ApacheCon Europe
Sinsheim, Germany 5th–8th November 2012

ApacheCon: Wicket
- where do we go from here?

Sven Meier

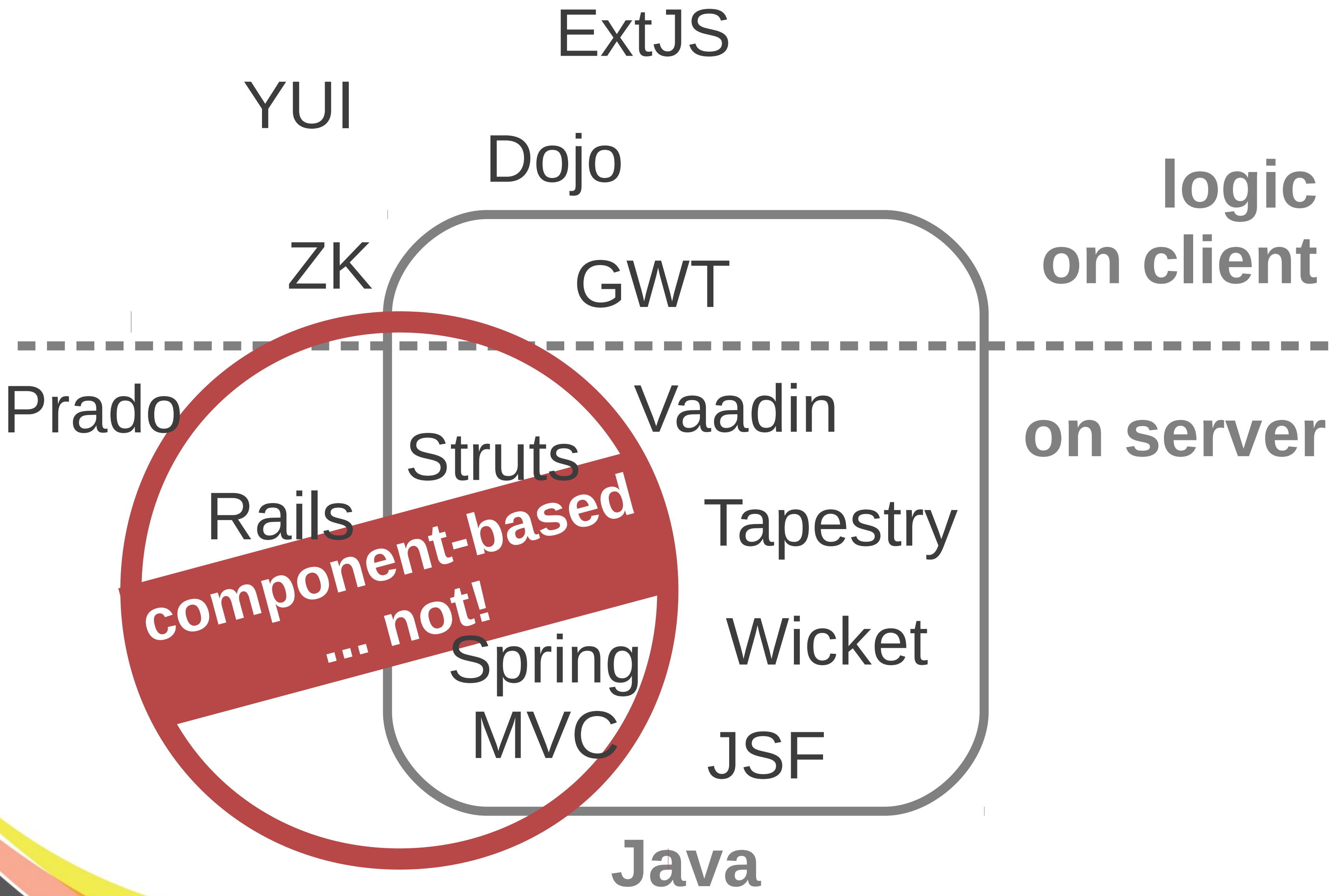
Introduction

- Sven Meier
  - freelancer and Java developer
  - Wicket user since 2005
  - Wicket committer since 2011
- Apache Wicket
  - Java web framework
  - started 2004 by Jonathan Locke
  - Apache top level project since 2007

Agenda

- Introduction ✓
- Web application
- Wicket
- ... where do we go from here?
  - Code + Markup
  - Component hierarchy
  - State
  - Integration
  - Optimization

Web applications



Wicket ...

- component oriented
- just Java and plain HTML
- Ajax built-in
- models
- bonus: Wicket 6 out!

... where do we go from here?

https://github.com/svenmeier/apachecon-wicket

Code + Markup

```
<!DOCTYPE html>
<html xmlns:wicket="http://wicket.apache.org">
  <body>
    <wicket:extend>
      <div wicket:id="feedback"></div>

      <form wicket:id="form">
        <div wicket:id="person"></div>

        <input type="submit" value="Save" wicket:id="save" />
      </form>
    </wicket:extend>
  </body>
</html>
```

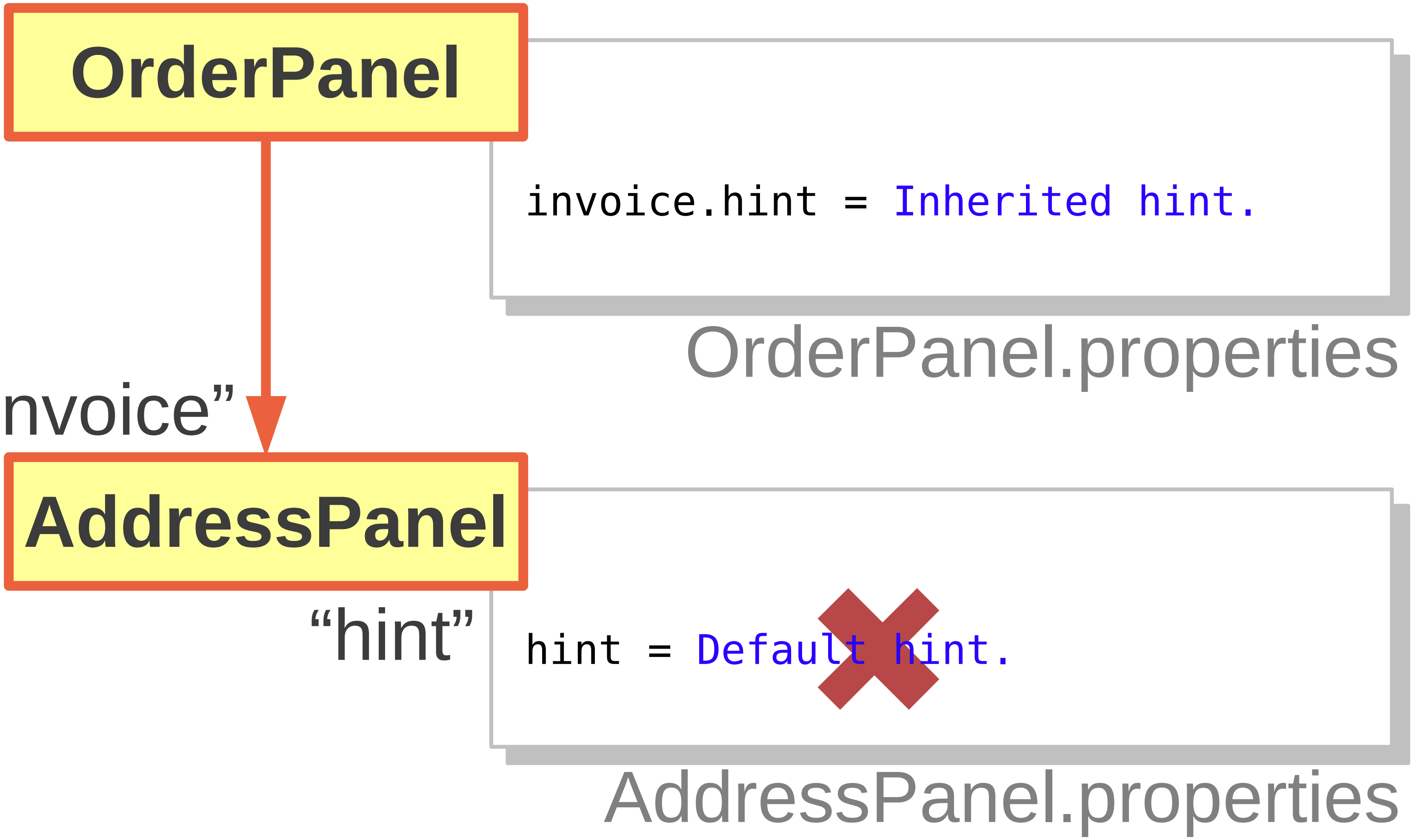
Code + Markup

- structure
  - separate UI code from other code
  - components with clear responsibility
  - create/configure/add
- names
  - type suffix for components
  - no suffix for models and component ids

Code + Markup

```
package eu.apachecon.customer.ui;

public class CustomerPage extends ApacheConPage {

  protected CustomerPage(final IModel<Person> person) {

    add(new FeedbackPanel("feedback"));

    Form<Void> form = new Form<Void>("form");
    add(form);

    form.add(new PersonPanel("person", person));

    form.add(new Button("save") {
      @Override
      public void onSubmit() {
      }
    });
  }
```

Component hierarchy

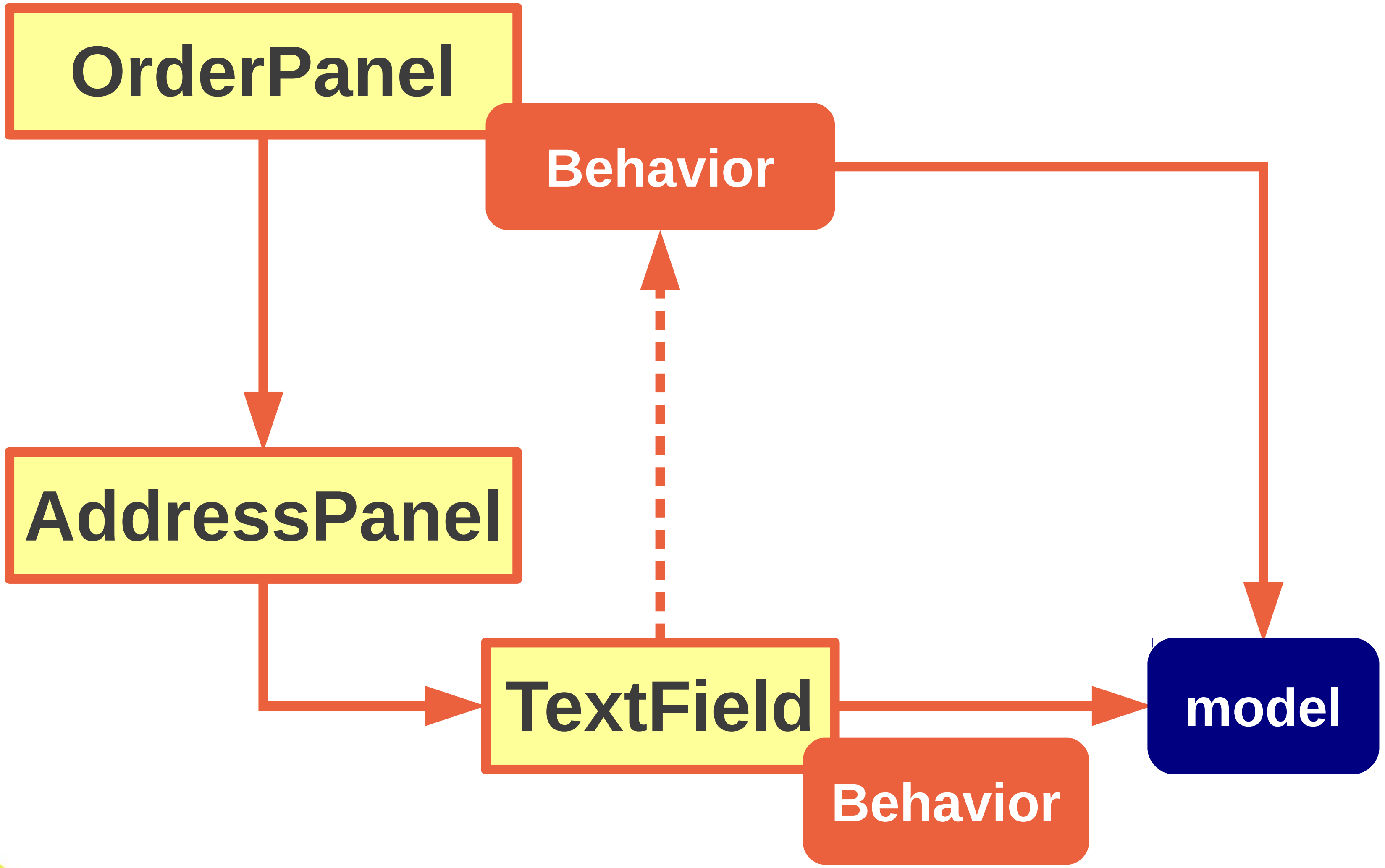


```
invoice.hint = Inherited hint.
```

```
hint = Default hint.
```

ApacheCon Europe
Sinsheim, Germany 5th–8th November 2012

Component hierarchy

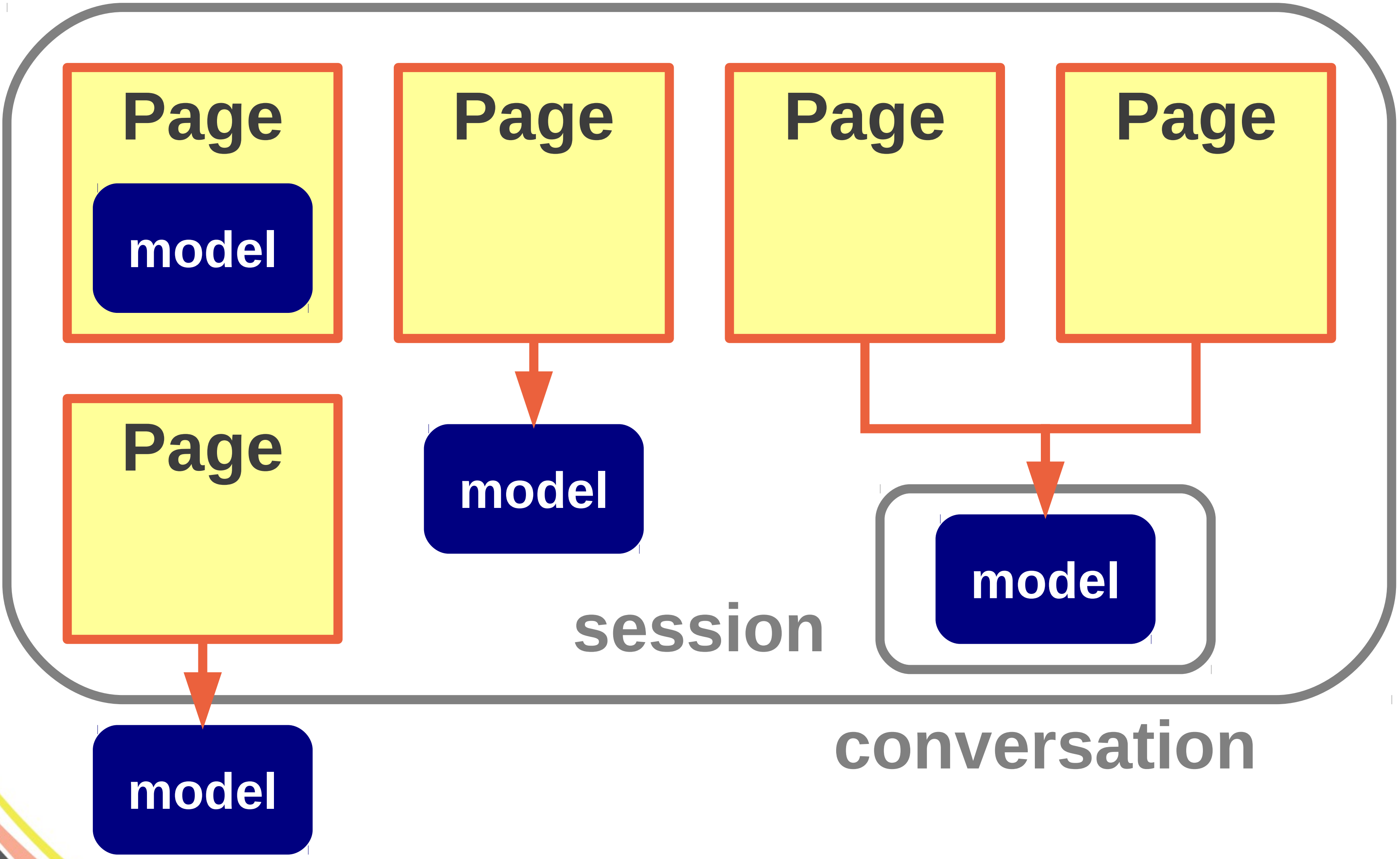


Component hierarchy

```
@Override
public void init() {
  super.init();

  InheritingBehavior.install(this);
}
```

```
final PropertyModel<String> invoicePostcode =
  new PropertyModel<String>(order, "invoice.postcode");

add(new InheritableBehavior(invoicePostcode,
  new Behavior() {
    @Override
    public void onConfigure(Component component) {
      ((FormComponent<?>) component).setRequired(true);
    }
  }
));
```

State


Page
model
Page
model
Page
Page
model
Page
model
session
conversation

Session state

```
public class CustomerSession extends WebSession {

  private Person customer;

  public Person getCustomer() {
    return this.customer;
  }

  public Person setCustomer(Person customer) {
    this.customer = customer;
  }
}
```

extensible
... not!

```
@Override
public Session newSession(Request request, Response response)
{
  return new CustomerSession(request);
}
```

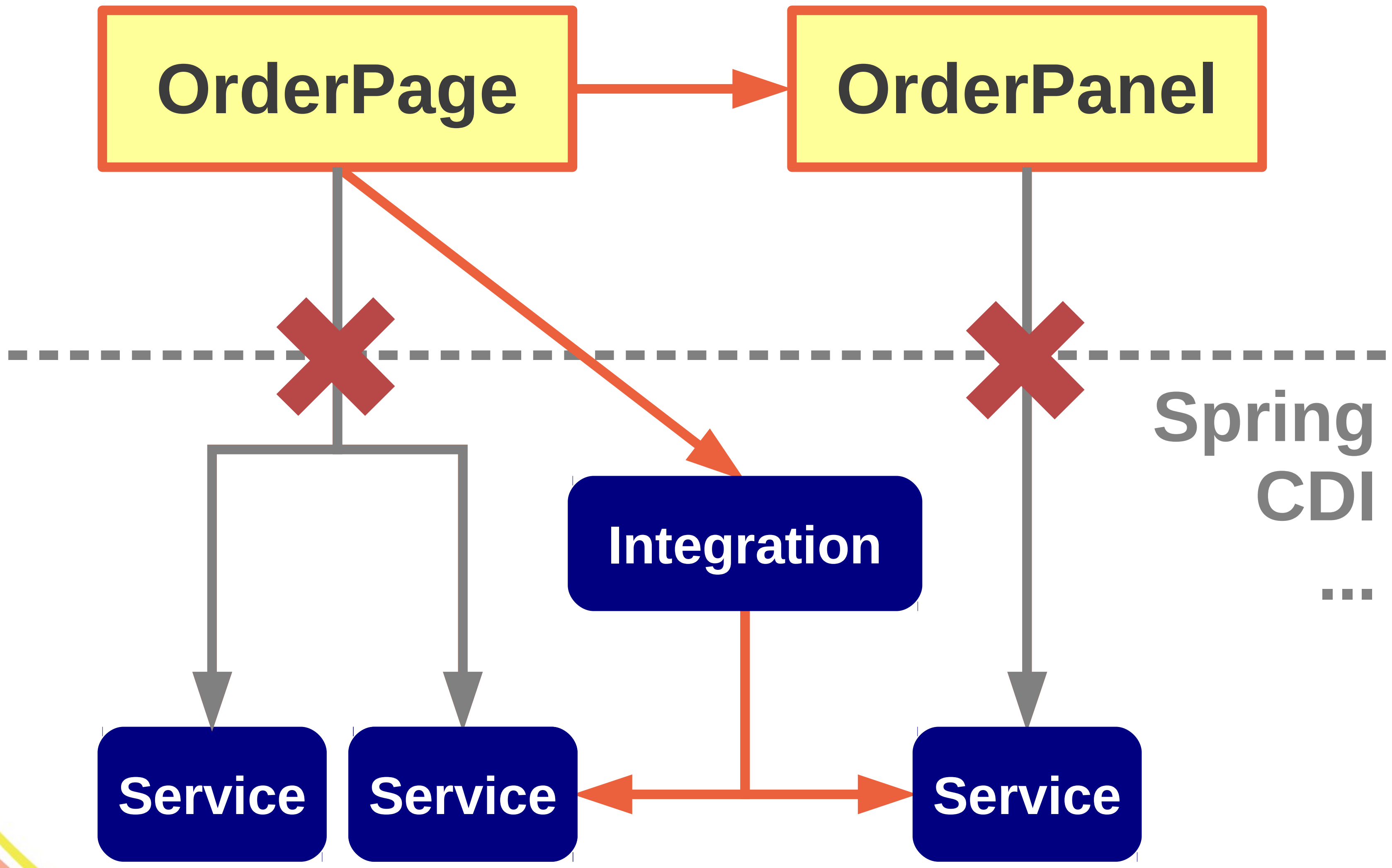
Session state

```
public class CustomerSessionModel
  extends LoadableDetachableModel<Person> {

  private static MetaDataKey<Person> KEY = new MetaDataKey(){};

  @Override
  protected Person load() {
    Person person = Session.get().getMetaData(KEY);
    if (person == null) {
      person = new Person();

      Session.get().setMetaData(KEY, person);
    }

    return person;
  }
}
```

Conversational state

```
public class OrderConversationModel
  extends LoadableDetachableModel<Order> {

  private static MetaDataKey<Order> KEY = new MetaDataKey(){};

  private Page page;

  public OrderConversationModel(Page page, Order order) {
    this.page = page;

    Conversation.create(page).setMetaData(KEY, order);
  }

  @Override
  protected Order load() {
    Conversation conversation = Conversation.get(page);

    return conversation.getMetaData(KEY);
  }
}
```

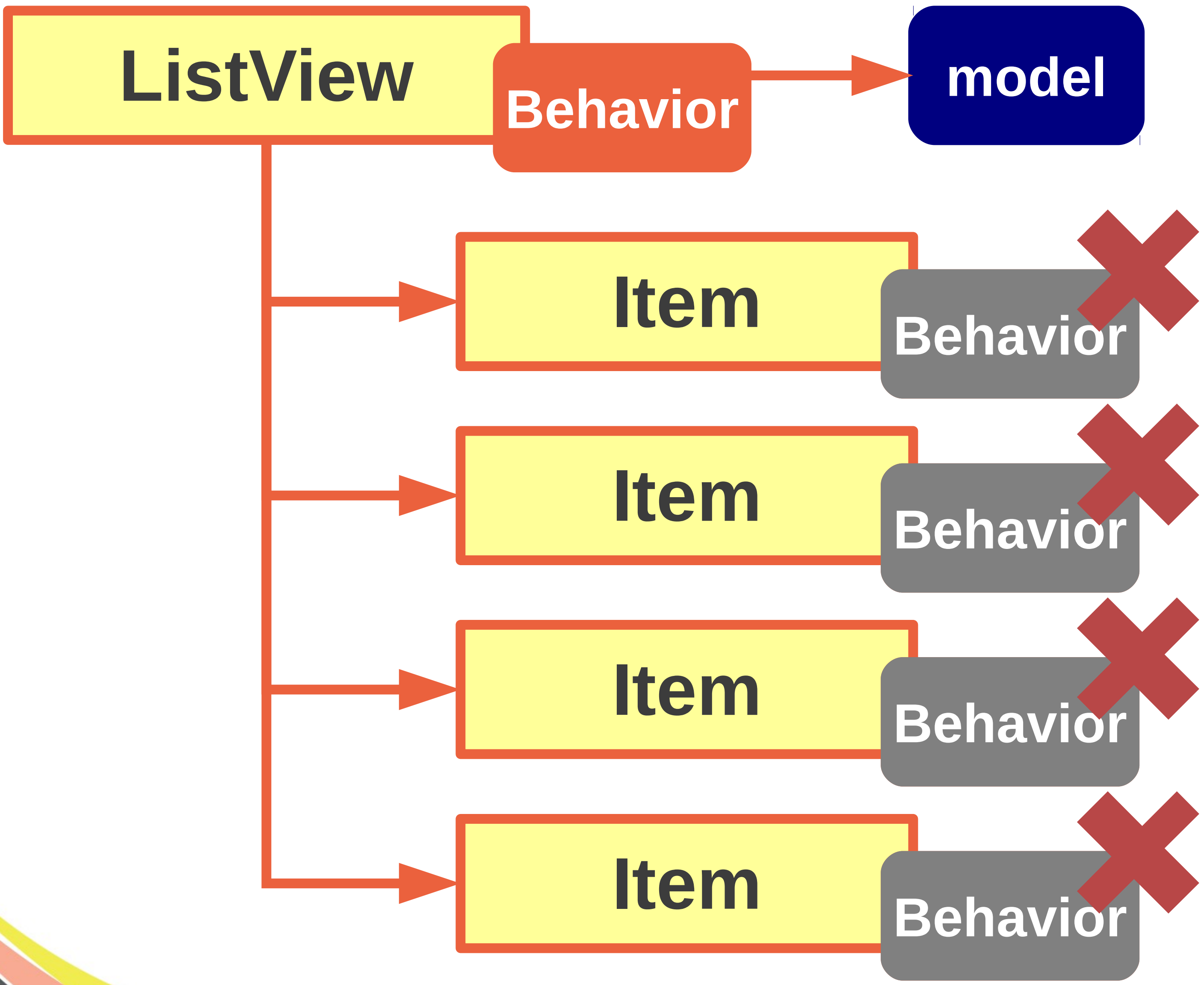
Integration



Integration

```
@SpringBean
private OrderIntegration integration;

private void init(final IModel<Order> order) {
  form.add(new OrderPanel("order", order, new PaymentsModel()));

  form.add(new Button("button") {
    @Override
    public void onSubmit() {
      integration.saveOrder(order.getObject());
    }
  });
}

private class PaymentsModel extends LoadableDetachableModel {
  @Override
  protected List<Payment> load() {
    return integration.getPayments();
  }
}
```

Optimization



Optimization

```
final OrderListModel orders = new OrderListModel();

WebMarkupContainer select = new WebMarkupContainer("select");
select.add(new IndexEventBehavior() {
  @Override
  protected void onItem(int i) {
    setResponsePage(new OrderPage(orders.getObject().get(i)));
  }
});
add(select);

select.add(new PropertyListView<Order>("orders", orders) {
  @Override
  protected void populateItem(ListItem<Order> item) {
    item.add(new Label("person.title"));
    item.add(new Label("person.surname"));
    item.add(new Label("person.name"));
    item.add(new Label("amount"));
  }
}
```

ApacheCon Europe
Sinsheim, Germany 5th-8th November 2012

Questions?

- now
- here at ApacheCon
- on mailing list
  - user@wicket.apache.org
  - dev@wicket.apache.org
- on your project
  - svenmeier@apache.org