Service Engineering

Modellgetriebene Komposition von Serviceangeboten

Die Inhalte der Vorlesung wurden primär auf Basis der angegebenen Literatur erstellt. Darüber hinaus finden sich vielfältige Beispiele aus dem Bereich der Telekommunikation.

Motivation und Einführung

Motivation

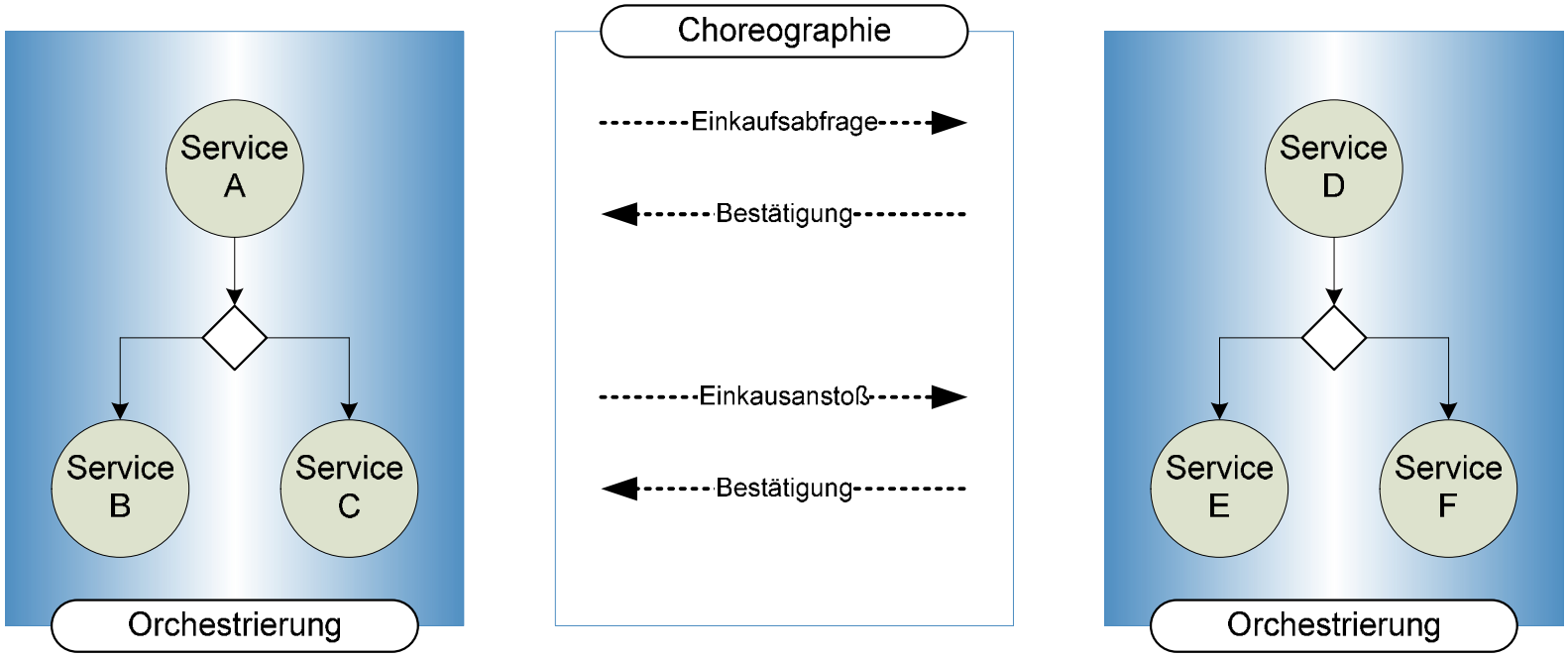
"In the past, we've been builders of custom software, or deployers of packages. In the new, agile application development, we'll find that reuse and assembly will be the keys. Application development organizations can't code themselves into the future!"

Quelle: Hotle, M.; Vecchio, D.: AD Scenario – Tying Agile Development, BPM and Architecture Together, Gartner Symposium ITxpo, Cannes/France 2005

Ziele einer Servicekomposition

- Produktivitätserhöhung bei der Erstellung einer Softwarelösung.
- Verwendung qualitativ gesicherter Serviceangebote.
- Möglichkeit zur agilen Umsetzung neuer Anforderungen.
- Industrialisierung mit Hilfe von Standardservices.
- Arbeitsteilige Implementierung in Bezug auf Zeit und Ort.
- Innovationen über Communities treiben.
- Unterstützung aktueller Trends (z.B. IoT, Big Data und Mobile)

Orchestrierung/Choreographie



In Anlehnung an: Peltz, Chris: Web Services Orchestration and Choreography. In: IEEE Computer (2003), Nr. 36, S. 46–52

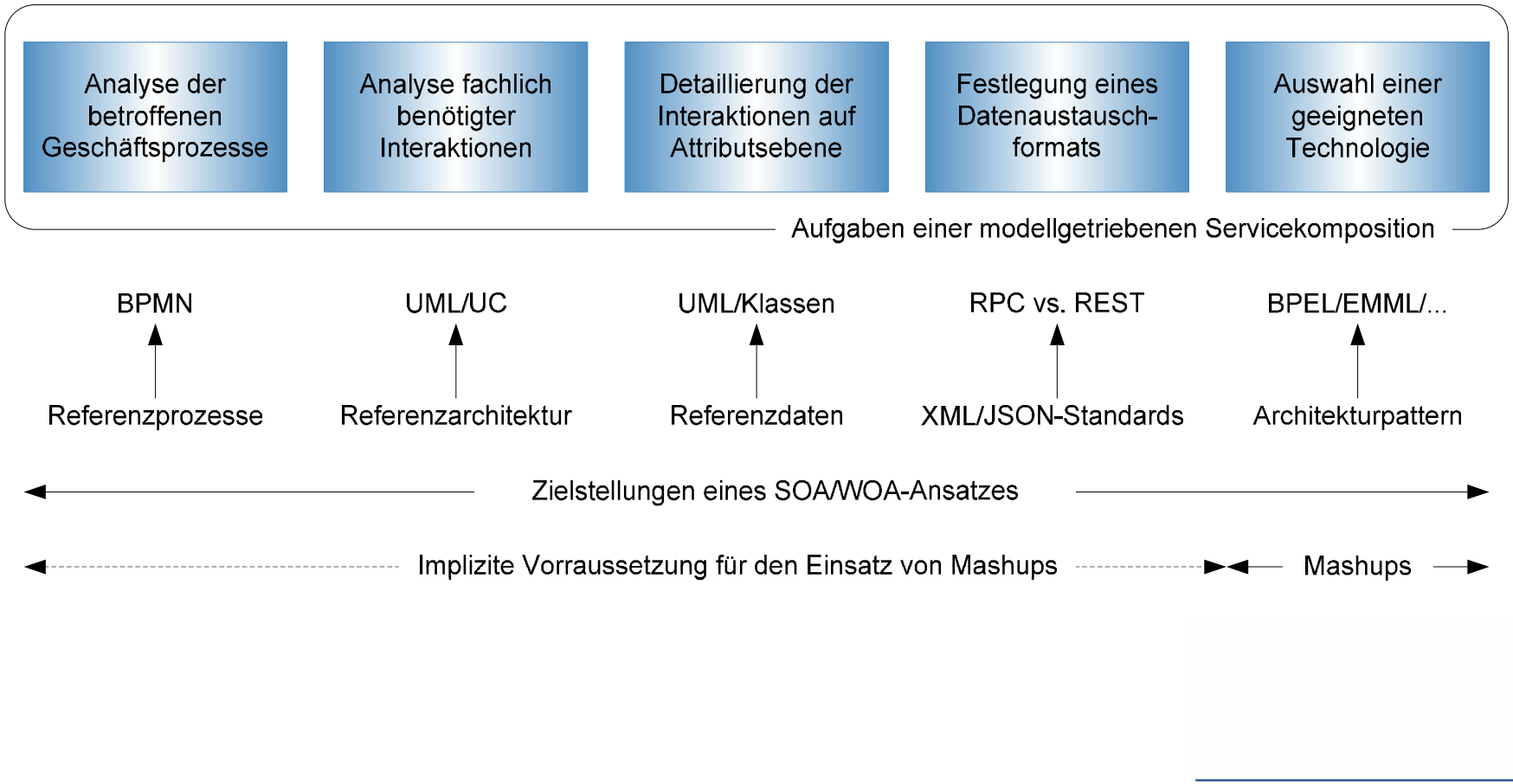
Kompositionsansätze

- Orchestrierung - ausführbare Aspekte eines Geschäftsprozesses
  - Abbildung der Prozesslogik auf auszutauschende Nachrichten.
  - Zentrale Einheit ruft Services mit entsprechenden Parametern auf.
  - Zentrale Einheit synchronisiert und koordiniert das Zusammenspiel.
- Choreografie - keine zentrale Koordination → Microservice-Idee
  - An einer Choreografie beteiligte Service arbeiten autonom.
  - Interaktionsfähigkeiten werden durch die Serviceschnittstelle festgelegt.
  - Gestaltung unternehmensübergreifender Interaktionen.

→ WS Choreography Description Language (WS-CDL) des W3C

→ WS Business Process Execution Language (WS-BPEL) der OASIS

Prozessgetriebene Komposition



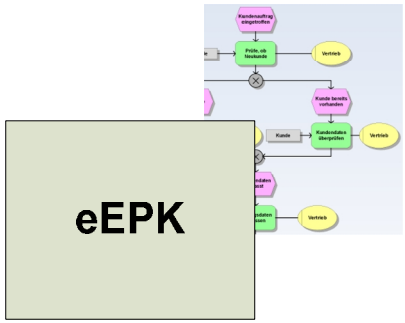
Klassische Integrationsprobleme

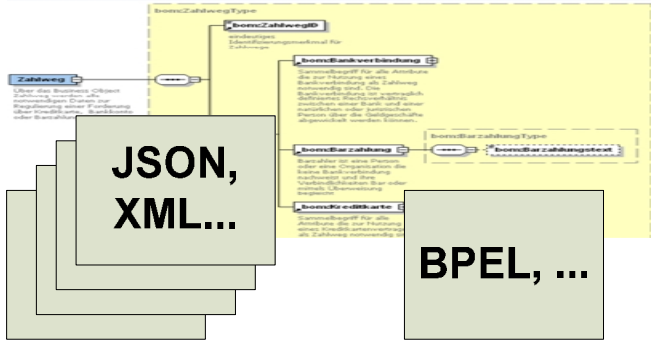
- Unterschiede beim semantischen Verständnis
- Beschreibungskonflikte (z.B. Homonyme und Synonyme)
- Heterogenitätskonflikte durch unterschiedliche Paradigmen
- Strukturelle Konflikte (z.B. Schemakonflikte)
- Inkonsistente und unkorrekte Daten
- …

In Anlehnung an: Conrad, S.: Föderierte Datenbanksysteme, Berlin, Springer 1997

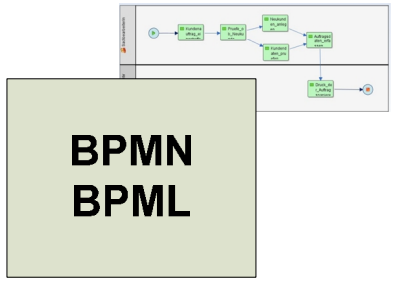
Prozess- und Datenmodell

Prozessgetriebene Servicekomposition

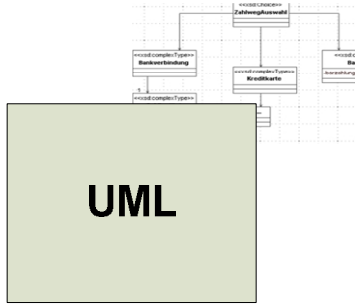




Datenaustausch-
formate

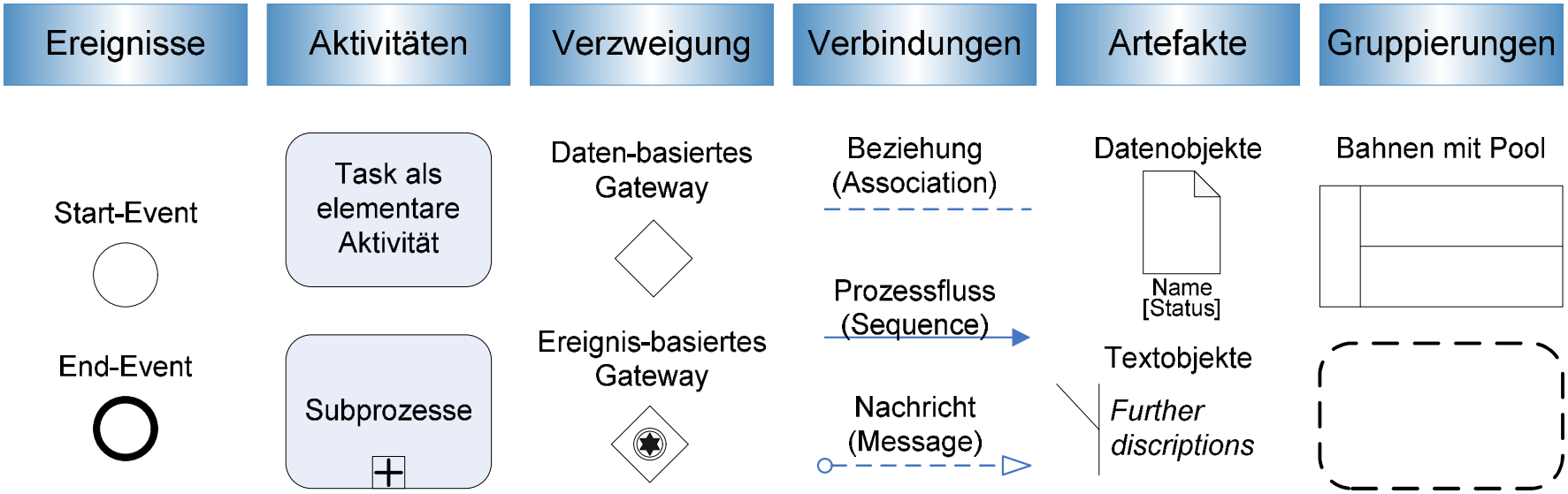


Process-
Implementierung



Process- und
Service-Engineering

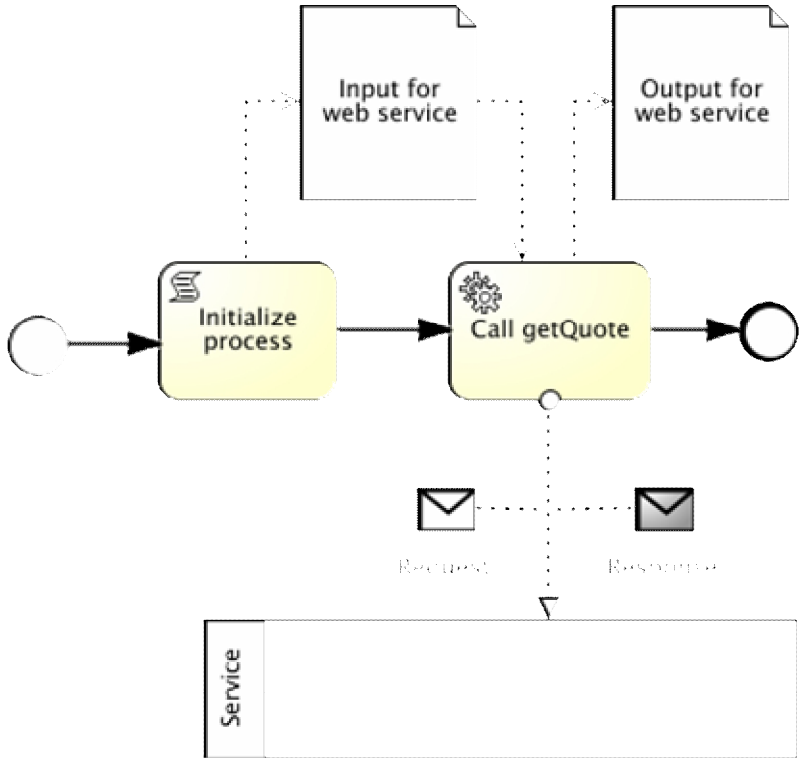
Basiselemente der BPMN-Notation



In Anlehnung an: http://www.bpmn.org/, Abruf Oktober 2016

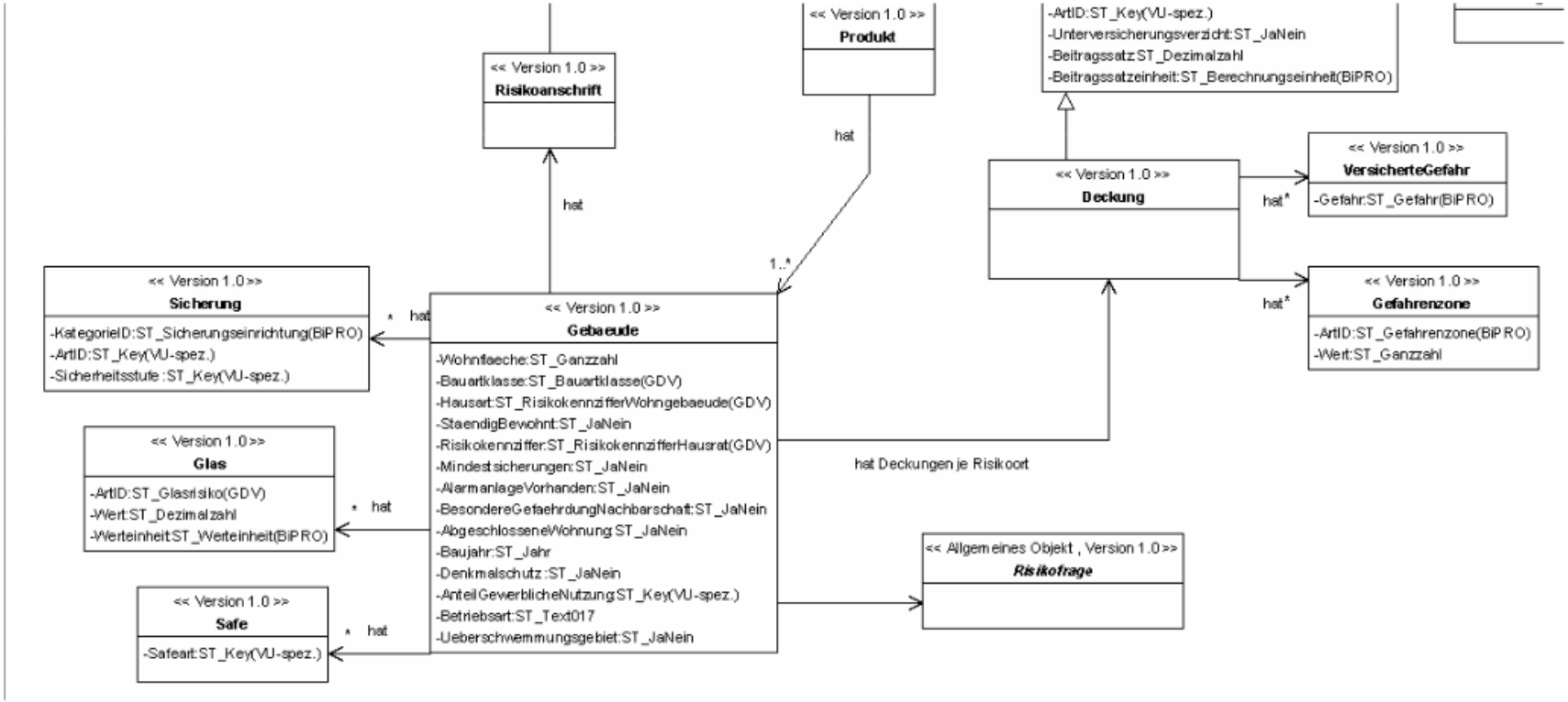
Mapping WSDL/BPMN

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions id="definitions" xmlns="http://www.omg.org/spec/BPMN/20100524/MODEL"
  xmlns:bpmn="http://schema.omg.org/spec/BPMN/2.0"
  xmlns:bpmndi="http://www.omg.org/spec/BPMN/20100524/DI"
  xmlns:dc="http://www.omg.org/spec/DD/20100524/DC"
  xmlns:di="http://www.omg.org/spec/DD/20100524/DI"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.pleus.net/example"
  xmlns:tns="http://www.pleus.net/example"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:nachrichten="http://www.bipro.net/namespace/nachrichten"
  xmlns:bipro="http://www.bipro.net/namespace"
  xsi:schemaLocation="http://www.omg.org/spec/BPMN/20100524/MODEL

  <!-- WSDL Import -->
  <import importType="http://schemas.xmlsoap.org/wsdl/"
          location="KompsitService_2.4.3.1.1.wsdl"
          namespace="http://www.bipro.net/namespace" />

   <!-- Item definition. Link to the external WSDL/XSD structure.
   <itemDefinition id="getQuoteRequestItem" structureRef="nachrich
   <itemDefinition id="getQuoteResponseItem" structureRef="nachric

   <!-- Message definitions. Link to the item definition. Can be v
   <message id="getQuoteRequestMessage" itemRef="tns:getQuoteReque
   <message id="getQuoteResponseMessage" itemRef="tns:getQuoteResp

   <!-- Interface definition. implementationRef = QName of WSDL Po
   <interface name="Komposit Interface" implementationRef="bipro:K
      <!-- Operation: implementationRef = QName of WSDL Operation
      <operation id="getQuoteOperation" name="getQuote Operation"
         <!-- Links to the message definitions -->
         <inMessageRef>tns:getQuoteRequestMessage</inMessageRef>
         <outMessageRef>tns:getQuoteResponseMessage</outMessageRef>
      </operation>
   </interface>
```



Quelle: Wolfgang …: How to Link WSDL-Services To BPMN-Processes, 2014, http://www.pleus.net/blog/?p=2335, Abruf Oktober 2016

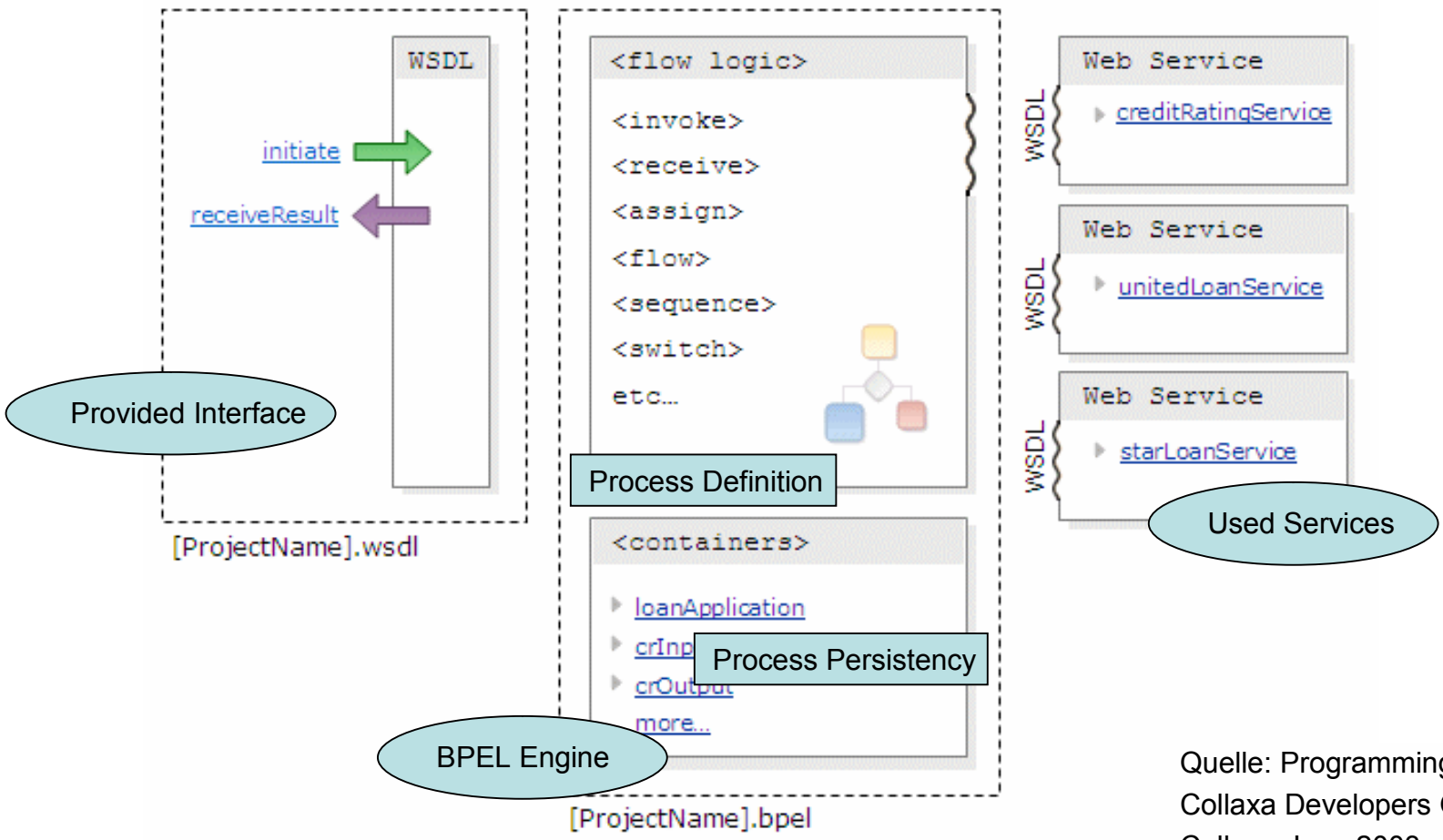
Auszug Datenmodell (hier BiPRO/Hausrat)



Quelle: Norm 421 – Private Sach-, Unfall, Haftpflicht-Versicherung, Release 1, Version 3.0, 2007 https://www.bipro.net/sites/default/files/uploads/norm/2010/11/05/bipro_norm421_r1_v3.0_on_taashu.pd, Abruf Oktober 2016

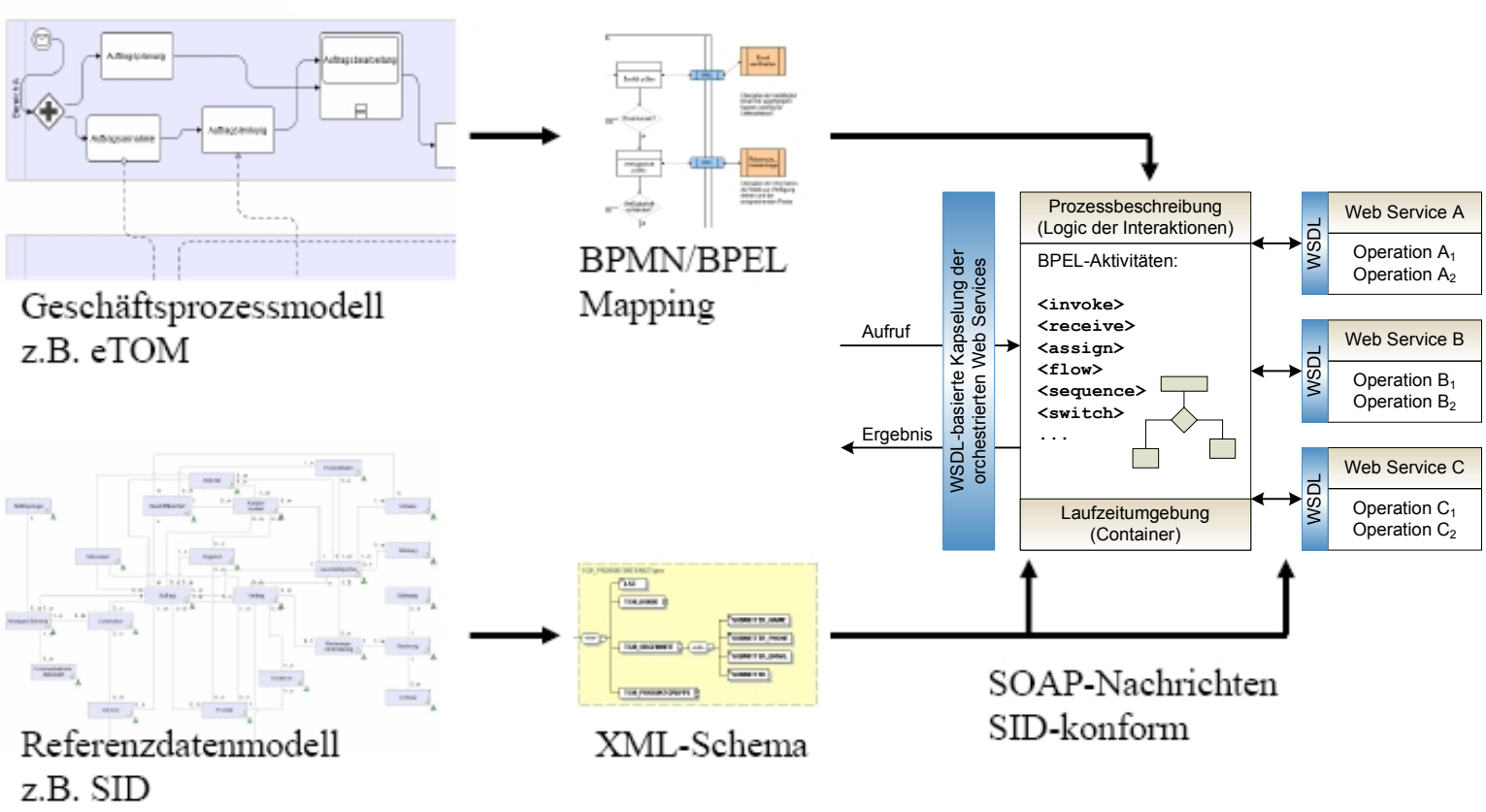
Orchestrierung mit BPEL

Servicekomposition mit Hilfe von BPEL



Quelle: Programming BPEL, Collaxa Developers Guide, Collaxa, Inc., 2003

Servicekomposition mit Hilfe von BPEL



Generische BPEL-Beschreibung

```
<process name="BPELProzess" targetNamespace="anyURI">
                                        <!-- Prozess -->
      <extensions>
                                        <!-- Erweiterunge
      </extensions>
      <imports />                       <!-- externe Abhä
      <partnerlinks>
                                        <!-- Dienste -->
      </partnerlinks>

      <variables>
                                        <!-- Variablen -->
      </variables>
      <faultHandlers>
                                        <!-- Fehlerbehanc
      </faultHandlers>

      <eventHandlers>
                                        <!-- Ereignisbehä
      </eventHandlers>
      <sequence name="Sequenz">
                                        <!-- prozedurale
            <recieve />
            <assign />
            <reply />
      </sequence>
</process>
```

```
<invoke name="openGeoDBAufruf"
        partnerLink="OpenGeoDBPartnerlink"
        operation="getLocationDetails"
        xmlns:impl="http://DefaultNamespace"
        portType="impl:openGeoDB"
        inputVariable="GetLocationDetailsIn"
        outputVariable="GetLocationDetailsOut"/>
```

```
<forEach  name="ForEach"
          parallel="no"
          counterName="ForEachCounter">
    <startCounterValue>0</startCounterValue>
    <finalCounterValue>5</finalCounterValue>
    <scope name="Scope1">
        <sequence name="Sequence2">
            <invoke name="Invoke2"/>
            <receive name="Receive2"/>
            <reply name="Reply2"/>
        </sequence>
    </scope>
</forEach>
```

Abgrenzung Microservices

- Bezugsbereich auf einen Aufgabenkontext (Modularisierung)
  - Fachliche Orientierung – vgl. Conways Law
  - Arbeitsteilung entlang fachlicher Fragen (nicht technisch!)
  - Orientierung an der kompletten Ineraktionskette (GUI – Logik – Persistenz)
- Agile Entwicklungs- und Deployment-Ansätze (Continuous …)
  - Vermeidung monolithischer Ansätze
  - Leichte Ersetzbarkeit von Microservices
  - Erfolg ist massiv abhängig von der fachl. Architektur
- Keinen Bezug zu einem fest geprägten Technologiestack – u.a. REST

Idee der Mashups

Mashup-Hype

„Eine neue Generation von Webapplikationen verändert die Softwareentwicklung und -anwendung grundlegend. Neue Entwicklungen, bekannt unter dem Begriff Enterprise Mashups, ermöglichen es Endnutzern, auf Basis existierender Webressourcen individuelle Applikationen innerhalb von Minuten zu erstellen."

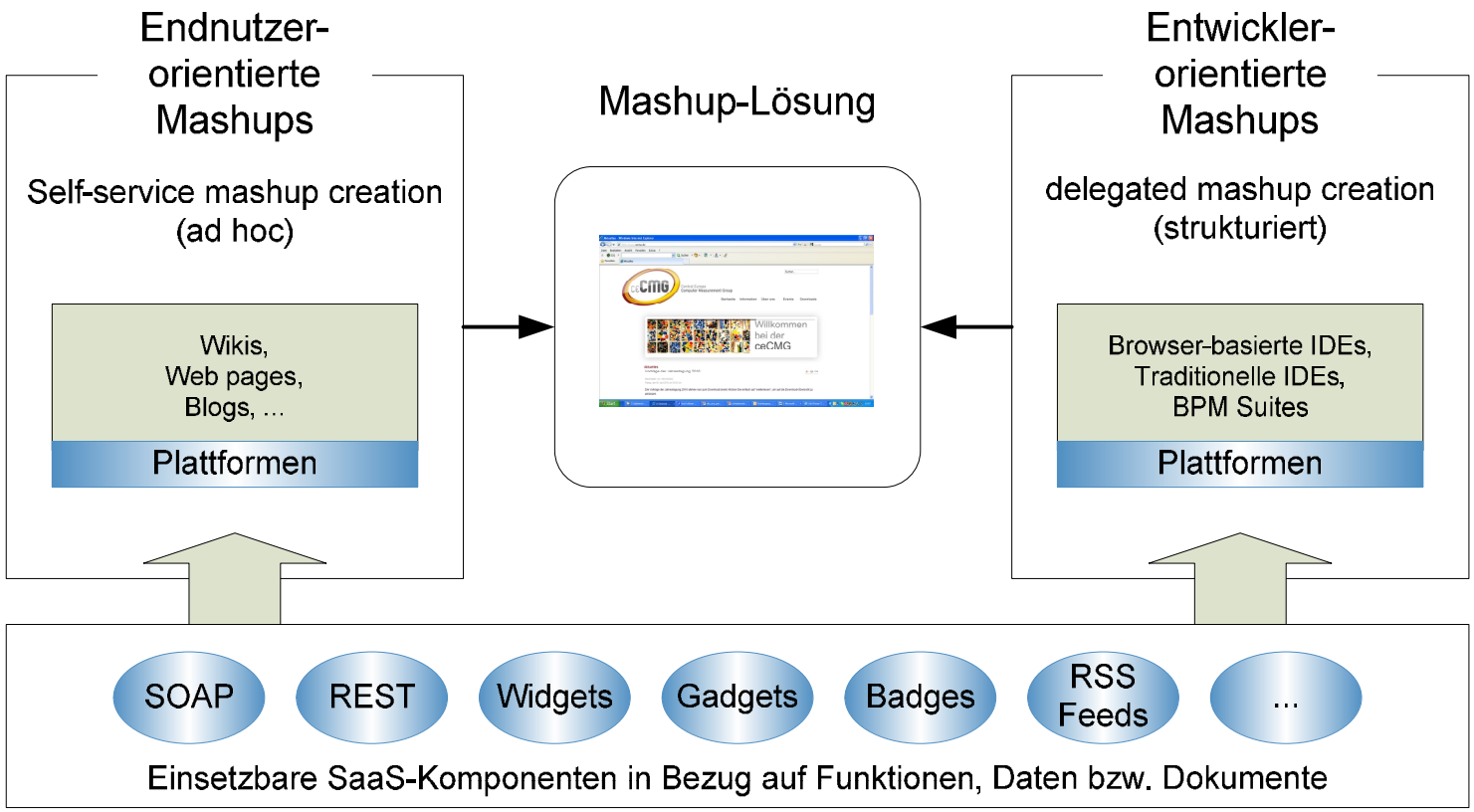
Quelle: Hoyer, V.; Stanoevska-Slabeva, K.: Enterprise Mashups – Neue Herausforderung für das Projektmanagement, HMD Praxis der Wirtschaftsinformatik, Heft 260, dpunkt.verlag, Heidelberg 2008

Herausforderung Mashup

„When mashups aggregate heterogeneous data sources it becomes difficult to convert, condense, and intelligibly communicate the summarization on a common web interface".

Quelle: Beemer, B.; Gregg, D.: Mashups: A Literature Review and Classification Framework, Future Internet 2009

Verwendungsbereiche von Mashups

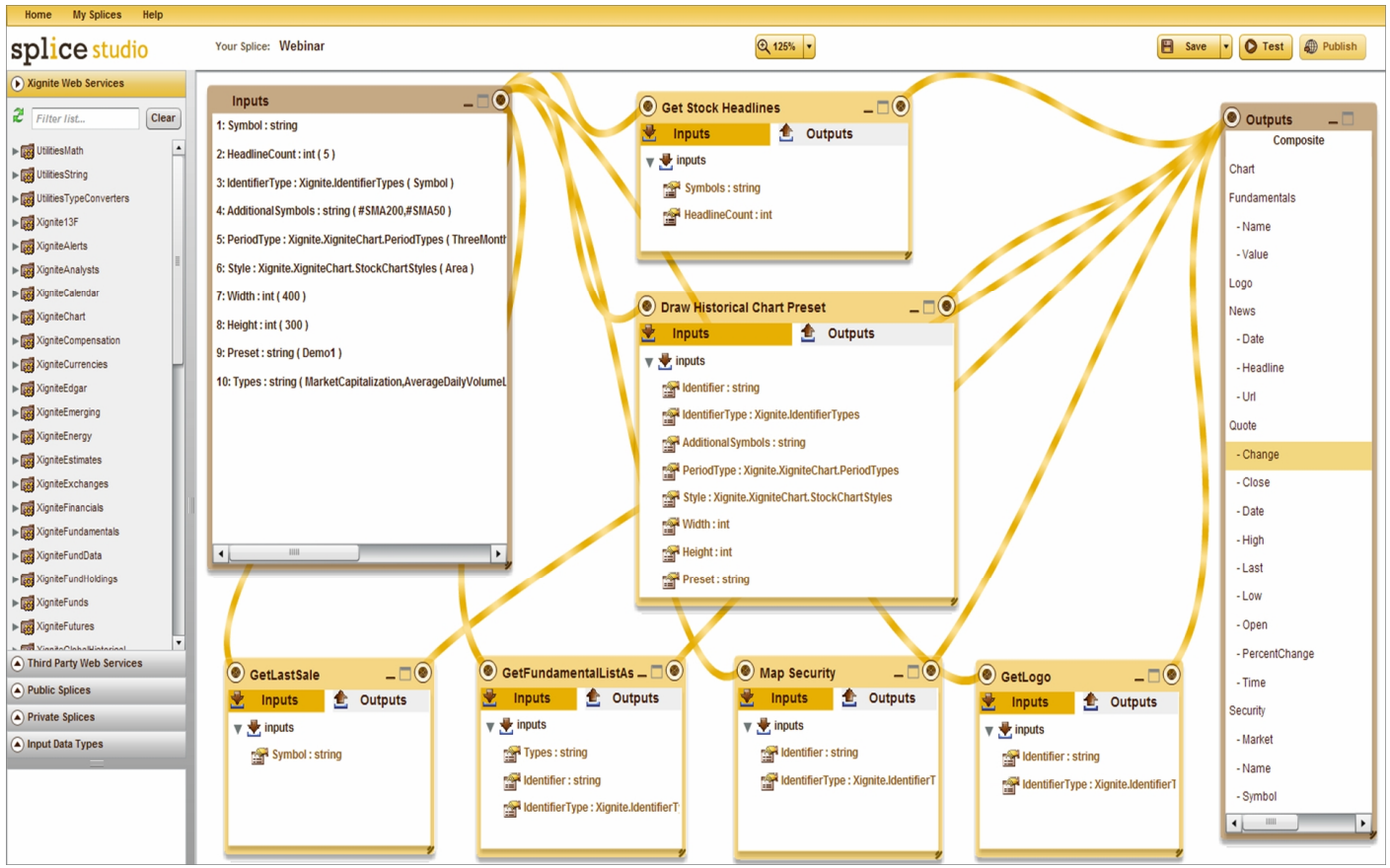


Typen von Mashups

- Endnutzerbezogene Mashups (Self-Service Mashups)
  - Meist browserbasierte Ansätze.
  - Integration von einfach strukturierten Informationsquellen.
  - Primäre lesende Zugriffe
- Fachspezifische Mashups (Business-driven Mashups)
  - Für „ad hoc“ benötigte IT-Lösungen bzw. Anpassungen.
  - In direkter Verantwortung des Informationsmanagements.
  - Bedarf standardisierter Fachfunktionen und Fachdaten.
- Entwicklerbezogene Mashups (Delegated Mashup Creation).
  - Enterprise Mashups als Ansatz zur agilen Entwicklung.
  - Ähnliche Ansätze im grafisch orientierten BPEL-Diskurs.
  - Gestaltung von Microservice-Architekturen

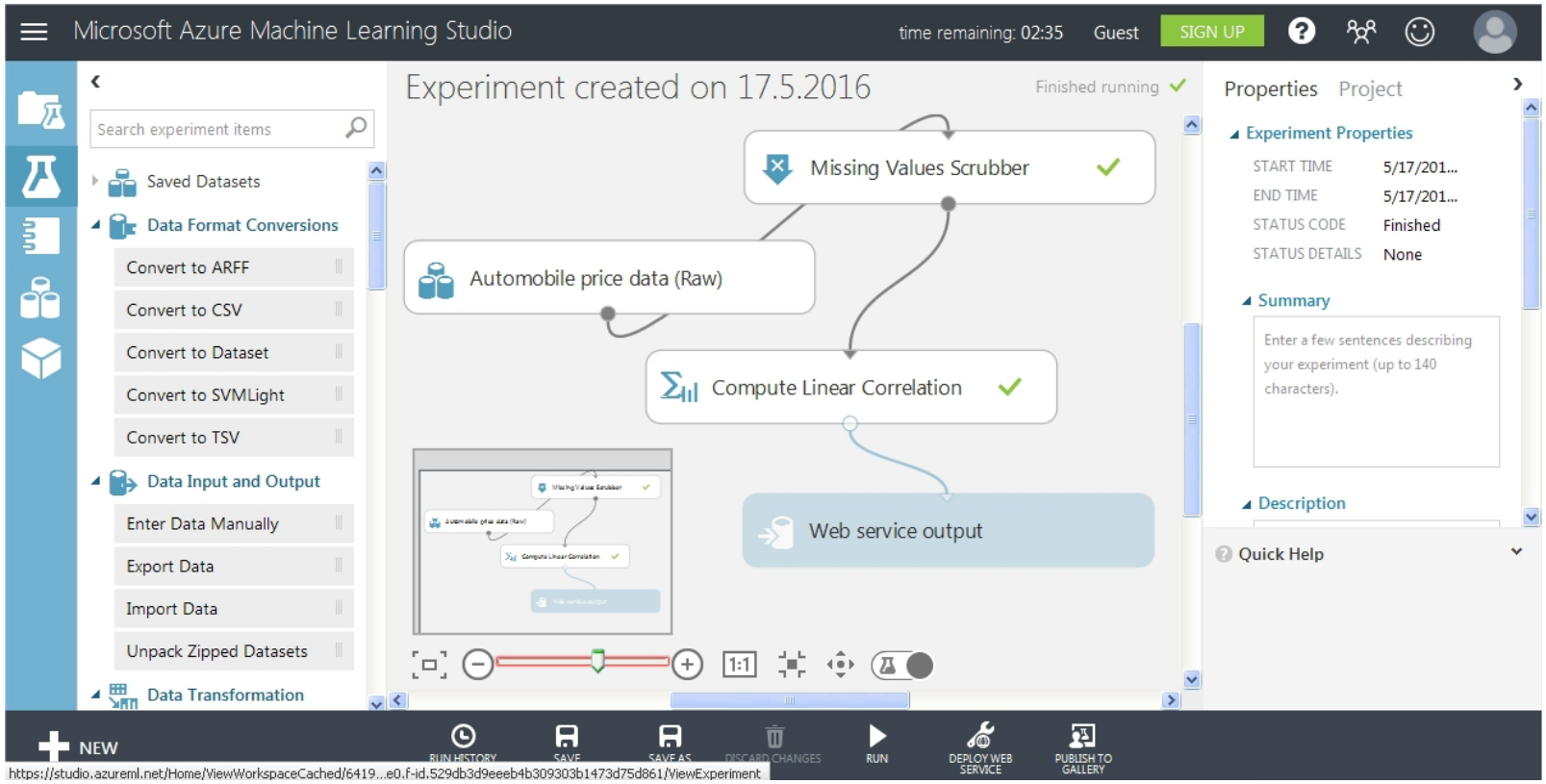
Praktische Beispiele

Beispiel splice studio



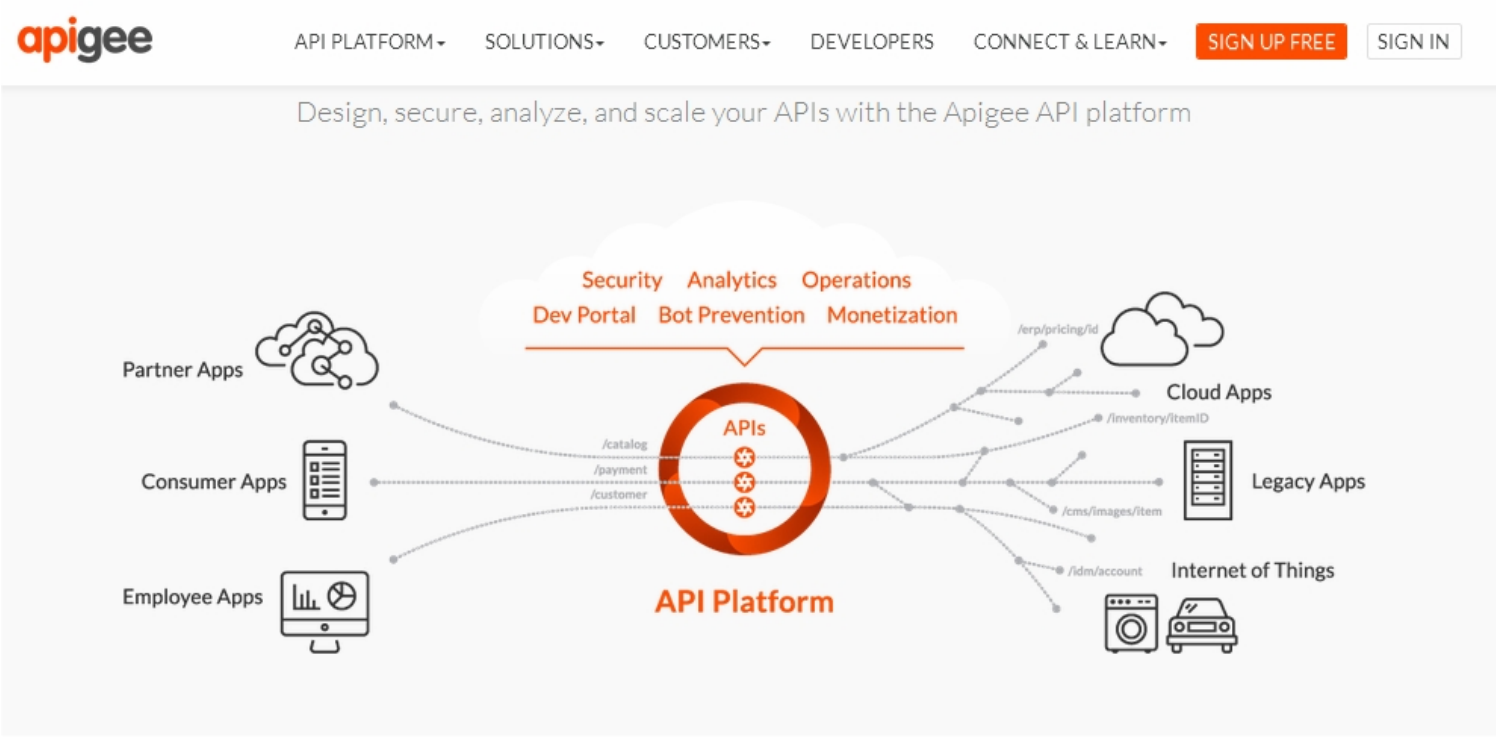
Quelle: http://xignite.web-services-blog.com/media/splice-web-services-mashup-big.png

Mashup-Lösung (MS Azure ML Studio)



In Quelle: https://studio.azureml.net, Erstellung und Abruf des Experiments, Abruf: Mai 2016

API-Framework (apigee)



Quelle: https://apigee.com/api-management/#/products?jump=slider_group_section1, Abruf: Oktober 2016

Literatur

Literaturhinweise

- Mayer, M. et al.: From SOA2WOA - Leitfaden, bitkom 2016, https://www.bitkom.org/Publikationen/2016/Leitfaden/From-SOA2WOA/160128-FromSOA2WOA-Leitfaden.pdf
- Lessen, T. v.; Lübke, D.; Nitzsche, J.: Geschäftsprozesse automatisieren mit BPEL, Januar 2011, dpunkt.verlag,
- Wolff, E.: Microservices – Grundlagen flexibler Softwarearchitekturen, 2016, dpunkt.verlag
- Josuttis, N.: SOA in der Praxis – Systemdesign für verteilte Geschäftsprozesse, 2009, dpunkt.verlag