

WS-Agreement Negotiation

Oliver Waeldrich, Wolfgang Ziegler, Philipp Wieder
OGF 30, SLAs in Grids Workshop
26. October 2010, Brussels

Agenda



- Motivation, Goals and Requirements
- Negotiation and Deployment Model
- Negotiation Factory Port Type
- Negotiation Port Type
- Negotiation Offer State Model
- Coupling Negotiation and Agreement Layer

Motivation, Goals and Requirements

WS-Agreement Negotiation

Motivation – What is missing in WS-Agreement



Service Parameterization

- Service descriptions might become very complex
- Service parameters may depend on each other
- Complex service parameterization can not easily be expressed in a template

Resource Bargaining

- Service properties of SLA might depend on highly dynamic data
- Service provider want to
 - Make special offers to good customers
 - Offer unused resources cheaply

Renegotiation of existing agreements

- Consumer requires more resources to cope with performance peaks

WS-Agreement Negotiation

Goals and Non-Goals



Requirements

- Must support bilateral negotiation of agreement offers
- Must allow negotiation of agreement offers for new and renegotiation of existing agreements
- Must support parallel negotiation of multiple offers
- Must provide a symmetric protocol
- Must build on top of the WS-Agreement specification
- Must provide a simple state machine

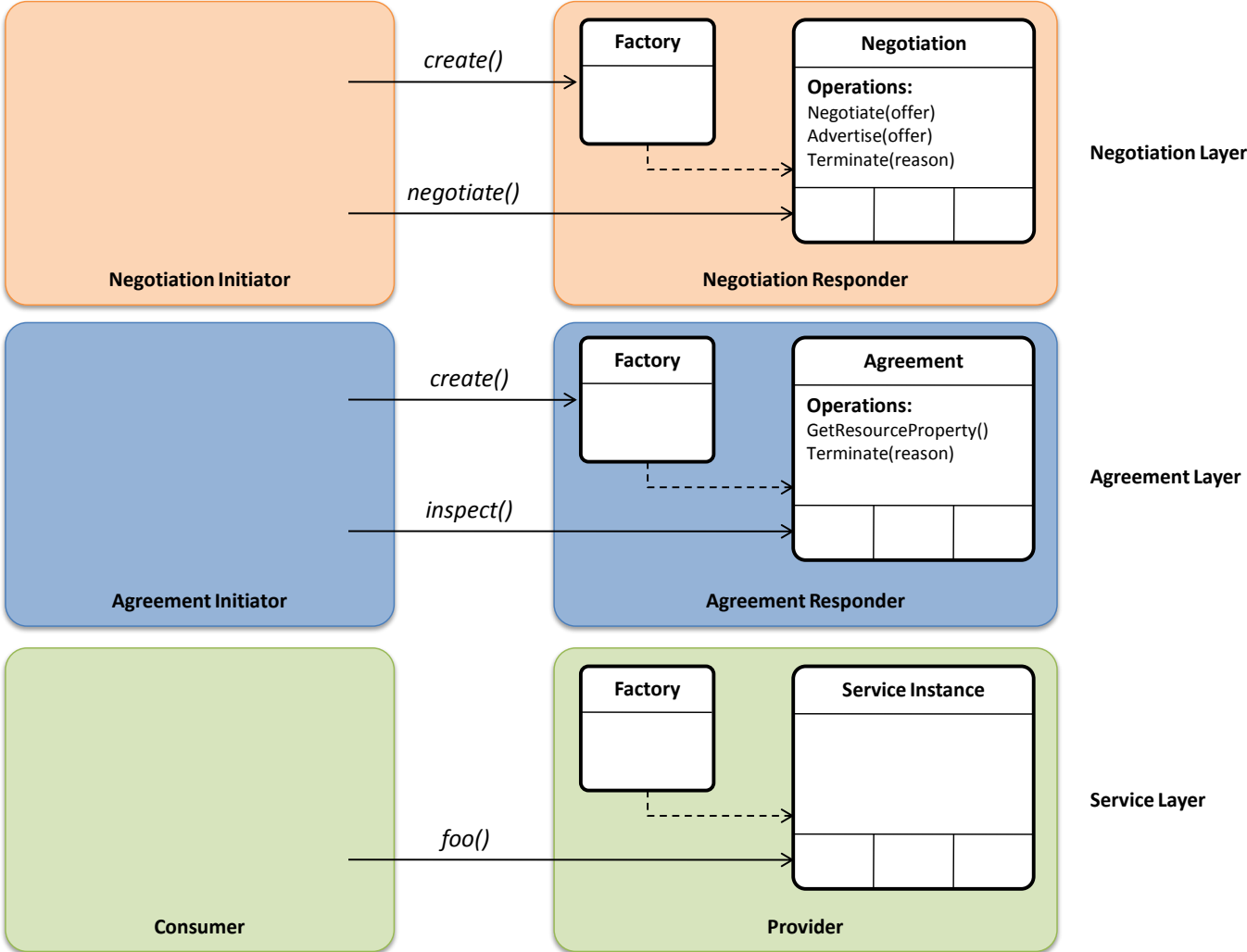
Out of Scope

- Binding negotiations and definition of compensation methods
- Definition of concrete negotiation strategies
- Auctions, 1-to-many negotiations with shared context

Negotiation and Deployment Model

WS-Agreement Negotiation

Layered Negotiation Model



WS-Agreement Negotiation

Basic Negotiation Model



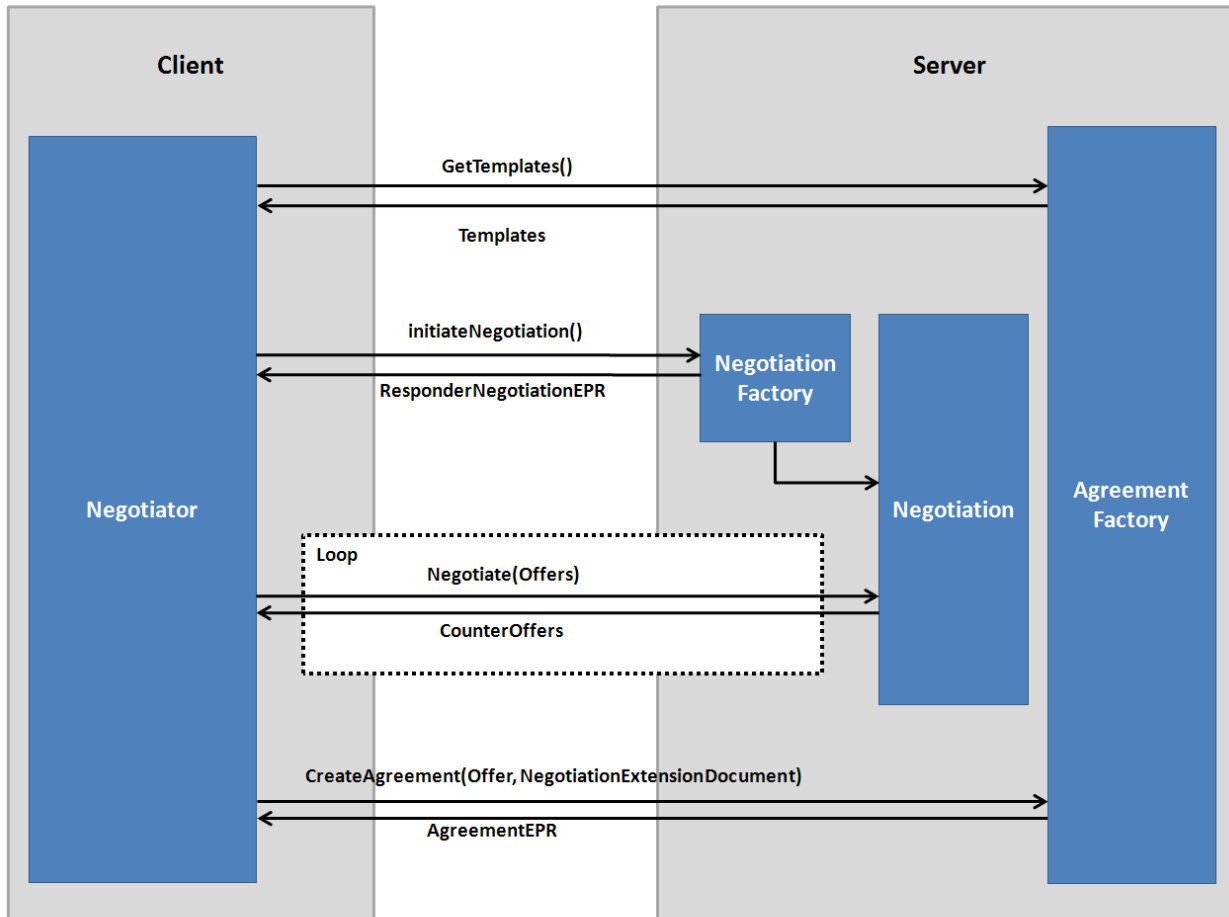
Offer/Counter Offer Model

- Implemented via Negotiation Port Type
- Exchange of dynamic information
- Counter offers are offers created based on offers of the opposite negotiation party
 - take constraints of negotiation participator into account

Notification

- Implemented via Advertisement Port Type
- Notification about new offers, state changes, etc.
- No counter offer expected

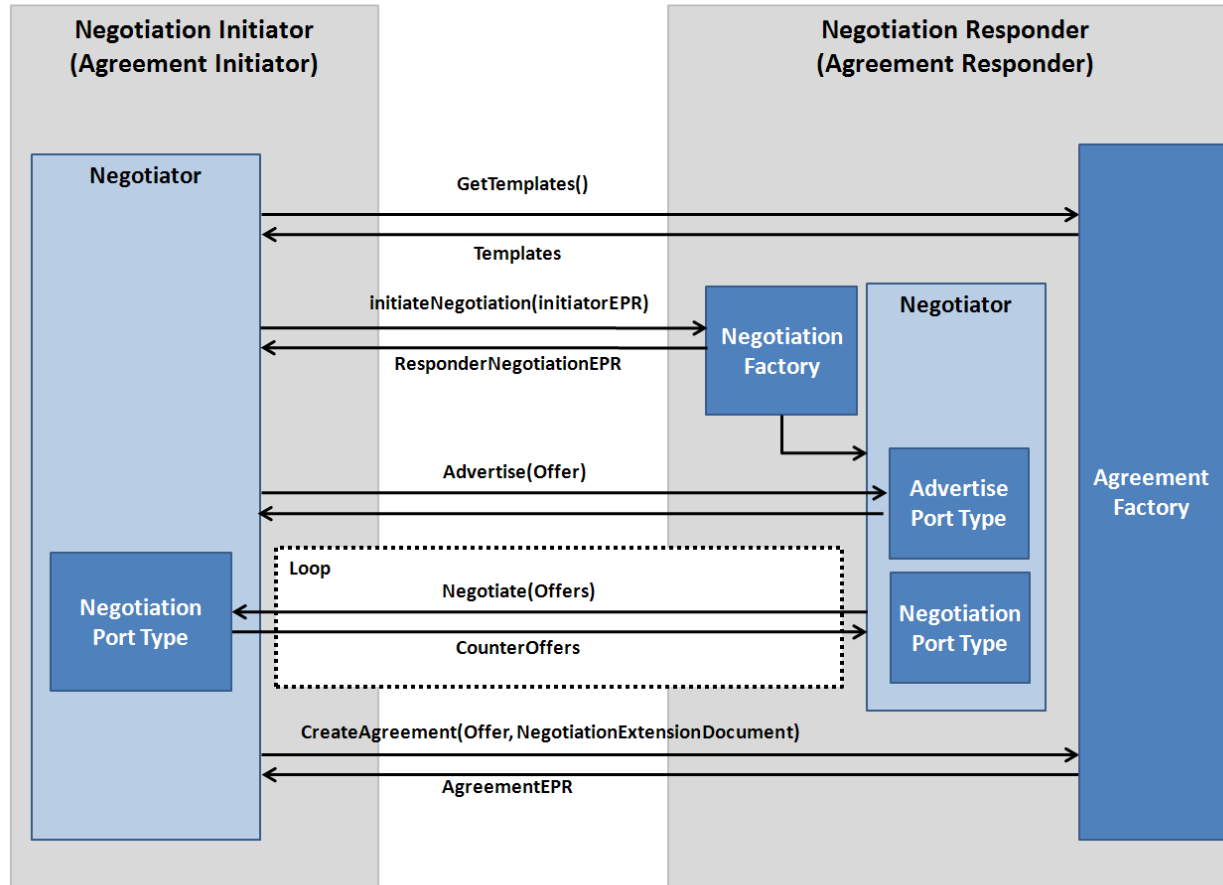
WS-Agreement Negotiation Deployment Scenarios



Simple Client-Server Negotiation

- Negotiation Responder implements the Negotiation Factory Port Type
- Negotiation Factory creates a new **Negotiation** instance that implements the *Negotiation Port Type*
- The **Negotiation** instance implements strategies to negotiate offers behalf of a specific agreement factory
- Negotiation Factory Port Type and Agreement Factory Port Type MAY be composed

WS-Agreement Negotiation Deployment Scenarios

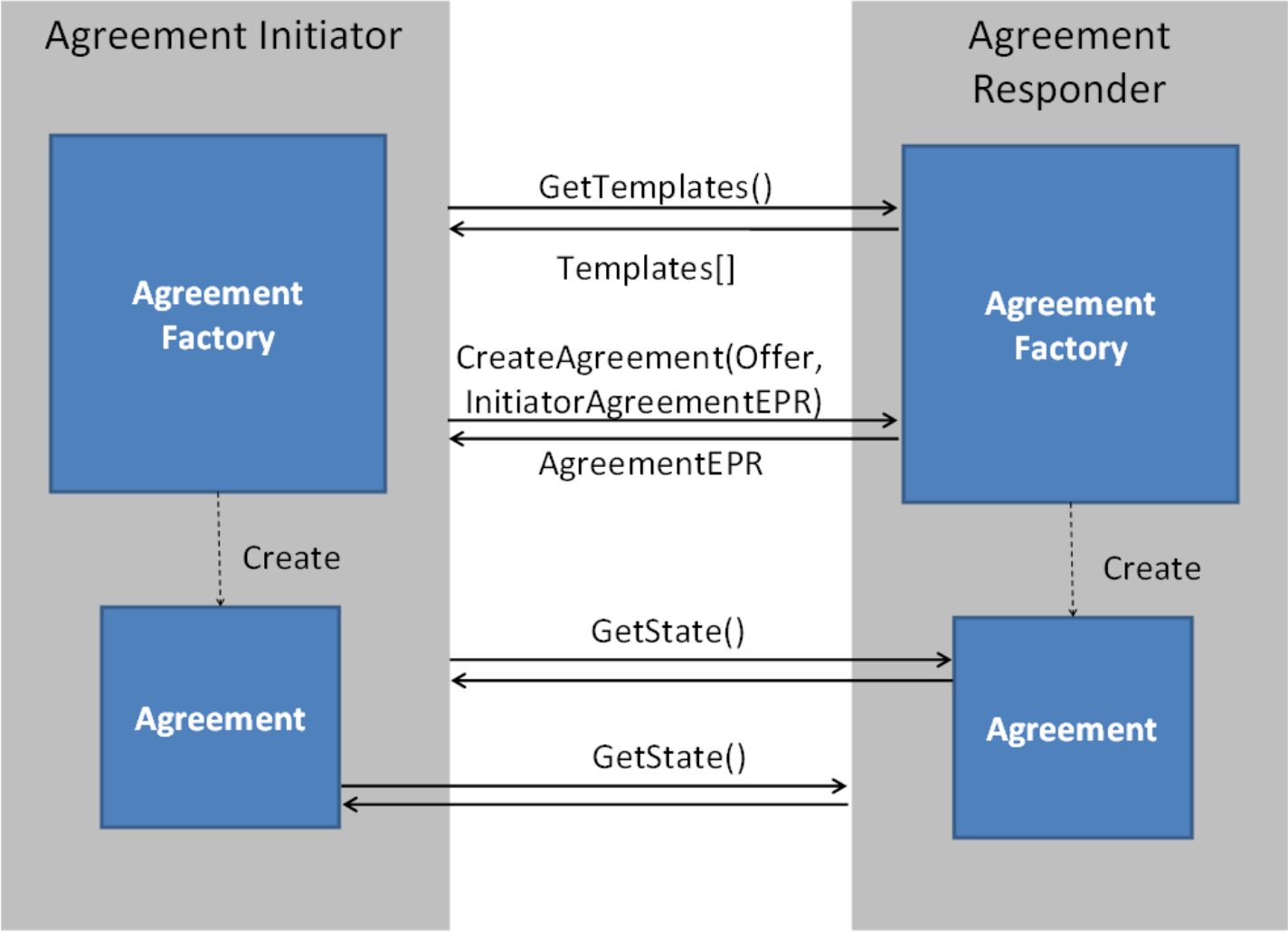


Bilateral Negotiation

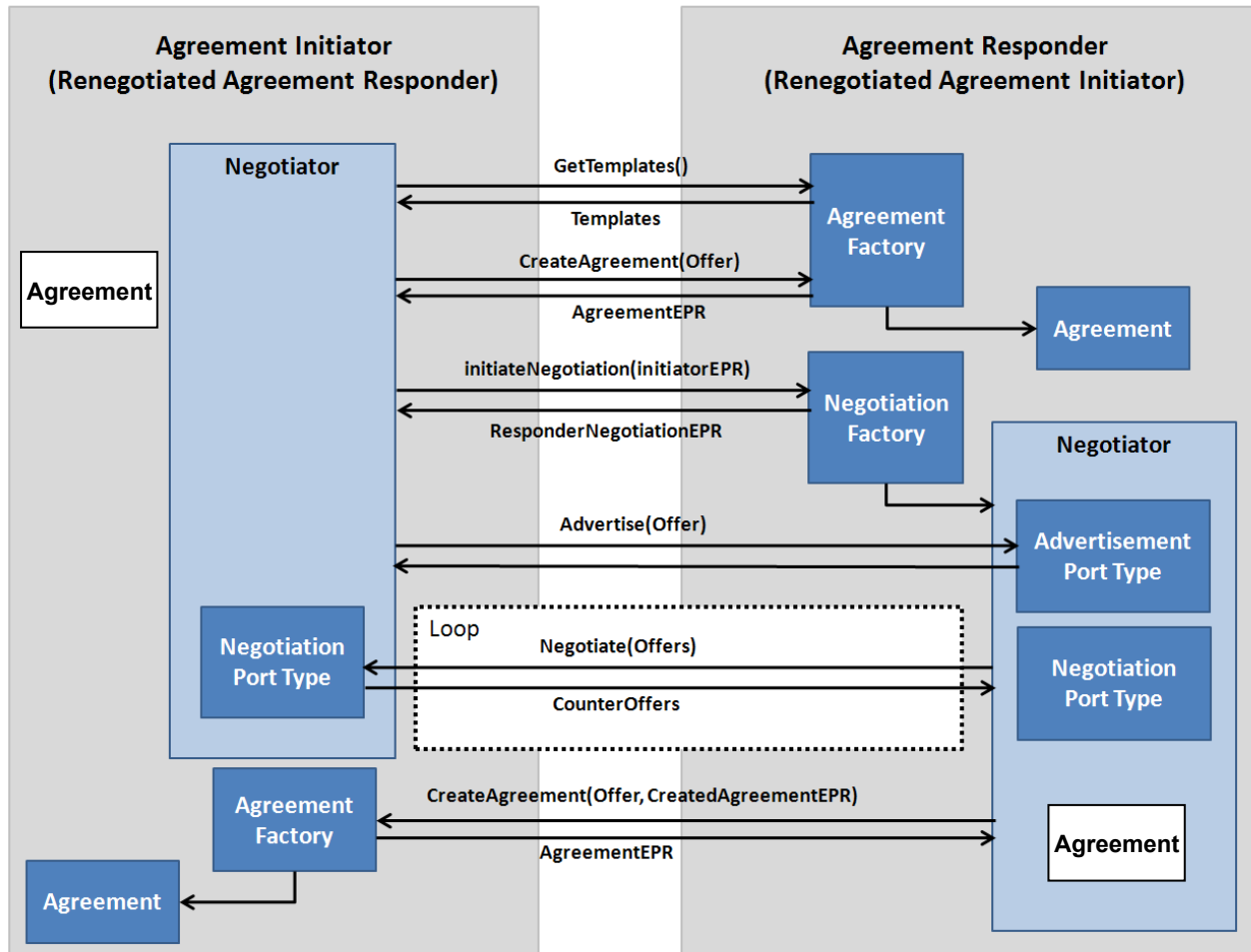
- Symmetric protocol layout on agreement layer
- Both negotiating parties can take an active role
- *Negotiate* method follows request/response pattern
- *Advertise* method allows asynchronous exchange of offers and implementation of negotiation handovers
- Both methods may be used in a multi-round negotiation

WS-Agreement Negotiation

WS-Agreement symmetric deployment



WS-Agreement Negotiation Deployment Scenarios



Bilateral Renegotiation of Agreements

- Symmetric agreement layer
- Decoupled negotiation and agreement layer
- Changing roles on agreement layer possible

Provider-driven Renegotiation of Agreements

- Service requirements of a consumer changes
- Consumer advertises new service requirements, also indicating that it expects the provider to create the renegotiated agreement
- Resource provider negotiates offers with consumer based on resource availability and finally creates agreement

Negotiation Factory Port Type

WS-Agreement Negotiation

Negotiation Factory



Negotiation Factory

- Follows Factory Pattern
- Creates a new negotiation instance
- Initiate relationship between two negotiating parties
 - Identify and associate roles Negotiation Initiator and Negotiation Responder
 - Define the context of the negotiation process

WS-Agreement Negotiation

NegotiationFactory Port Type

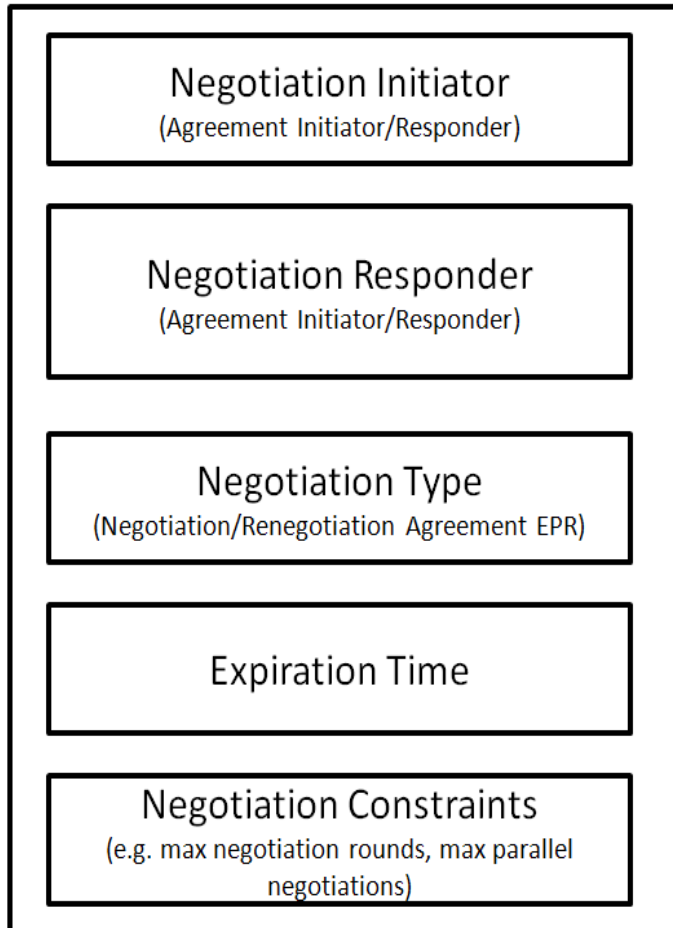


Negotiation Factory

- `<<create>> initiateNegotiation(Context) : Negotiation`
- Negotiation Context
 - identifies the negotiating parties
 - context scopes negotiation process
 - Negotiation of only specific templates
 - Renegotiation of existing agreements
- Critical and Noncritical extensions
 - extension on the protocol layer
 - binding negotiations
 - negotiation of pre-established agreements
- Resource Properties: Template for Negotiation Context

WS-Agreement Negotiation

NegotiationContext Structure



Negotiation Context

- Metadata associated with specific negotiation process
- Definition of Negotiation Roles
- Type of Negotiation
- Constraint Section
 - Max. number of rounds, max. counter offers per offer, ...

Negotiation Roles

- Negotiation Initiator
- Negotiation Responder
- Identify the negotiating parties in the context of a negotiation
- May be endpoint references, DNs, etc.

Negotiation Port Type

WS-Agreement Negotiation

Negotiation and Role Model



Negotiation

- Implements Offer/Counter Offer Model
- Parallel negotiation of multiple offers
- Must adhere to negotiation context
 - Binding negotiations (e.g. definition of compensation methods)
 - Renegotiation of existing agreements

Negotiation Offer Context

- Specify metadata of a negotiation offer
 - Originating offer
 - creator of negotiation offer
 - Offer state

WS-Agreement Negotiation

Negotiation Port Type / Advertisement Port Type



Negotiation Port Type Operations

- Negotiate(Offer[]) : CounterOffer[]
- Terminate(Reason) : void

Advertisement Port Type Operations

- Advertise(Offer[]) : void

Negotiation Port Type Resource Properties

- Negotiation Context
- NegotiationOffer

WS-Agreement Negotiation

NegotiationOffer Structure



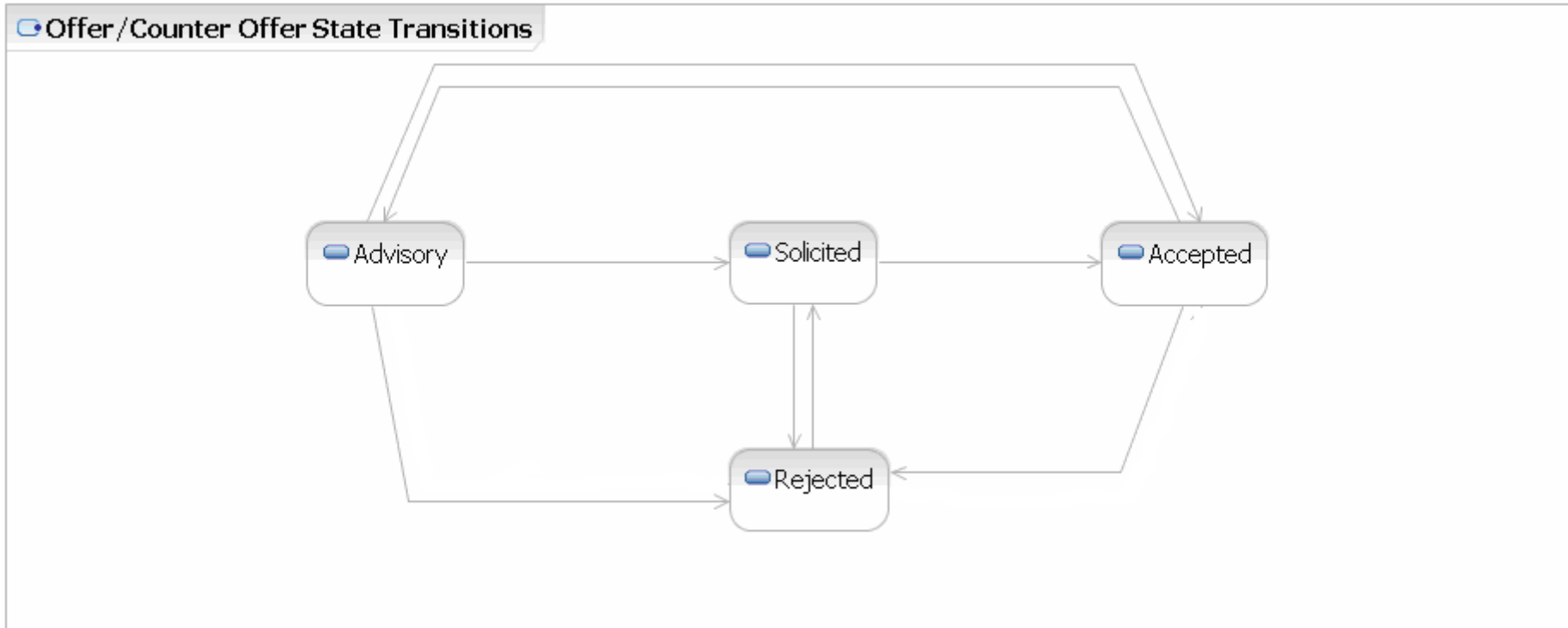
NegotiationOffer Structure

- wsag:AgreementType with additional elements
- NegotiationOfferId uniquely identifies an offer in the context of a negotiation
- NegotiationContext contains meta data related to an offer
 - the Id of the original offer
 - the creator the offer (optional)
 - the expiration time of the offer (optional)
 - the state of the offer
 - any other domain specific data
- Negotiation constraints define negotiation goals a counter offer must take into account

Negotiation Offer State Model

WS-Agreement Negotiation

Offer/Counter Offer State Transitions



Unidirectional transitions: advisory → solicited → Accepted

- Provides a path for negotiation convergence

Otherwise bidirectional state transitions

- Offers flexibility in the negotiation process

States may include domain specific data, e.g. propagating reasons for rejection, etc.

Coupling of Negotiation and Agreement Layer

WS-Agreement Negotiation

Coupling of Negotiation and Agreement Layer



Profile for WS-Agreement-Factory

- Extension documents for creation of (re-)negotiated agreements
- (Non)critical extension for negotiated agreements
- Critical extension for renegotiated agreements
- Comprises references to negotiation instances and/or EPRs original agreements

→ *Loose coupling of negotiation layer and agreement layer*

```
<wsag-neg:NegotiationExtension>
  <wsag-neg:ResponderNegotiationEPR>
    wsa:EndpointReferenceType
  </wsag-neg:ResponderNegotiationEPR> ?
  <wsag-neg:InitiatorNegotiationEPR>
    wsa:EndpointReferenceType
  </wsag-neg:InitiatorNegotiationEPR> ?
  <xsd:any /> *
</wsag-neg:NegotiationExtension>
```

WS-Agreement Negotiation

Coupling of Negotiation and Agreement Layer



```
<wsag-neg:RenegotiationExtension>
  <wsag-neg:ResponderAgreementEPR>
    wsa:EndpointReferenceType
  </wsag-neg:ResponderAgreementEPR>
  <wsag-neg:InitiatorAgreementEPR>
    wsa:EndpointReferenceType
  </wsag-neg:InitiatorAgreementEPR> ?
  <wsag-neg:ResponderNegotiationEPR>
    wsa:EndpointReferenceType
  </wsag-neg:ResponderNegotiationEPR>
  <wsag-neg:InitiatorNegotiationEPR>
    wsa:EndpointReferenceType
  </wsag-neg:InitiatorNegotiationEPR> ?
  <xsd:any /> *
</wsag-neg:RenegotiationExtension>
```

Negotiation Constraints and Negotiation Convergence

WS-Agreement Negotiation

Negotiation Constraints and Convergence



- Enable the negotiating parties to express restrictions for counter offers
- Mechanism to converge a negotiation process
- Extend Negotiation Item with domain specific convergence model
- Negotiation items provide extension point for domain specific elements
- Include annotation constraints to steer negotiation process

```
<wsag-neg:NegotiationOffer>
  ...
  <wsagneg:NegotiationConstraints>
    <wsag:Item>...</wsag:Item> *
    <wsag:Constraint>...</wsag:Constraint> *
  </wsag-neg:NegotiationConstraints> ?
</wsag-neg:NegotiationOffer>
```

WS-Agreement Negotiation

Converging Negotiation and Constraint Annotations



OpenGridForum

```
<wsag:Item>
  <wsag:Location>//jsdl:IndividualCPUCount</wsag:Location>
  <wsag:Constraint>
    <xsd:sequence>
      <xsd:element name="Exact" minOccurs="1" >
        <xsd:simpleType>
          <xsd:restriction base="xsd:double">
            <xsd:enumeration value="1"/>
            <xsd:enumeration value="2"/>
            <xsd:enumeration value="4"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </wsag:Constraint>
</wsag:Item>
```

WS-Agreement Negotiation

Converging Negotiation and Constraint Annotations



```
<wsag:Item name="CPU">
  <wsag:Location>//jsdl:IndividualCPUCount</wsag:Location>
  <wsag:Constraint>...<wsag:Constraint>
    <ca:ConstraintAnnotation xmlns:ca="...">
      <cs:ConstraintType>SIMPLETYPE_ENUMERATION</cs:ConstraintType>
      <cs:SpecificationLevel>REQUIRED</cs:SpecificationLevel>
    </ca:ConstraintAnnotation>
  </wsag:Item>
```

```
<wsag:Item name="STARTTIME">
  <wsag:Location>//ar:StartTime</wsag:Location>
  <wsag:Constraint>...<wsag:Constraint>
    <ca:ConstraintAnnotation xmlns:ca="...">
      <cs:ConstraintType>SIMPLETYPE_DATETIME</cs:ConstraintType>
      <cs:SpecificationLevel>OPTIONAL</cs:SpecificationLevel>
    </ca:ConstraintAnnotation>
  </wsag:Item>
```

WS-Agreement Negotiation

Converging Negotiation and Negotiation Priorities



```
<wsag:Item Name="xs:string">
  <wsag:Location>xs:anyType</wsag:Location>
  <wsag:ItemConstraint>
    <xs:restriction>
      xs:simpleRestrictionModel
    <xs:restriction> ?
    <xs:group>xs:groupRef</xs:group> ?
    <xs:all>xs:all</xs:all> ?
    <xs:choice>xs:explicitGroup</xs:choice> ?
    <xs:sequence>xs:explicitGroup</xs:sequence>?
  </wsag:ItemConstraint>
  <xs:any>any#other</xs:any> *
</wsag:Item>
```

WS-Agreement Negotiation

Open Questions



Profiling the WS-Agreement Factory Layer

- Description of the of the profile sufficient?
- Explicitly show other approaches, e.g. for negotiation of agreements?

State machine

- Does it meet all our requirements?

Extension of the Negotiation Constraints to converge negotiation

- Include it in the standard? Leave it to implementations?
- Solve this issue via profiling?

Other open issues?

Thank you!