



Vorlesung "Service and Cloud Computing"

6. SaaS – WS-Erweiterungen

Dr.-Ing. Iris Braun

- WS-* Erweiterungen
 - Adressierung
 - Zuverlässige Nachrichtenübertragung
 - Asynchrone Nachrichtenübertragung
 - Transaktionen
 - Festlegung von Richtlinien zur Dienstnutzung

Geschäftsprozess

BPEL, WS-CDL,
BPML, WSCI

Verzeichnis

UDDI

Dienstbeschreibung

WSDL

OWL-S, WSMO

Nachrichtenformat

XML

SOAP

Nachrichtenprotokoll

HTTP

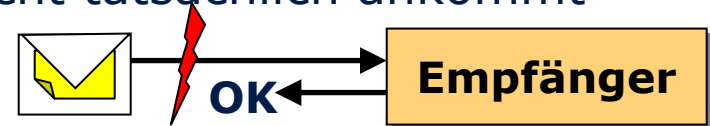
TCP/IP

Problem: Grundlegende WS-Spezifikationen nicht ausreichend, um alle Anforderungen der Geschäftsprozess-Schicht zu erfüllen!

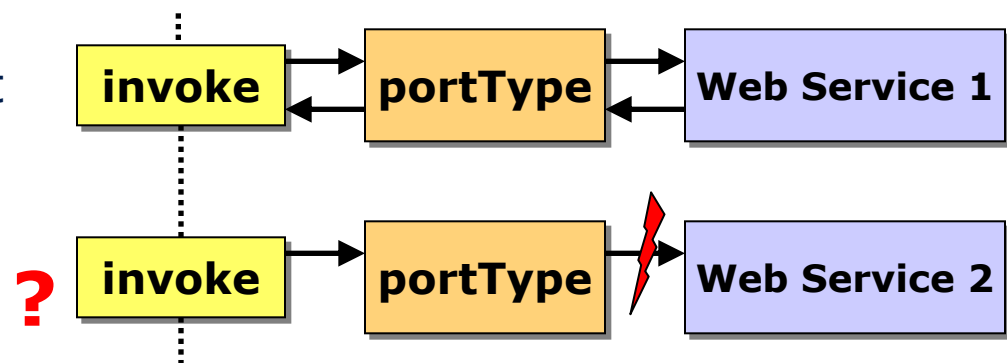
Wie kann man in einer SOAP-Nachricht angeben, von wem sie kommt, wer sie empfangen soll, und an wen die Antwort geschickt werden soll?

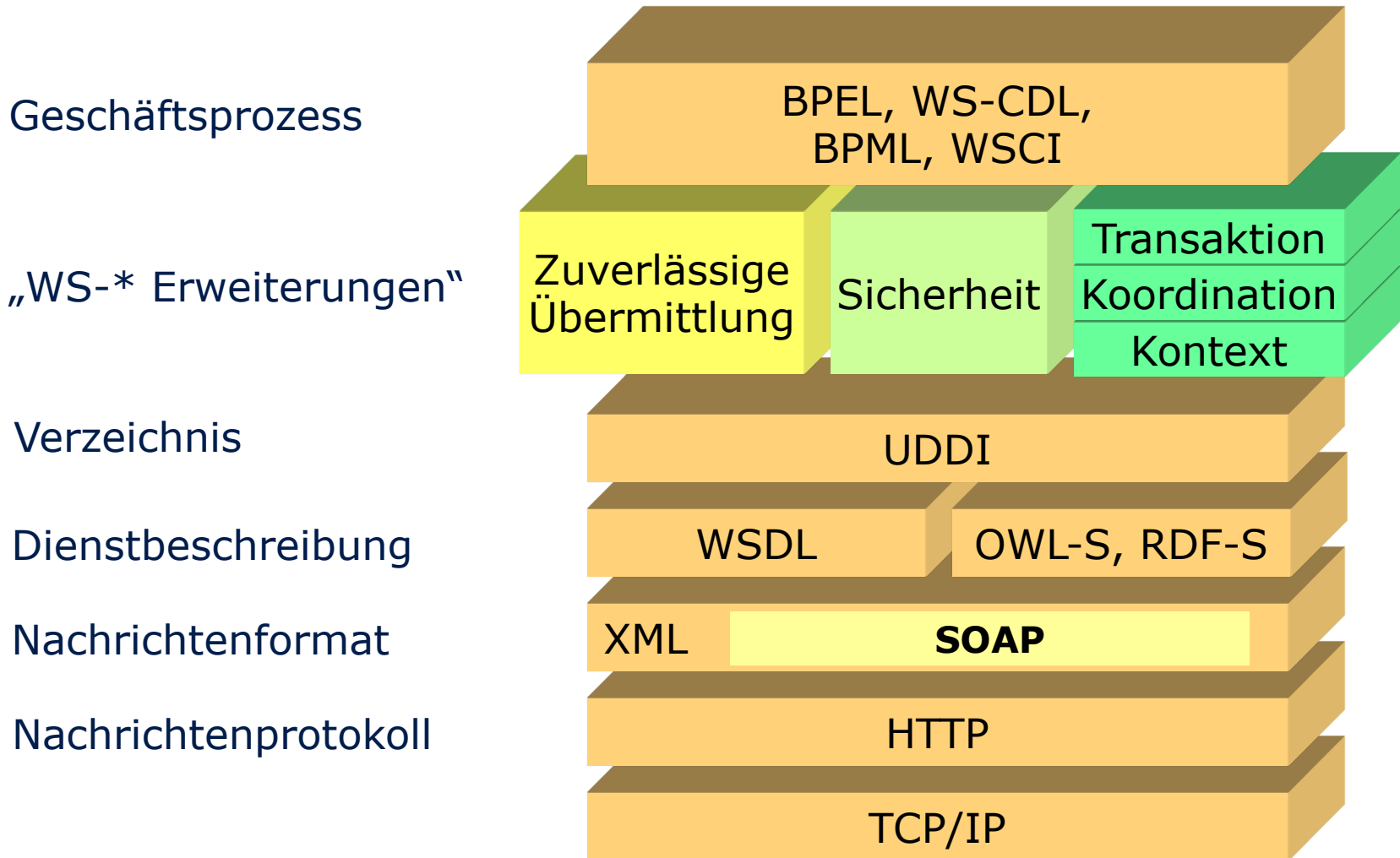


Wie wird sichergestellt, dass eine Nachricht tatsächlich ankommt und nur einmal verarbeitet wird?



Was passiert in einem Geschäftsprozess mit einzelnen Aktivitäten, die erfolgreich ausgeführt werden, während andere Aktivitäten fehlschlagen?



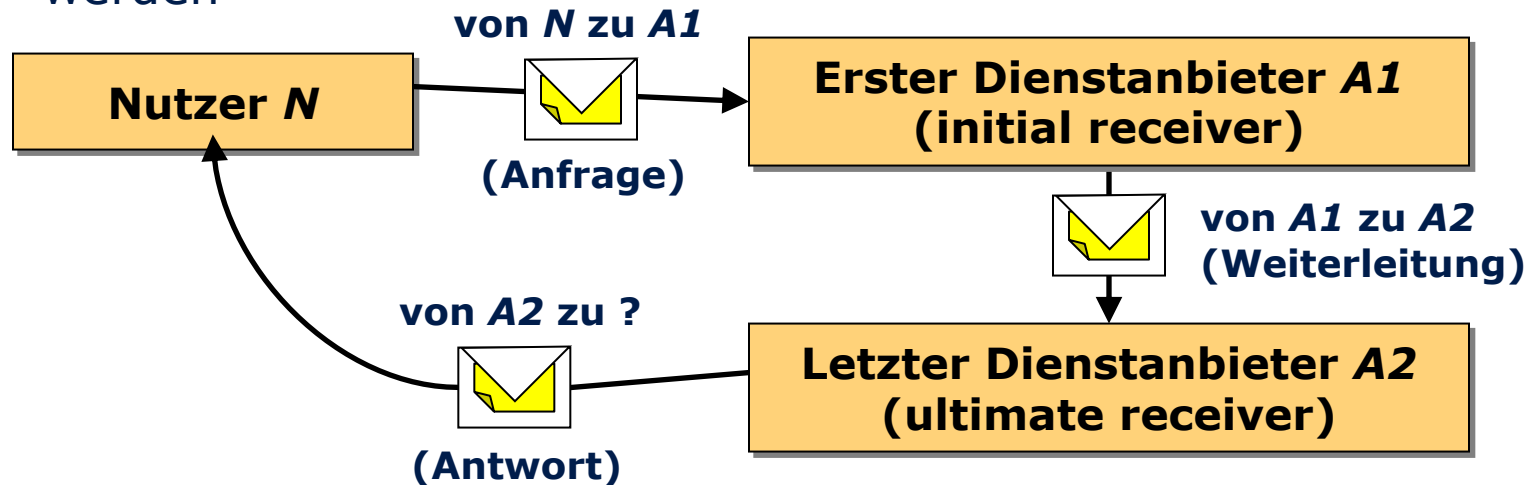


„WS-* Erweiterungen“



- Grundlagen für komplexe Geschäftsprozesse definieren
- Standardisierung, Unabhängigkeit von Transportprotokollen
- *Heute:* Adressierung, zuverlässige Nachrichtenübermittlung, Benachrichtigung, Transaktionen

- Szenario: Mehrere Web-Service-Anbieter arbeiten zusammen
 - Anfragekette zwischen Diensteanbietern, Antwort soll von letztem Anbieter *A2* in dieser Kette zum Nutzer *N* übermittelt werden



-> *A2* muss dafür die Adresse des ursprünglichen Dienstanwenders *N* kennen

- Adressierung auch hilfreich für zuverlässige Nachrichtenübertragung bei Problemen (Netzwerkfehler, Antwort-Verlust, ...)

- W3C-Standard, Version 1.0 (2006)
- Ersetzt frühere Spezifikationen WS-Referral und WS-Routing
- Transport-neutrale Methode zum Ansprechen von Objekten (Endpunkt-Referenz)
- Erweiterung für SOAP-Header, definiert
 - Endpunkt-Beschreibung (Zugriffspunkt eines Dienstes im Netzwerk, zusätzliche zur Interaktion nötige Metadaten)
 - Message Information Headers (Elemente zur Adressierung)
- Abstraktion von Transportprotokoll, erleichtert Implementierung für mehrere Transportprotokolle („multi-transport“)
- WS-Addressing Action ersetzt das Action-Attribut des SOAP-HTTP-Bindings (in SOAP 1.2 entfernt)
 - Ermöglicht Aufruf eines generischen Dienstes („Bank“) mit einer bestimmten Operation (Action, z.B. „Überweisen“)

```
<soap:Header>
```

```
  <wsa:ReplyTo>
```

```
    <wsa:Address>
```

```
      http://www.user.de/testuser/TestSearchRequest
```

```
    </wsa:Address>
```

```
  </wsa:ReplyTo>
```

```
  <wsa:To>
```

```
    http://api.holtin.com/hss
```

```
  </wsa:To>
```

```
  <wsa:Action>
```

```
    http://api.holtin.com/hss/search
```

```
  </wsa:Action>
```

```
</soap:Header>
```



Antwort-Adresse

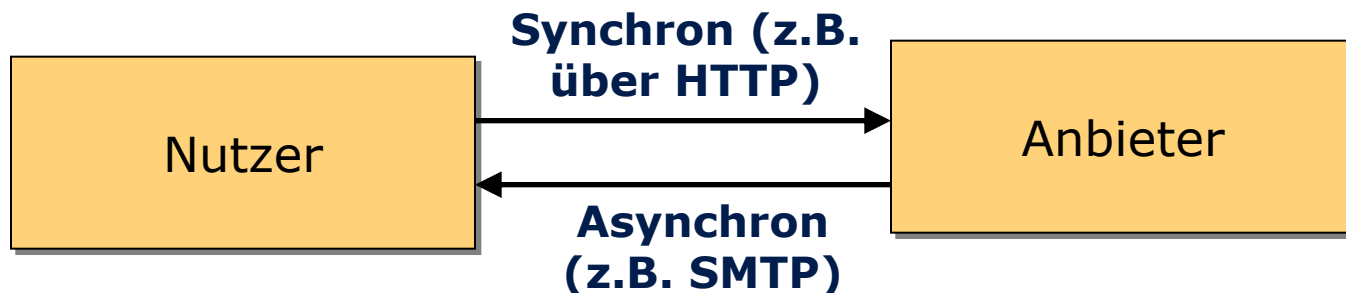


**Adresse des
Dienstanbieters**



**Auszuführende
Operation**

- Spezifikation 2004 von BEA Systems, Microsoft, IBM und Tibco an W3C übergeben, Version 1.2 (2009)
- Definiert
 - Nachrichtenidentifikation
 - Abstrakte Übertragungseigenschaften
 - Nachrichtenreferenzierung für MEP (Nachrichtenaustauschmuster, z.B. Callback, Broadcast)
- Ziel: Abstraktion der SOAP-Nachricht von Transportprotokollen
- Ermöglicht Referenz auf Nachrichten, Verwendung von verschiedenen Protokollen innerhalb eines MEP



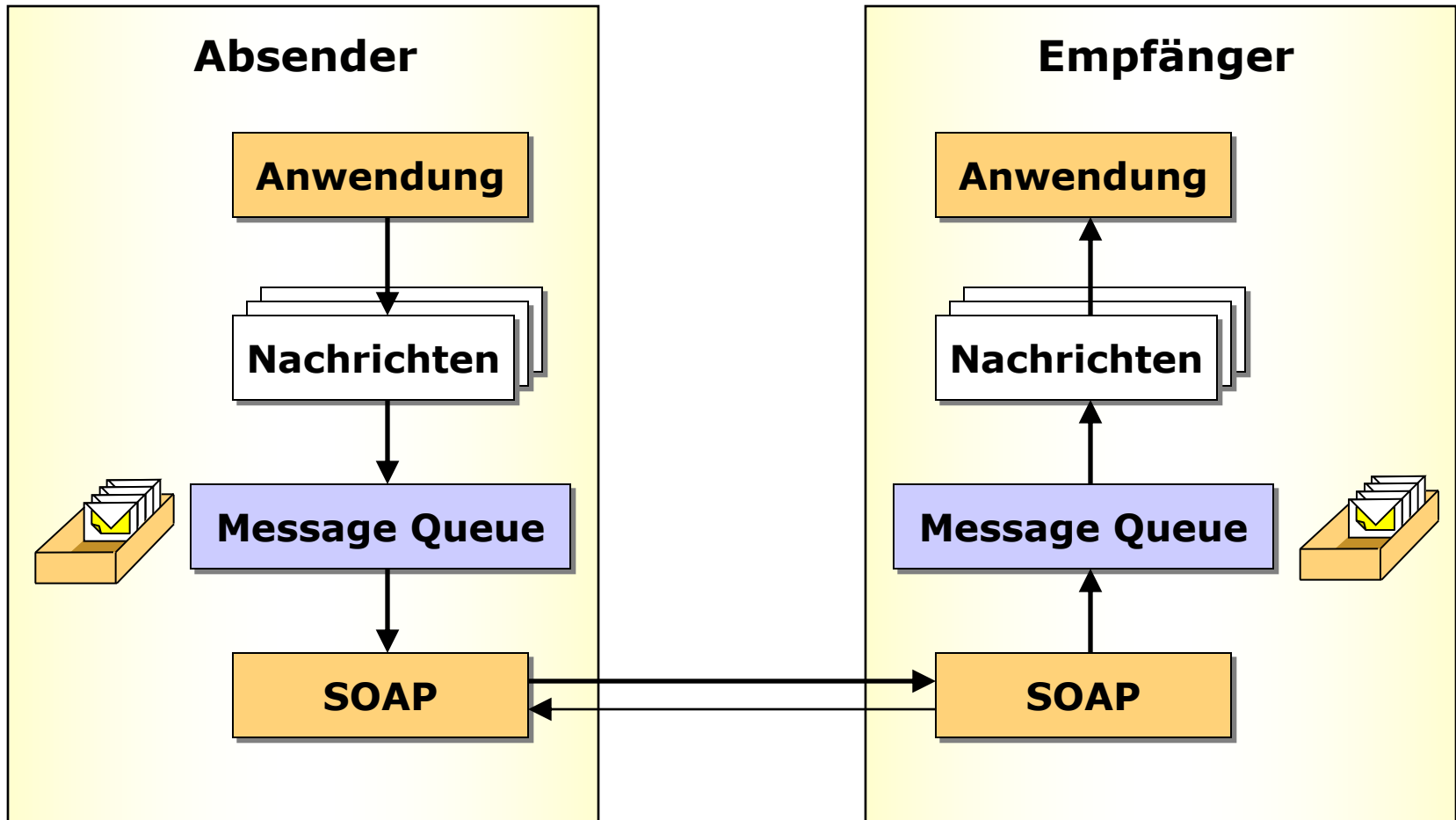
```
... xmlns:wsmid="http://www.w3.org/2004/04/ws-messagedelivery"  
<soap:Header>  
  <wsmid:MessageOriginator ... >  
    <wsdl:service ... > <wsdl:port ... >  
      <soapbind:address location="http://..."/>  
    </wsdl:port> </wsdl:service>  
  </wsmid:MessageOriginator>  
  <wsmid:MessageDestination>  
    <wsmid:uri>http://example.com/Client-A/reply</wsmid:uri>  
  </wsmid:MessageDestination>  
  <wsmid:MessageID>uuid:5a389ad2-22dd-11d1-aa77-002035b29092  
  </wsmid:MessageID>  
  <wsmid:MessageReference wsmid:reason="&wsmid;/reason/response">  
    uuid:58f202ac-22cf-11d1-b12d-002035b29092</wsmid:Message...>  
  <wsmid:OperationName>myRequestResponseOperation</wsmid:Op...>  
  ...
```

WS-Addressing	WS-MessageDelivery
From (EndPointRef)	MessageOriginator (wsdlLocation service uri serviceQname)
To (URI)	MessageDestination (wsdlLocation service uri serviceQname)
ReplyTo (EPR)	ReplyDestination (wsdlLocation service uri serviceQname)
FaultDestination (EPR)	FaultTo (wsdlLocation service uri serviceQname)
MessageID (URI)	MessageId (URI)
RelatesTo (URI)	MessageReference (URI)
Action (URI)	OperationName (NCName)

➔ **WS-Addressing gebräuchlicher, Basis weiterer Standards (WS-ReliableMessaging, WS-Policy, WS-Security,...)**

- Zuverlässige Übermittlung
 - Sehr wichtig für unternehmenskritische Anwendungen
 - Middleware-Transportprotokolle: zuverlässig (WebSphere MessageQueue (MQ), Java Message Service (JMS), Apache ActiveMQ, MQTT)

- Erweiterte MEPs
 - Ereignisbenachrichtigung
 - Publish-/Subscribe-Mechanismen
 - Techniken für mobile Anwendungen



Grundlegende Reliable-Messaging-Konzepte:

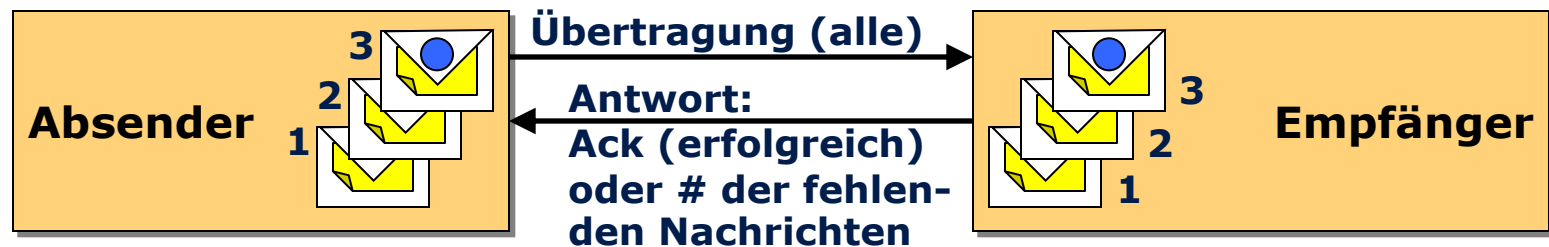
- Garantierte Nachrichtenübertragung
- Benachrichtigung über den Nachrichtenstatus
- Eliminierung von Duplikaten (doppelt empfangene Nachrichten)
- Sortierung der Nachrichten (Reihenfolge Sender = Empfänger)

Kardinalitäts-Garantien:

- Mindestens einmal (at least once)
- Genau einmal (exactly once)
- Höchstens einmal (at most once)

„Normales“ SOAP über HTTP: erfüllt keines dieser Konzepte!

- Software-Infrastruktur, installiert sowohl bei Nachrichten-Sender als auch bei Nachrichten-Empfänger
- Behandelt Fehler in der Nachrichtenübertragung auf einem möglicherweise unzuverlässigen Netzwerk
- Üblich: symmetrische (Sender- = Empfänger-)Implementierung
- Grundidee:
 - Sequenz-ID für Gruppe zusammenhängender Nachrichten
 - Nachrichten-ID für einzelne Nachrichten dieser Gruppe
 - Absender markiert letzte Nachricht
 - Empfänger bestätigt



- Losere Kopplung
- Trennung der Verantwortlichkeiten („separation of concerns“) führt zu einfacherem Anwendungscode
 - Reliable Messaging in der Infrastruktur, Anwendungscode enthält nur noch Geschäftslogik
- Einfachere Administration
 - Wartung von integrierten Anwendungen: während ein Teil A offline ist, läuft Teil B einer anderen Organisation weiterhin
- Ermöglicht asynchrone Kommunikation
- Sichert Unabhängigkeit vom Transportprotokoll
- Reduziert Komplexität bei Nutzung unzuverlässiger Netzwerke

- **WS-Reliability**: Spezifikation von Oracle, Sun u.a., 2004 bei OASIS eingereicht, aktuelle Version 1.1
- **WS-ReliableMessaging**: Entwicklung von Microsoft, IBM, BEA u.a., wurde 2005 ebenfalls zur Standardisierung an OASIS übergeben, aktuelle Version 1.2
- Sehr ähnliche Standards, aber nicht zueinander kompatibel

Unterstützte Anforderungen (WS-R und WS-RM):

- Mindestens einmalige Übertragung
- Höchstens einmalige Übertragung
- Eliminierung von Duplikaten \Rightarrow genau einmalige Übertragung
- Garantierte Nachrichtenreihenfolge

Feature	WS-ReliableMessaging	WS-Reliability
Garantierte Übertragung, Duplikateliminierung, Sortierung, SOAP-Binding	√	√
Nachrichten-Status	√	×
HTTP-Binding	√	×
SMTP-Binding	×	×
Fehlerbehandlung	√	√ (SOAP-Faults)
Sicherheit	√ über WS-Security und andere Spezifikationen	√ über WS-Security
Anforderungen (Policies)	√ WS-RM Policy Assertion, über WS-Policy	√ über WS-Policy

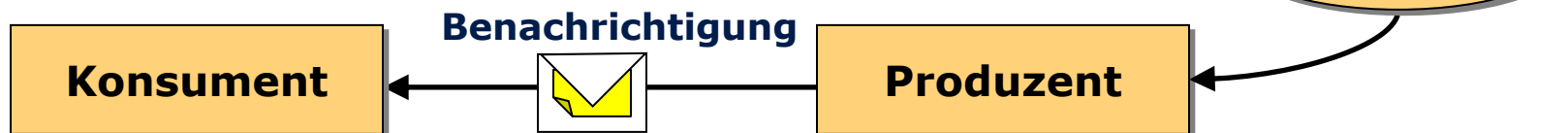
➔ **WS-RM hat breitere Industrieunterstützung (IBM, SAP, Microsoft, ...)**

Pub-Sub-Pattern

- Nachrichtenkonsument registriert sich (*Subscribe*)



- Nachrichtenproduzent sendet Nachrichten über Ereignisse an die registrierten Konsumenten (*Publish*)



2 wichtige Standards:

- **WS-Eventing**: BEA, IBM, MS, Sun u.a., einfache Spezifikation
- **WS-Notification**: OASIS-Standard, aktuelle Version 1.3, von IBM, SAP, Tibco, HP u.a. entwickelt, deutlich komplexer

<SOAP:Header>

<wsa:Action>

<http://schemas.xmlsoap.org/ws/2004/01/eventing/Subscribe>

</wsa:Action>

<wsa:ReplyTo>

<wsa:Address>

<http://api.reisebuero.com/HotelStatusSink>

</wsa:Address>

</wsa:ReplyTo>

<wsa:To>

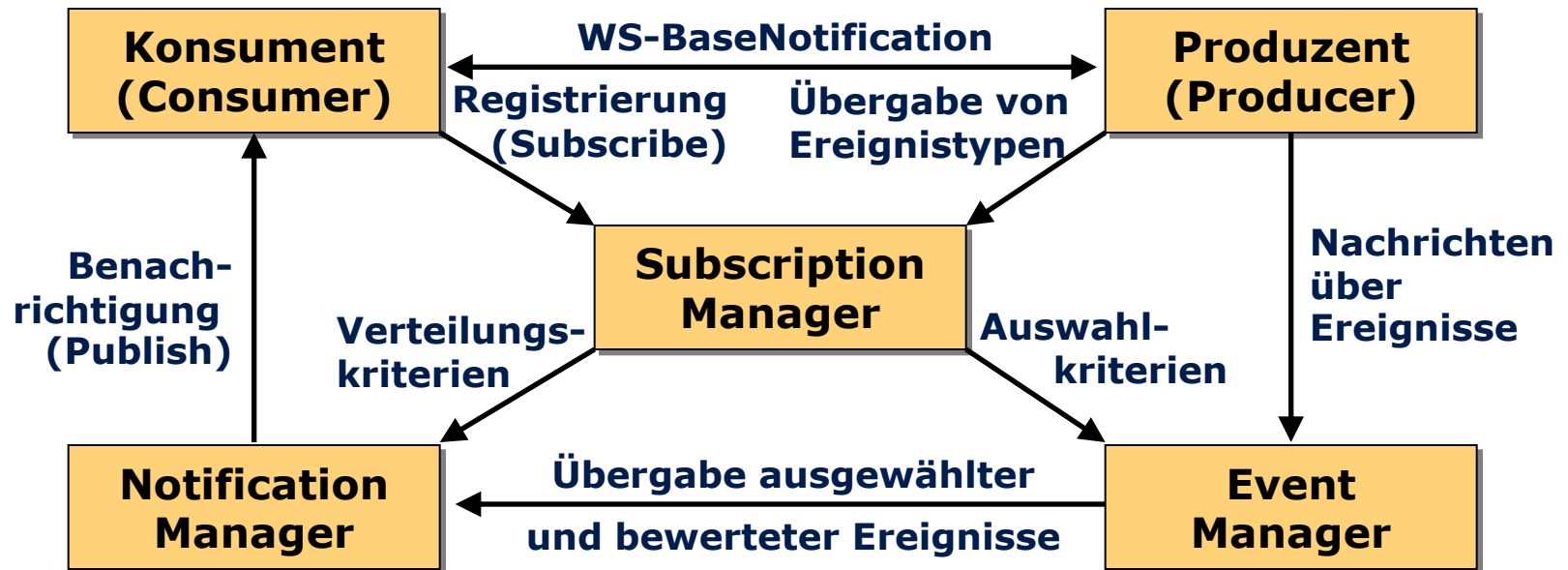
<http://api.holtin.com/HotelStatusSource>

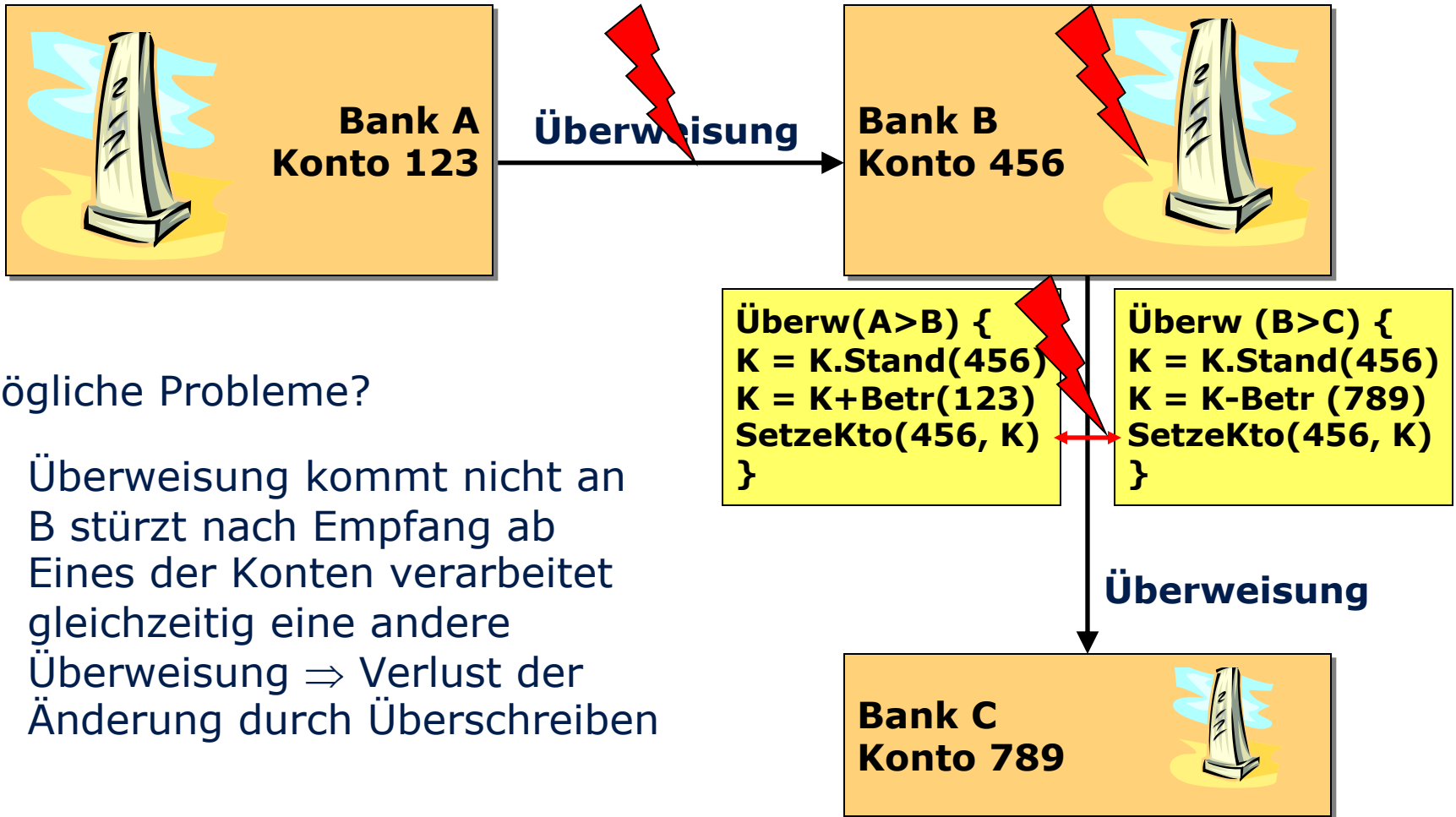
</wsa:To>

</SOAP:Header>

Familie von 3 Spezifikationen:

- WS-BaseNotification: Producer & Consumer, Point-to-Point-Not.
- WS-Topics: Kategorisierung und Organisation von Informationen
- WS-BrokeredNotification: Schnittstelle des „Managers“



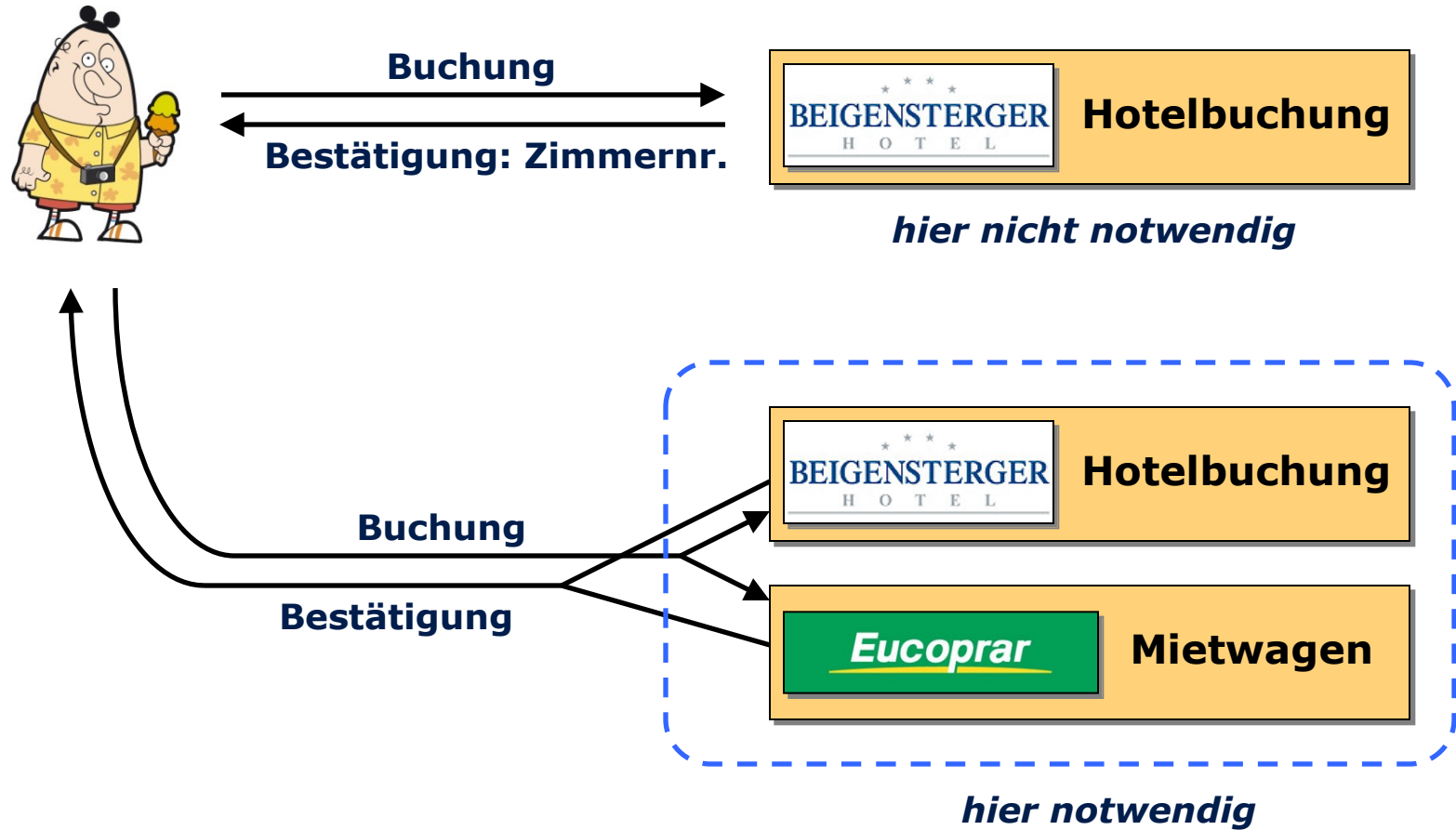


Mögliche Probleme?

- Überweisung kommt nicht an
- B stürzt nach Empfang ab
- Eines der Konten verarbeitet gleichzeitig eine andere Überweisung \Rightarrow Verlust der Änderung durch Überschreiben

- Nicht nur typisches Banküberweisungsszenario, z.B. auch:
 - Terminvereinbarungen
 - Reiseplanung
- Eigenschaften von Transaktionen:
 - Beteiligung mehrerer Teilnehmer
 - Vorher festgelegte Prozedur für jeden Teilnehmer
 - Transaktion hat bestimmtes Ergebnis
 - Ende: Teilnehmer kennen Ergebnis, stimmen diesem ggf. zu
- Definition: Eine Transaktion ist eine Menge von Arbeitsschritten mit einem konsistenten Ergebnis.

Muss ein WS Transaktionen unterstützen?



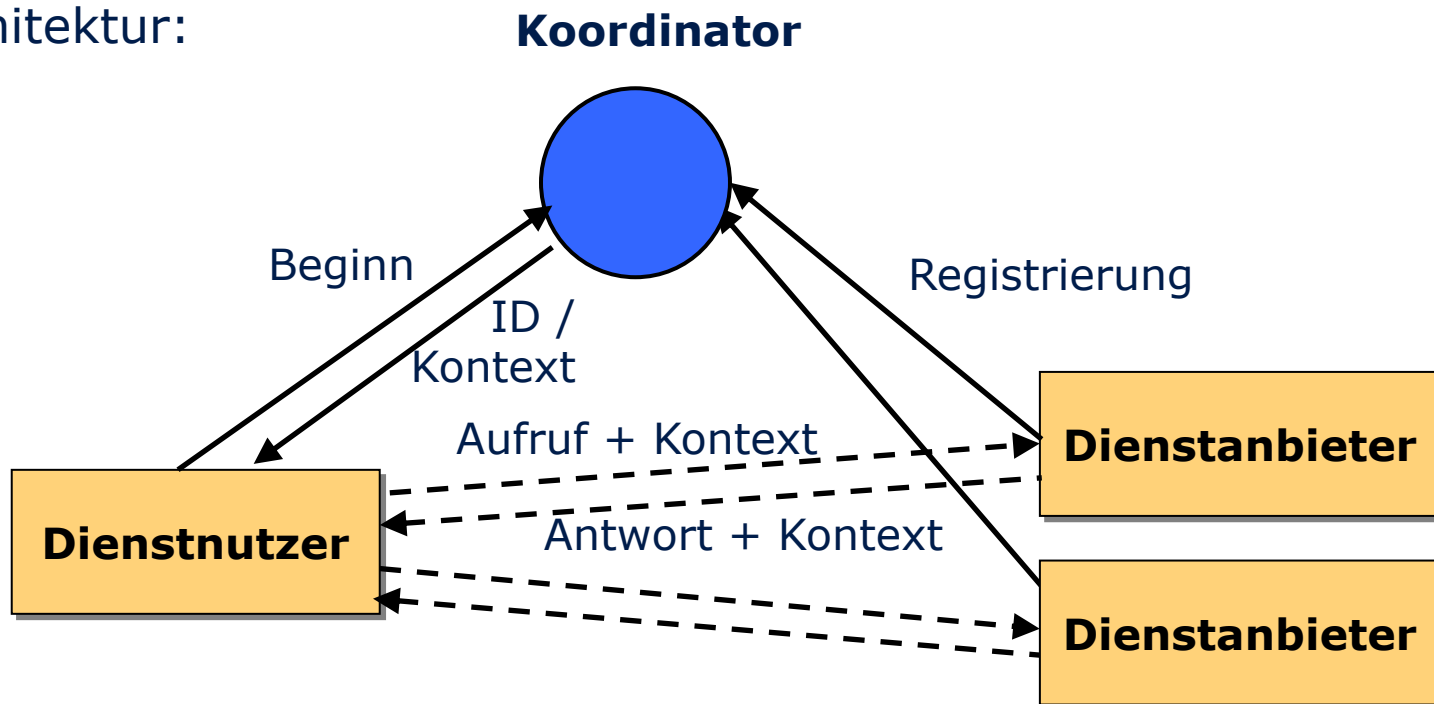
Ziel:

- Interoperabilität zwischen Ausführungsumgebungen
 - Verteilte Transaktionsunterstützung
 - *Keine Implementierung* des Transaktionsparadigmas bei den einzelnen Teilnehmern
- ⇒ Nur im Umfeld von Technologien nützlich, die die Anforderungen der Protokolle implementieren

Einfluss von Transportprotokollen:

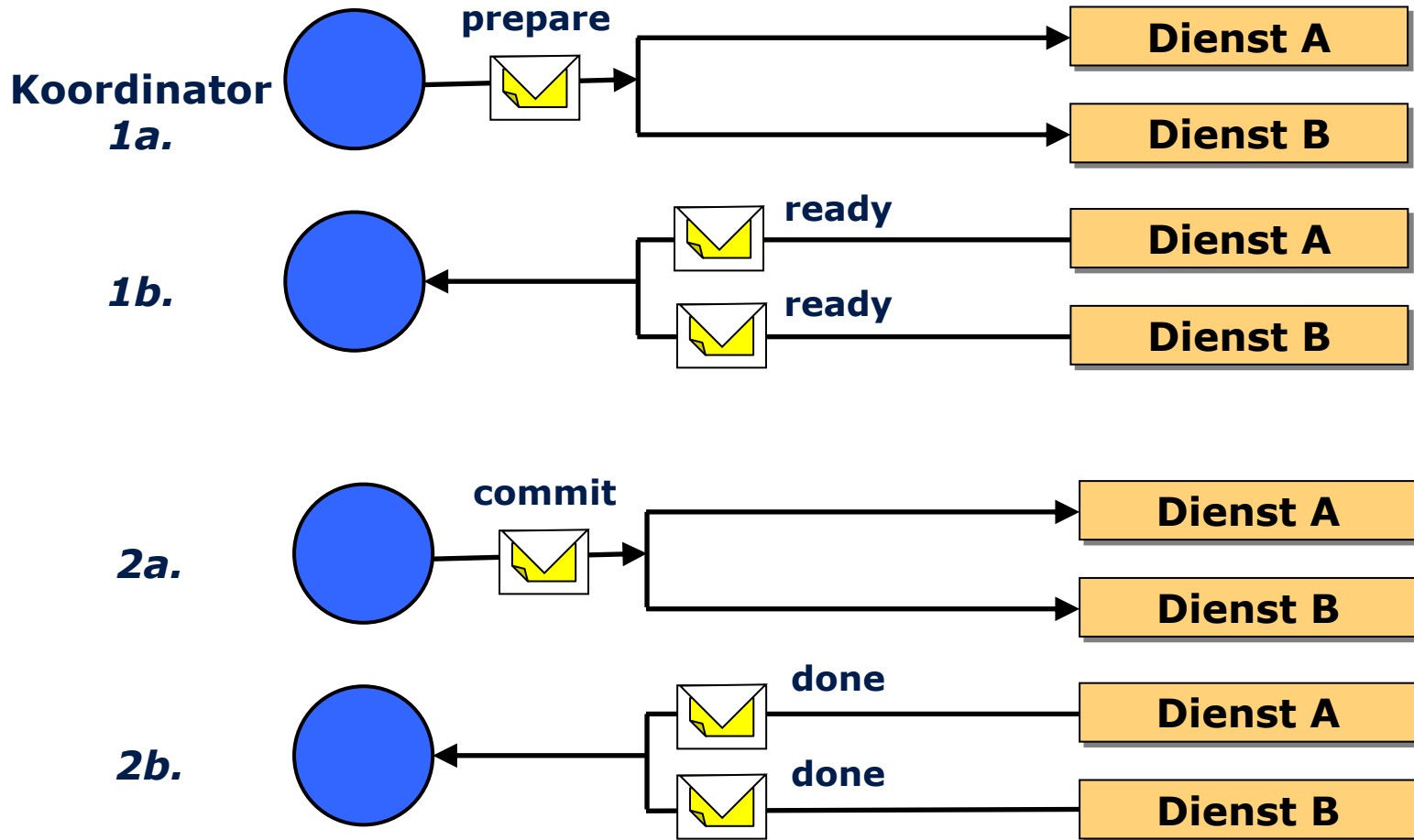
- HTTP: Nur einmaliger Request/Response-Nachrichtenaustausch
- ⇒ Transaktionskontext muss bei jedem Request gesendet werden
- ⇒ Kontext-Management für asynchrone Protokolle notwendig

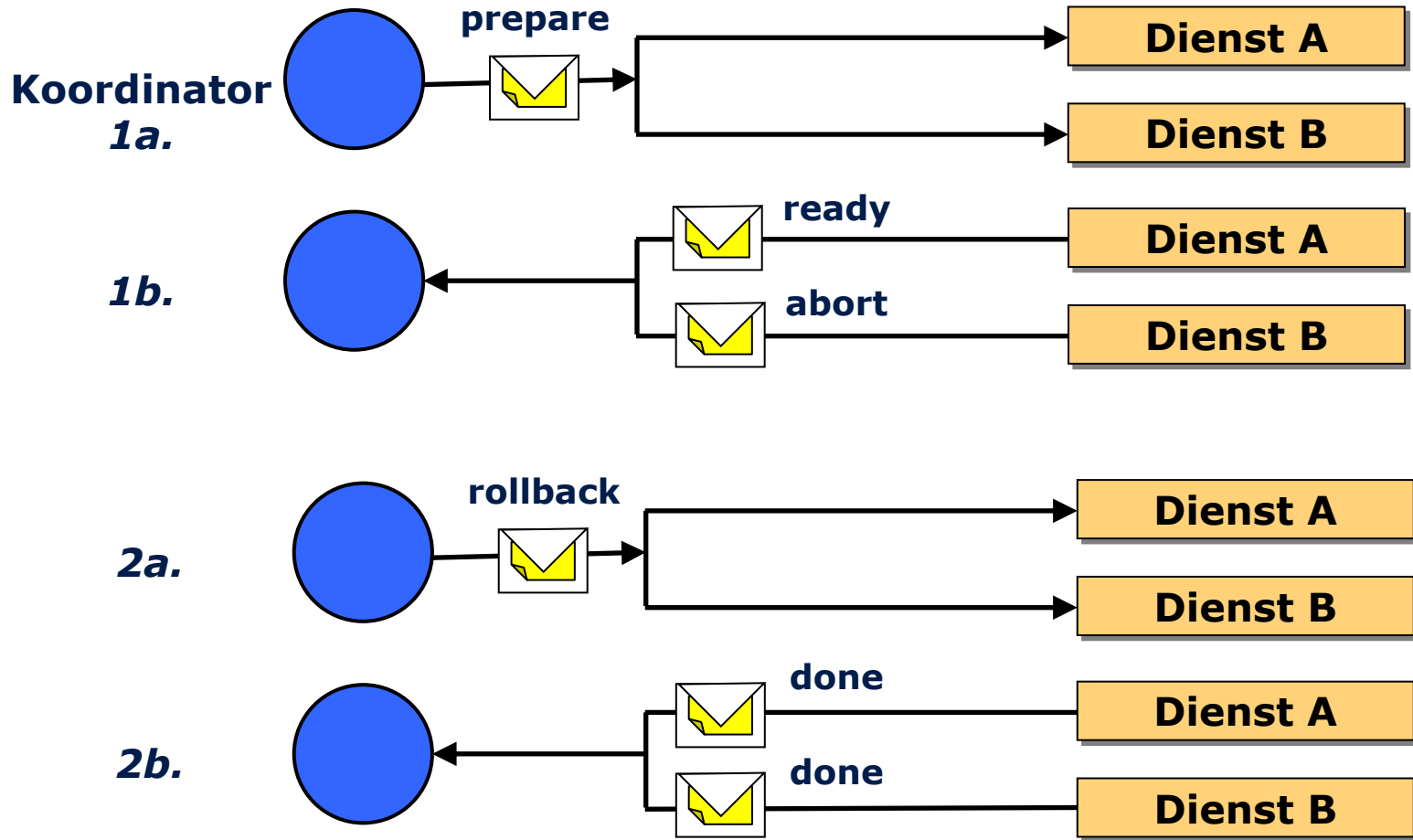
- Ziel: alle beteiligten Dienste verwenden gleichen Transaktions-Kontext
- Zentraler WS-Transaktionskoordinator: generisch
 - Verwendung verschiedener Protokoll- und Kontexttypen
- Architektur:



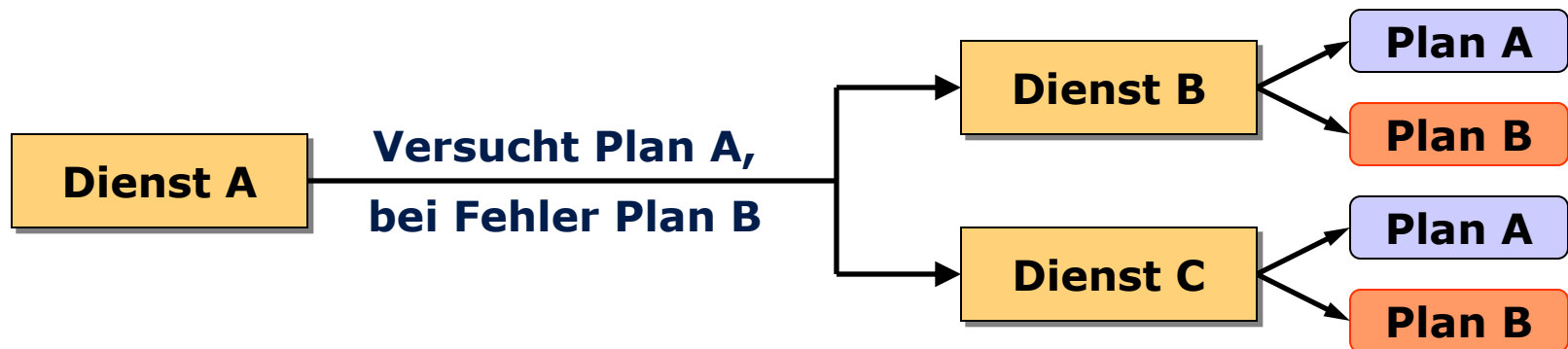
- Verschiedene Anforderungen und Umgebungen
⇒ Unterschiedliche Protokolle
- Für WS-Transaktionen entwickelte Protokolle:
 - **Kontextmanagement:** generisch, trennt Kontext von Transaktionsprotokollen
 - **Atomare Transaktionen:** klassisches 2-Phasen-Commit-Protokoll, stellt ACID-Eigenschaften sicher
 - Beispiel: WS-AtomicTransactions
 - **Kompensationsbasierte / langlebige Transaktionen:** ermöglichen zwischenzeitliche Commits, die über Kompensation anstatt Rollback rückgängig gemacht werden können
 - Beispiel: WS-BusinessActivity
 - BPM: Überbrückung anderer Protokolle für Transaktions-Koordination in Geschäftsprozessen
 - Bsp.: Business Process in WS-TransactionManagement

- **Atomarität**: Alles-oder-Nichts-Prinzip, eine Menge aller Arbeitsschritte einer Transaktion wird entweder vollständig oder gar nicht ausgeführt
- **Konsistenz** (**C**onsistency): Informationssystem wird von einer Transaktion von einem in einen anderen konsistenten Zustand überführt
- **Isolation**: Durchführung einer Transaktion erfolgt isoliert von anderen gleichzeitig ausgeführten Transaktionen (T. ist völlig unbeeinflusst von anderen parallel arbeitenden T.s)
- **Persistenz** (**D**urability): bei Ende der Transaktion dauerhafte Speicherung des Transaktionsergebnisses, z.B. auf Festplatte





- Im täglichen Leben oft anzutreffen (Beispiel: Rückgabe eines Produktes, Kompensation: Geldrückgabe und Hinzufügen des Produktes zum Geschäftsbestand)
- Nachteile:
 - Rekursive Kompensation nicht möglich
 - Fehler bei der Kompensation – Fehlerbehandlung?
 - Kompensation für manche Operationen nicht möglich
 - Beispiel: Raketenabschuss, Ausdruck eines Dokuments



- Aufgabe: Beschreibung von Transaktionsvoraussetzungen
- Ziel: Interoperabilität
- Antworten auf Fragen wie
 - Setzt ein Web Service immer einen Transaktionskontext voraus?
 - Wird ein Transaktionskontext optional unterstützt?
 - Werden überhaupt Transaktionen unterstützt?
- Beispiel: WS-AtomicTransaction-Protokoll muss vom Dienstonutzer unterstützt werden:

<wsp:SpecVersion

wsp:URI="http://schemas.xmlsoap.org/ws/2003/09/wsat"

wsp:Usage="wsp:Required"/>

- **WS-Transactions:** BEA, IBM, Microsoft (2002)
 - WS-AtomicTransactions (WS-AT)
 - WS-BusinessActivity (WS-BA)
 - WS-Coordination (WS-C)
- **WS-Composite Application Framework (WS-CAF):** Oracle, Sun, Fujitsu (2005)
 - WS-Context (WS-CTX)
 - WS-CoordinationFramework (WS-CF)
 - WS-TransactionManagement (WS-TXM)
- *WS-C* und *WS-CF*: Koordinationsframeworks, die *WS-AT* und *WS-BA* unterstützen
 - *WS-CF* unterstützt zusätzlich die *WS-TXM*-Protokolle
 - *WS-C* dient auch zur Kontextverwaltung

- Web-Service-Konversation: Abfolge des Nachrichtenaustauschs zwischen Dienstnutzer und Anbieter eines bestimmten WS
- Web-Service-Koordinationsprotokoll: spezifiziert eine bestimmte Menge korrekter und akzeptierter WS-Konversationen
- Koordination: 3 Phasen
 - Aktivierung (Erstellung des Transaktionskontextes)
 - Registrierung (Web Services bestätigen ihre Teilnahme an der Transaktion)
 - Koordination (der Ergebnisse der teilnehmenden WS)

Geeignet für

- Kurzzeitige Transaktionen
- Transaktionen innerhalb einer Anwendung

- WS-AT unterstützt 2 Versionen von 2PC (2-Phasen-Commit):
 - volatile (flüchtig, d.h. Operationen auf RAM) und
 - durable (persistente Speicherung, z.B. auf Festplatte)
- Ziel dieser Unterscheidung: Koordination von Operationen auf flüchtigem Speicher vor persistenten Operationen

Geeignet für

- Langandauernde, verteilte Anwendungen, die viele Ressourcen und atomare Transaktionen verwenden könnten
- Anwendungen, die bei Bedarf die Transaktion auch ohne Koordinator beenden müssen
- Erlaubt sowohl ACID-Transaktionen als auch Kompensation
- Definition von Business-Logik für Ausnahmefälle möglich
- Kann als Wrapper für existierende, proprietäre Workflow- und Business-Process-Systeme dienen ⇒ Interoperabilität

- Zuverlässige Übermittlung:
 - WS-Addressing, WS-MessageDelivery
 - WS-ReliableMessaging, WS-Reliability
- Benachrichtigung: WS-Eventing, WS-Notification
- Transaktion, Koordination, Kontext:
 - WS-Transactions (Coordination, AtomicTransaction, BA)
 - WS-CAF (Context, CoordinationFramework, TransactionM.)

