



Seminar: Service-orientierte Architektur (SOA)

Thema: WS-Coordination: Zielsetzung,
Lösung und Beispiel

Seminarleiter: Sebastian Iwanowski
Fachhochschule Wedel



Inhaltsangabe

- Einleitung
 - Was ist WS-Coordination?
 - Zielsetzung WS-Coordination
 - WS-Coordination – Kontext und Model
- Coordination service
- Komponenten WS-Coordination
 - Überblick WS-Coordination
 - CoordinationContext
 - Activation service
 - CreateCoordinationContext
 - CreateCoordinationContextResponse
 - Registration service
 - Register Message
 - RegistrationResponseMessage
 - Importieren einer activity
 - Coordination Faults



1.1 Was ist WS-Coordination?

- Spezifikation, welche im Rahmen des WS-* Workshopprozesses entwickelt wurde
- Momentaner Status: *working draft*
- erweiterbares Framework
- Framework hält Protokolle bereit, welche Aktionen einzelner Applikationen koordinieren



1.1 Was ist WS-Coordination?

- ausserdem sind Standardmechanismen vorhanden, um WebServices zu entwickeln und zu registrieren
- entwickelt von der Arjuna Technologies, Ltd., BEA Systems, Hitachi, International Business Machines (IBM) Corporation und der Microsoft Corporation

1.2 Zielsetzung WS-Coordination

- Durch Nutzung von WebServices wird es Applikationen ermöglicht, verschiedenste Dienste plattformübergreifend via Internet zu nutzen
- dabei entstehen Einheiten verteilt ablaufender Verarbeitung, mit beliebig großer Teilnehmerzahl
- resultierende Einheiten sind häufig komplex in ihrer Struktur mit ebenso komplexen Verhältnissen zwischen ihren Teilnehmern
- Eine solche Einheit wird als *activity* bezeichnet



1.2 Zielsetzung WS-Coordination

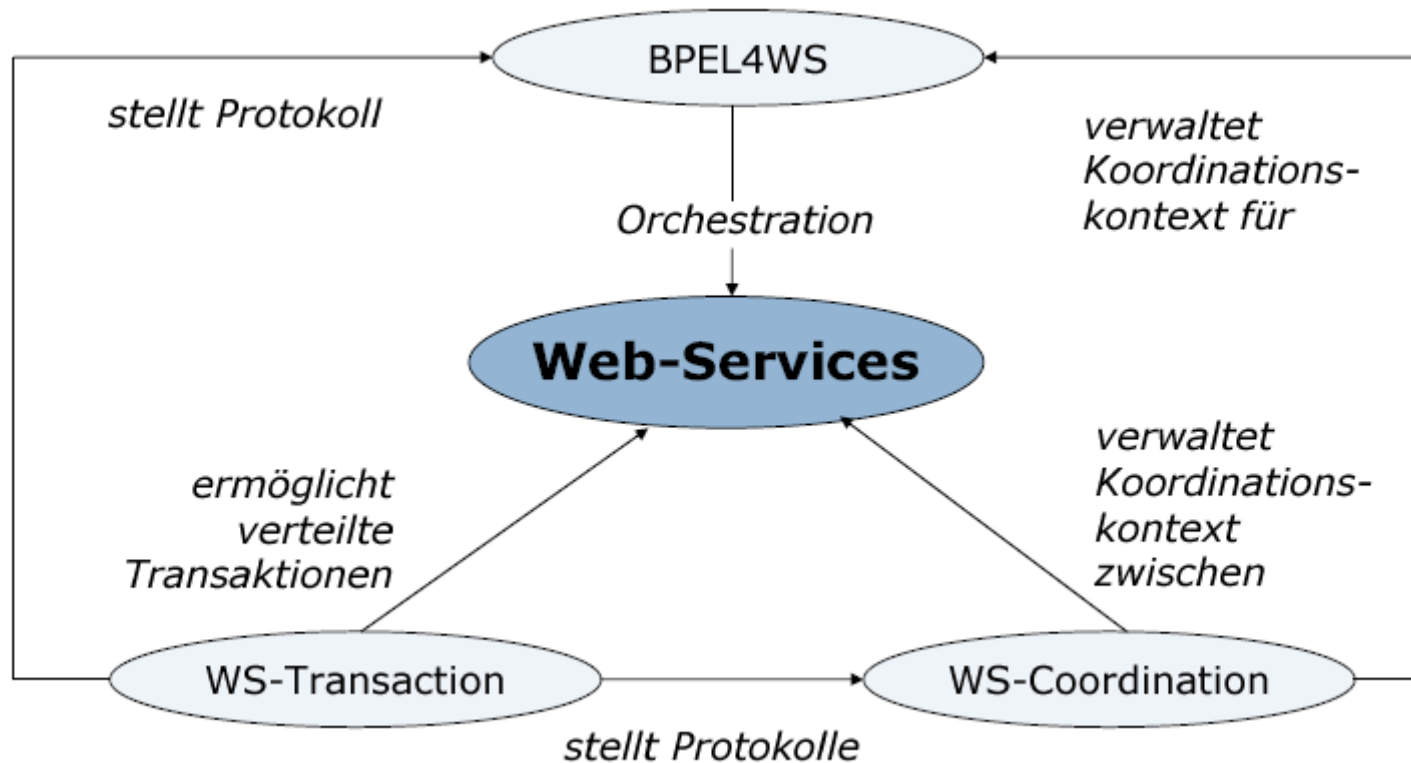
- WS-Coordination bildet ein erweiterbares Framework als Basis für Protokolle zur Sicherstellung konsistenter verteilter Zustandsübergänge im Rahmen einer *activity*
- Durchführung solcher Tätigkeiten nimmt häufig viel Zeit in Anspruch
- dies liegt einerseits an der Benutzerinteraktion, andererseits an der Latenz bei der Aufgabenbearbeitung



1.2 Zielsetzung WS-Coordination

- Die WS-Coordination-Spezifikation definiert daher dieses Framework, um die bestehende Komplexität von WebServices zu reduzieren
- Spezifikation kann als Grundlage für die Integration von Transaktionen in eine Web-Service-Umgebung dienen
- ist gleichzeitig ausreichend allgemein, um auch andere Formen der Koordination zu unterstützen, die eine wechselseitige Einigung aller Teilnehmer bezüglich des Ausgangs einer *activity* ermöglichen

1.3 WS-Coordination – Kontext und Model



2. Coordination Service

- WS-Coordination beschreibt Framework für einen *coordination service* (oder einen coordinator)
- Service besteht aus folgenden Komponenten:
 - Activation service
 - ermöglicht Initiator eine *activity* zu beginnen und veranlasst damit den *coordinator* eine neue Koordinationsinstanz und einen Koordinationskontext zu erzeugen
 - Registration service
 - ermöglicht es einem Web Service sich für die Teilnahme an einer laufenden *activity* zu registrieren
 - Satz von Koordinationsspezifischen Protokollen

2. Coordination Service

- Ein *coordination type* beinhaltet verschiedene *coordination protocols*
 - eine Menge wohldefinierter Nachrichtenformate und Regeln für den Austausch von konkreten Nachrichten zwischen dem Koordinator und den Teilnehmern einer *activity*
- Die Arbeitsweise des *coordination protocol services* wird in der Spezifikation des verwendeten *coordination type* (z.B. WS-AtomicTransaction) beschrieben

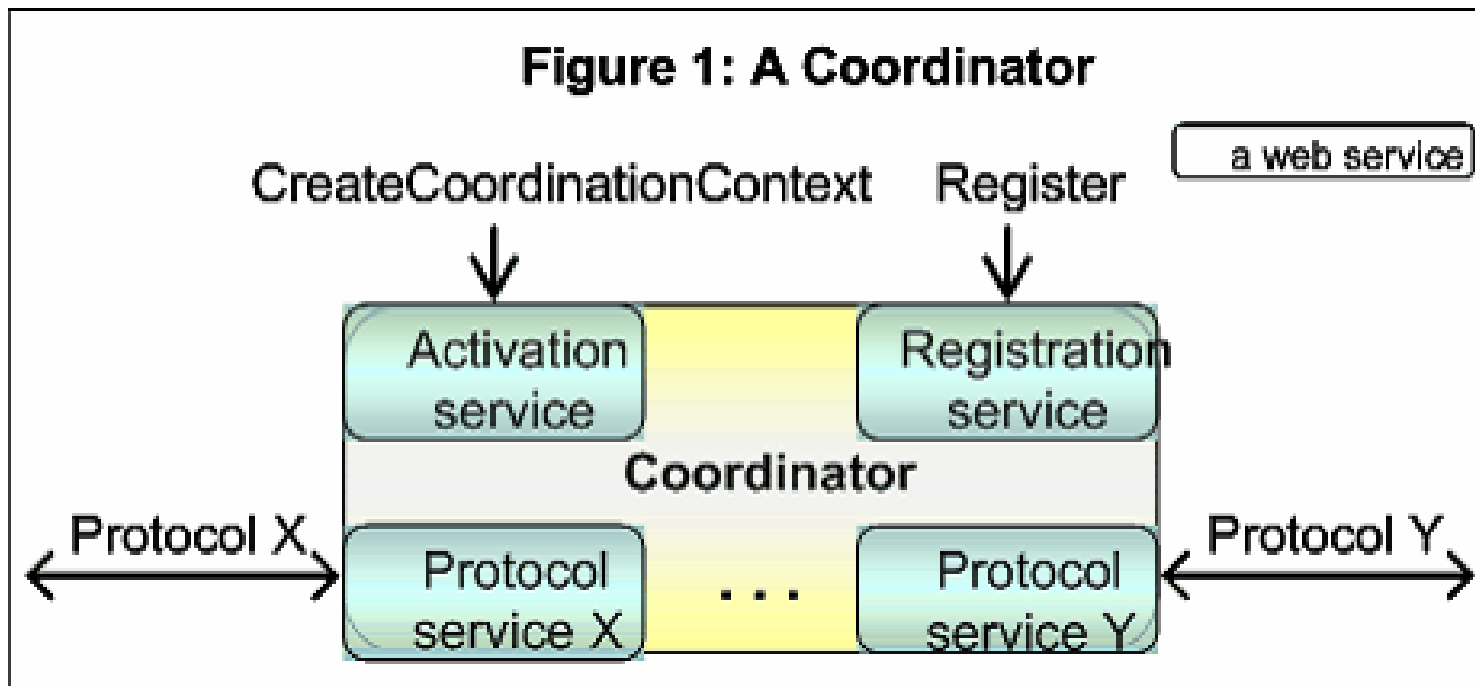
2. Coordination Service

- Applikationen benutzen den *activation service* um den `CoordinationContext` zu erstellen
- Der `CoordinationContext` enthält die benötigten Informationen, um sich in einer *activity* zu registrieren
- dabei wird das Koordinationsverhalten spezifiziert, welchem die Applikation folgen wird -> Wahl eines `coordination type`

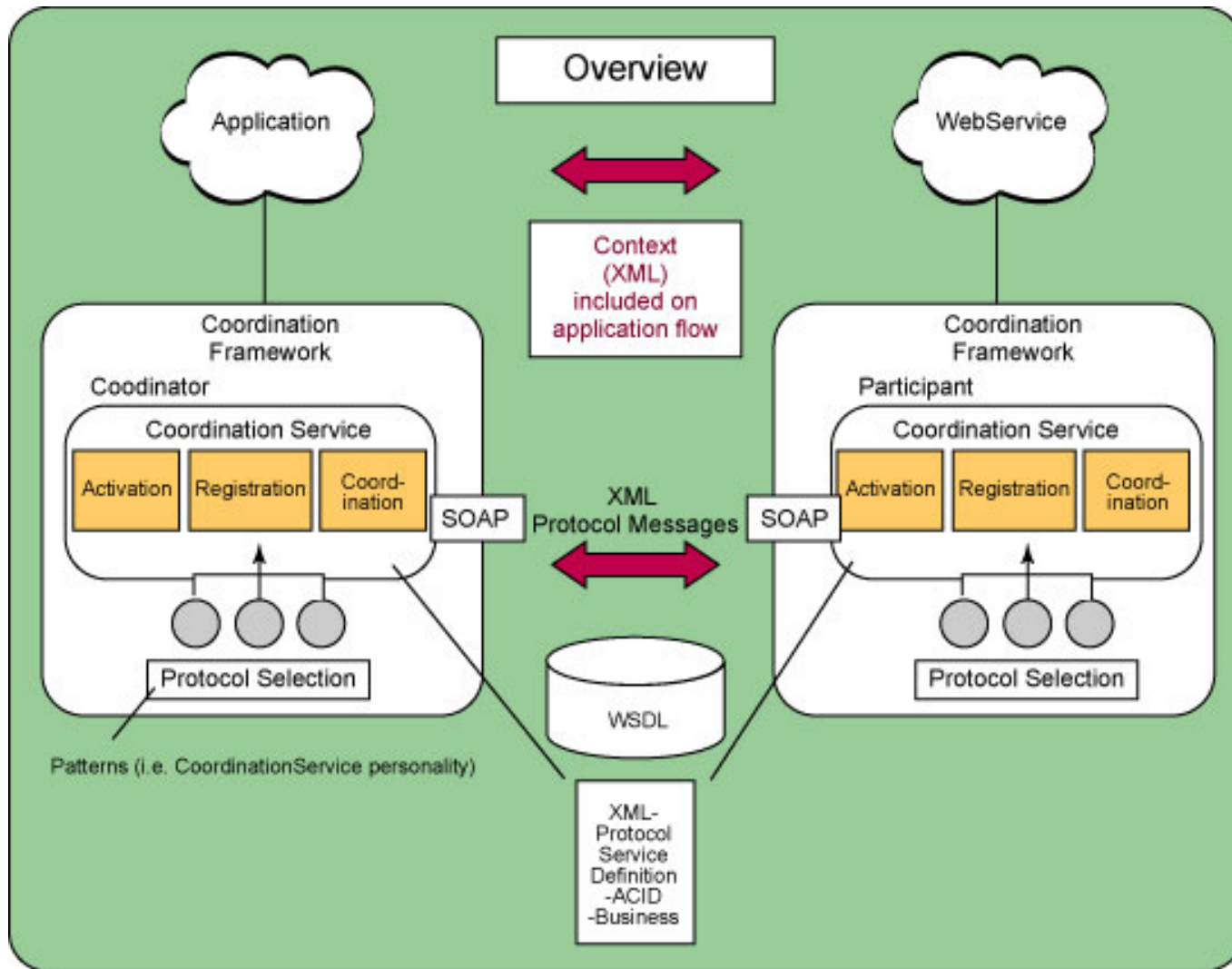
2. Coordination Service

- empfängt eine Applikation einen CoordinationContext, kann sie den Registrationservice der sendenden Applikation benutzen
- es steht ihr aber auch frei einen anderen Koordinator zu wählen (Importieren einer activity, vgl. Kap. 3.5)

2. Coordination Service



3.1 Überblick WS-Coordination



3.2 CoordinationContext

- Ein CoordinationContext wird von einer Applikation mittels der CreateCoordinationContext-Operation des *activation service* erzeugt
- der Kontext wird, eingebettet im Header-Element, mit allen SOAP-Nachrichten im Rahmen einer *activity* verschickt
- Rückblick *endpoint reference*: eine *endpoint reference* ermöglicht die Nutzung eines Web Service, indem sie folgende Informationen bereitstellt:
 - die URI eines Web Service
 - beliebige zusätzliche Daten, unverständlich für jeden, außer für den Erzeuger selbst.



3.2 CoordinationContext

- Diese Daten müssen in allen Nachrichten an den Web Service enthalten sein
- sie werden in der Regel genutzt, um bestimmte Ressourcen in diesem anzusprechen

3.2 CoordinationContext

- der CoordinationContext ist ein Kontexttyp der dazu benutzt wird Koordinationsinformation an die im *coordination service* involvierten Parteien zu vermitteln
- Ein CoordinationContext stellt Zugriff auf den *registration service*, einen Koordinationstypen (vgl. WS-AtomicTransaction, etc.) und relevante Erweiterungen bereit

3.2 CoordinationContext

```
<soap:Envelope xml-ns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    ...
    <wscoor:CoordinationContext
      xml-ns:wsu="http://schemas.xmlsoap.org/ws/2002/07/utility"
      xml-ns:wscoor="http://schemas.xmlsoap.org/ws/2003/09/wscoor"
      xml-ns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing"
      xml-ns:myApp="http://myApplication.de/myApp"

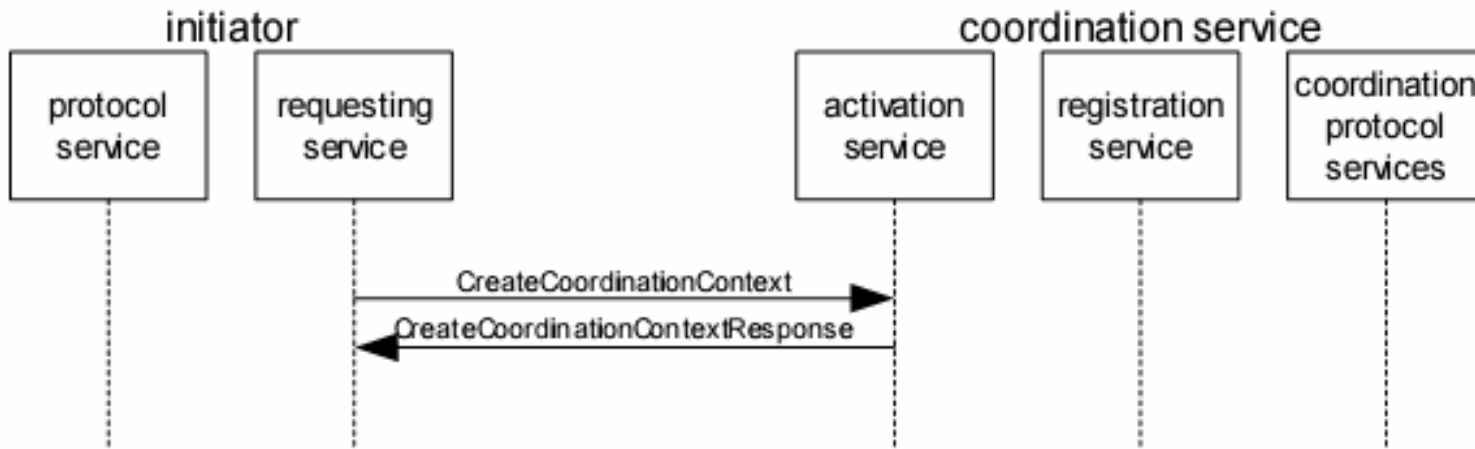
      soap:mustUnderstand="true">

      <wsu:Expires>
        2004-05-31T12:30:00Z
      </wsu:Expires>
      <wsu:Identifier>
        http://myCoordinator.de/activity1
      </wsu:Identifier>
      <wscoor:CoordinationType>
        http://schemas.xmlsoap.org/ws/2003/09/wsat
      </wscoor:CoordinationType>
      <wscoor:RegistrationService>
        <wsa:Address>
          http://myCoordinator.de/registrationService
        </wsa:Address>
        <wsa:ReferenceProperties>
          <TransactionId="c fb01dc0-5073-405a-a3aea6038ecc476e">
        </wsa:ReferenceProperties>
        </wscoor:RegistrationService>
      </wscoor:CoordinationContext>
      <myApp:IsolationLevel>
        RepeatableRead
      </myApp:IsolationLevel>
      ...
    </soap:Header>
    <soap:Body>
      ...
    </soap:Body>
  </soap:Envelope>
```

3.3 Activation service

- Um eine neue *activity* zu beginnen, sendet eine Applikation eine `CreateCoordinationContext`-Nachricht an den *activation service* eines ihr bekannten Koordinators
- diese Applikation wird als Initiator bezeichnet
- der Koordinator erzeugt einen `CoordinationContext` und liefert diesen mittels einer `CreateCoordinationContextResponse`-Nachricht an die Applikation zurück.

3.3 Activation service



3.3.1 CreateCoordinationContext

- Eine CreateCoordinationContext-Nachricht ist nach folgendem Schema aufgebaut:

```
<CreateCoordinationContext ...>  
  <CoordinationType> ... </CoordinationType>  
  <wsu:Expires> ... </wsu:Expires>  
  <CurrentContext> ... </CurrentContext>  
  ...  
</CreateCoordinationContext>
```

3.3.2 CreateCoordinationContextResponse

- Die CreateCoordinationContext-Nachricht wird vom ActivationService mit einer CreateCoordinationContextResponse-Nachricht beantwortet
- Der Aufbau der Nachricht sieht folgendermaßen aus:

```
<CreateCoordinationContextResponse ...>  
  <CoordinationContext> ... </CoordinationContext>  
  ...  
</CreatCoordinationCotextResponse>
```

3.3.2 CreateCoordinationContextResponse

- Das einzig geforderte Element enthält den erzeugten CoordinationContext
- hier jedoch Versand im Body einer SOAP-Nachricht, nicht im Header
- Die Verwendung von beliebigen zusätzlichen Elementen und Attributen zum Versenden weiterer Informationen ist ebenfalls möglich



3.3.2 CreateCoordinationContextResponse

```
<CreateCoordinationContextResponse>
  <CoordinationContext>
    <Identifier>
      http://Business456.com/tm/context1234
    </Identifier>
    <CoordinationType>
      http://schemas.xmlsoap.org/ws/2004/10/wsat
    </CoordinationType>
    <RegistrationService>
      <wsa:Address>
        http://Business456.com/tm/registration
      </wsa:Address>
      <wsa:ReferenceProperties>
        <myapp:PrivateInstance>
          1234
        </myapp:PrivateInstance>
      </wsa:ReferenceProperties>
    </RegistrationService>
  </CoordinationContext>
</CreateCoordinationContextResponse>
```

3.4 Registration service

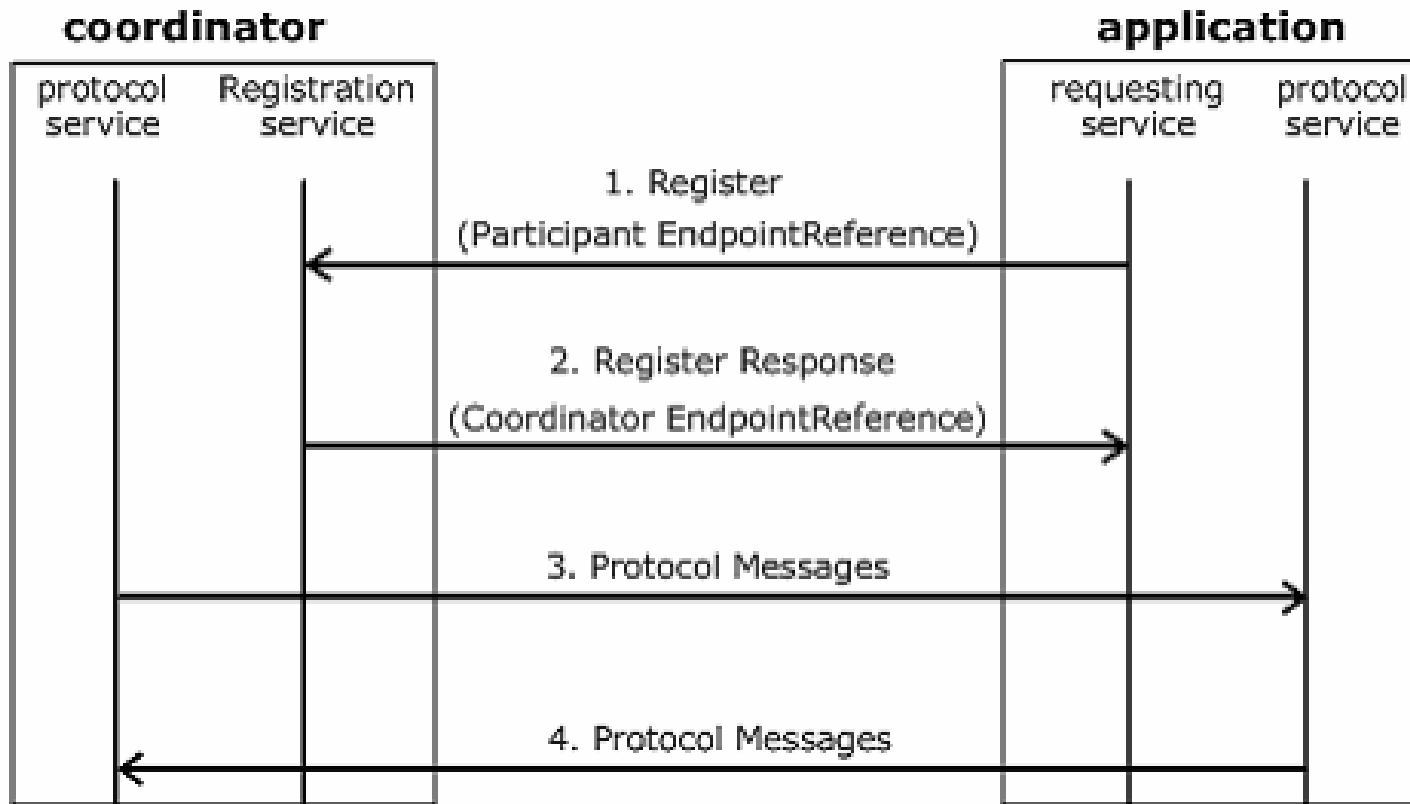
- ist der Activation service abgeschlossen, kann sich der Initiator selbst in der *activity* als Teilnehmer registrieren
- dies geschieht, indem der Initiator aus dem Koordinationskontext die *endpoint reference* auf den *registration service* des Koordinators ausliest und an diese URI eine *register*-Nachricht sendet
- der Initiator hat die Möglichkeit sich für mehrere Protokolle zu registrieren, indem er die *register*-nachricht mehrmals sendet.



3.4 Registration service

- Der *registration service* des Koordinators antwortet auf die *register-nachricht* mit einem *registration response*
- der Koordinator gibt in einer *register response-nachricht* eine *endpoint reference* zurück
- an diese URI sollen ab sofort alle Protokollnachrichten der Applikation versendet werden

3.4 Registration service





3.4.1 Register Nachricht

- Die Registrierungsanfrage wird folgenderweise verwendet:
 - Teilnehmerauswahl und Registration durch den *coordination service*, für ein bestimmtes Koordinationsprotokoll, welches der gegenwärtige *coordination type* unterstützt
 - Austausch der *endpoint references* durch beide Seiten (sowohl Initiator, als auch Koordinator)

3.4.1 Register Nachricht

```
<Register ...>
  <ProtocolIdentifier> ... </ProtocolIdentifier >
  <ParticipantProtocolService> ... </ParticipantProtocolService>
  ...
</Register>
```

- Register/ProtocolIdentifier
 - Die Applikation entscheidet sich für ein bestimmtes *coordination protocol*, das vom *coordination type* der *activity* unterstützt wird und übergibt dessen Identifikator
- Register/ParticipantProtocolService
 - Die Applikation übergibt außerdem eine *endpoint reference*, an die der Koordinator künftige Protokollnachrichten senden soll



3.4.1 Register Nachricht

```
<Register>
  <ProtocolIdentifier>
    http://schemas.xmlsoap.org/ws/2004/10/wsat/Volatile2PC
  </ProtocolIdentifier>
  <ParticipantProtocolService>
    <wsa:Address>
      http://Adventure456.com/participant2PCservice
    </wsa:Address>
    <wsa:ReferenceProperties>
      <BetaMark> AlphaBetaGamma </BetaMark>
    </wsa:ReferenceProperties>
  </ParticipantProtocolService>
</Register>
```

3.4.2 RegistrationResponse Nachricht

- Die Antwort auf einer Registernachricht enthält die *endpoint references* des Koordinators

```
<RegisterResponse ...>  
  <CoordinationProtocolService> ... </CoordinationProtocolService>  
  ...  
</RegisterResponse>
```

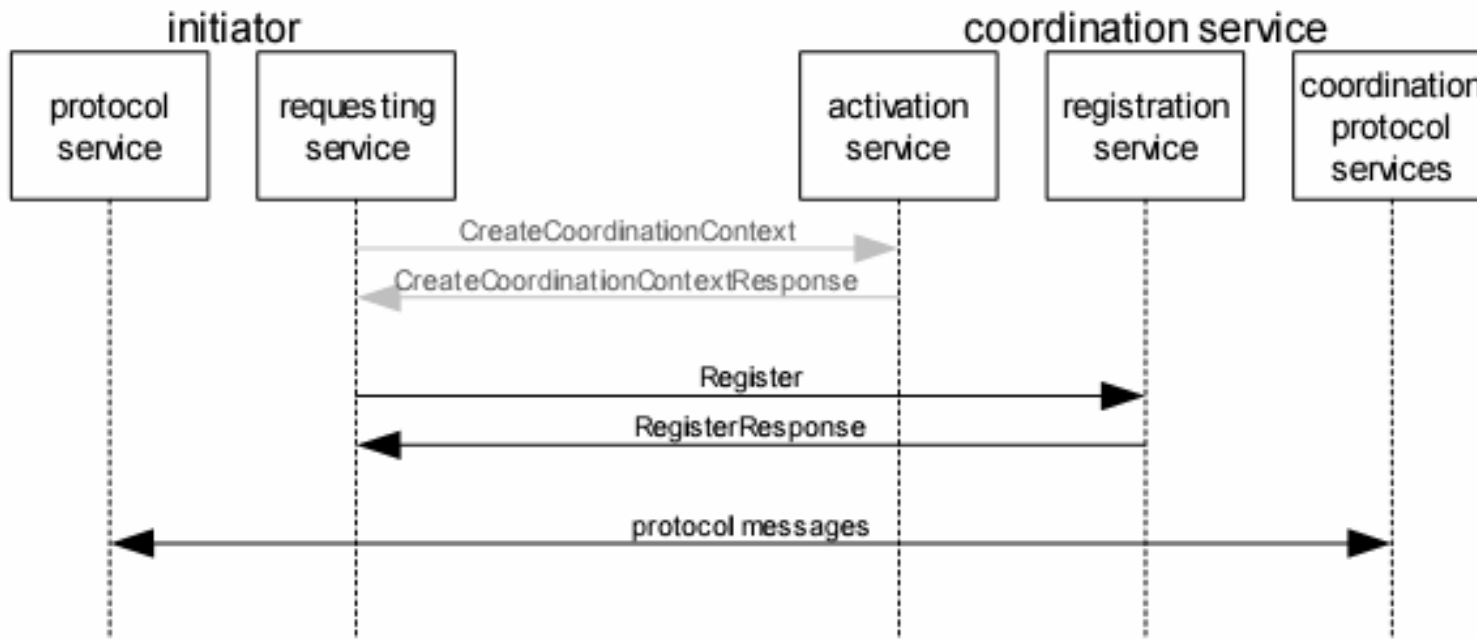
- RegisterResponse/CoordinatorProtocolService
 - Die vom *coordination service* gewählte *endpoint reference*, die der Teilnehmer (die Applikation) für das Koordinations-Protokoll verwenden soll



3.4.2 RegistrationResponse Nachricht

```
<RegisterResponse>
  <CoordinatorProtocolService>
    <wsa:Address>
      http://Business456.com/mycoordinationservice/coordinator
    </wsa:Address>
    <wsa:ReferenceProperties>
      <myapp:MarkKey> %%F03CA2B%% </myapp:MarkKey>
    </wsa:ReferenceProperties>
  </CoordinatorProtocolService>
</RegisterResponse>
```

3.4.2 RegistrationResponse Nachricht





3.5 Importieren einer activity

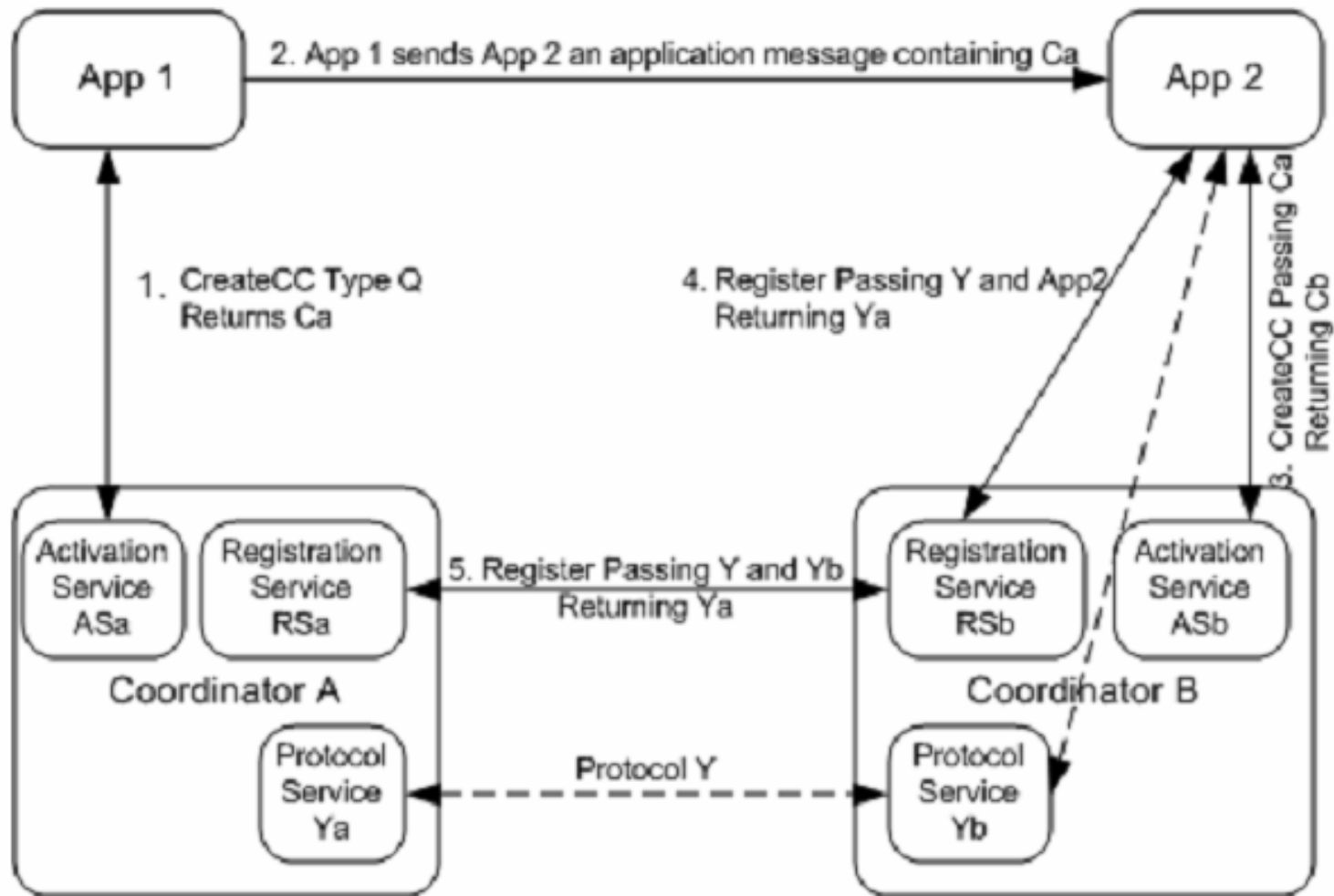
- Der *activation service* kann genutzt werden, um einen Koordinator in eine bestehenden *activity* einzubinden
- erhält ein Web Service eine Applikationsnachricht, die einen `CoordinationContext` beinhaltet, kann er sich beim Koordinator registrieren, der diesen erzeugt hat



3.5 Importieren einer activity

- es gibt jedoch auch Gründe, einen eigenen Koordinator heranzuziehen, da dieser beispielsweise eine höhere Leistungsfähigkeit aufweist oder als vertrauenswürdiger gilt
- dieser Vorgang wird „importieren“ oder „interposing“ genannt

3.5 Importieren einer activity





3.5 Importieren einer activity

- Es entsteht eine neue Protokoll Instanz, in der A die Rolle des Koordinators und B die Rolle eines Teilnehmers zukommt
- Koordinator B wird nun als *interposed* oder *subordinate coordinator* bezeichnet
- es können hierarchische Strukturen beliebiger Tiefe gebildet werden
- Ein Koordinator kann hierbei auch mehrere *subordinate coordinators* unter sich haben.



3.6 Coordination Faults

- WS-Coordination faults müssen als [action] Eigenschaft folgende *fault action* URI einbinden:
 - <http://schemas.xmlsoap.org/ws/2004/10/wscoor/fault>
- Die Definitionen der Störungen besitzen folgende Eigenschaften:
 - [Code] Code der Störung
 - [Subcode] Subcode der Störung
 - [Reason] Der Grund der Störung (in Englischer Sprache)
 - [Detail] Detaillierte Angaben zum aufgetretenen Fehler

3.6 Coordination Faults

- Bei SOAP 1.2, muss [Code] entweder "Sender" oder "Receiver" sein
 - Diese Eigenschaften werden in text XML in dieser Reihenfolge wie folgt dargestellt:

SOAP Version	Sender	Receiver
SOAP 1.2	S:Sender	S:Receiver



3.6 Coordination Faults

```
<S:Envelope>
  <S:Header>
    <wsa:Action>
      http://schemas.xmlsoap.org/ws/2004/10/wscoor/fault
    </wsa:Action>
    <!-- Headers elided for clarity. -->
  </S:Header>
  <S:Body>
    <S:Fault>
      <S:Code>
        <S:Value>[Code]</S:Value>
        <S:Subcode>
          <S:Value>[Subcode]</S:Value>
        </S:Subcode>
      </S:Code>
      <S:Reason>
        <S:Text xml:lang="en">[Reason]</S:Text>
      </S:Reason>
      <S:Detail>
        [Detail]
        ...
      </S:Detail>
    </S:Fault>
  </S:Body>
</S:Envelope>
```

3.6 Coordination Faults

- die Darstellung in SOAP 1.1:

```
<S11:Envelope>
  <S11:Body>
    <S11:Fault>
      <faultcode>[Subcode]</faultcode>
      <faultstring xml:lang="en">[Reason]</faultstring>
    </S11:Fault>
  </S11:Body>
</S11:Envelope>
```

3.6 Coordination Faults

- Invalid State

- Diese Fehlermeldung wird entweder vom Koordinator oder einem Teilnehmer gesendet, um anzuzeigen, dass die *endpoint reference*, die den Fehler gemeldet hat, eine Nachricht bekommen hat, die für den momentanen Status ungültig ist.
- Dies ist ein unbehebbarer Zustand.
- Eigenschaften:
 - [Code] Sender
 - [Subcode] wscoor:InvalidState
 - [Reason] The message was invalid for the current state of the activity.
 - [Detail] unspecified

3.6 Coordination Faults

- Invalid Protocol
 - Diese Fehlermeldung wird entweder vom Koordinator oder von einem Teilnehmer gesendet, um anzuzeigen, dass die *endpoint reference*, die den Fehler erzeugt hat, eine Nachricht von einem ungültigen Protokoll empfing.
 - Dies ist ein unbehebbarer Zustand.
 - Eigenschaften:
 - [Code] Sender
 - [Subcode] wscoor:InvalidProtocol
 - [Reason] The protocol is invalid or is not supported by the coordinator.
 - [Detail] unspecified

3.6 Coordination Faults

- Invalid Parameters

- Diese Störung wird entweder vom Koordinator oder von einem Teilnehmer gesendet, um anzuzeigen, dass die *endpoint reference* die den Fehler erzeugt hat unzulässige Parameter auf oder innerhalb einer Nachricht empfing.
- Dies ist ein unbehebbarer Zustand.
- Eigenschaften:
 - [Code] Sender
 - [Subcode] wscoor:InvalidParameters
 - [Reason] The message contained invalid parameters and could not be processed.
 - [Detail] unspecified



3.6 Coordination Faults

- No Activity
 - Diese Störung wird vom Koordinator gesendet, wenn der Teilnehmer für zu lange inaktiv gewesen ist und vermutet wird, dass er die *activity* beendet hat.
 - Eigenschaften:
 - [Code] Sender
 - [Subcode] wscoor:NoActivity
 - [Reason] The participant is not responding and is presumed to have ended.
 - [Detail] unspecified

3.6 Coordination Faults

- Context Refused

- Einem Koordinator wird diese Störung geschickt, um anzuzeigen, dass die *endpoint reference* einen Kontext nicht annehmen kann, den er übergeben bekam.
- Eigenschaften:
 - [Code] Sender
 - [Subcode] wscoor:ContextRefused
 - [Reason] The CoordinationContext that was provided could not be accepted.
 - [Detail] unspecified

3.6 Coordination Faults

- **Already Registered**

- Einem Teilnehmer wird diese Störung geschickt, wenn der Koordinator feststellt, dass der Teilnehmer versucht hat, sich für das gleiche Protokoll der gleichen Tätigkeit mehr als einmal zu registrieren.
- Eigenschaften:
 - [Code] Sender
 - [Subcode] wscoor:AlreadyRegistered
 - [Reason] The participant has already registered for the same protocol.
 - [Detail] unspecified



Ende

- Vielen Dank für Eure und Ihre Aufmerksamkeit!
- Fragen? Diskussionsbedarf?