

# Wie testen Sie Ihre Prozesse?

Wolfgang Strunk (Talend) und Johannes Rost (C1 WPS)

WAM Workshop 2011

# Talend is a global Leader in Integration



Venture-backed

**SILVERLAKE**



balderton capital

**idinvest**  
PARTNERS

Galileo  
PARTNERS

## Adoption rates

- 10 million downloads
- 600,000 users
- 2,500 customers

## Company facts

- Leading company for Open Source integration technology
- Roots in France and Germany
- Currently 450 employees, planned 550 until end of 2011

**San Francisco** (Los Altos)  
Corporate

**Orange County** (Irvine)  
Business  
R&D  
Tech Support

**New York** (Tarrytown)  
Business  
Tech Support

**Boston**  
Business  
Tech Support

**Paris** (Suresnes)  
Corporate  
Business  
R&D  
Tech Support

**Utrecht**  
Business

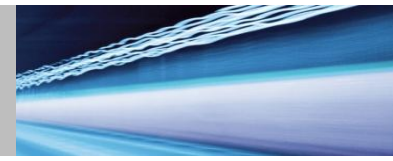
**Milan** (Curno)  
Business  
Tech Support

**London** (Maidenhead)  
Business  
Tech Support

**Bonn, Düsseldorf, Nuremberg, Munich**  
Corporate  
Business  
R&D  
Tech Support

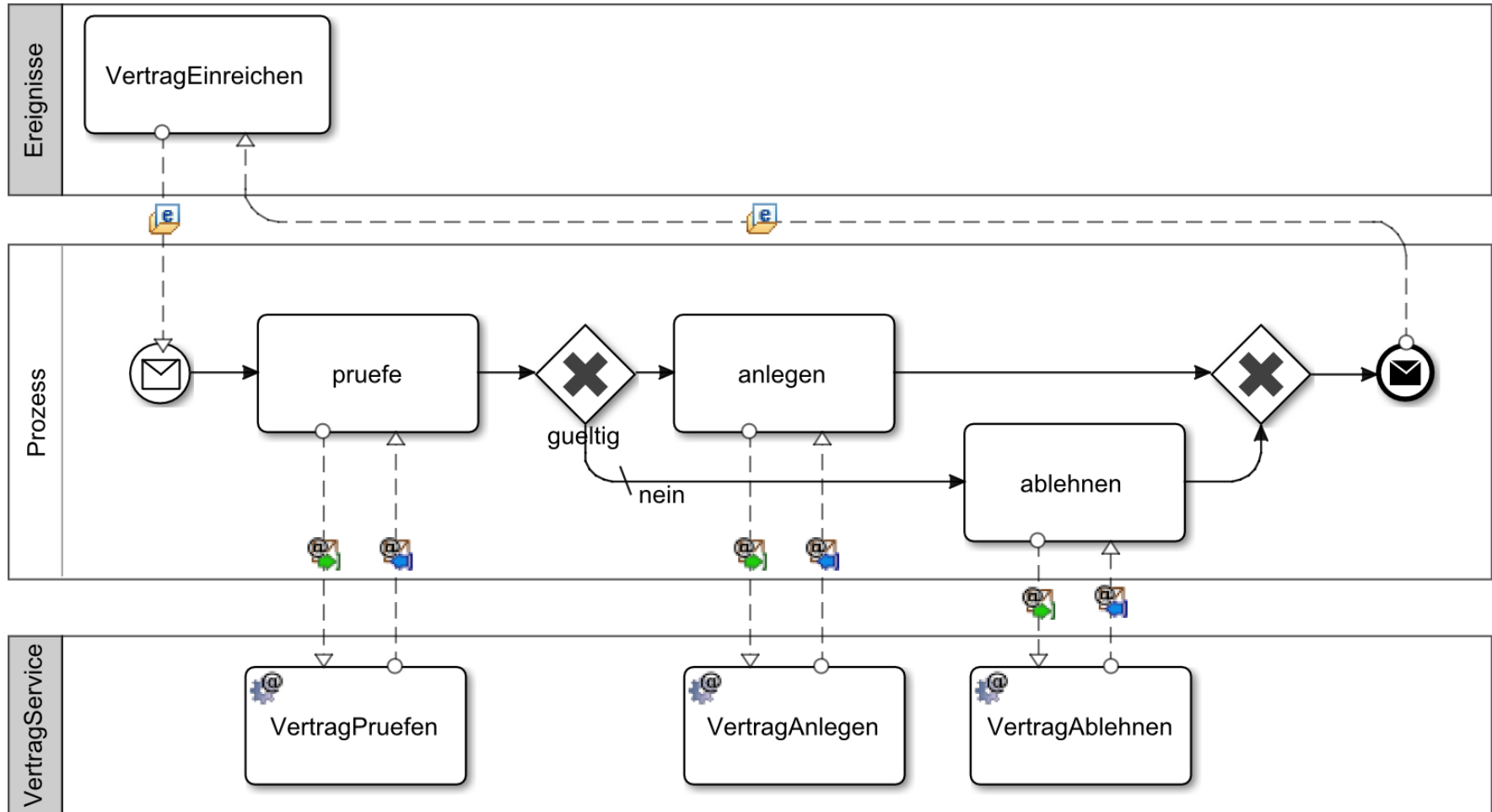
**Tokyo**  
Business  
Tech Support

**Beijing**  
R&D  
Tech Support

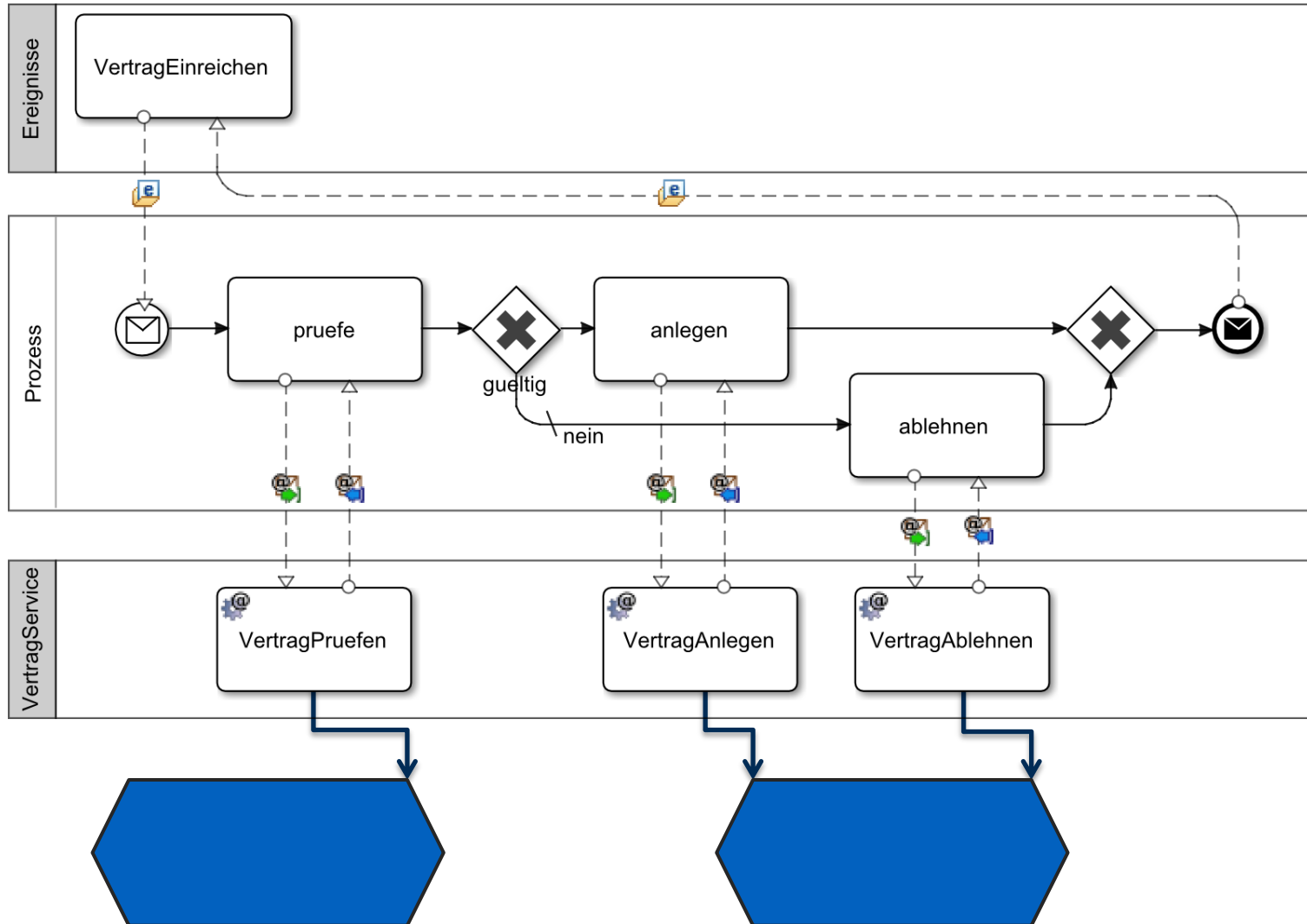


1. Möglicher Prozess auf Basis einer SOA/BPM-Architektur
2. Definition eines Prozesstests
3. Schrittweise Reduktion der Komplexität des Testfalls
4. Anforderungen an einen ESB aus Sicht von Testfällen
5. Vorstellung des Talend ESB
6. Beschreibung des Testframeworks
7. Besonderheiten in der neuen Version des ESB

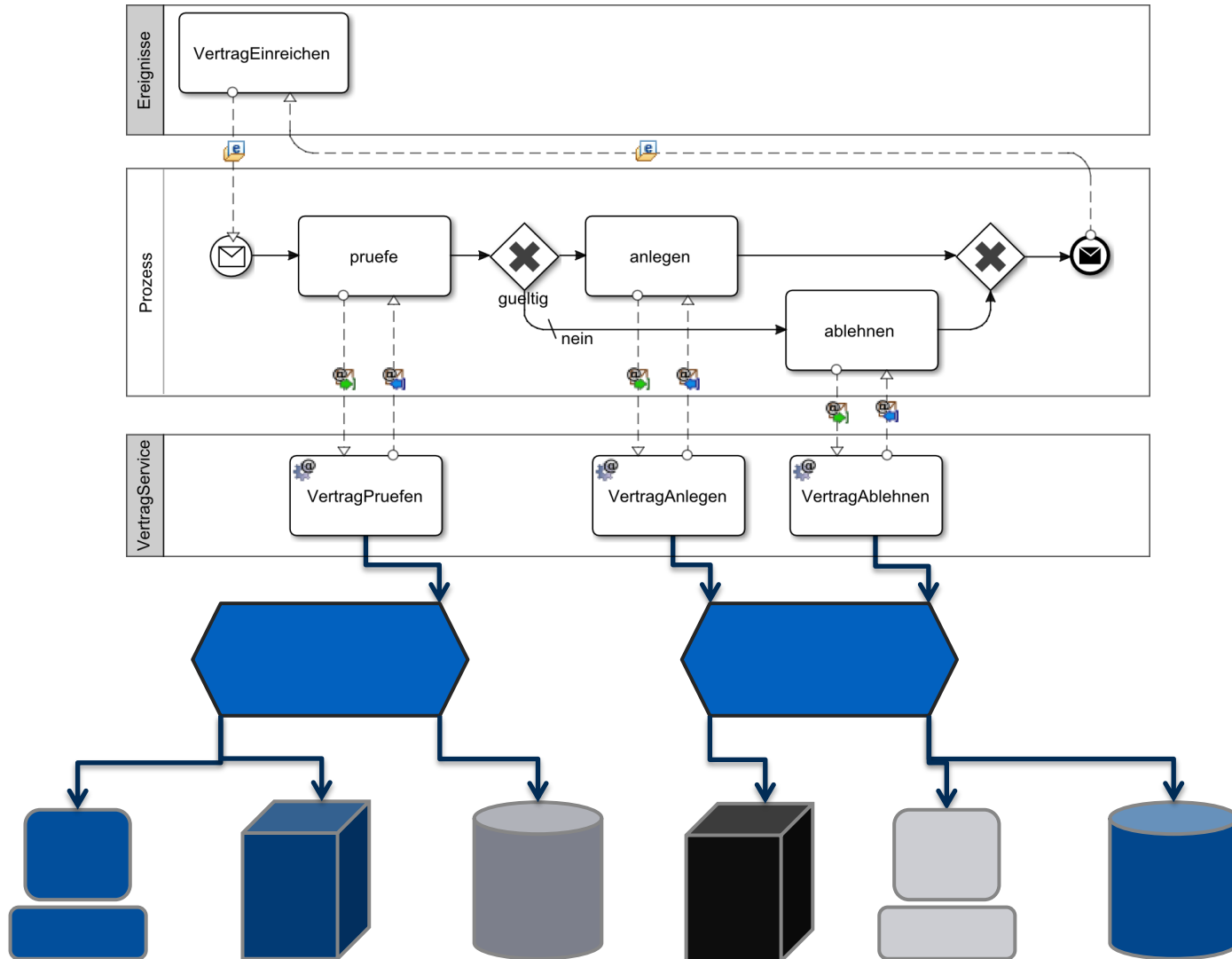
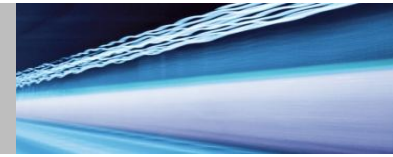
# Ein einfacher Prozess...



# ... einige Services ...



# ... viele Systeme

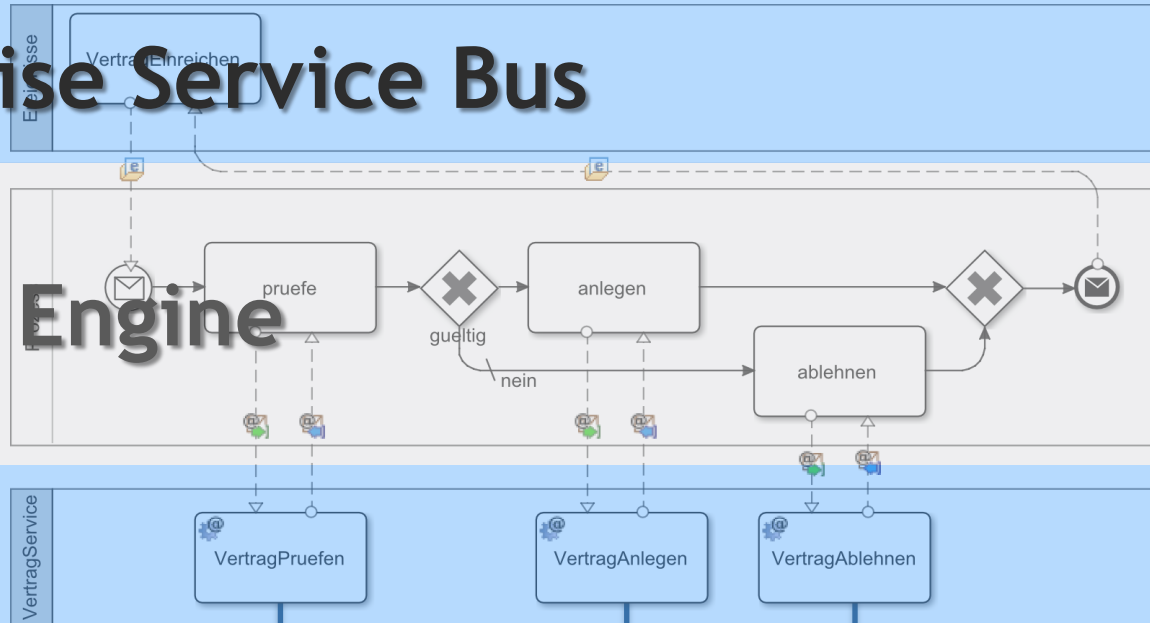


... viele Systeme



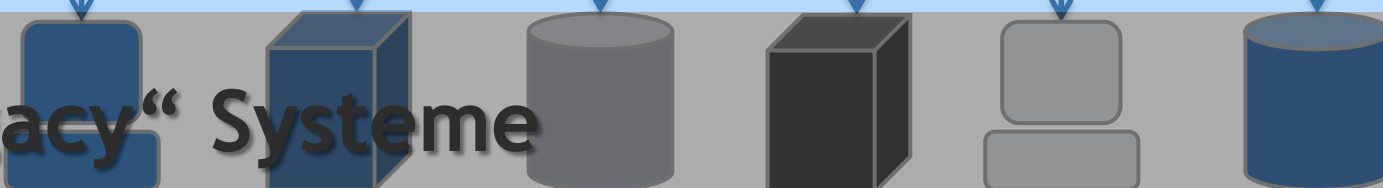
# Enterprise Service Bus

# Prozess Engine

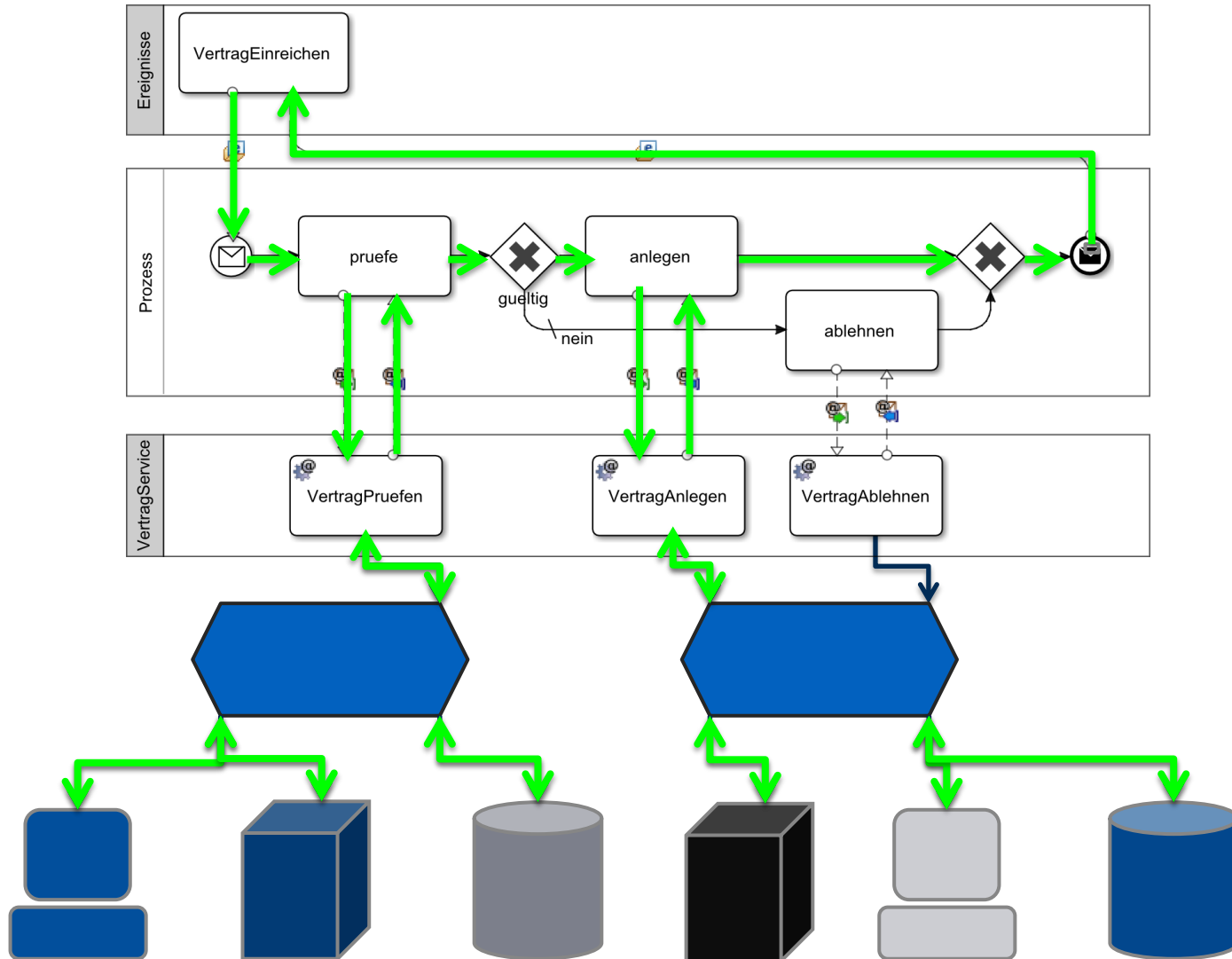
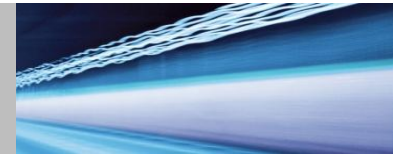


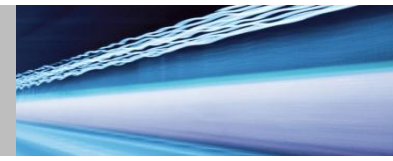
# Enterprise Service Bus

# „Legacy“ Systeme

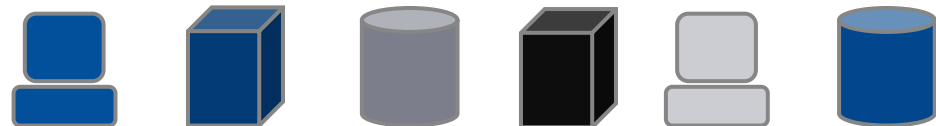


# Ein möglicher Testfall

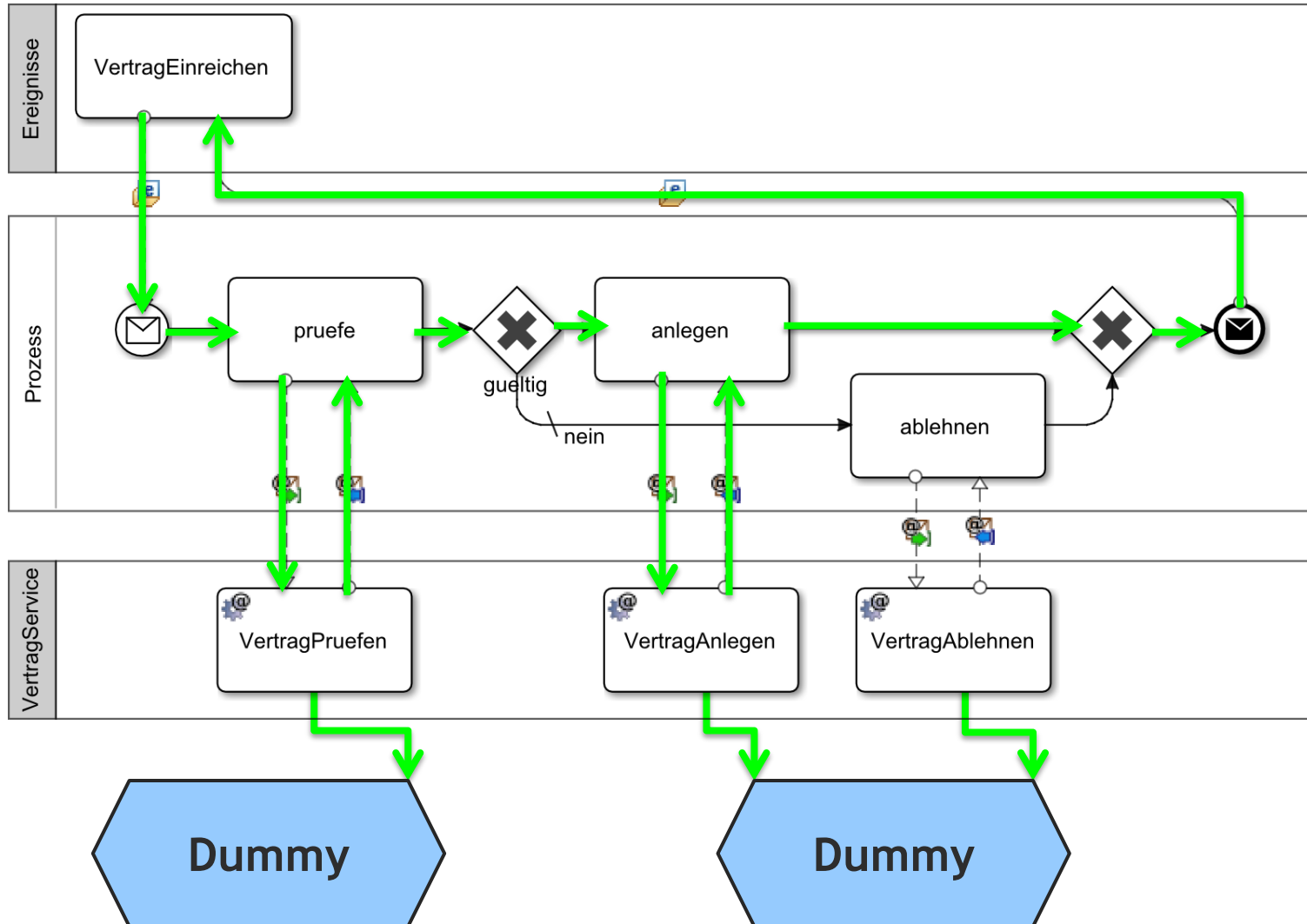
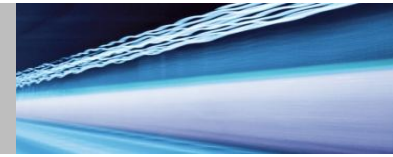




- Es ist sehr aufwendig einen Test auf Basis der richtigen Systeme durchzuführen
  - Alle verwendeten Systeme müssen in der richtigen Version verfügbar sein. Unabhängige Entwicklung der einzelnen Prozesse und Services wird erschwert.
  - Passende Testdaten müssen in allen Systemen vorliegen und werden durch einen Durchlauf unter Umständen „verbraucht“.
  - Bestimmte Fehlersituationen sind unter Umständen schwer zu erzeugen.
- Auftretende Fehler lassen sich aufgrund der zahlreichen involvierten Systeme schwer lokalisieren

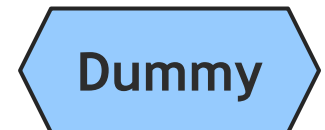


# Dummies nehmen einen Teil der Komplexität weg

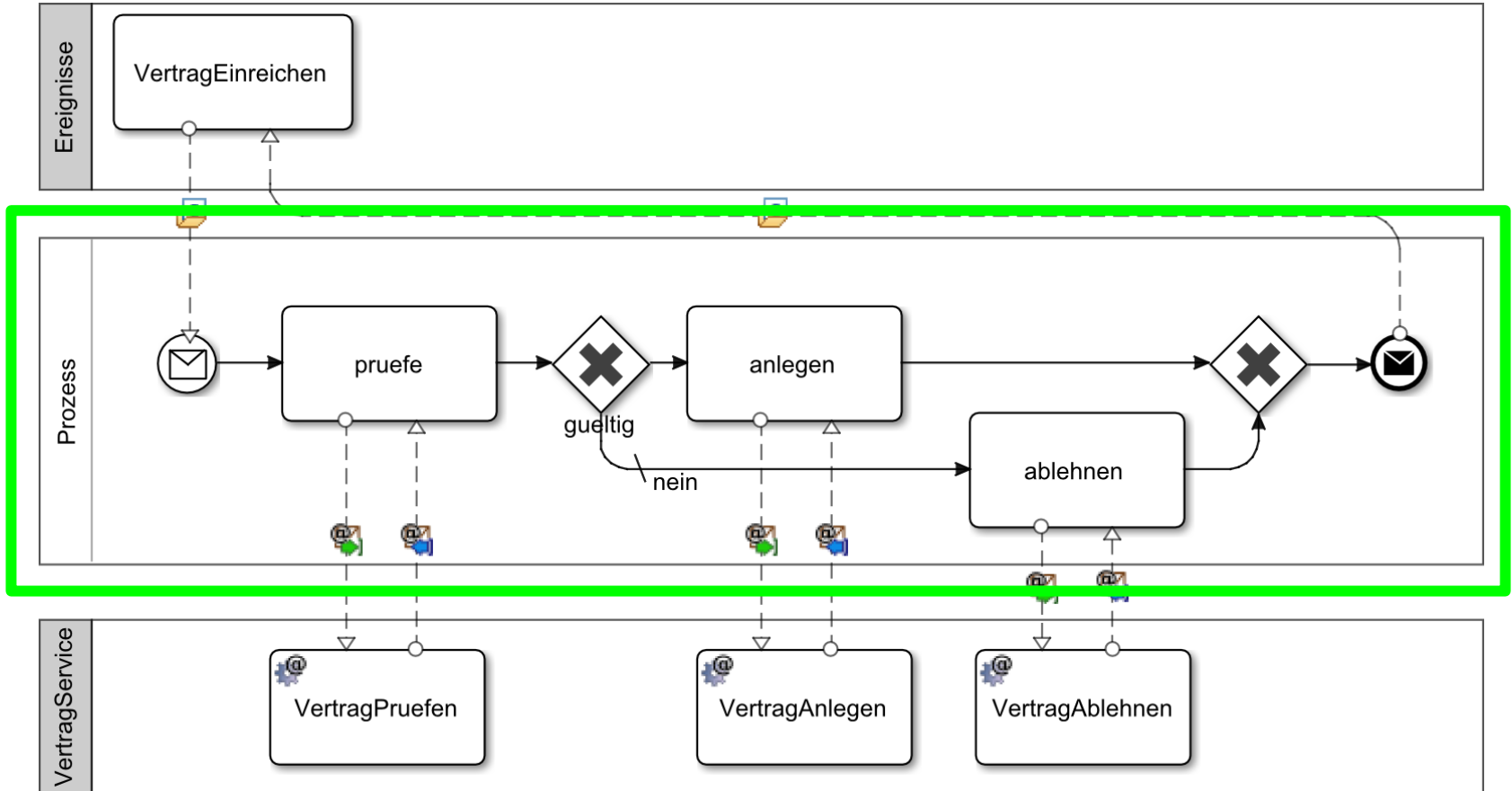
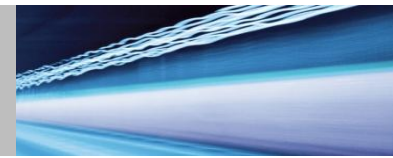




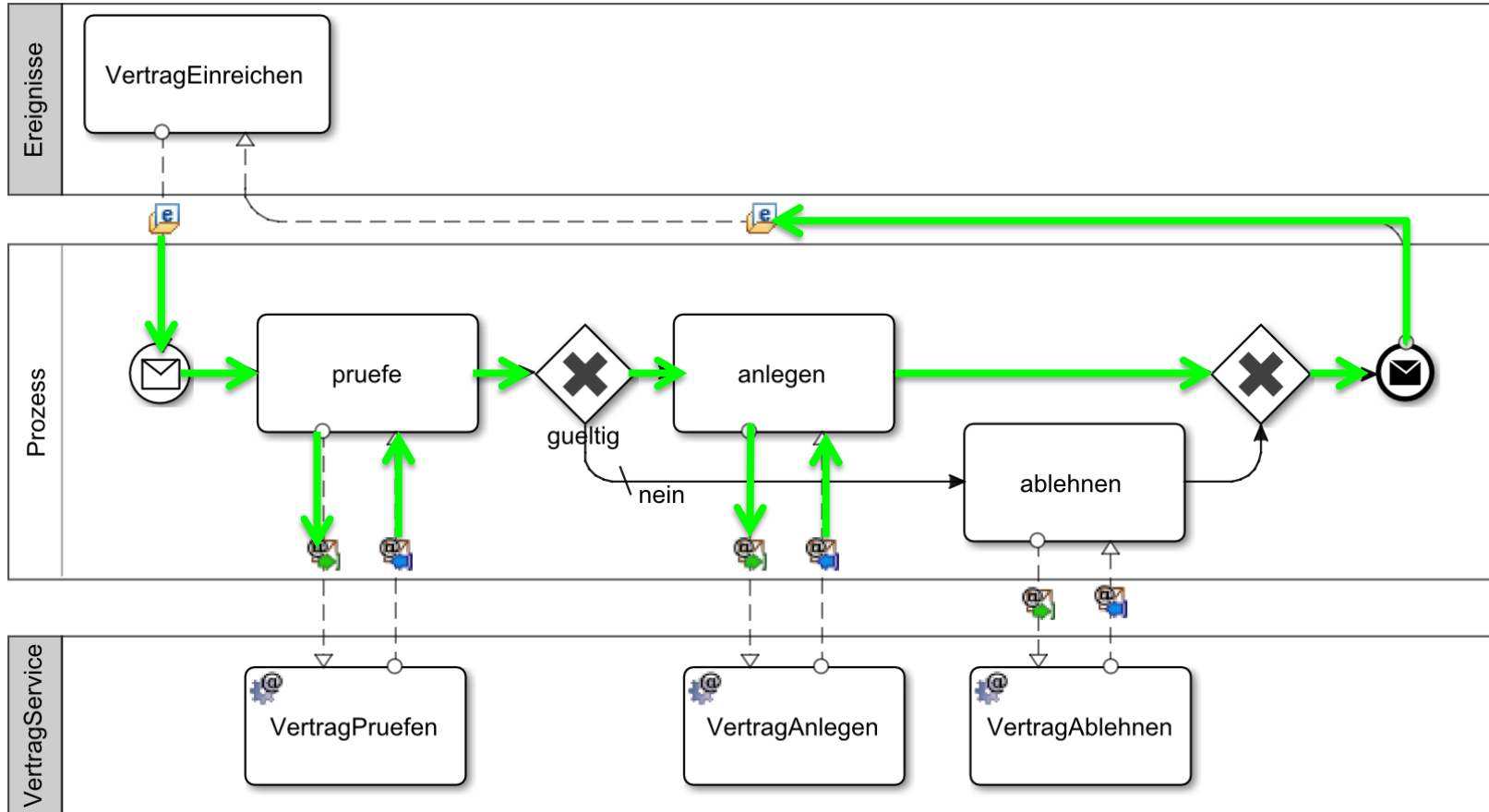
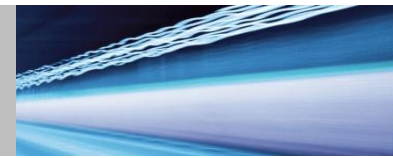
- Für Prozesstest mit Dummies müssen immer noch mehrere Systeme zusammenspielen.
- Für unterschiedliche Testfälle braucht man unter Umständen unterschiedliche Dummies.
- In einem Test mit Dummies wird immer noch zu viel getestet und es gibt zu viele Fehlerquellen neben dem zu testenden Prozess.
- Durch die Implementierung von Service-Dummies entsteht zusätzlicher Quellcode, der ebenfalls gewartet werden muss.



# Der eigentliche Prozess...



# ... „endet“ mit den ausgetauschten Nachrichten



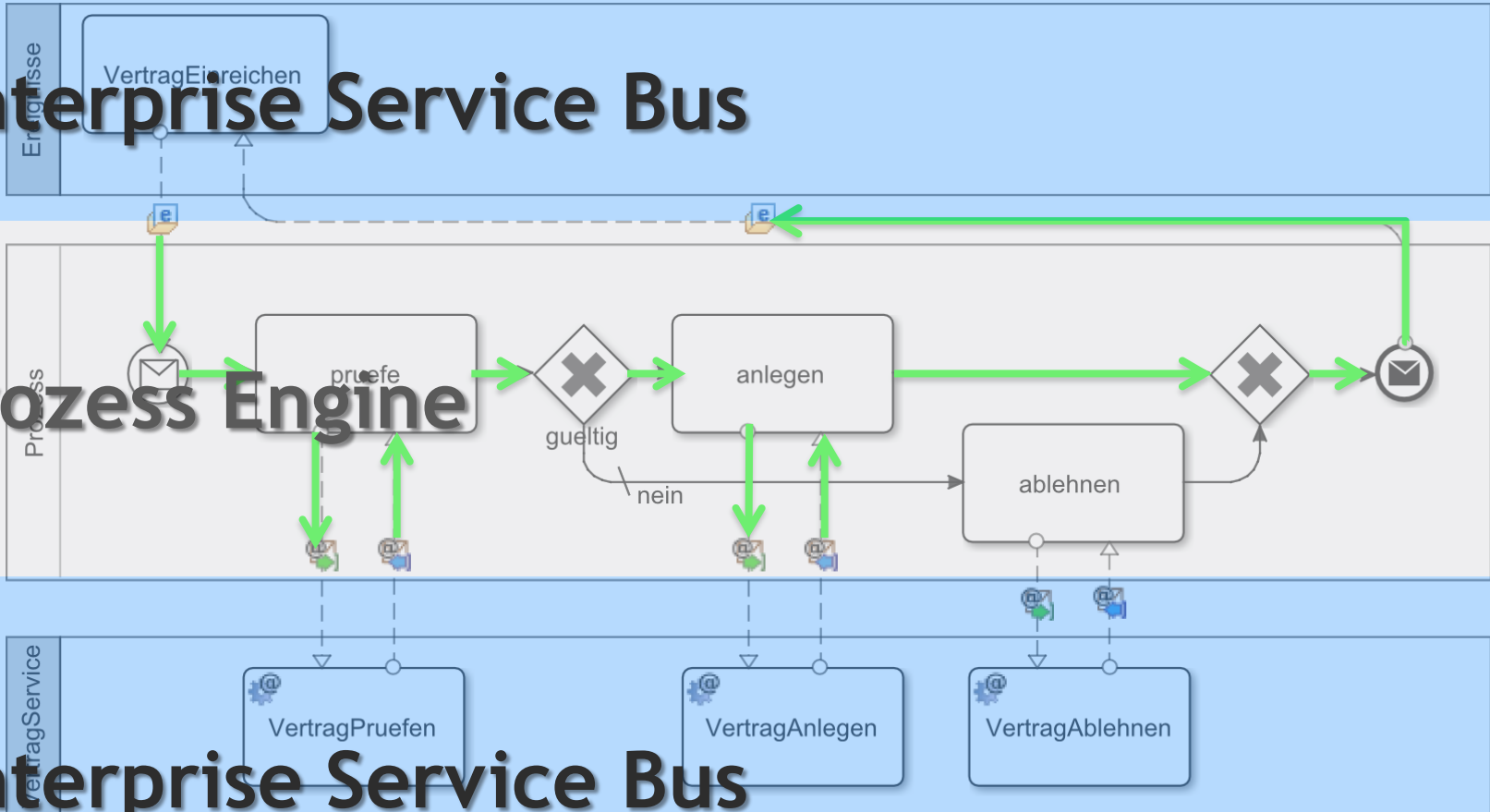
... „endet“ mit den ausgetauschten Nachrichten

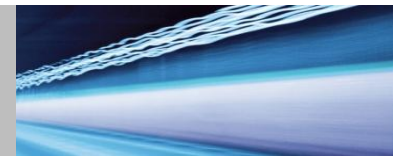


# Enterprise Service Bus

# Prozess Engine

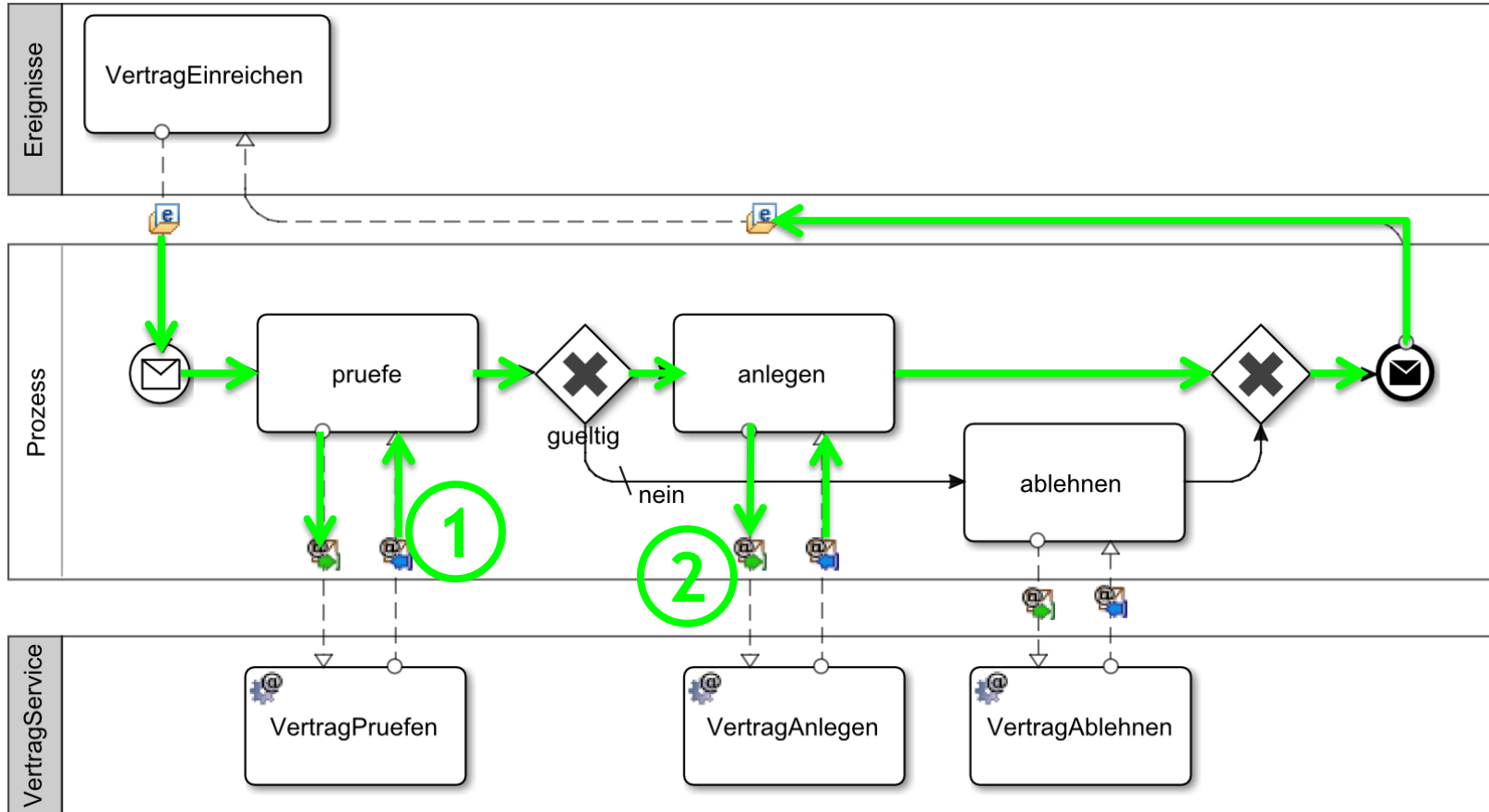
# Enterprise Service Bus



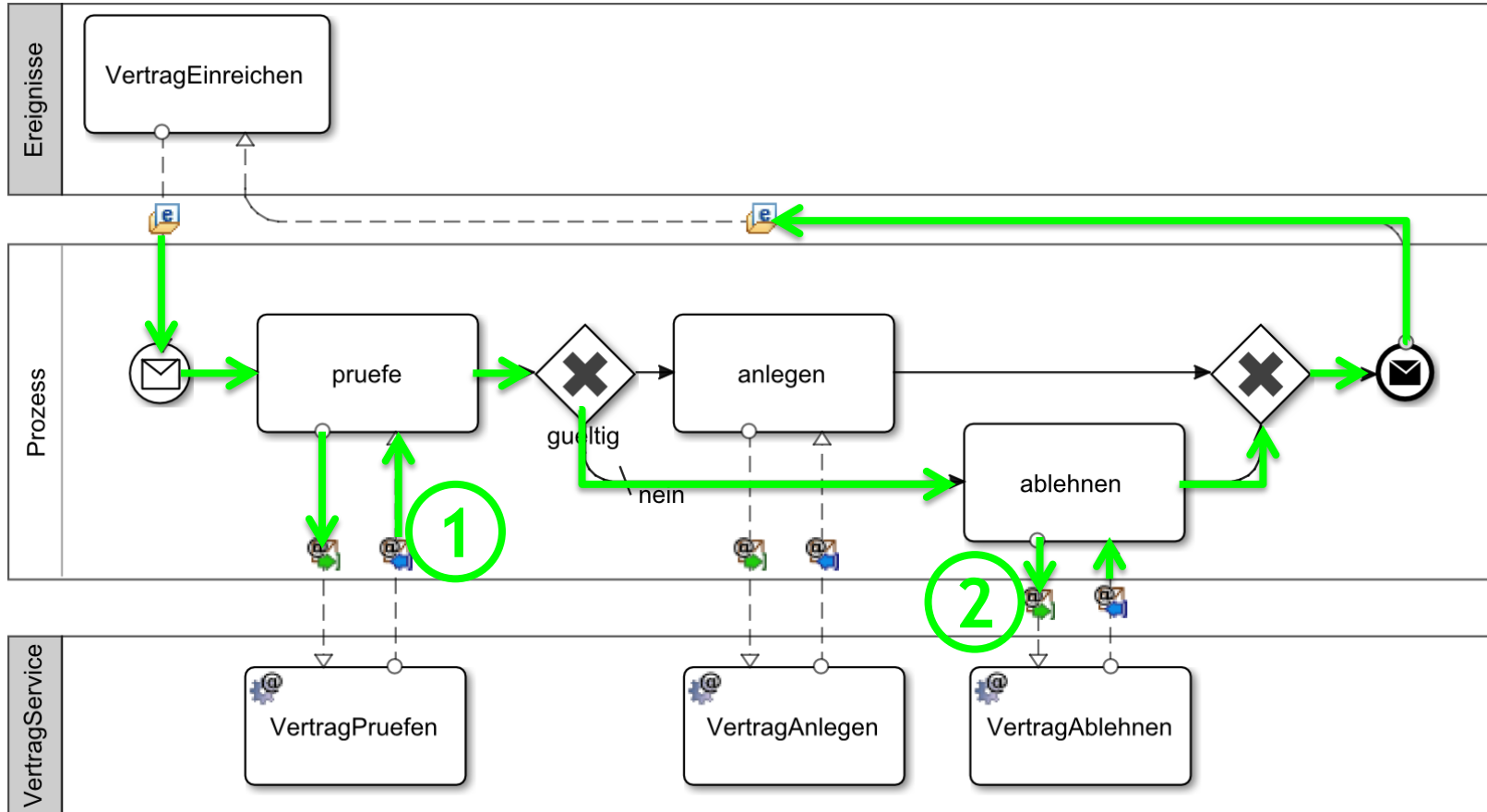
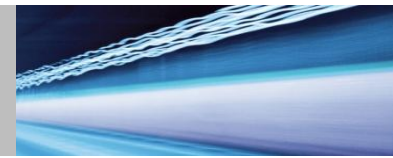


- Wenn der ESB die Möglichkeit bietet Testfälle an beliebigen Schnittstellen einsetzen zu können...
- ... muss ein Prozesstest nur aus einem definierten Satz Nachrichten bestehen:
  - Die Nachrichten können an den Prozess gesendet werden...
  - Oder werden vom Prozess erwartet
- Einzelne Testfälle unterscheiden sich durch den Austausch der verwendeten Nachrichten:
  - Testen des Kontrollflusses durch die Art der Nachrichten
  - Testen des Datenflusses durch den Inhalt der Nachrichten

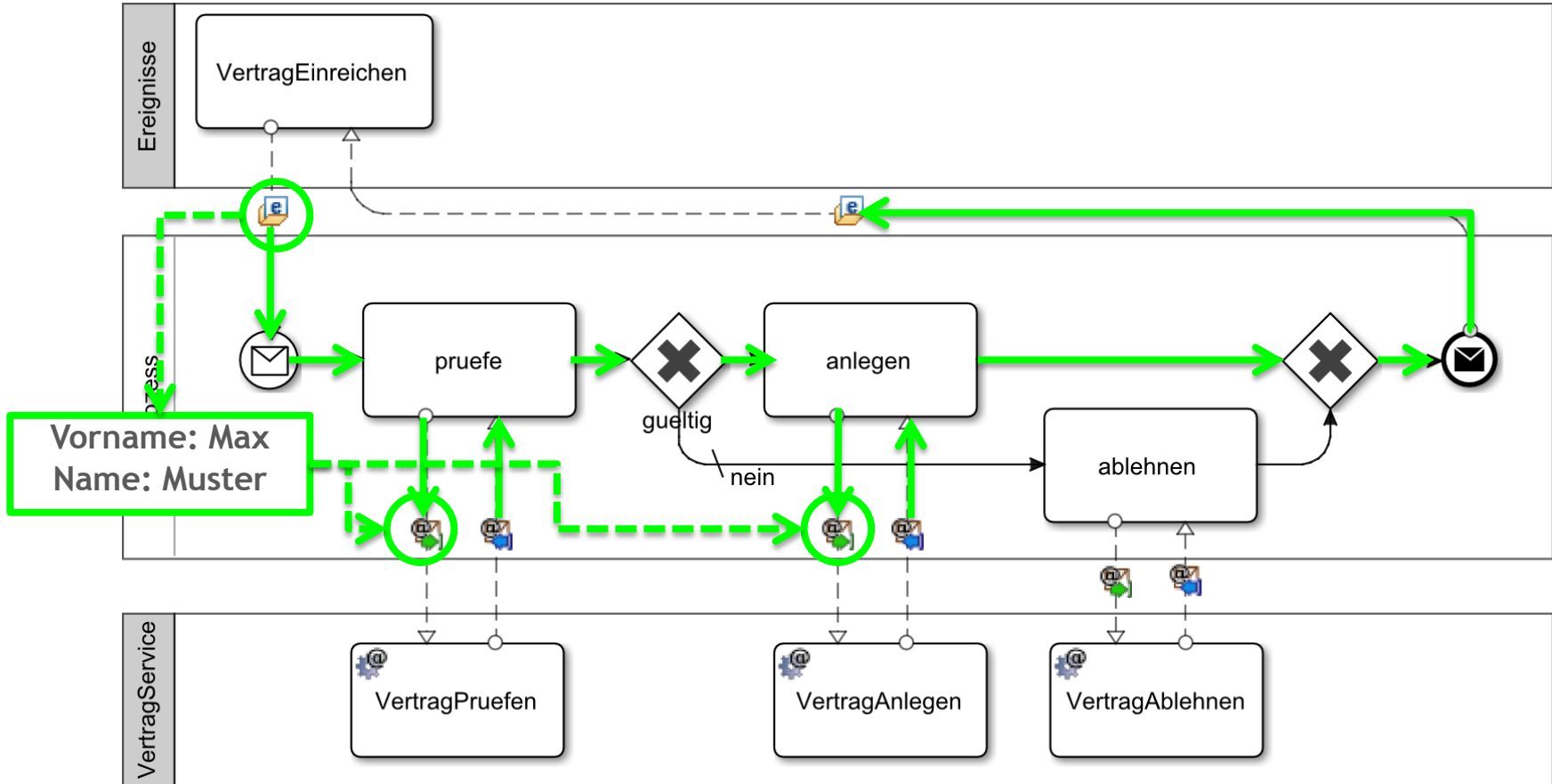
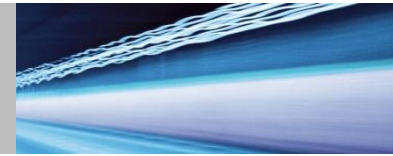
# Testen des Kontrollflusses: Variante A

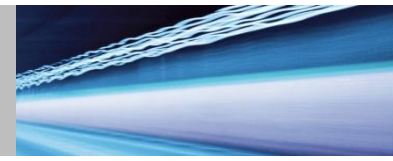


# Testen des Kontrollflusses: Variante B



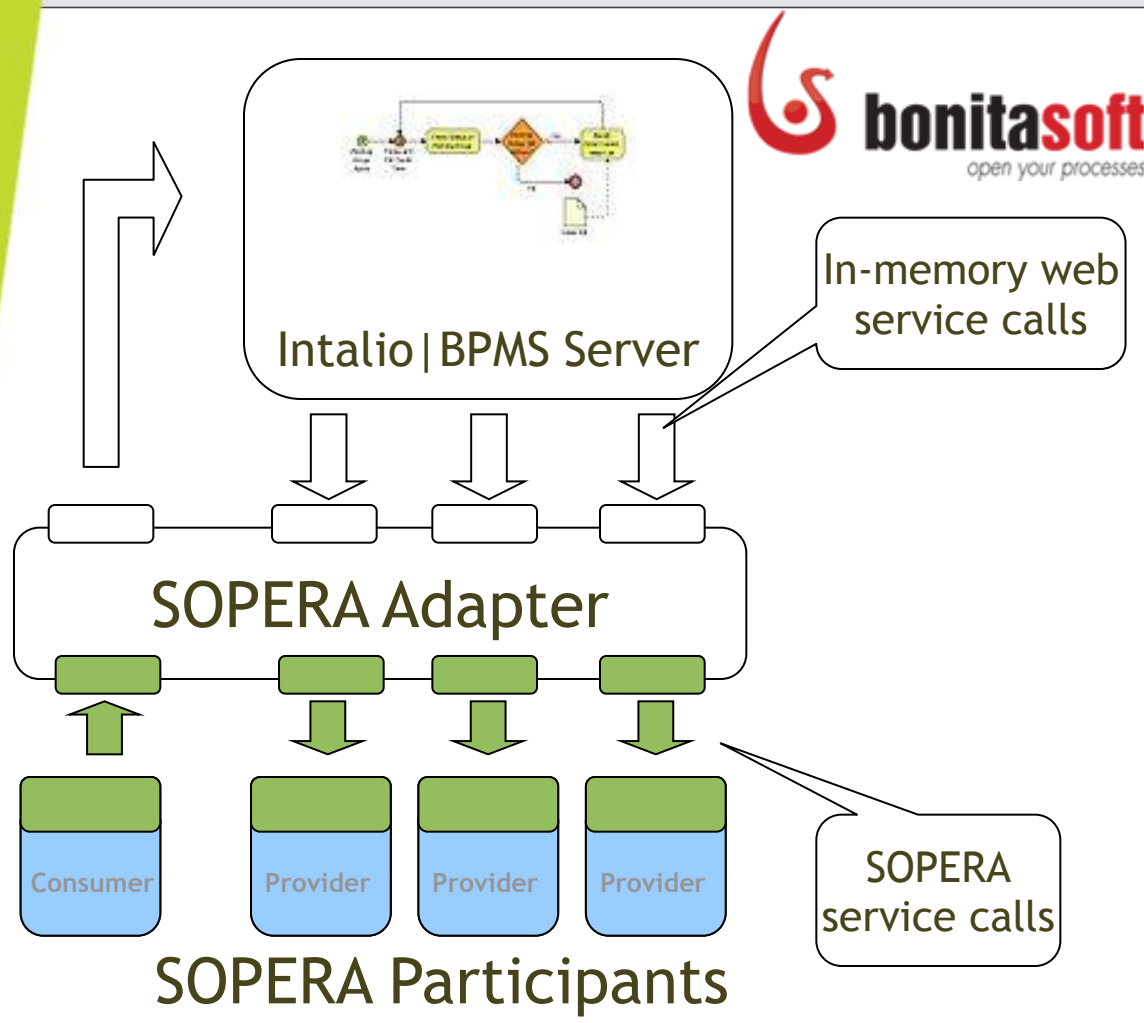
# Testen des Datenflusses durch Nachrichteninhalt





- Der Prozess sollte in der normalen Umgebung ausgeführt werden. Der ESB muss es zulassen, dass sich der Testfall anstelle der „echten“ Services in den Nachrichtenfluss einklinken kann.
- Sätze von Nachrichten müssen als Testfälle konfiguriert werden können:
  - Welche Nachrichten gesendet werden
  - Welche Nachrichten erwartet werden
- Einzelne Werte in den Nachrichten sollten als Variablen exportiert werden:
  - Steigerung Wiederverwendbarkeit von Testnachrichten
  - Ermöglicht eindeutige Werte über alle Testläufe (zum Beispiel für Korrelation)

# BPM Integration with SOPERA ASF



- Call SOPERA Providers in Business Processes
- Call Business Processes from SOPERA Consumers
- Human Interaction and Workflow
- Dynamic Provider Lookup & Location Independence
- Service Activity Monitoring
- Security & Policy Handling
- Registry Integration (Drag&Drop)

## Prozesstests: ein erster Ansatz mittels Testservices

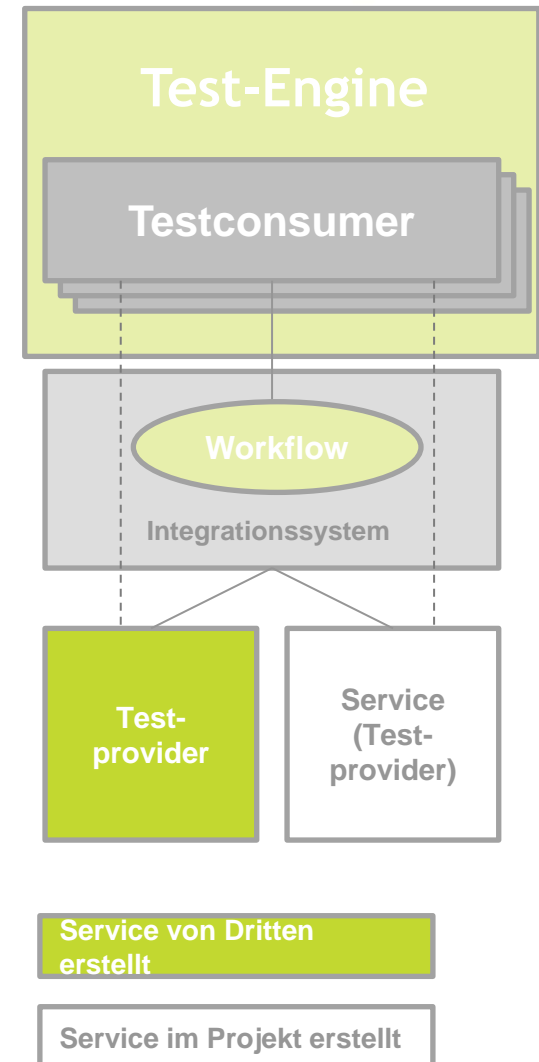
- Services können neben der vollen Implementierung auf Basis von Hintergrundsystemen noch eine **Testimplementierung** bereitstellen
- Dadurch werden die Abhängigkeiten zum Zustand der Hintergrundsysteme entfernt
- Diese Tests können zum Testen von Erreichbarkeit von Services, korrekter Anbindung an den Prozess, Berechtigungen, usw. genutzt werden.
- Nachteile:
  - Es entsteht zusätzlicher Aufwand für die Testimplementierung
  - Das Setup für solche Integrationstests ist immer noch komplex (z.B. Deployment der Services)

# Das Test-Framework

... ist ein Black-Box-Test auf Basis der fachlichen Prozesse

... basiert auf Testszenarien pro zu testendem Workflow

- Gliederung in Testfälle und Testschritte
- Testfälle decken unterschiedliche Ausführungspfade im Workflow ab (z.B. Genehmigung von Urlaub, Ablehnung von Urlaub, Fehler bei Urlaubsbuchung)
- Testschritte definieren einzelne Aktionen zur Durchführung und Prüfung der Testfälle
- ... wird unterstützt durch den generischen Testconsumer und den generischen Testprovider
- Der Consumer ruft die Prozesse auf und kann mit dem Workflow über den TMS interagieren.



## Prüfen des Nachrichteninhalts

- je nach Request wird eine definierte Response als Antwort gesendet
- Modifizieren der Responses durch feldweises Kopieren von Werten
- Der erwartete Nachrichteninhalt für vom Prozess gesendete Requests wird ebenfalls in der Konfiguration des Testfalls festgelegt.
- Die Vorbedingungen lassen sich als XPath-Ausdrücke formulieren, z.B. `/Brief/Absender/Name=„Müller“`
- Um die Testfälle flexibler zu machen, werden Variablen eingeführt, die in der Konfiguration verwendet werden und vom Testfall gesetzt werden:

### Nachricht:

```
<Brief>
  <Absender>
    <Name>${name}</Name>
  ...
```

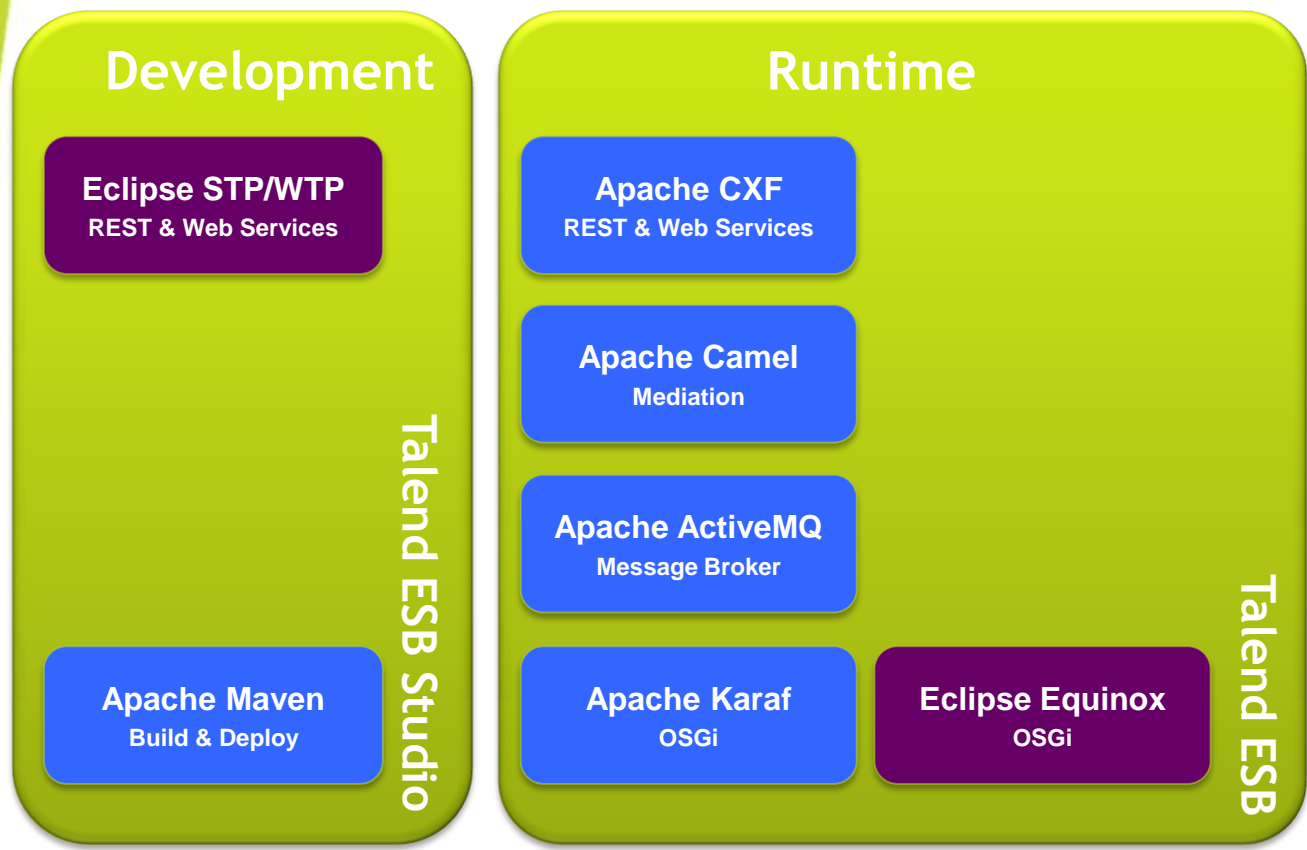
### Vorbedingung:

```
/Brief/Absender/Name=„${name}“
```

# Einsatzszenarien

- Service-Simulation zur Entwicklungszeit
  - Parallele Entwicklung von Service-Konsumenten und Services
  - Konsumenten können gegen Mockups getestet werden
- Test-Konsument zum Schnittstellen-Test
  - Schrittweise Entwicklung von Test-Konsumenten
  - Regressionstests der Service Implementierung über einen Test-Konsumenten
- Workflow-Test , Prozess-Test
  - In einem Workflow-Test können Services durch Mockups ersetzt werden.
  - Die Simulation von Benutzeraktionen an der Benutzerschnittstelle übernimmt ein Test-Konsument.

# Talend ESB Components



Project Management

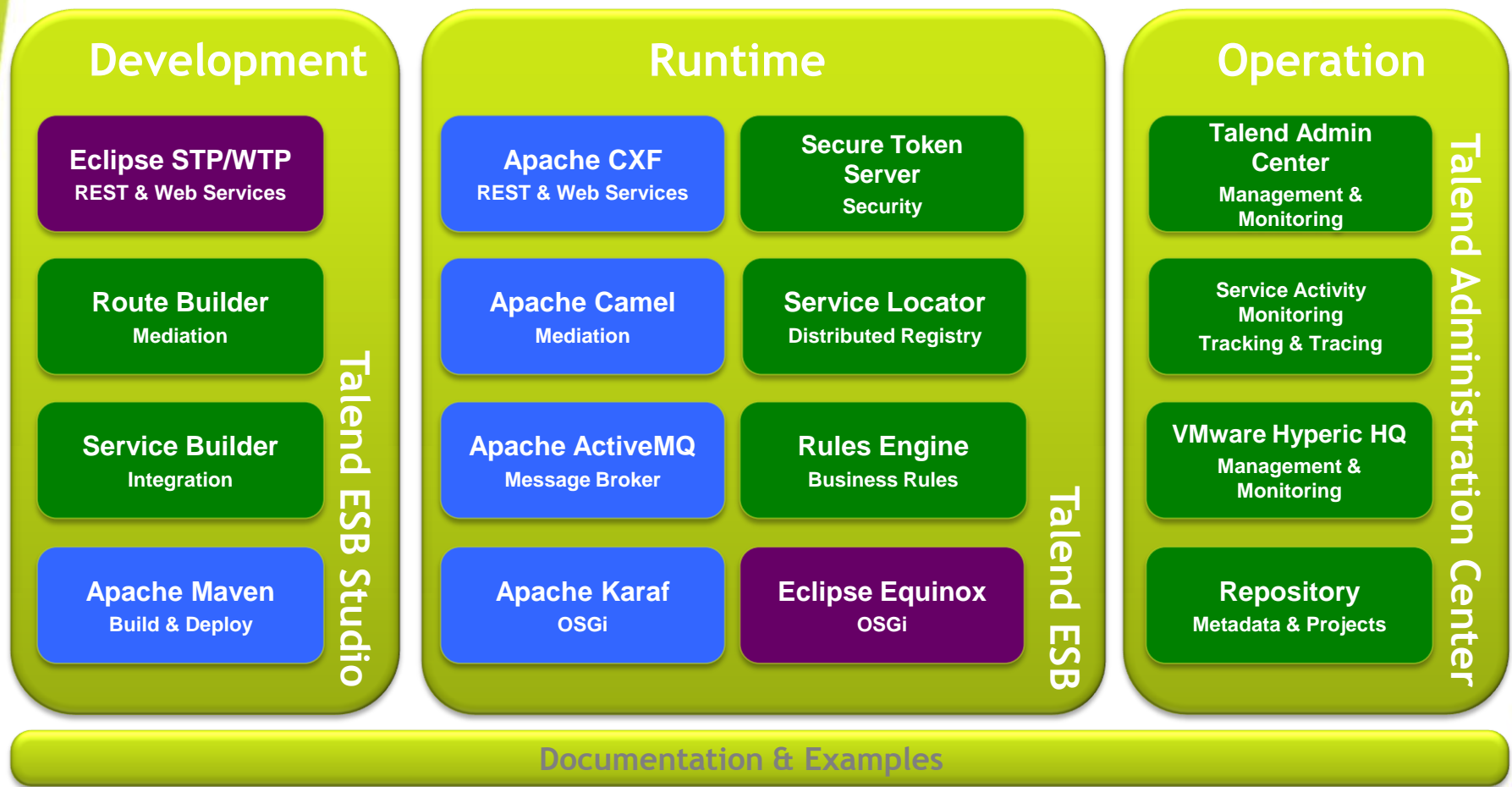


Strategic Board Member



Expert Group Member

# Talend ESB Components



Project Management  
Chairs



Strategic Board  
Member



Expert Group  
Member

# Das Apache Camel Projekt

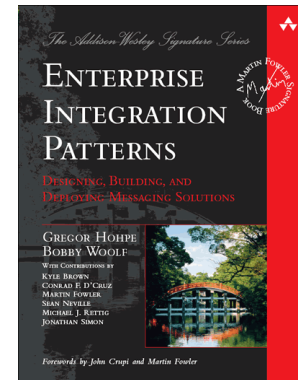
## Leistungsfähiges Integrationsframework

- Open Source Apache License v2
- Homepage des Projekts: <http://camel.apache.org>
- Aktive und wachsende Community



## Camel basiert auf EIPs!

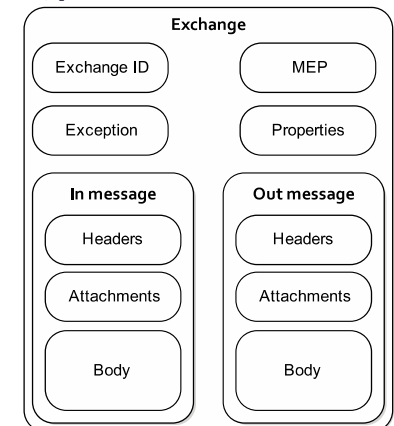
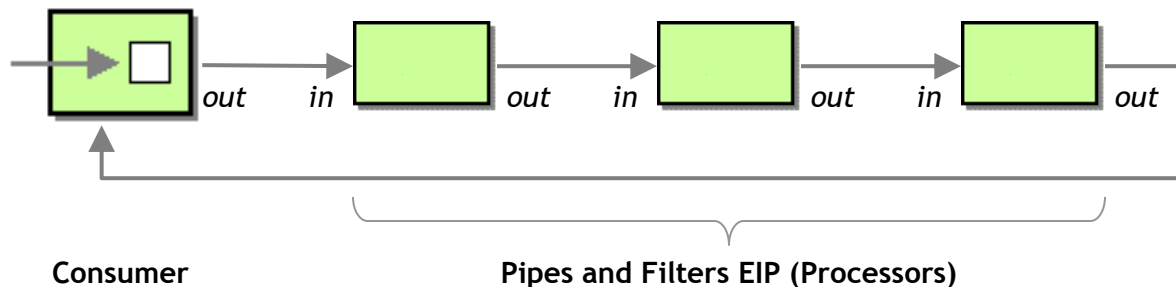
- Messages und Exchanges
- Processors
- Endpoints
- Producers und Consumers
- Languages, Expressions und Predicates
- TypeConverters
- Routes
- CamelContext, Services und Registries



# Routing in Camel

## Konvention vor Konfiguration!

- Eine Route ist (vereinfacht) ein Consumer, der mit einem zusammengesetzten Processor verbunden ist
- Ein Processor verarbeitet einen Exchange, indem er die eingehende Message nutzt und optional eine ausgehende Message erzeugt (oder eine Exception)
- Die out Message des vorherigen Processor ist die in Message des jeweils nächsten
- Gibt der vorherige Processor keine out Message zurück, wird die vorige in Message zur out Message
- Am Ende der Verarbeitung wird die letzte out (oder in) Message zur Rückgabemessage und vom Consumer zurückgeschickt



# Routen debuggen

Um Routen zu debuggen, können Sie...

- ...mit einem vereinfachten Testfall beginnen
- ...Unit Tests (JUnit) nutzen
- ...das Test Kit von Camel nutzen

*<http://camel.apache.org/testing.html>*

- ...“mock://...” Endpoints nutzen
- ...Logdateien und den “log://...” Endpoint nutzen
- ...Camel Tracer Interceptor nutzen

*<http://camel.apache.org/tracer.html>*

- ...den Java Debugger einer IDE (z.B. Eclipse) nutzen
- ...Camel Debugger nutzen

*<http://camel.apache.org/debugger.html>*

- ...die Funktionen von JMX und Notifications nutzen

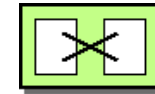


# Message Transformation

## Message Translator

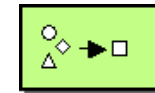
“Wie können Applikationen, die unterschiedliche Datenformate nutzen, miteinander kommunizieren?”

- Implizit mittels TypeConverter
- Explizit - in er DSL deklariert



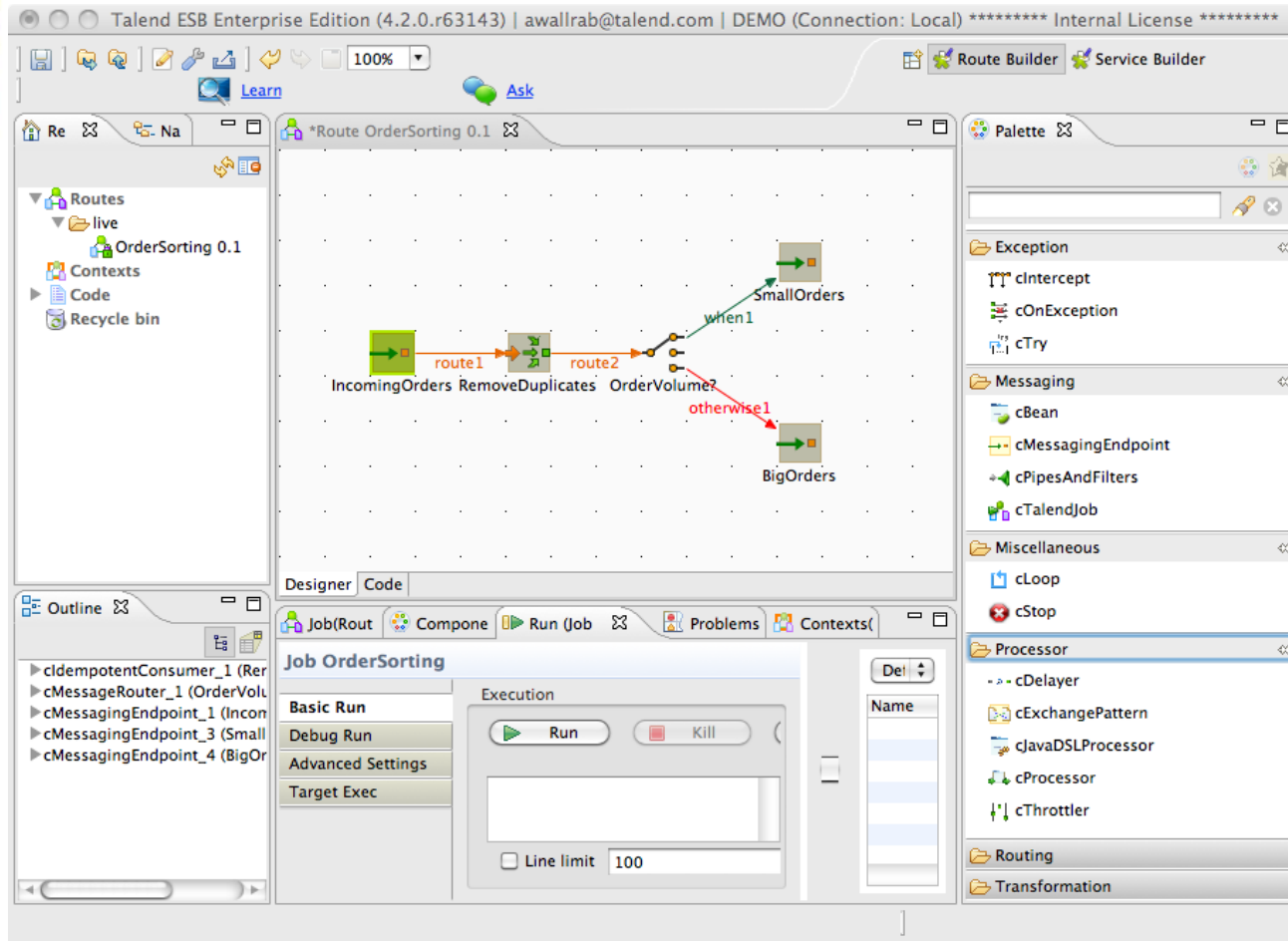
## Normalizer

“Wie werden Nachrichten verarbeitet, die semantisch gleich sind, aber unterschiedliche Formate haben?”



```
from("direct:set-body").setBody().constant("Goodbye World")
from("direct:bean").to("bean:translatorBean")
from("direct:processor").process(translatorProcessor)
from("direct:template").to("velocity:templates/report.vm")
```

# Talend ESB Route Builder



## Route Builder

- Endpoints
- EIPs
- **Processors**
- Custom components

## Configuration

- Components
- Endpoints

## Code Generation

- 100% Java
- Camel Code
- **Packaged as OSGi Bundles**

## Execution in the IDE

- Debugging
- Live statistics
- Short dev cycles

## Q &amp; A

Thank you! Merci! Vielen Dank!

Use our forums <http://www.talendforge.org/forum/>  
to ask questions or email [wstrunk@talend.com](mailto:wstrunk@talend.com) and  
we will get back to you via email