

# eXTra-Client Open-Source: Anforderungen, Architektur und Implementierung

Köln, November 2012

# Agenda

- Anforderungen an die OSS-eXTra-Client-Implementierung
- Architektur
- Implementierung
- Demo
- Open-Source Status
- Ausblick
- Q&A

# Agenda

- **Anforderungen an die OSS-eXTra-Client-Implementierung**
- Architektur
- Implementierung
- Demo
- Open-Source Status
- Ausblick
- Q&A

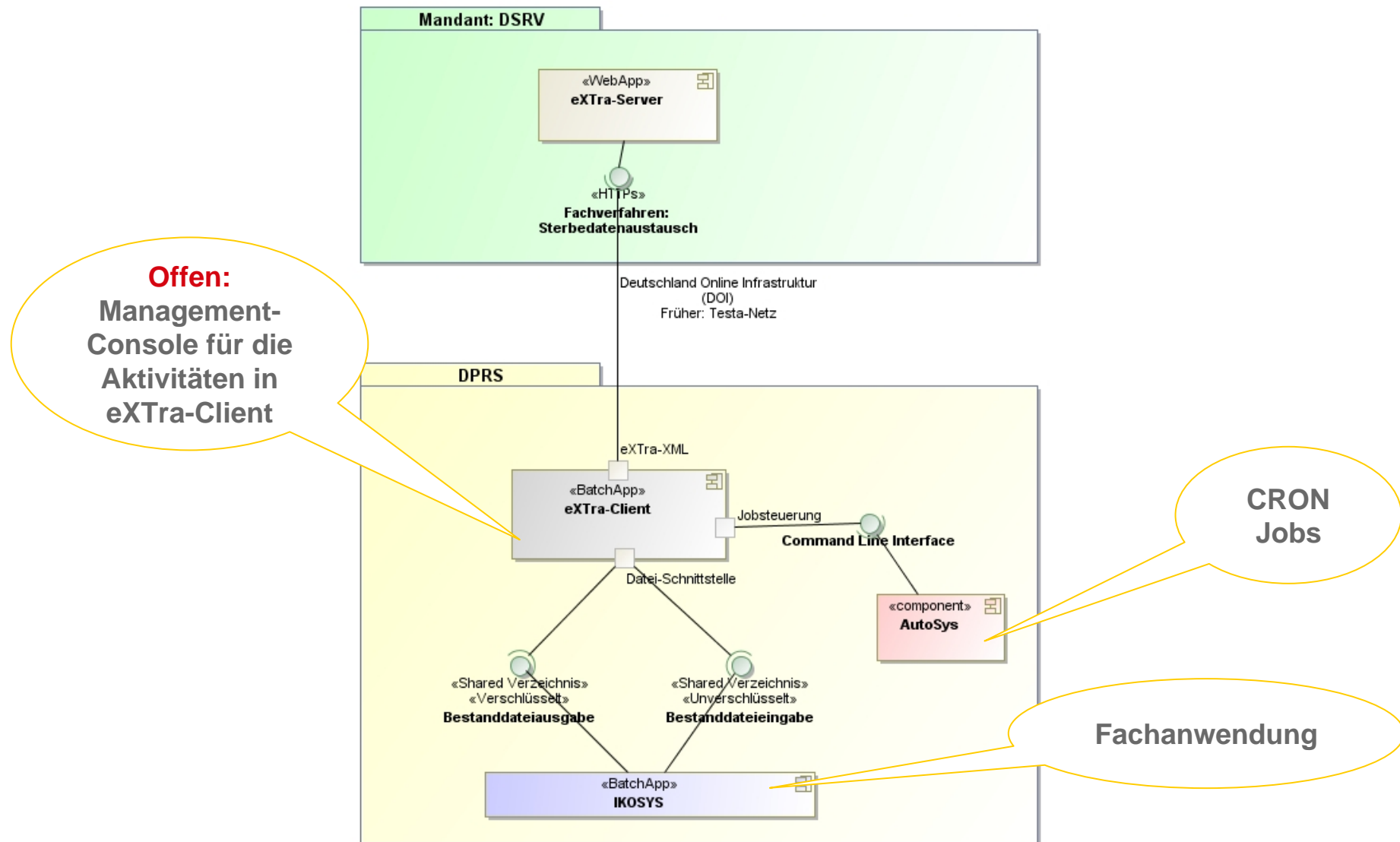
## Anforderungen eXTra-Client

- Referenzimplementierung für eXTra 1.3
- Produktionsumgebung tauglich ⇒ Analog zu Apache Tomcat für Referenzimplementierung eines Servlet-Containers
- Flexibel und generisch
  - Unterstützung unterschiedlicher Mandanten und Fachverfahren
  - Minimalem Implementierungsaufwand (ausschließlich konfigurative Tätigkeiten) bei neuen Mandanten und Fachverfahren  
⇒ Analog zu FTP-Client
- Sichere, effiziente und nachvollziehbare Datenübermittlung
  - RequestID und ResponseID sollten konsequent gespeichert werden, um Vorgänge nachvollziehen zu können.
- Individuelle Anpassung an System-Umgebung
  - Unterschiedliche Betriebssysteme
  - Unterschiedliche Datenbanken

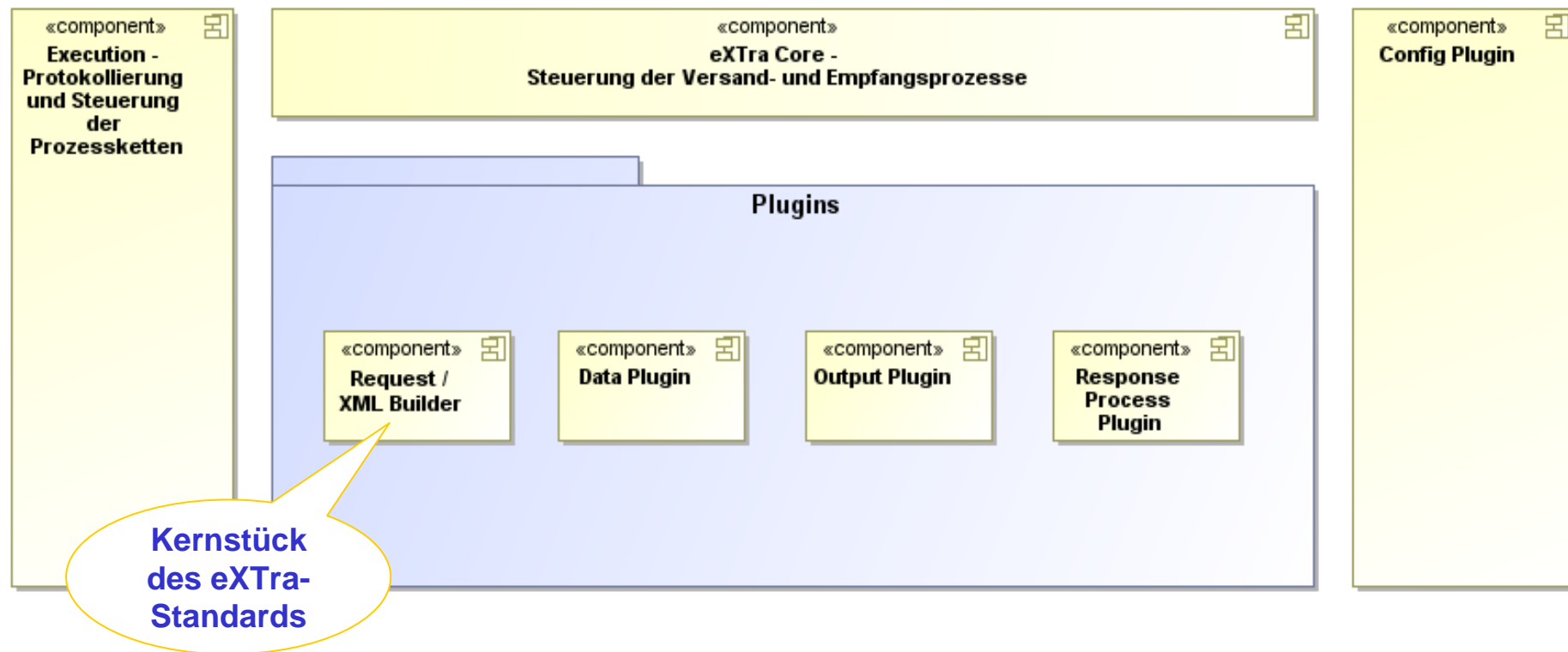
# Agenda

- Anforderungen an die OSS-eXTra-Client-Implementierung
- **Architektur**
- Implementierung
- Demo
- Open-Source Status
- Ausblick
- Q&A

# Architektur: Kontextsicht



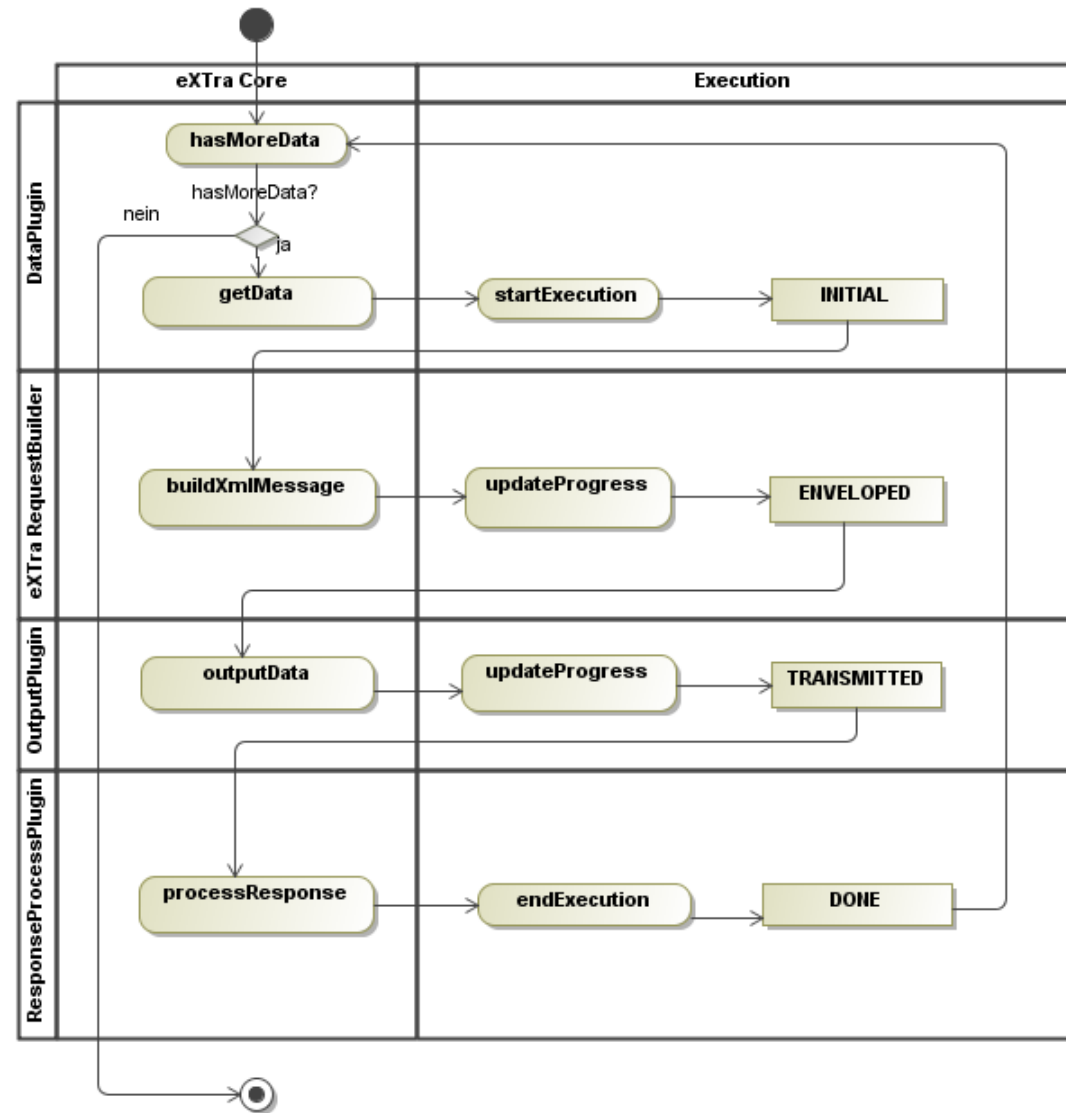
## Architektur: Bausteinsicht



**Plugins in der Anwendung eXtra-Client != Plugins im eXtra-Briefumschlag (Standard)**

# Architektur: Laufzeitsicht

- Inputdaten entgegen nehmen
- Inputdaten in eXTra-Umschlag (XML) verpacken
- eXTra-Umschlag an einen eXTra-Server versenden
- Response des eXTra-Servers empfangen und verarbeiten

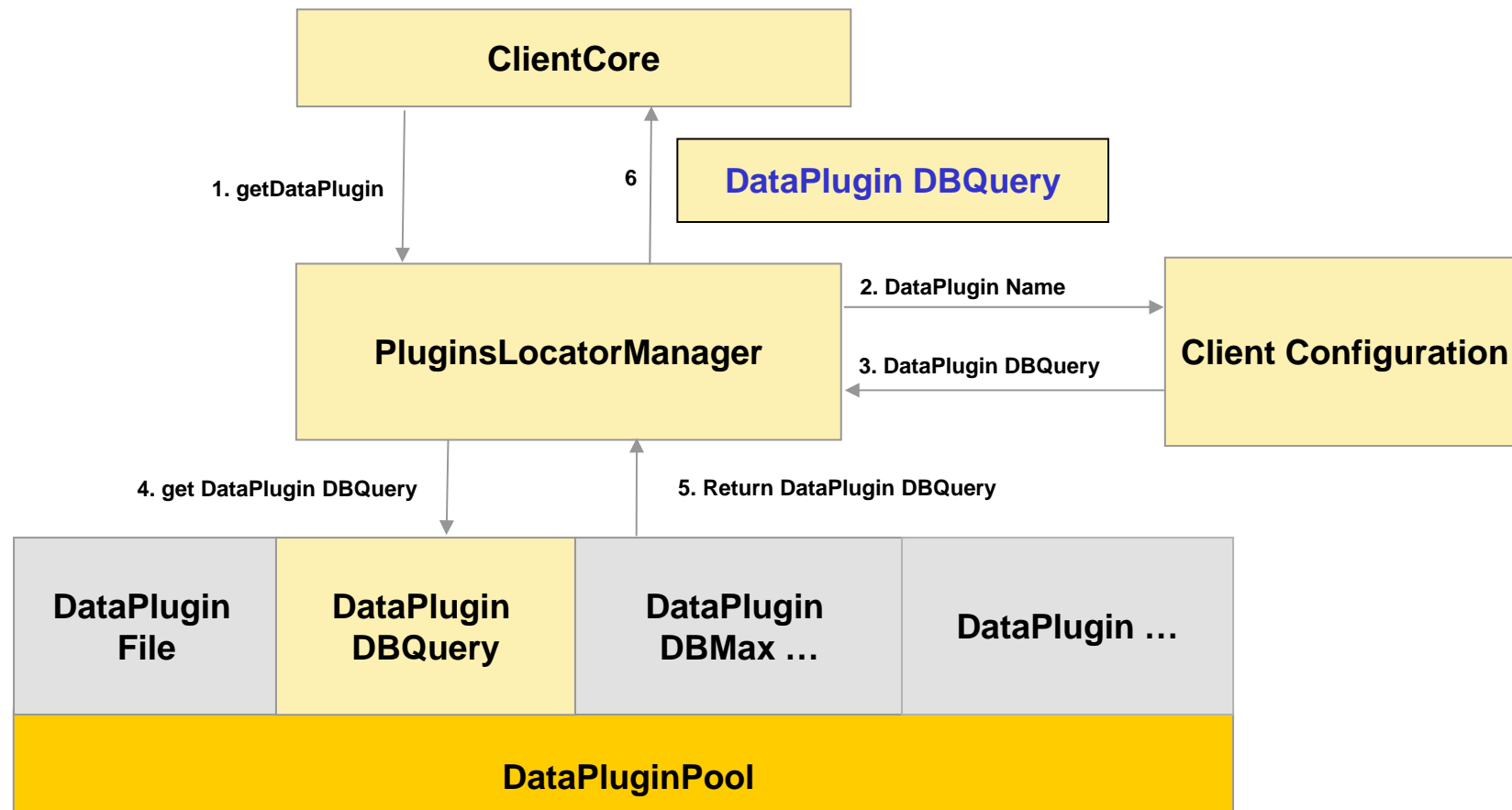




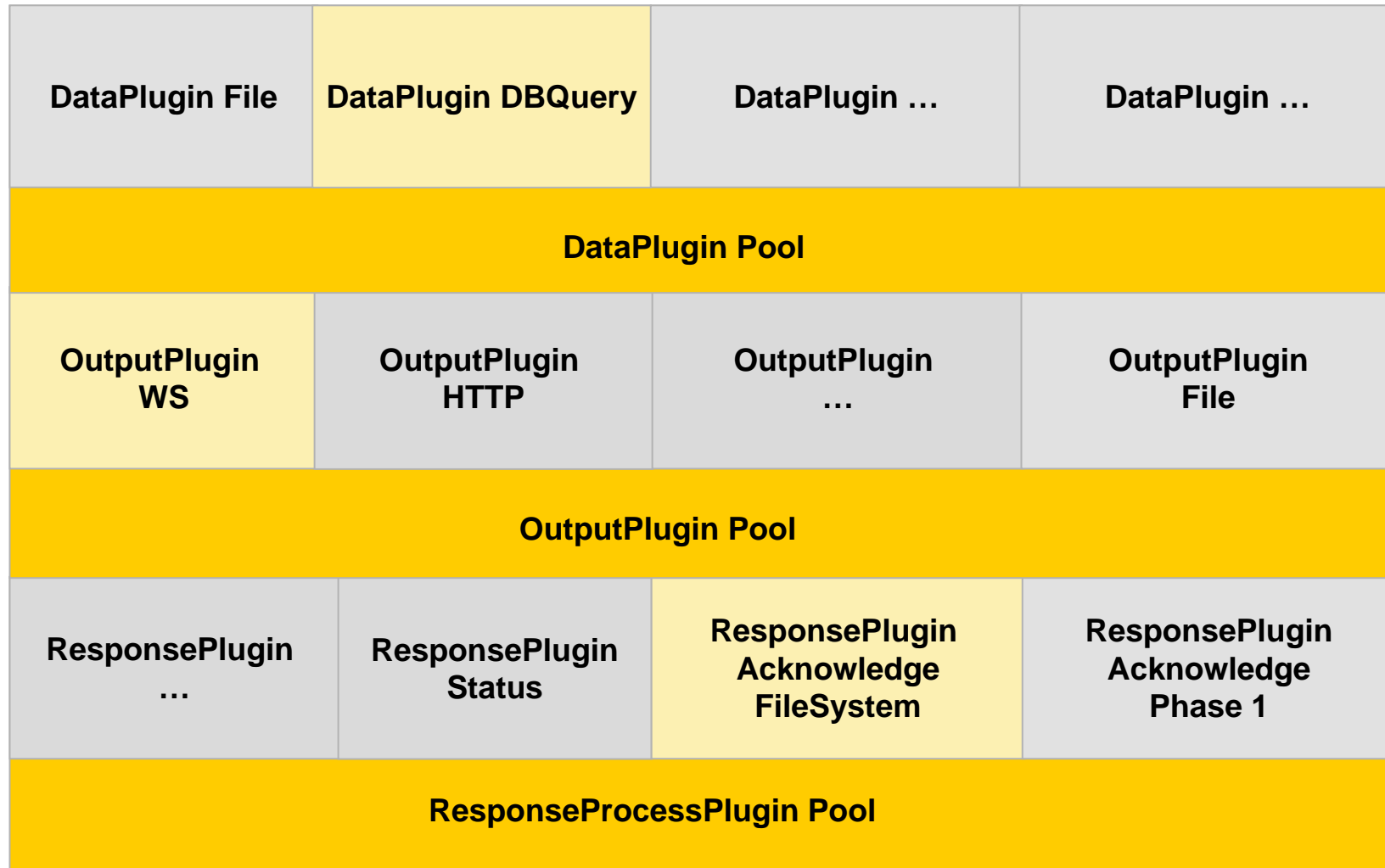
# Agenda

- Anforderungen an die OSS-eXTra-Client-Implementierung
- Architektur
- **Implementierung**
- Demo
- Open-Source Status
- Ausblick
- Q&A

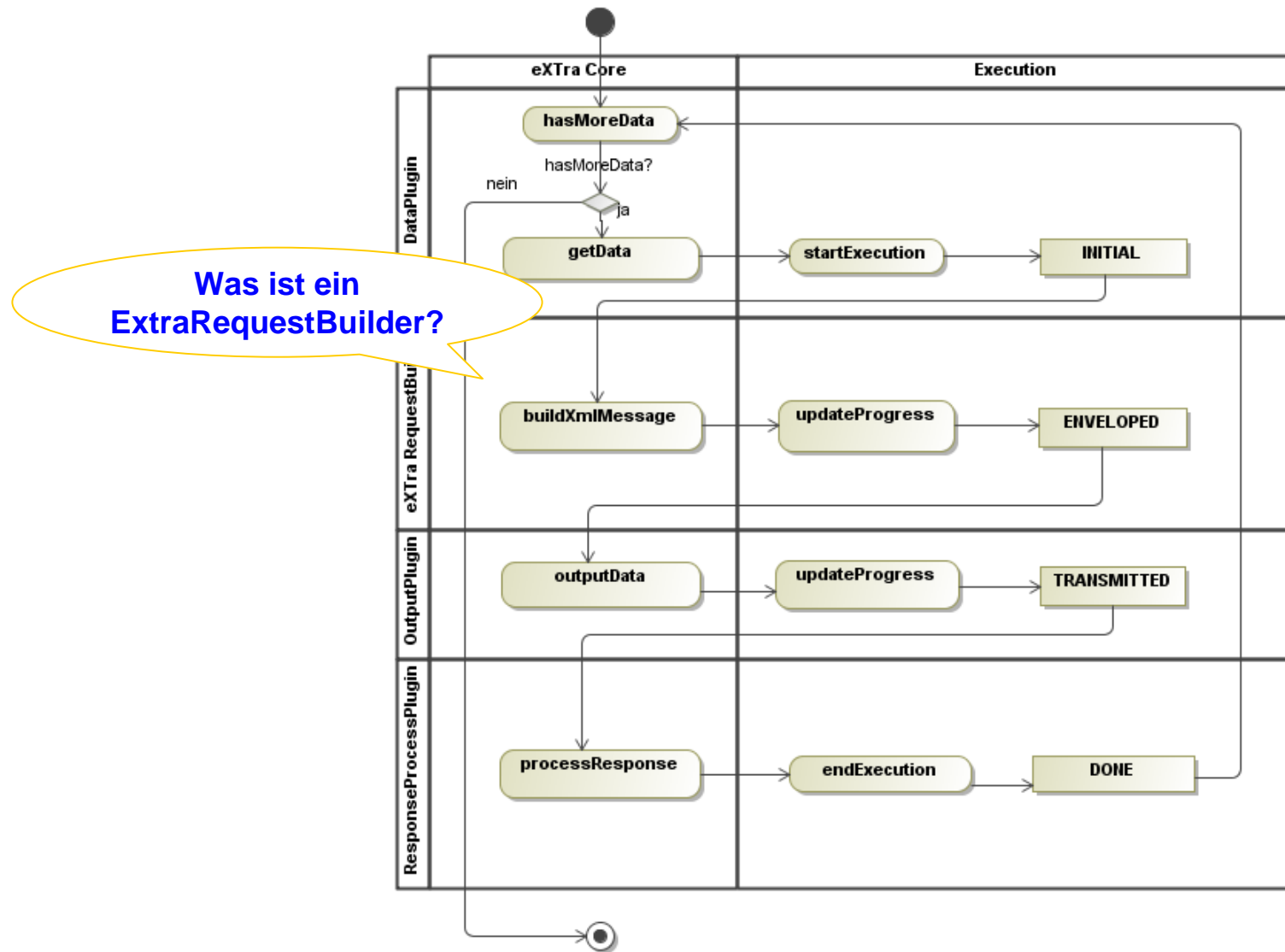
# Implementierung: Wie wird ein Plugin instanziiert?



## Implementierung: Baukasten-Prinzip

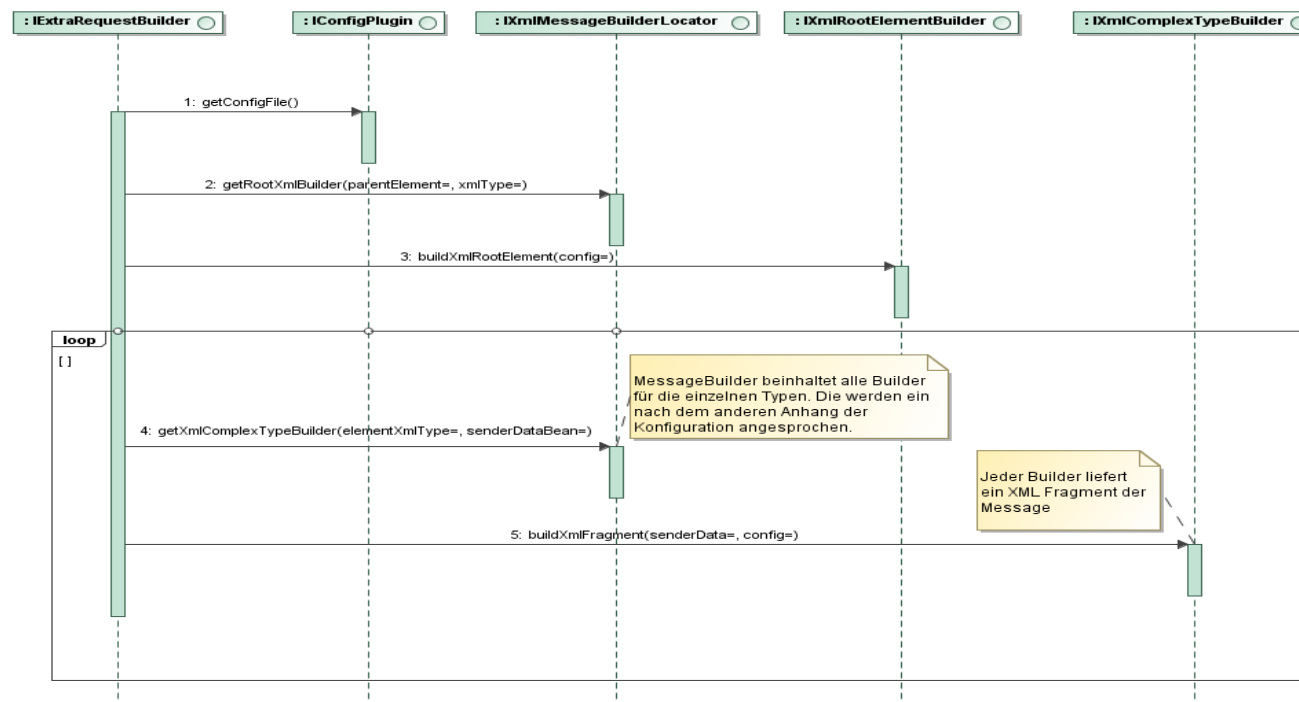


# Implementierung: ExtraRequestBuilder



## Implementierung: ExtraRequestBuilder

- ExtraRequestBuilder bereitet mit Hilfe von **XML-Buildern** einen eXTra-XML-Umschlag vor
- eXTra-XML wird aus einer Profildatei, Konfigurationsdatei und DataPlugin erstellt.
- Die XML-Fragments in einem eXTra-XML-Umschlag werden durch wiederverwendbare und konfigurierbare Komponenten (**XML-Builder**) erstellt.
- Baukasten-Prinzip für **XML-Builder**: Analog zur Instanziierung der Plugins



# Implementierung: ExtraRequestBuilder - Instanziierung

Beispiel:

Profil-Datei

```
<element>
  <Name>xcpt:Sender</Name>
  <Elternelement>TransportHeader</Elternelement>
</element>
```

Ergebnis der  
Profilierung

Properties-Datei

```
message.builder.header.senderId.class=
message.builder.header.senderId.value=SENDER-ID
message.builder.header.senderNameValue=SENDER NAME
```

Konfiguration

Erzeugter XML-Abschnitt

```
<xcpt:Sender>
  <xcpt:SenderID>SENDER-ID</xcpt:SenderID>
  <xcpt:Name>SENDER NAME</xcpt:Name>
</xcpt:Sender>
```

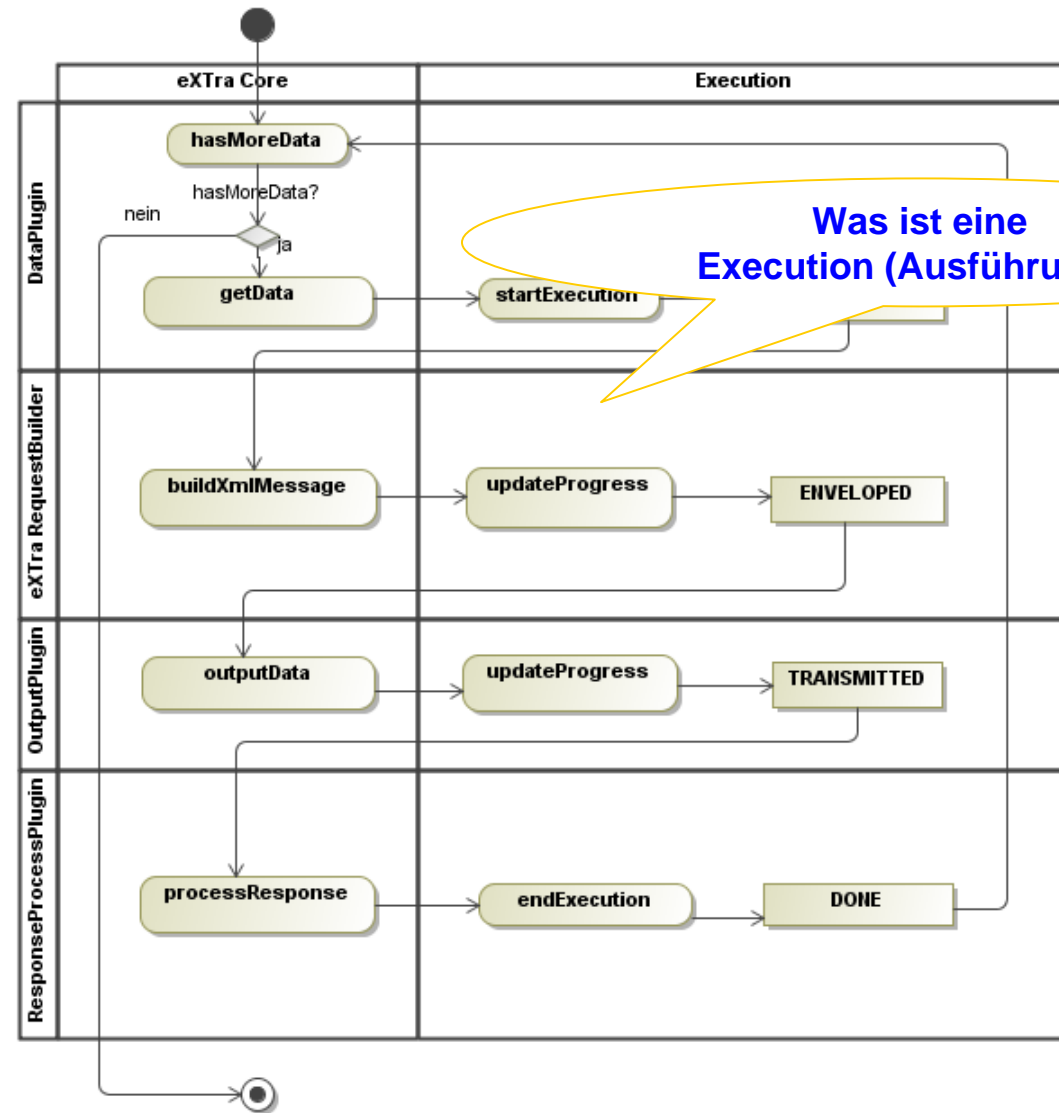
Ergebnis

# Implementierung: ExtraRequestBuilder - XML-Builder-Klassen

## Übersicht über alle XML-Builder

Name	XML-Element	Klasse
configurableSMTPContactsPluginsBuilder	xplg:Contacts	de.extra.client.core.builder.impl.plugins.ConfigurableSMTPContactsPluginsBuilder
dataSourceConfigurablePluginsBuilder	xplg:DataSource	de.extra.client.core.builder.impl.plugins.DataSourceConfigurablePluginsBuilder
dataSourceInputDataPluginsBuilder	xplg:DataSource	de.extra.client.core.builder.impl.plugins.DataSourceInputDataPluginsBuilder
dataSourcePluginsBuilder	xplg:DataSource	de.extra.client.core.builder.impl.plugins.DataSourcePluginsBuilder
dataTransformConfigurablePluginsBuilder	xplg:DataTransforms	de.extra.client.core.builder.impl.plugins.DataTransformConfigurablePluginsBuilder
dataTransformPluginsBuilder	xplg:DataTransforms	de.extra.client.core.builder.impl.plugins.DataTransformPluginsBuilder
requestTransportBodyBuilder	Siehe eXTra-OSS-Wiki!	
requestTransportBuilder		
requestTransportHeaderBuilder		
transportBodyCharSequenceBuilder		
transportBodyDataBuilder	xcpt:Data	de.extra.client.core.builder.impl.components.TransportBodyDataBuilder
transportBodyFileInputBase64CharSequenceBuilder	xcpt:Base64CharSequence	de.extra.client.core.builder.impl.components.TransportBodyFileInputBase64CharSequenceBuilder
transportBodyRequestConfirmationOfReceiptSequenceBuilder	xcpt:ElementSequence	de.extra.client.core.builder.impl.components.TransportBodyRequestConfirmationOfReceiptSequenceBuilder
transportBodyRequestQueryElementSequenceBuilder	xcpt:ElementSequence	de.extra.client.core.builder.impl.components.TransportBodyRequestQueryElementSequenceBuilder
transportHeaderReceiverBuilder	xcpt:Receiver	de.extra.client.core.builder.impl.components.TransportHeaderReceiverBuilder
transportHeaderRequestDetailsBuilder	xcpt:RequestDetails	de.extra.client.core.builder.impl.components.TransportHeaderRequestDetailsBuilder
transportHeaderSenderBuilder	xcpt:Sender	de.extra.client.core.builder.impl.components.TransportHeaderSenderBuilder
transportHeaderTestIndicatorBuilder	xcpt:TestIndicator	de.extra.client.core.builder.impl.components.TransportHeaderTestIndicatorBuilder
transportPluginsBuilder	req:TransportPlugins	de.extra.client.core.builder.impl.plugins.TransportPluginsBuilder

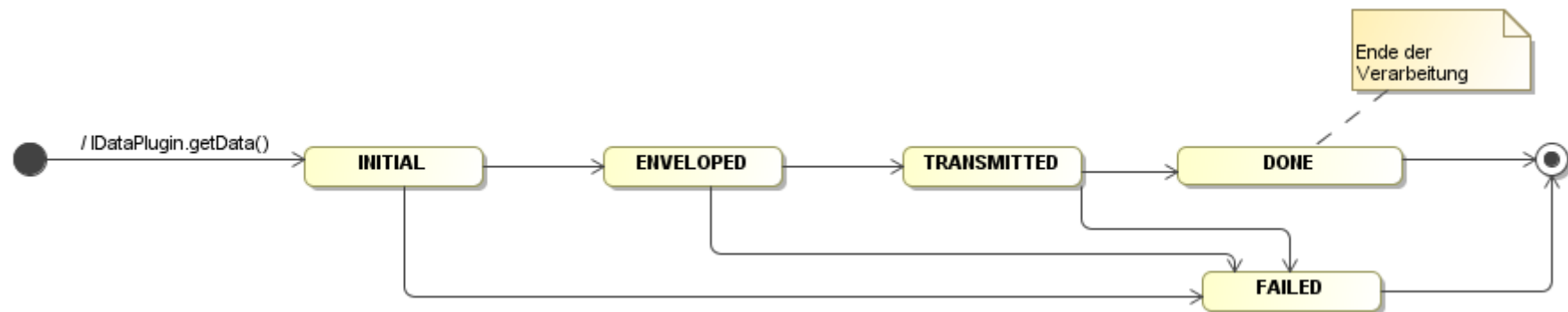
# Implementierung: Execution (Ausführung)





## Implementierung: Execution (Ausführung)

- Nachvollziehbarkeit für jeden eXTra-XML-Umschlag:
  - Stati von Request- und Response-Nachrichten
  - Protokollierung der fachlichen und technischen Fehler



## Implementierung: Domänen-Modell

- Domänen-Modell wird in der Datenbank persistiert.
- Konfiguration für die Mandantenfähigkeit und mehrstufiges Fachverfahren
- Gesamtes Kommunikationsprotokoll



## Implementierung: Verwendete Technologien

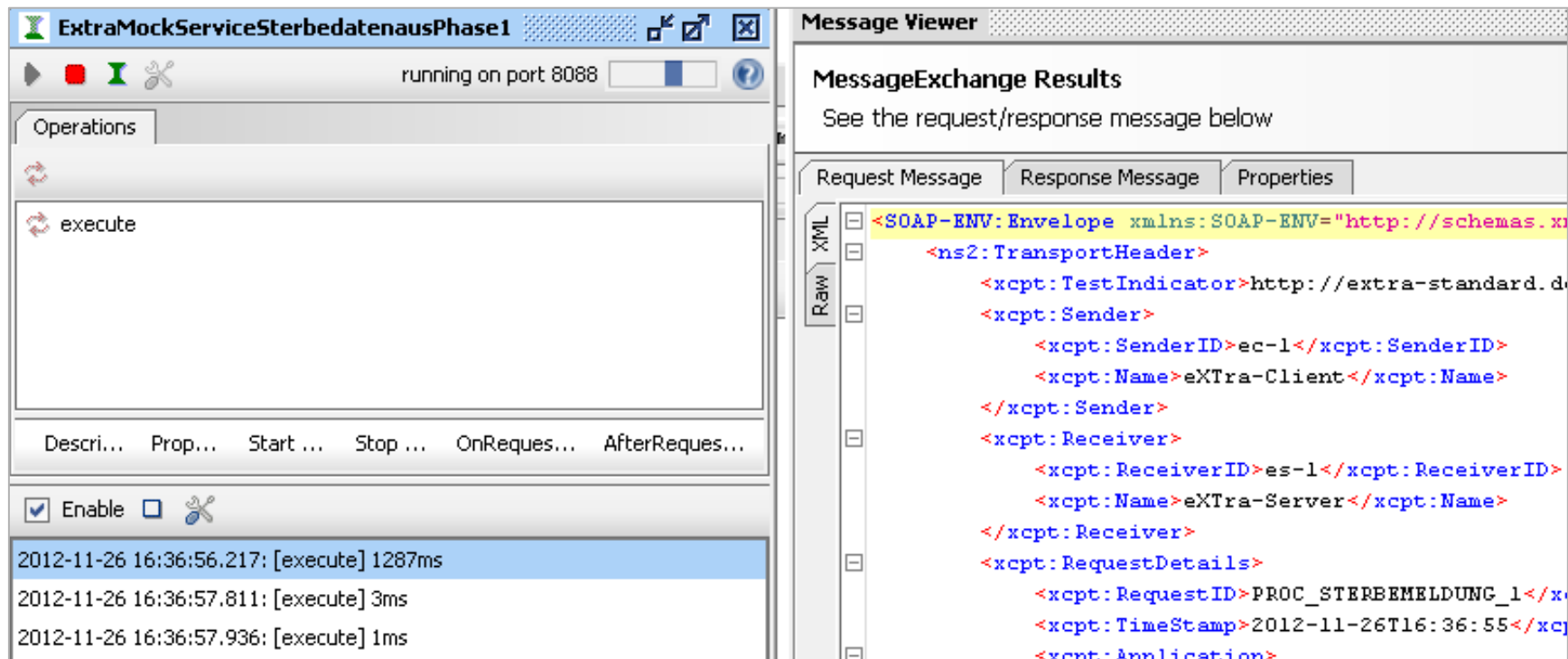
- Maven 3.x als Buildsystem
- Flyway 2.x als Datenbank-Migration-Tool (Versionierung der DB)
- Hibernate 4.x als JPA-Provider (Java Persistence API)  
⇒ Datenbankunabhängig
- SpringFramework 3.x:
  - Spring Web Services: Webservice
  - Spring Data JPA: JPA Repository Zugriff
  - Spring AspectJ: Domain Driven Design
- JAXB 2.x für Transformation XSD ⇒ Java-Objekte

Domain Driven Design: [http://de.wikipedia.org/wiki/Domain-Driven\\_Design](http://de.wikipedia.org/wiki/Domain-Driven_Design)

# Agenda

- Anforderungen an die OSS-eXTra-Client-Implementierung
- Architektur
- Implementierung
- **Demo**
- Open-Source Status
- Ausblick
- Q&A

# Demo



The screenshot displays the 'ExtraMockServiceSterbedatenausPhase1' application window, which is running on port 8088. The interface includes a toolbar with icons for play, stop, and other controls. Below the toolbar, there is a section for 'Operations' with a list of actions, including 'execute'. A status bar at the bottom shows the application is 'Enable' and provides a list of recent log entries with timestamps and durations.

**Message Viewer**

**MessageExchange Results**  
See the request/response message below

Request Message | Response Message | Properties

Raw XML

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <ns2:TransportHeader>
    <xcpt:TestIndicator>http://extra-standard.de/</xcpt:TestIndicator>
    <xcpt:Sender>
      <xcpt:SenderID>ec-1</xcpt:SenderID>
      <xcpt:Name>eXTra-Client</xcpt:Name>
    </xcpt:Sender>
    <xcpt:Receiver>
      <xcpt:ReceiverID>es-1</xcpt:ReceiverID>
      <xcpt:Name>eXTra-Server</xcpt:Name>
    </xcpt:Receiver>
    <xcpt:RequestDetails>
      <xcpt:RequestID>PROC_STERBEMELDUNG_1</xcpt:RequestID>
      <xcpt:TimeStamp>2012-11-26T16:36:55</xcpt:TimeStamp>
      <xcpt:Application>
```

# Agenda

- Anforderungen an die OSS-eXTra-Client-Implementierung
- Architektur
- Implementierung
- Demo
- **Open-Source Status**
- Ausblick
- Q&A

## Open-Source Status: Ohloh.net

- Ohloh.net: <https://www.ohloh.net/p/extra-standard>



# Agenda

- Anforderungen an die OSS-eXTra-Client-Implementierung
- Architektur
- Implementierung
- Demo
- Open-Source Status
- **Ausblick**
- **Q&A**

## Ausblick und Herausforderungen (1/2)

- ExtraRequestBuilder mit den konfigurativen XML-Buildern:  
**Aufwändig!**
  - Neue Anforderung == neuer XML-Builder
  
- Unterstützung anderer Datenbanken
  - Technisch möglich durch JPA, jedoch **noch nicht explizit ausprobiert!**
  
- Domänen-Modell zum Code  $\Rightarrow$  **Nicht synchron!**
  - **Modellgetriebene Softwareentwicklung mit UML**

## Ausblick und Herausforderungen (2/2)

- Wir haben gerade **ein Fachverfahren SA** implementiert.
  - Allgemeingültigkeit?
  - Zusätzlicher Status für Request- und Response-Nachrichten notwendig?
  - Keine Zusatzimplementierung nötig bzw. alles per Konfiguration möglich?
- **Management-Console** für eXTra-Client **Datenbank** fehlt!
- Vorgehen zur Entwicklung eines neuen Fachverfahrens **nicht optimal!**
  - Viele Schritte und Hintergrundwissen
  - Viele Konfigurationen
  - Je nachdem neue Implementierungen XML-Builder, Plugins, usw.

## Q&A



# Vielen Dank für Ihre Aufmerksamkeit.

**Leonid Potap, Michael Werner und Dr. Lofi Dewanto**

Deutsche Post AG  
NL Renten Service  
Abteilung Systementwicklung gesetzliche Rente (SEG)  
Team SEG4

Venloer Str. 151-153  
50672 Köln