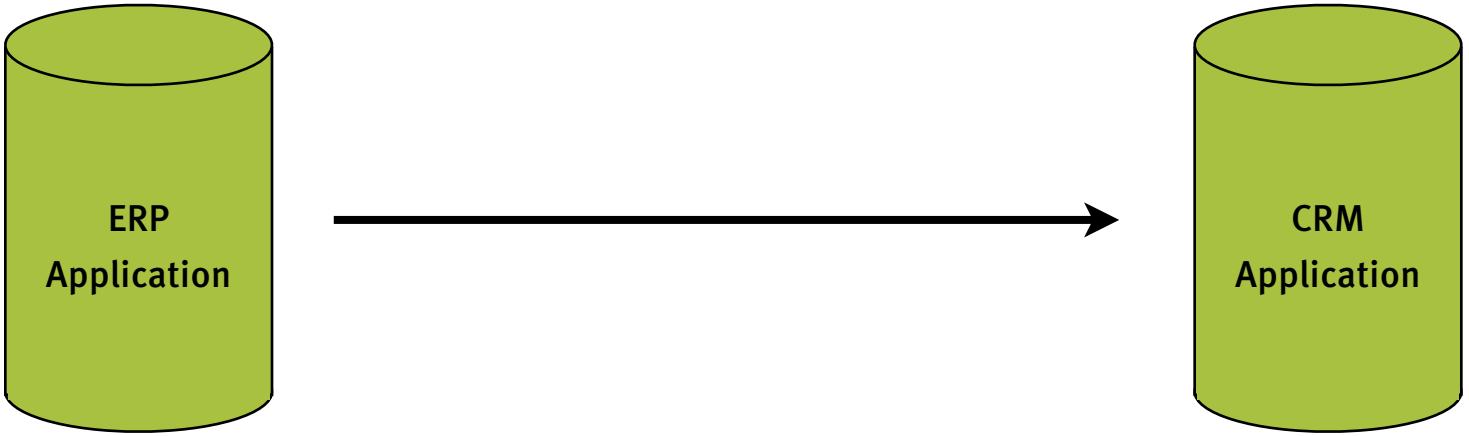
Integration is everywhere

When applications communicate



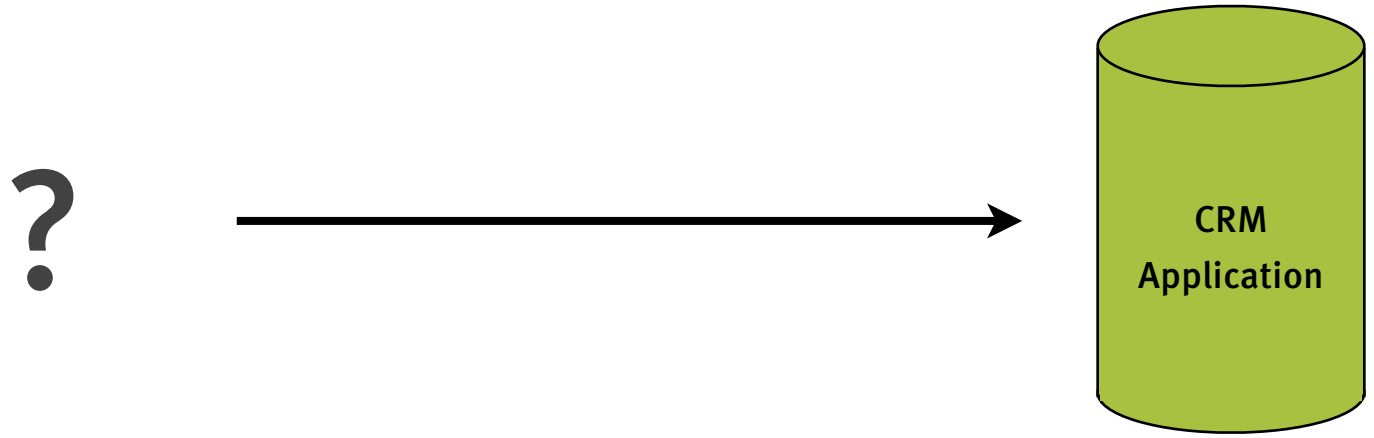
Integration is everywhere

When applications communicate



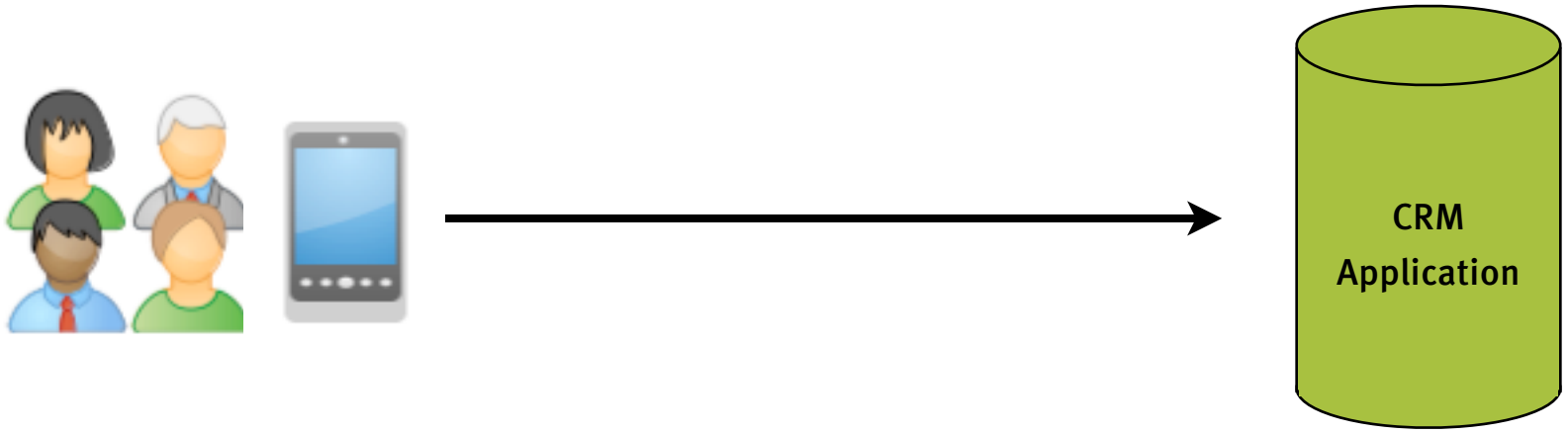
Integration is everywhere

When applications communicate



Integration is everywhere

When applications communicate



Enterprise Integration Pattern

innoQ

Enterprise Integration Pattern

innoQ

Patterns for EAI

innoQ

Patterns for EAI

Design Patterns (Gamma et al), 1994

innoQ

Patterns for EAI

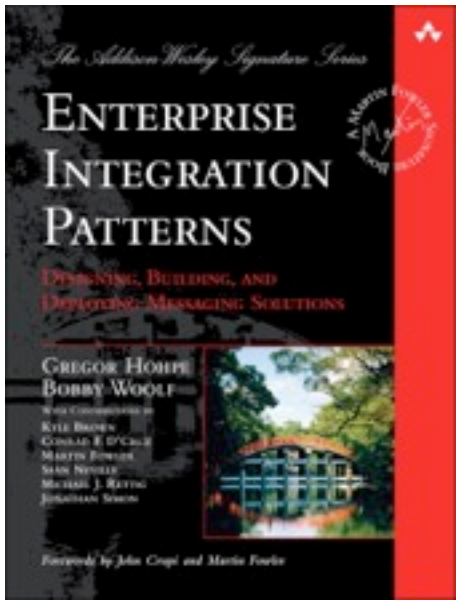
Design Patterns (Gamma et al), 1994

Proven solutions for common problems

Patterns for EAI

Design Patterns (Gamma et al), 1994

Proven solutions for common problems



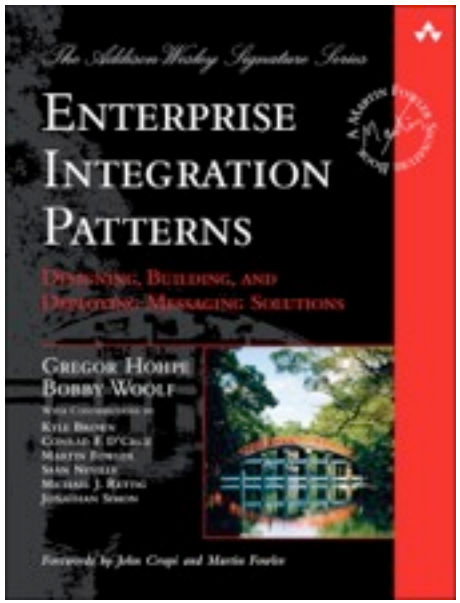
Enterprise Integration Patterns (Hohpe & Woolf), 2003

innoQ

Patterns for EAI

Design Patterns (Gamma et al), 1994

Proven solutions for common problems



Enterprise Integration Patterns (Hohpe & Woolf), 2003

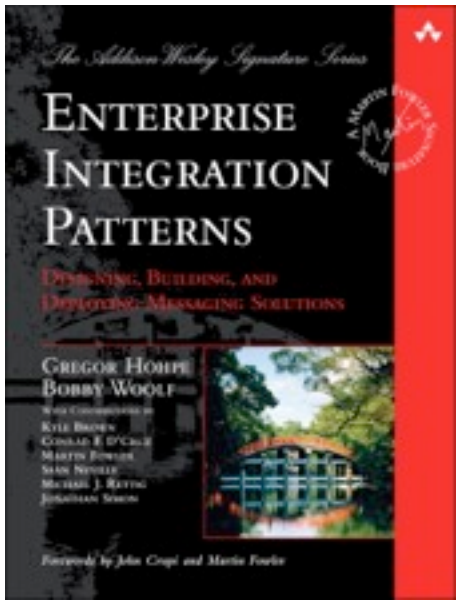
Swiss-army knife for asynchronous messaging

innoQ

Patterns for EAI

Design Patterns (Gamma et al), 1994

Proven solutions for common problems



Enterprise Integration Patterns (Hohpe & Woolf), 2003

Swiss-army knife for asynchronous messaging

www.eaipatterns.com

innoQ

“We all believe that asynchronous messaging carries the greatest promise.”

- Martin Fowler (Enterprise Integration Patterns, 2003)

innoQ


http://bit.ly/PtrhWy
REICHSPOST
Deutsche Post
Variationen
eines
Klassikers
MESSAGING?

Benefits of async. Messaging

innoQ

Benefits of async. Messaging

decoupled

innoQ

Benefits of async. Messaging

decoupled

integrated platforms/ languages

innoQ

Benefits of async. Messaging

decoupled

integrated platforms/ languages

reliable communication

innoQ

Benefits of async. Messaging

decoupled

integrated platforms/ languages

reliable communication

disconnected

innoQ

Benefits of async. Messaging

decoupled

integrated platforms/ languages

reliable communication

disconnected

throttled

innoQ

But why those Patterns?

We take care of it - personally!

innoQ

Thoughts on EAI patterns

innoQ

Thoughts on EAI patterns

not invented, but observed

innoQ

Thoughts on EAI patterns

not invented, but observed

don’t solve everything

innoQ

Thoughts on EAI patterns

not invented, but observed

don’t solve everything

provide ideas

innoQ

Thoughts on EAI patterns

not invented, but observed

don’t solve everything

provide ideas

good ones evolve

innoQ

Thoughts on EAI patterns

not invented, but observed

don’t solve everything

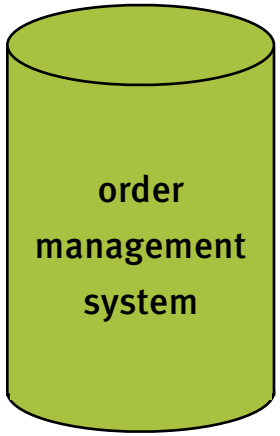
provide ideas

good ones evolve

changes are incorporated at eaipatterns.com

Real Life Scenario

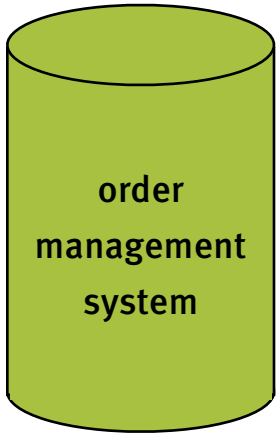
Simple order management system (CRUD)



Real Life Scenario

Simple order management system (CRUD)

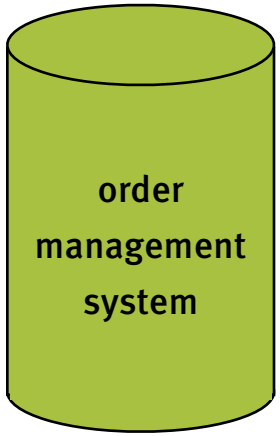
- basic Java EE



Real Life Scenario

Simple order management system (CRUD)

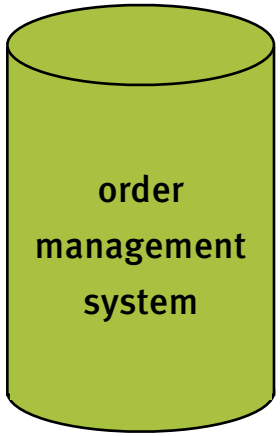
- basic Java EE
- Just CRUD operations



Real Life Scenario

Simple order management system (CRUD)

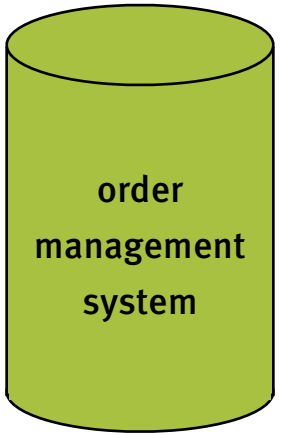
- basic Java EE
- Just CRUD operations
- “information silo”



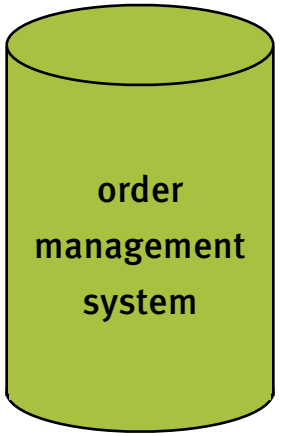
Real Life Scenario

Simple order management system (CRUD)

- basic Java EE
- Just CRUD operations
- “information silo”

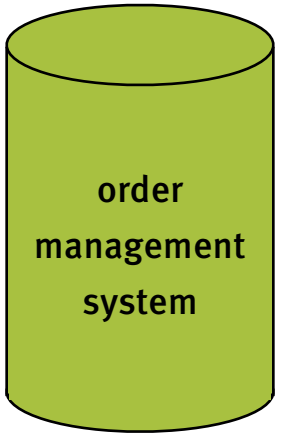


Simple Change



Simple Change

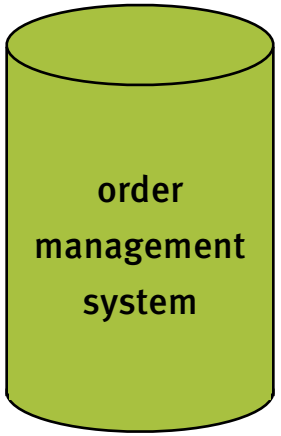
Enhance the system with an importer!



Simple Change

Enhance the system with an importer!

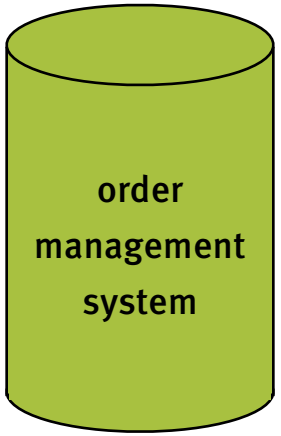
- Receive CSV-data via HTTP POST



Simple Change

Enhance the system with an importer!

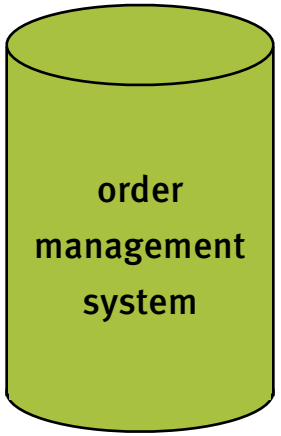
- Receive CSV-data via HTTP POST
- CSV contains header and detail records



Simple Change

Enhance the system with an importer!

- Receive CSV-data via HTTP POST
- CSV contains header and detail records
- Import data into database



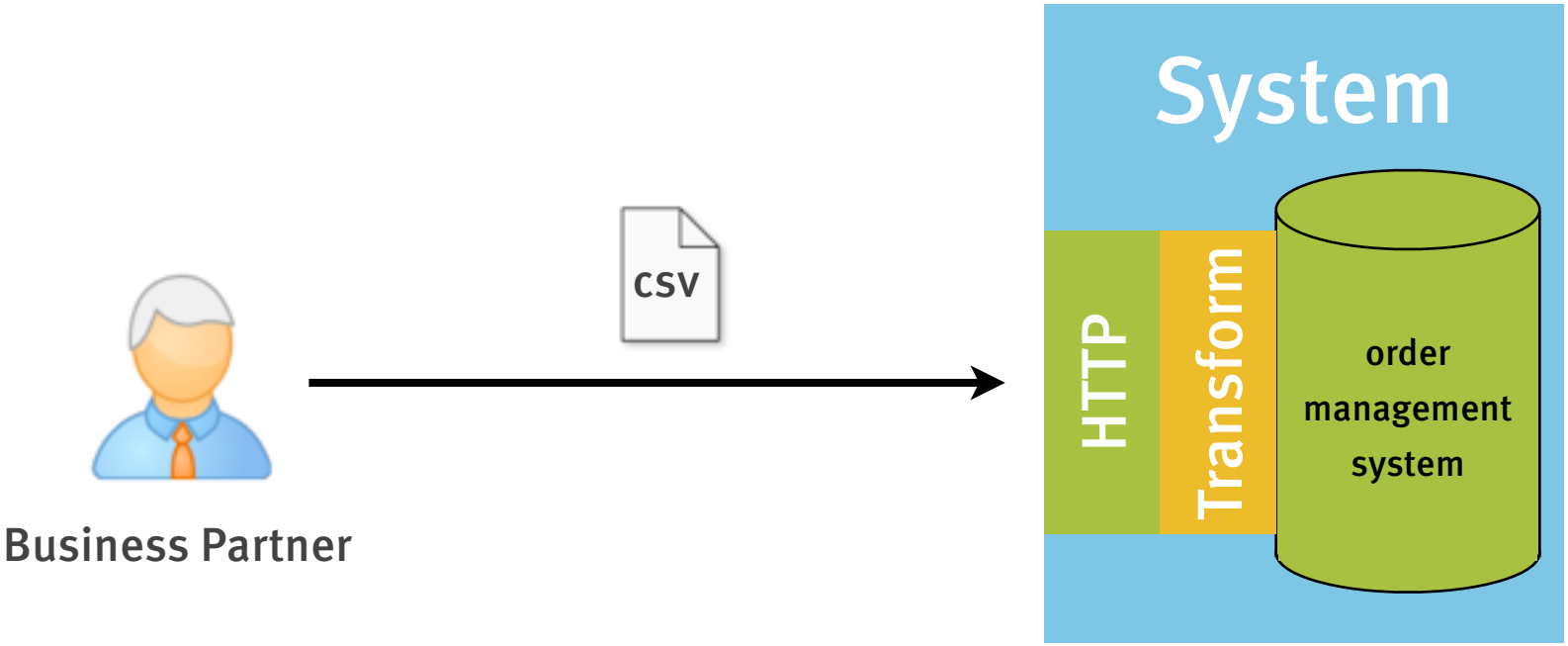
innoQ

A Simple Java EE sample

```
@WebServlet(urlPatterns = {"/order/import"})
public class ImporterServlet extends HttpServlet {

    @Inject OrderBean bean;
    @Inject CsvDataParser parser;

    @Override
    protected void doPost(HttpServletRequest request,
            HttpServletResponse response)
            throws ServletException, IOException {
        String data = request.getParameter("csvdata");
        List<Order> orders = parser.transform(data);
        for (Order order : orders) {
            bean.persist(order);
        }
    }
}
```

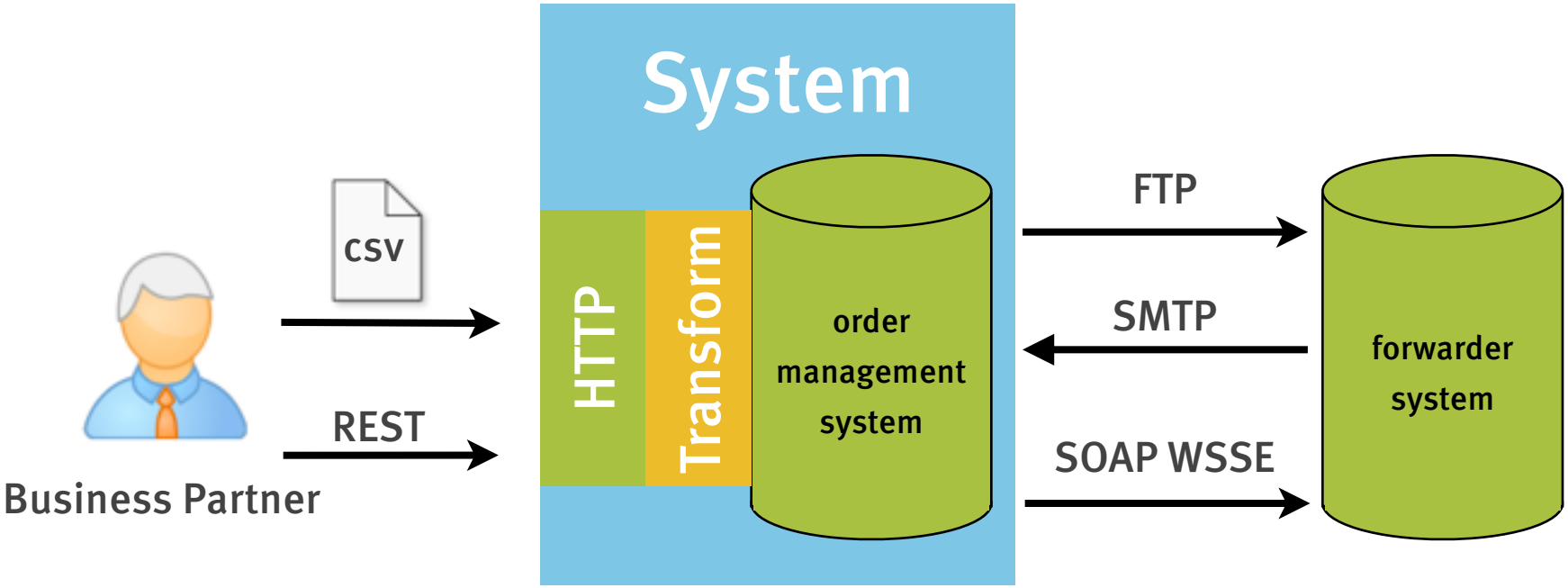
Simple Change done

Enhanced the system with an importer!



Change happens

Integrate the system of a forwarder



Recognize an integration task

when it‘s staring in your face?


http://www.flickr.com/photos/wespionage/184793114
EASY!

Build an easy and maintainable solution

for multiple integration challenges?


http://www.flickr.com/photos/mroach/3922903520/
NO PROBLEM!

Something wrong here?

```
@WebServlet(urlPatterns = {"/order/import"})
public class ImporterServlet extends HttpServlet {

    @Inject OrderBean bean;
    @Inject CsvDataParser parser;

    @Override
    protected void doPost(HttpServletRequest request,
            HttpServletResponse response)
            throws ServletException, IOException {
        String data = request.getParameter("csvdata");
        List<Order> orders = parser.transform(data);
        for (Order order : orders) {
            bean.persist(order);
        }
    }
}
```

Something wrong here?

```
@WebServlet(urlPatterns = {"/order/import"})
public class ImporterServlet extends HttpServlet {

    @Inject OrderBean bean;
    @Inject CsvDataParser parser;

    @Override
    protected void doPost(HttpServletRequ
            HttpServletResponse response)
            throws ServletException, IOEx
        String data = request.getParameter("csvdata");
        List<Order> orders = parser.transform(data);
        for (Order order : orders) {
            bean.persist(order);
        }
    }
}
```

testable?

Something wrong here?

```
@WebServlet(urlPatterns = {"/order/import"})
public class ImporterServlet extends HttpServlet {

    @Inject OrderBean bean;
    @Inject CsvDataParser parser;

    @Override
    protected void doPost(HttpServletRequ
            HttpServletResponse response)
            throws ServletException, IOEx
        String data = request.getParameter("csvdata");
        List<Order> orders = parser.transform(data);
        for (Order order : orders) {
            bean.persist(order);
        }
    }
}
```

testable?

reusable?

Something wrong here?

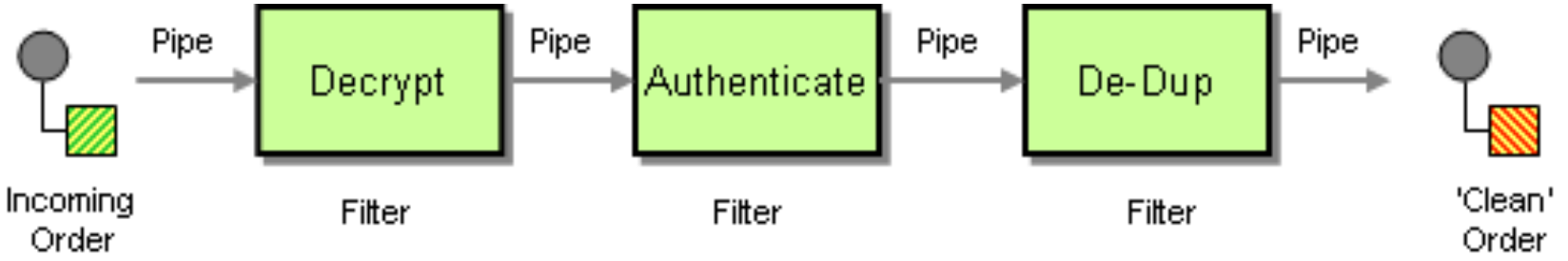
```
@WebServlet(urlPatterns = {"/order/import"})
public class ImporterServlet extends HttpServlet {

    @Inject OrderBean bean;
    @Inject CsvDataParser parser;

    @Override
    protected void doPost(HttpServletRequ
            HttpServletResponse response)
            throws ServletException, IOEx
        String data = request.getParameter("csvdata");
        List<Order> orders = parser.transform(data);
        for (Order order : orders) {
            bean.persist(order);
        }
    }
}
```

testable?

reusable?

extensible?

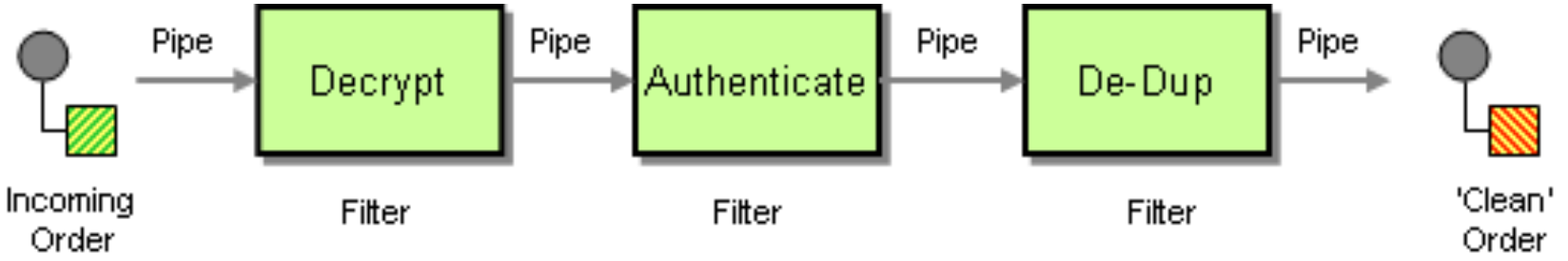
Pipes and Filters



http://www.enterpriseintegrationpatterns.com/PipesAndFilters.html

Pipes and Filters

divide your task into small steps



http://www.enterpriseintegrationpatterns.com/PipesAndFilters.html

Apply pipes & filters pattern

Apply pipes & filters pattern

Servlet

Apply pipes & filters pattern

Servlet

Custom

Apply pipes & filters pattern

Servlet Custom JPA

When you got the right tools to do the job?

innoQ

http://www.flickr.com/photos/ohskylab/4547693502

INFALLIBLE!

Is there something simpler?

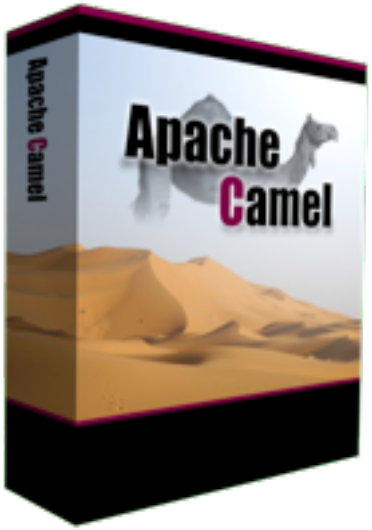
innoQ

Property of ChinBuye Limited. http://bit.ly/xBKwGw

ADAPTERS?

innoQ

EAI Frameworks



Apache Camel

Spring Integration

A simple camel route

```
from("jetty:http://localhost:9080/order/import")
        .routeId(“orderimport”)
        .process(new PostParameterProcessor(“csvdata”))
        .process(new StringRemover(new String[]{"\""}))
        .unmarshal().csv()
        .split(body(List.class))
        .aggregate(header("orderId"), new
StringAggregationStrategy()).completionTimeout(3000)
        .to("seda:singlepartsroute");
```

Spring Integration sample

```
<int-http:inbound-channel-adapter
    id="orderImportHttpAdapter"
    path="/order/import” supported-methods="POST"
    channel="ordersIn"/>

<int:channel="ordersIn"/>

<int:splitter input-channel="ordersIn" ref="orderSplitter"
   method="split" output-channel="ordersToDB" />
<bean id="orderSplitter"
   class="com.innoq.samples.eai.OrderSplitter"/>

<int:channel="ordersToDB"/>

<int-jpa:updating-outbound-gateway
  request-channel="ordersToDB"
  entity-class="com.innoq.samples.eai.domain.Order"
  entity-manager="em"/>
```

Should I always use an EAI-Framework?

innoQ

Should I always use an EAI-Framework?

It depends!

innoQ

Further tips considering EAI

Further tips considering EAI

small building blocks

Further tips considering EAI

small building blocks

no shared mutable state

innoQ

Further tips considering EAI

small building blocks

no shared mutable state

use immutable DTOs

innoQ

Further tips considering EAI

small building blocks

no shared mutable state

use immutable DTOs

=› increased scalability

Further tips considering EAI

small building blocks

no shared mutable state

use immutable DTOs

=› increased scalability

advanced: see “SEDA” & “actor model”

Thanks for your attention!

Alexander Heusingfeld, **@goldstift**

alexander.heusingfeld@innoq.com

http://www.innoq.com/de/talks

twitter #j1eai - speakerdeck

We take care of it - personally!

innoQ