ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Εισαγωγή στα Δίκτυα Υπηρεσιών

Διάλεξη 14η: WS Policy

Χρήστος Νικολάου
Τμήμα Επιστήμης Υπολογιστών

Introduction to Service Networks

WS-Policy, WS-AtomicTransaction, WS-BusinessActivity

Source: WS Policy Primer, WS-AT, WS-BusinessActivity Specs

Christos Nikolaou

Transformation Services Laboratory
CSD/UoC

Introduction

Web Services Policy is a machine-readable language for representing the capabilities and requirements of a Web service. These are called 'policies'.

Web Services Policy offers mechanisms to represent consistent combinations of capabilities and requirements, to determine the compatibility of policies, to name and reference policies and to associate policies with Web service metadata constructs such as service, endpoint and operation.

Web Services Policy is a simple language that has four elements - `Policy, All,`
`ExactlyOne` and `PolicyReference` - and two attributes - `wsp:Optional`
and `wsp:Ignorable`.

Why do we need WS-Policy?

```
<soap:Envelope>
<soap:Header>
    <wsa:To>http://x.example.com/realquote</wsa:To>
    <wsa:Action>http://x.example.com/GetRealQuote</wsa:Action>
</soap:Header>
<soap:Body>...</soap:Body>
</soap:Envelope>
```

This message uses message addressing headers. The `wsa:To` and `wsa:Action` header blocks identify the destination and the semantics implied by this message respectively.

Justification for WS-Policy

Assume a Web service developer is building a client application that retrieves real time stock quote information from Company-X, Ltd. Company-X supplies real time data using Web services.

The developer has Company-X's advertised WSDL description of these Web services.

Company-X requires the use of addressing headers for messaging. Just the WSDL description is not sufficient for the developer to enable the interaction between her client and these Web services. WSDL constructs do not indicate requirements such as the use of addressing.

Providers have the option to convey requirements, such as the use of addressing, through word-of-mouth and documentation – as they always have. To interact successfully with this service, the developer may have to read any related documentation, call someone at Company-X to understand the service metadata, or look at sample SOAP messages and infer such requirements or behaviors.

Web Services Policy is a machine-readable language for representing these Web service capabilities and requirements as policies. Policy makes it possible for providers to represent such capabilities and requirements in a machine-readable form. For example, Company-X may augment the service WSDL description with a policy that requires the use of addressing. The client application developer can use a policy-aware client that understands this policy and engages addressing automatically.

A Policy Expression

```
<Policy>
<wsam:Addressing>…</wsam:Addressing>
</Policy>
```

Attached to binding:

```
<wsdl:binding name="AddressingBinding" type="tns:RealTimeDataInterface" >
<Policy>
    <wsam:Addressing>…</wsam:Addressing>
</Policy>
…
</wsdl:binding>
```

The `wsam:Addressing` element is a policy assertion.

Another Example with Security

```
<soap:Envelope>
<soap:Header>
    <wss:Security soap:mustUnderstand="1" >
        <wsu:Timestamp wsu:Id="_0">
            <wsu:Created>2006-01-19T02:49:53.914Z</u:Created>
            <wsu:Expires>2006-01-19T02:54:53.914Z</u:Expires>
        </wsu:Timestamp>
    </wss:Security>
    <wsa:To>http://x.example.com/quote</wsa:To>
    <wsa:Action>http://x.example.com/GetRealQuote</wsa:Action>
</soap:Header>
<soap:Body>...</soap:Body>
</soap:Envelope>
```

A policy statement would be needed:

```
<Policy>
<wsam:Addressing>...</wsam:Addressing>
<sp:TransportBinding>...</sp:TransportBinding>
</Policy>
```

Policy Assertions

Who defines policy assertions? Where are they?

Policy assertions are defined by Web services developers, product designers, protocol authors and users. Like XML Schema libraries, policy assertions are a growing collection. Several WS-* protocol specifications and applications define policy assertions:

- Web Services Security Policy
- Web Services Reliable Messaging Policy
- Web Services Atomic Transaction
- Web Services Business Activity Framework
- Devices Profile for Web Services
- ...

Combining Policy Assertions

```
<All>
<wsam:Addressing>…</wsam:Addressing>
<sp:TransportBinding>…</sp:TransportBinding>
</All>
```

```
<All>
<wsam:Addressing>...</wsam:Addressing>
<ExactlyOne>
    <sp:TransportBinding>...</sp:TransportBinding>
    <sp:AsymmetricBinding>...</sp:AsymmetricBinding>
</ExactlyOne>
</All>
```

Asymmetric Binding Imposes Message Level Security

Optional Policy Assertion

```
<All>
<mtom:OptimizedMimeSerialization wsp:Optional="true"/>
<wsam:Addressing>...</wsam:Addressing>
<ExactlyOne>
    <sp:TransportBinding>¼</sp:TransportBinding>
    <sp:AsymmetricBinding>¼</sp:AsymmetricBinding>
</ExactlyOne>
</All>
```

Ignorable Policy Expressions

Suppose Company-X decides that it will log SOAP messages sent and received in an exchange. This behavior has no direct impact on the messages sent on the wire, and does not affect interoperability.

Some parties might have a concern about such logging and might decide not to interact with Company-X knowing that such logging is performed. To address this concern, Company-X includes a Logging assertion in its policy to enable such parties to be aware of logging.

By marking the Logging assertion with the **`wsp:Ignorable`** attribute with a value of "true" Company-X indicates that a requester may choose to either ignore such assertions or to consider them as part of policy intersection. An assertion that may be ignored for policy intersection is called an ignorable assertion.

```
<log:Logging wsp:Ignorable="true" />
```

Atomic Transaction Context

Atomic Transaction builds on WS-Coordination, which defines an activation and a registration service.

The Atomic Transaction coordination context must flow on all application messages involved with the transaction.

Atomic Transaction adds the following semantics to the CreateCoordinationContext operation on the activation service.

• If the request **includes** the CurrentContext element, the target coordinator is interposed as a subordinate to the coordinator stipulated inside the CurrentContext element.

• If the request **does not include** a CurrentContext element, the target coordinator creates a new transaction and acts as the root.

A coordination context may have an Expires attribute. This attribute specifies the earliest point in time at which a transaction may be terminated solely due to its length of operation. From that point forward, the transaction manager may elect to

unilaterally roll back the transaction, so long as it has not transmitted a Commit or a Prepared notification.

The Atomic Transaction protocol is identified by the following coordination type:
http://schemas.xmlsoap.org/ws/2004/10/wsat

Atomic Transaction Protocols

This specification defines the following protocols for atomic transactions.

• **Completion:** The completion protocol initiates commitment processing. Based on each protocol's registered participants, the coordinator begins with Volatile 2PC then proceeds through Durable 2PC. The final result is signaled to the initiator.

• **Two-Phase Commit (2PC)**: The 2PC protocol coordinates registered participants to reach a commit or abort decision, and ensures that all participants are informed of the final result. The 2PC protocol has two variants:

• **Volatile 2PC:** Participants managing volatile resources such as a cache should register for this protocol.

• **Durable 2PC:** Participants managing durable resources such as a database should register for this protocol.

A participant can register for more than one of these protocols by sending multiple Register messages.

Completion Protocol

The Completion protocol is used by an application to tell the coordinator to either try to commit or abort an atomic transaction. After the transaction has completed, a status is returned to the application.

An initiator registers for this protocol using the following protocol identifier:

http://schemas.xmlsoap.org/ws/2004/10/wsat/Completion
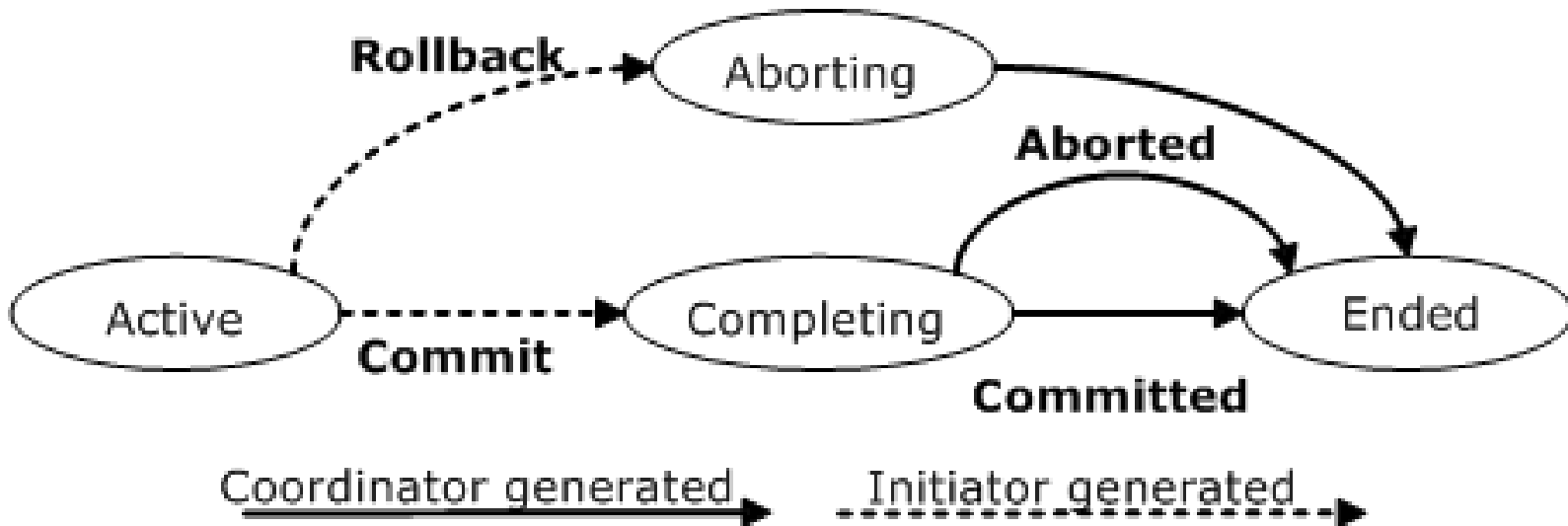
Two Phase Commit Protocol
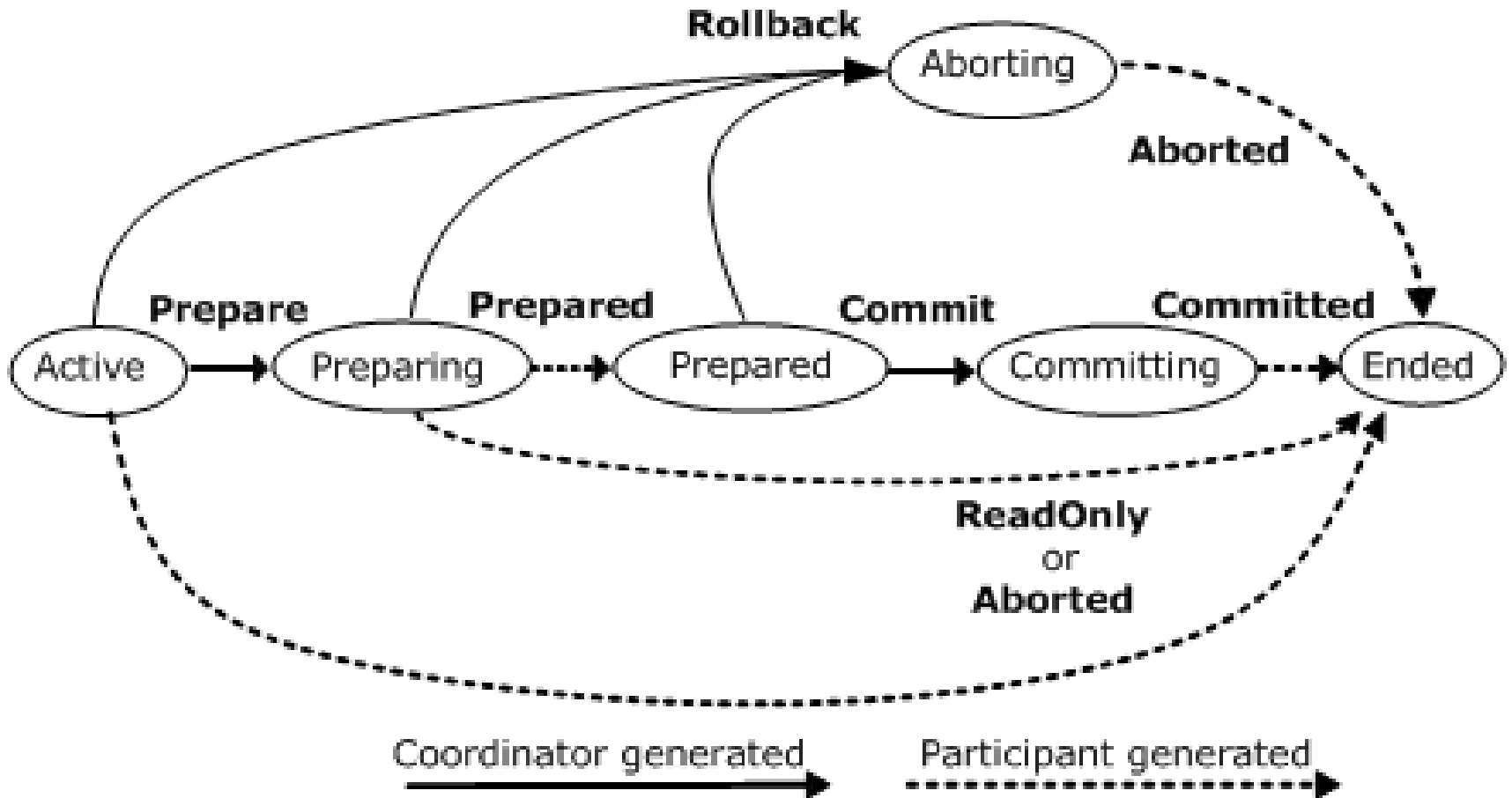
AT Policy Assertion

The AT policy assertions are provided by a web service to qualify the transactional processing of messages associated with the particular operation to which the assertions are scoped. The AT policy assertions indicate:

1. whether a requester MAY, MUST or SHOULD NOT include an AtomicTransaction CoordinationContext flowed with the message.

2. the capability of the target service to process the message under an atomic transaction regardless of whether the requester supplies an AtomicTransaction CoordinationContext.

The AT policy assertions are semantically independent of one another, and may be used together or in isolation.

```
<wsat:ATAssertion [wsp:Optional="true"]? ... >
...
</wsat:ATAssertion>
```

ATAlwaysCapability

<wsat:ATAlwaysCapability ... />

A policy assertion that specifies a capability of the target service indicating that a requester's message will be processed transactionally regardless of whether the requester supplies an AtomicTransaction CoordinationContext. If an AtomicTransaction context is provided by the requester, it will be used. Otherwise the processing of the message will be within a transaction implicitly started and ended by the target service's environment as part of the processing of that message.

Assertion Example

```
(01) <wsdl:definitions
(02)   targetNamespace="bank.example.com"
(03)   xmlns:tns="bank.example.com"
(04)   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
(05)   xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
(06)   xmlns:wsat="http://schemas.xmlsoap.org/ws/2004/10/wsat"
(07)   xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd" >
(08)
(09) <wsp:Policy wsu:Id="TransactedPolicy1" >
(10)       <wsat:ATAssertion wsp:optional="true" />
(11)       <!-- omitted assertions -->
(12) </wsp:Policy>
```

Example cont.

```
(13) <wsp:Policy wsu:Id="TransactedPolicy2" >
(14)        <wsat:ATAlwaysCapability />
(15)        <!-- omitted assertions -->
(16) </wsp:Policy>
(17) <!-- omitted elements -->
(18) <wsdl:binding name="BankBinding" type="tns:BankPortType" >
(19)        <!-- omitted elements -->
(20)        <wsdl:operation name="QueryBalance" >
(21)                <wsp:PolicyReference URI="#TransactedPolicy2" wsdl:required="true" />
(22)                <!-- omitted elements -->
(23)        </wsdl:operation>
(24)        <wsdl:operation name="TransferFunds" >
(25)                <wsp:PolicyReference URI="#TransactedPolicy1" wsdl:required="true" />
(26)                <!-- omitted elements -->
(27)        </wsdl:operation>
(28) </wsdl:binding>
(29) </wsdl:definitions>
```

WS-BusinessActivity Coordination Context

A business activity uses the WS-Coordination CoordinationContext with the CoordinationType set to one of the following URIs:

http://schemas.xmlsoap.org/ws/2004/10/wsba/AtomicOutcome

http://schemas.xmlsoap.org/ws/2004/10/wsba/MixedOutcome

A coordination context may have an Expires attribute. This attribute specifies the earliest point in time at which a long-running activity may be terminated solely due to its length of operation. A participant could terminate its participation in the long running activity using the Exit protocol message.

A CoordinationContext can have additional elements for extensibility.

Due to the extensibility of WS-Coordination it is also possible to define a coordination protocol type that, in addition to specifying the agreement protocol between a coordinator and a participant, also specifies the behavior of the coordination logic. For example, it may specify that the coordinator will act in an all-or-nothing manner to determine its outcome based on the outcomes communicated by its participants, or that it will use a specific majority rule when determining its final outcome based on the outcomes of its participants.

Coordination Types and Protocols

A coordinator for an AtomicOutcome coordination type must direct all participants to close or all participants to compensate. A coordinator for a MixedOutcome coordination type may direct each individual participant to close or compensate. All coordinators MUST implement the AtomicOutcome coordination type. Any coordinator MAY implement the MixedOutcome coordination type.

The Coordination protocols for business activities are summarized below with names relative to the wsba base name:

• **BusinessAgreementWithParticipantCompletion**: A participant registers for this protocol with its coordinator, so that its coordinator can manage it. A participant must know when it has completed all work for a business activity.

• **BusinessAgreementWithCoordinatorCompletion**: A participant registers for this protocol with its coordinator, so that its coordinator can manage it. A participant relies on its coordinator to tell it when it has received all requests to perform work within the business activity.

BusinessAgreementWithParticipantCompletion
Abstract State Diagram
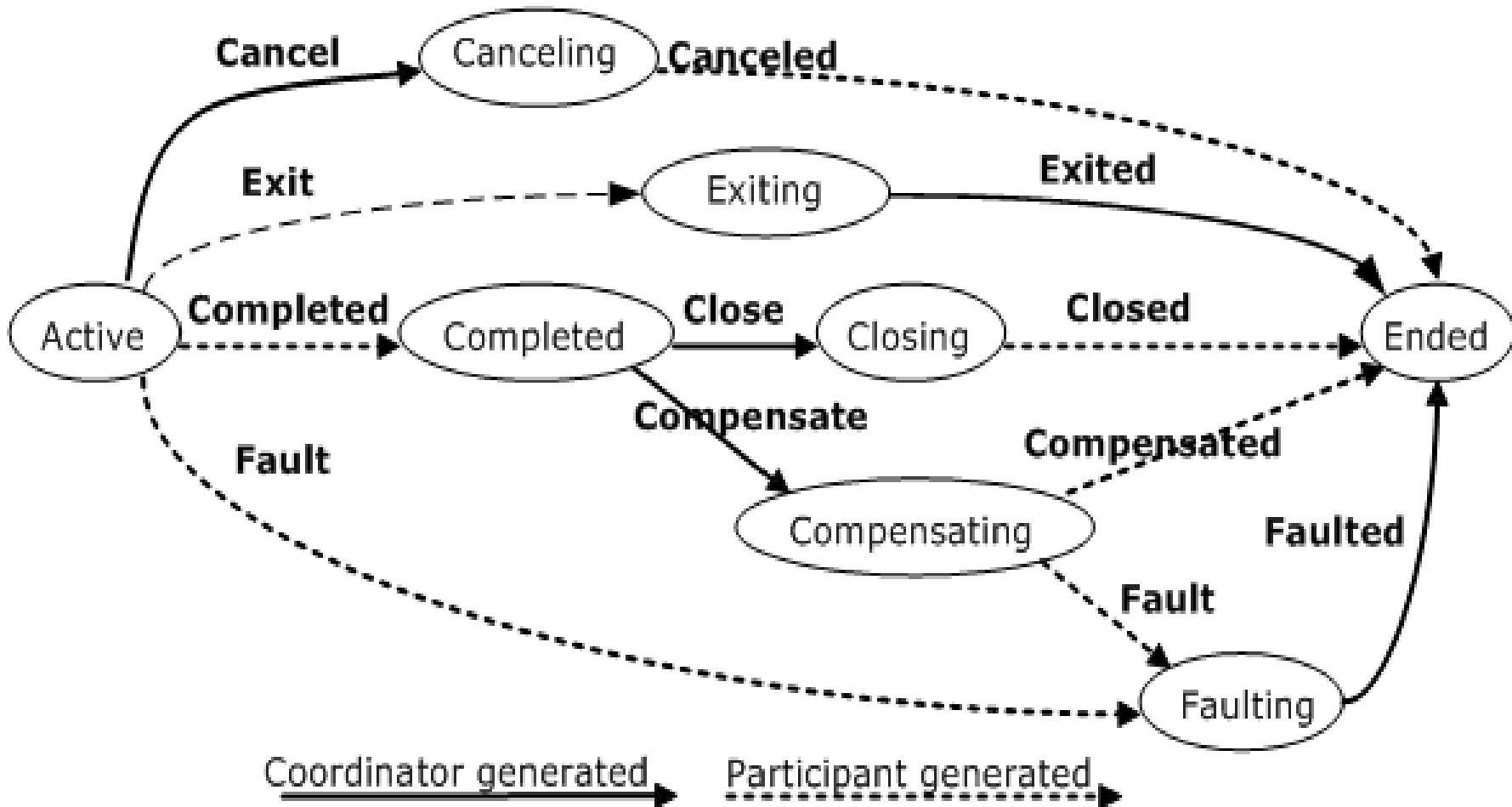
BusinessAgreementWithCoordinatorCompletion
Abstract State Diagram
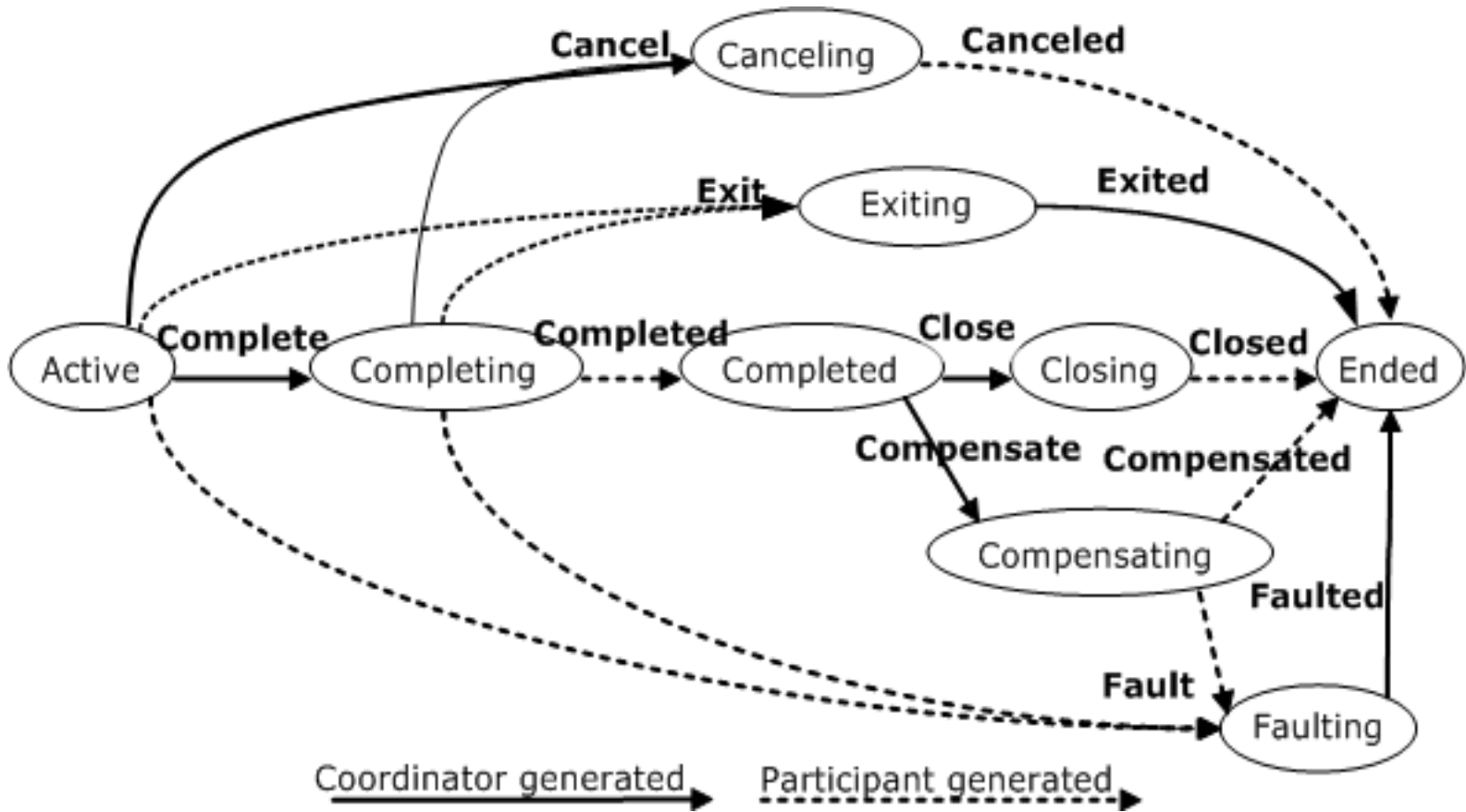
Τέλος Ενότητας

Χρηματοδότηση

Σημειώματα

Σημείωμα αδειοδότησης

Σημείωμα Αναφοράς