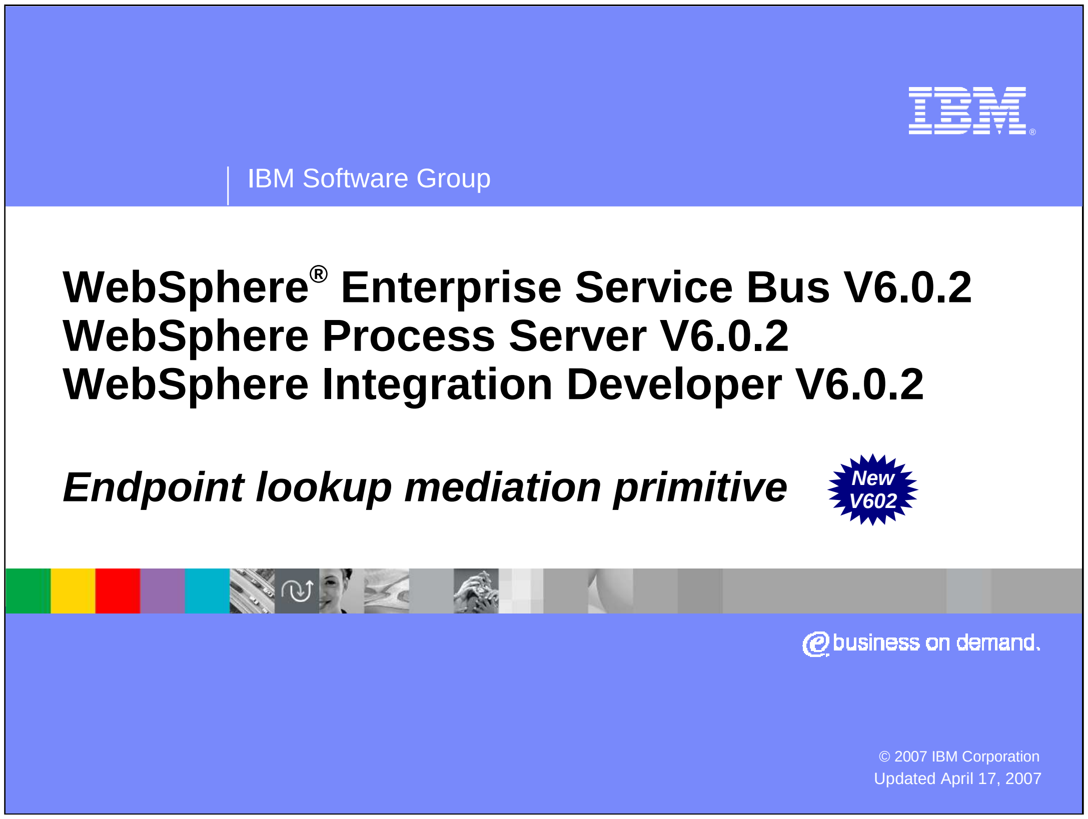


IBM Software Group

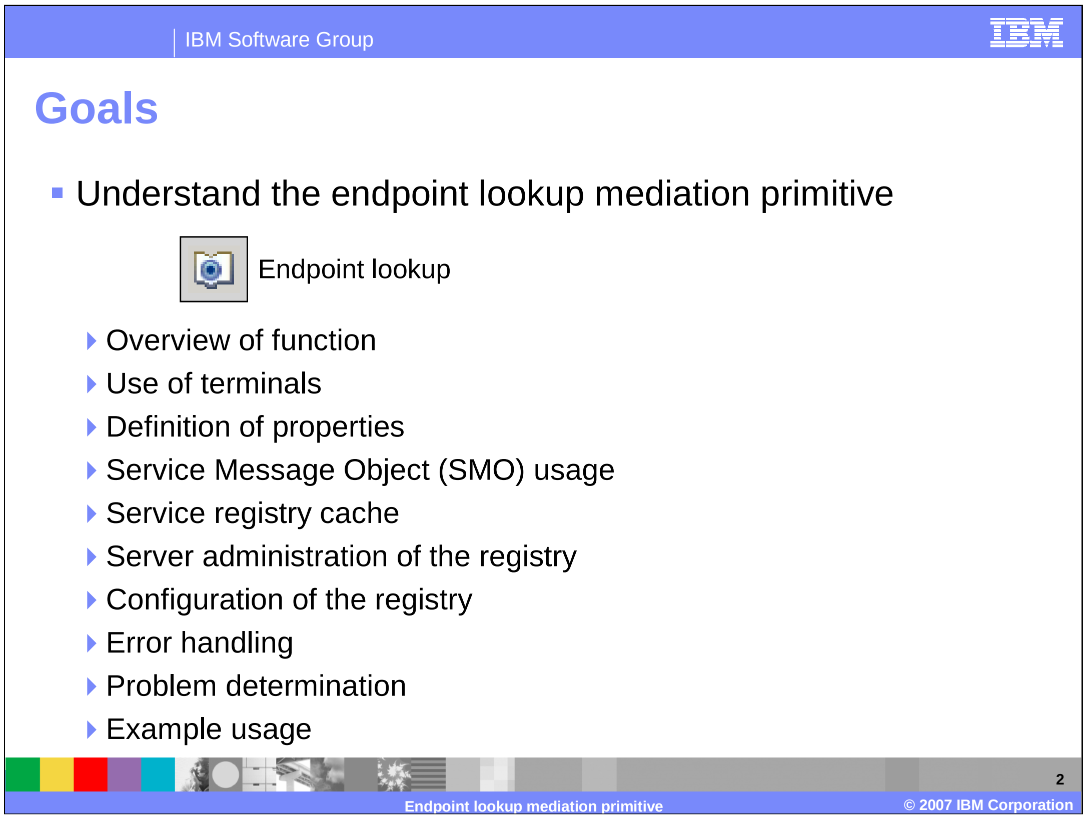
WebSphere® Enterprise Service Bus V6.0.2
WebSphere Process Server V6.0.2
WebSphere Integration Developer V6.0.2

Endpoint lookup mediation primitive

New V602

business on demand.

This presentation provides a detailed look at the endpoint lookup mediation primitive that was introduced in version 6.0.2.



Goals

- Understand the endpoint lookup mediation primitive

  Endpoint lookup

  - Overview of function
  - Use of terminals
  - Definition of properties
  - Service Message Object (SMO) usage
  - Service registry cache
  - Server administration of the registry
  - Configuration of the registry
  - Error handling
  - Problem determination
  - Example usage

The goal of this presentation is to provide you with a full understanding of the endpoint lookup mediation primitive.

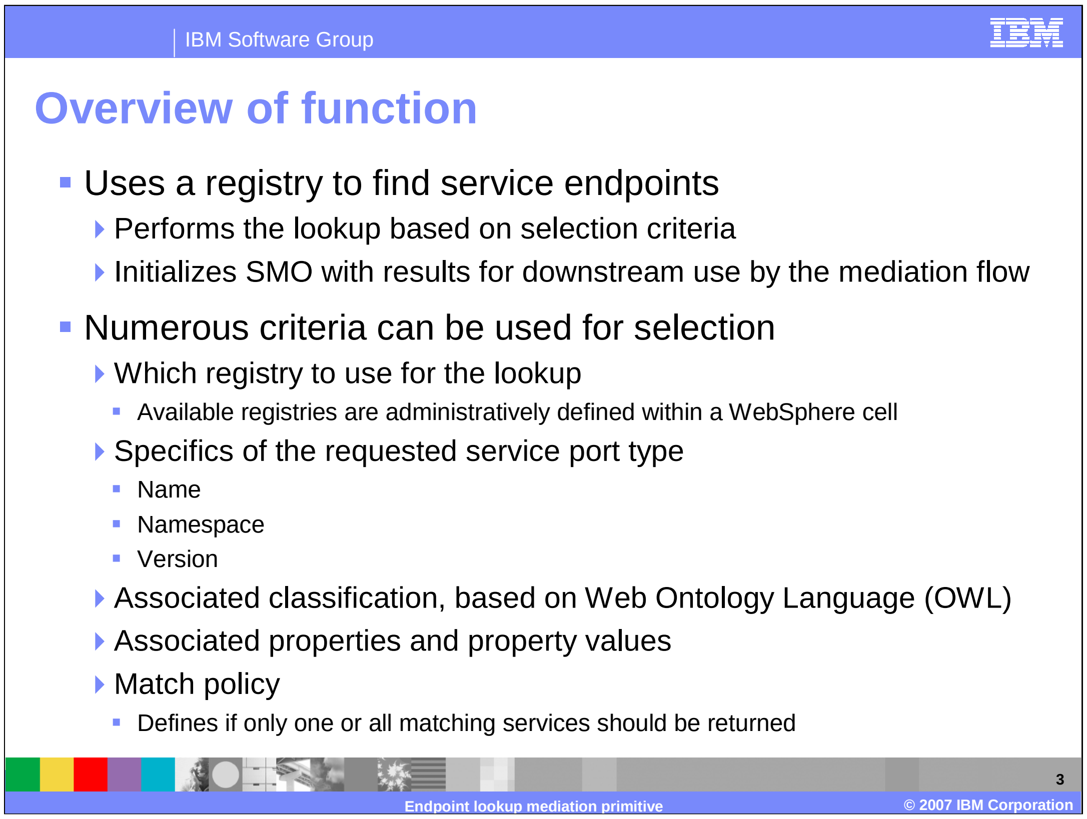
The presentation assumes that you are already familiar with the material presented in the **Mediation Primitive Common Details** presentation and the **Common Details – Promoted Properties** presentation. These two presentations serve as a base for understanding mediation primitives in general.

An overview of the function provided by the endpoint lookup primitive is presented, along with information about the primitive's use of terminals and its properties.

This primitive has a special relationship to the Service Message Object (SMO). The presentation reviews those elements in the SMO context that are specifically used by endpoint lookup primitives.

In order to understand the endpoint lookup primitive, it is necessary to understand more than the primitives behaviour and how it fits into a mediation flow. The endpoint lookup primitive interfaces with the WebSphere Service Registry and Repository, which is referred to as the registry in this presentation. Use of the registry is enabled through capabilities provided by the WebSphere Enterprise Service Bus and the WebSphere Process Server, which are referred to as the server in this presentation. You learn about the service registry cache and administration of registries, capabilities that are provided by the server. You are also introduced to an approach for configuring the registry.

The presentation then returns to looking specifically at the endpoint lookup primitive, covering error handling, problem determination and an example of its usage.


IBM Software Group

Overview of function

- Uses a registry to find service endpoints
  - Performs the lookup based on selection criteria
  - Initializes SMO with results for downstream use by the mediation flow
- Numerous criteria can be used for selection
  - Which registry to use for the lookup
    - Available registries are administratively defined within a WebSphere cell
  - Specifics of the requested service port type
    - Name
    - Namespace
    - Version
  - Associated classification, based on Web Ontology Language (OWL)
  - Associated properties and property values
  - Match policy
    - Defines if only one or all matching services should be returned

3

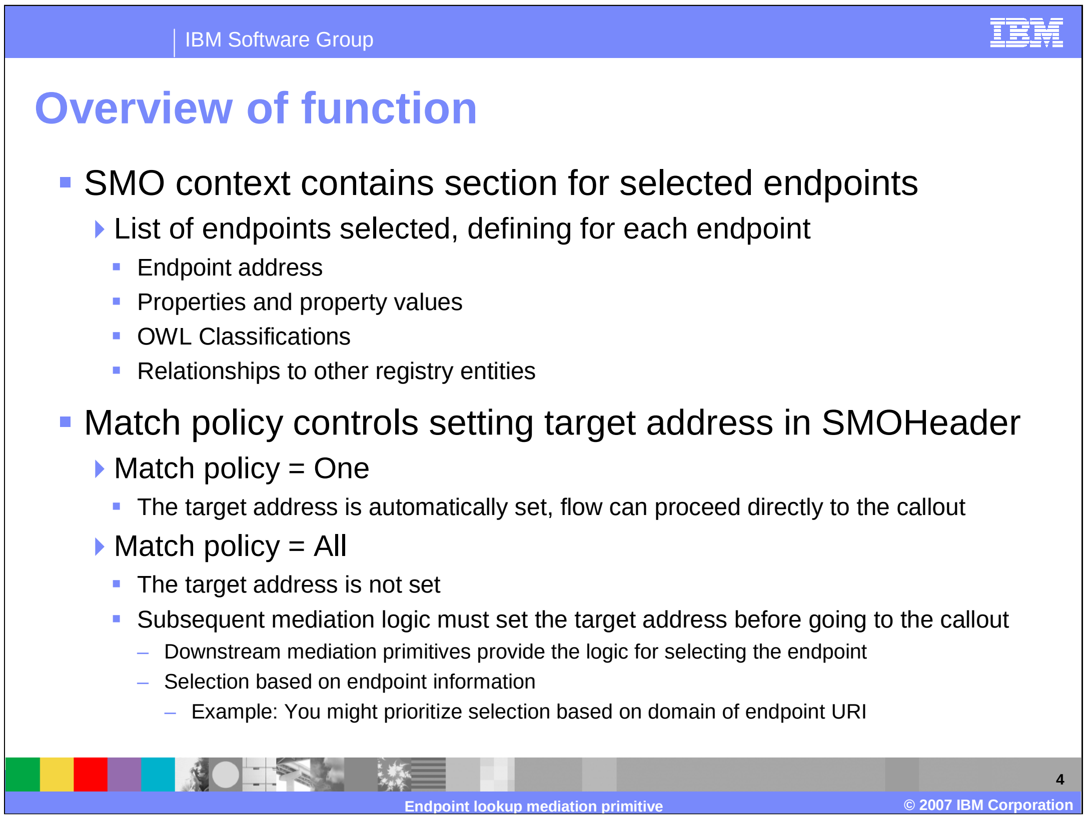
Endpoint lookup mediation primitive

© 2007 IBM Corporation

The endpoint lookup primitive uses a registry to find service provider endpoints based on a set of selection criteria. The results of the lookup are reflected in the Service Message Object which allows them to be used downstream in the mediation flow.

There are many different criteria that can be used for selection. Multiple registries can be configured for use by the servers within a WebSphere cell and the endpoint lookup can specify which of the registries should be used. The service port type can be qualified based on name, namespace and version. The Web Ontology Language (OWL), provides a classification system which can be utilized as part of the selection criteria. Registered services can be associated with name value pairs which can also be used as part of the selection criteria. These selection criteria reflect the underlying capabilities of the WebSphere Service Registry and Repository.

The match policy specifies if only one service should be returned or if all services matching the selection criteria should be returned.

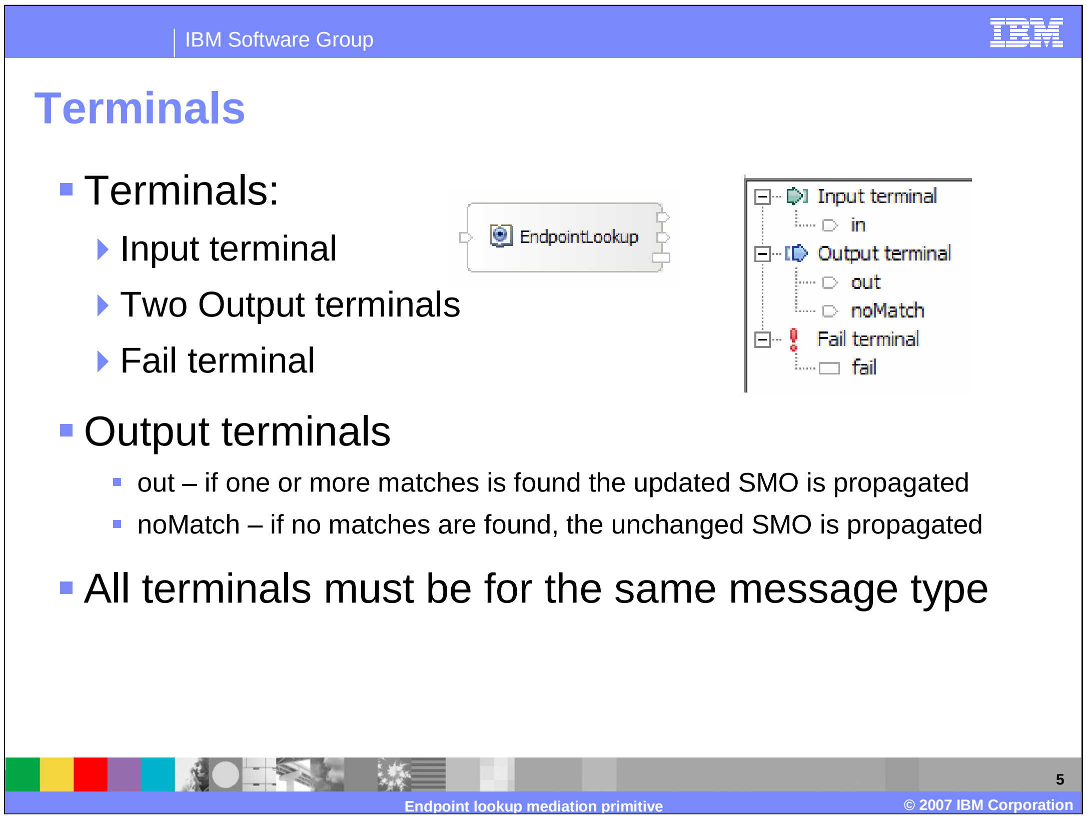


Overview of function

- SMO context contains section for selected endpoints
  - List of endpoints selected, defining for each endpoint
    - Endpoint address
    - Properties and property values
    - OWL Classifications
    - Relationships to other registry entities
- Match policy controls setting target address in SMOHeader
  - Match policy = One
    - The target address is automatically set, flow can proceed directly to the callout
  - Match policy = All
    - The target address is not set
    - Subsequent mediation logic must set the target address before going to the callout
      - Downstream mediation primitives provide the logic for selecting the endpoint
      - Selection based on endpoint information
        - Example: You might prioritize selection based on domain of endpoint URI

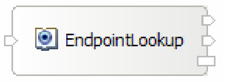
Within the context section of the SMO is a section in which the selected endpoints are placed. For each endpoint there is the endpoint address, a list of associated properties and property values, the OWL classifications and a list of relationships between the endpoint and other registry entries.

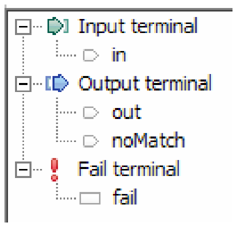
The use of dynamic callouts in mediation flows make use of a target address element in the SMOHeader. When the match policy is one, the target address element is automatically populated with the endpoint address for the selected service. If the match policy is all, the target address element is not set, even if only one endpoint was returned. Therefore, with a match policy of all, downstream processing in the mediation must perform some logic that selects an endpoint and places its address into the target address element of the SMOHeader. An example of how this might be used would be to look for an endpoint address that was within the same domain.



Terminals

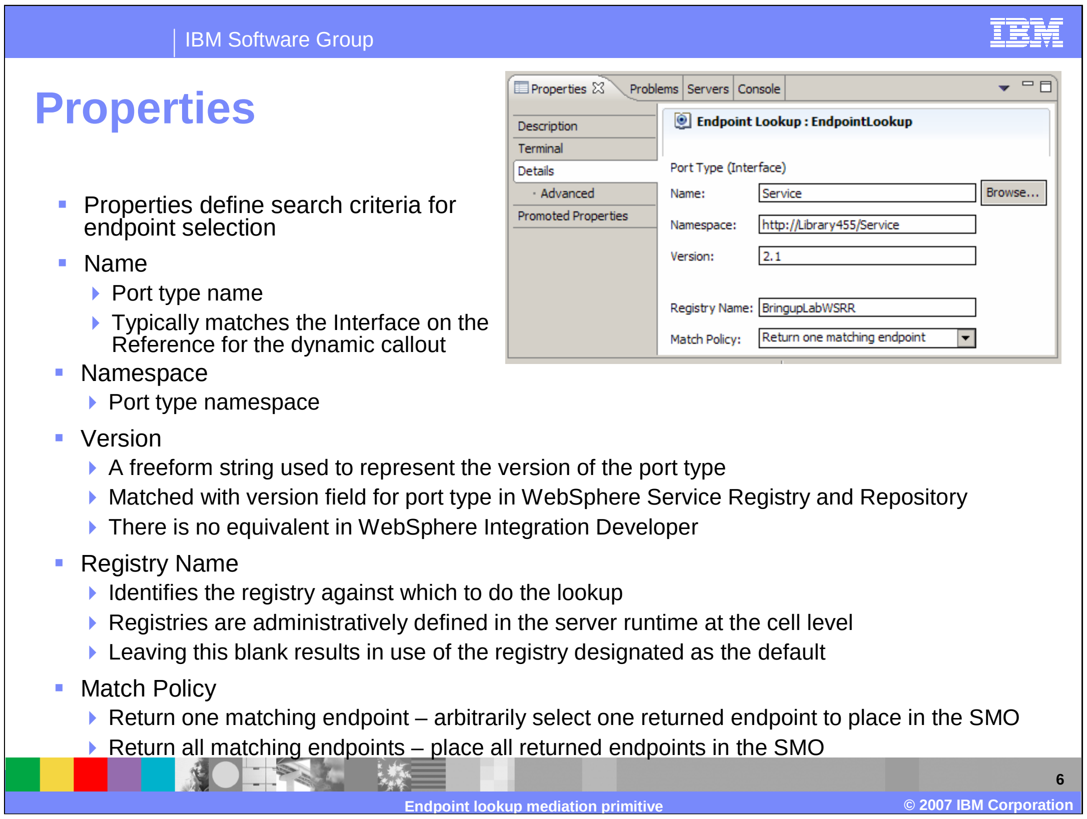
- Terminals:
  - Input terminal
  - Two Output terminals
  - Fail terminal



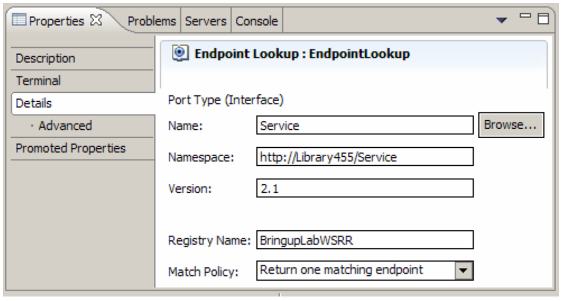


- Output terminals
  - out – if one or more matches is found the updated SMO is propagated
  - noMatch – if no matches are found, the unchanged SMO is propagated
- All terminals must be for the same message type

The endpoint lookup primitive has one input terminal, two output terminals and a fail terminal. There is one output terminal named **out** used when the endpoint lookup is successful and another output terminal named **noMatch** used when there was no service endpoint that satisfied the selection criteria. The output terminals must be for the same message type as the input terminal because the endpoint lookup primitive does not modify the message body structure. The slide shows a endpoint lookup primitive with its terminals and the terminals as seen in the properties view.



Properties



Endpoint Lookup : EndpointLookup

Port Type (Interface)

Name: Service

Namespace: http://Library455/Service

Version: 2.1

Registry Name: BringupLabWSRR

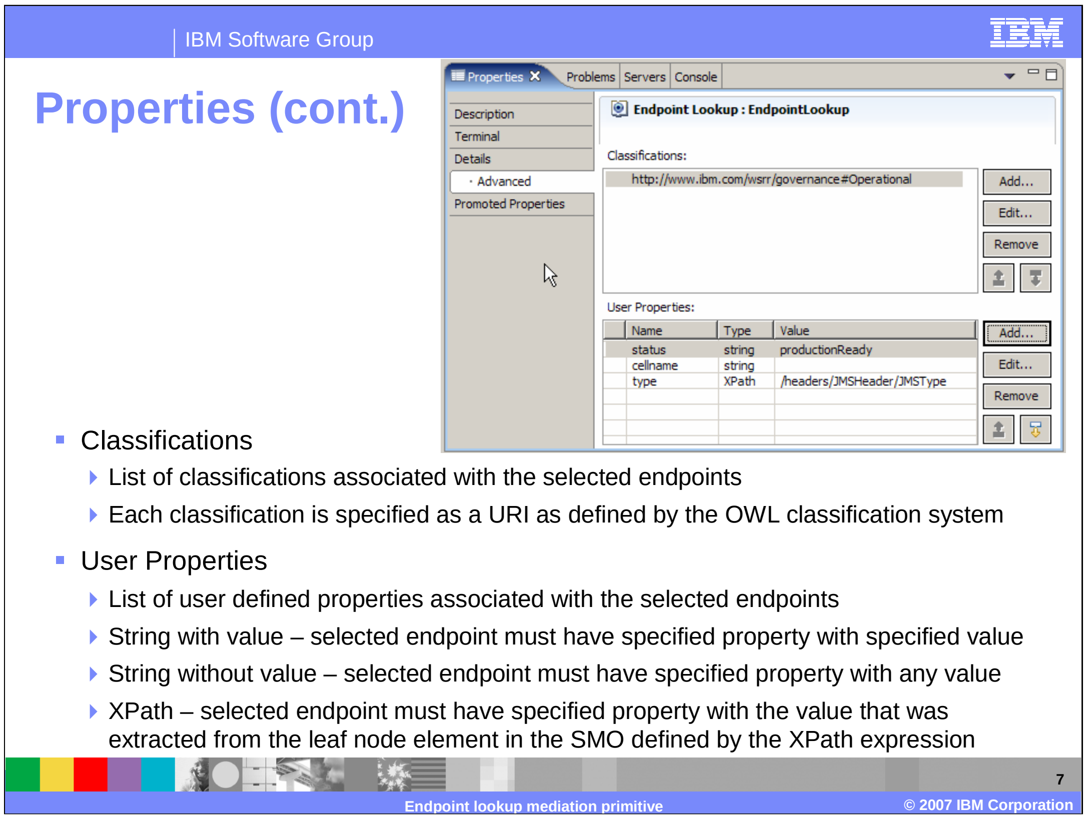
Match Policy: Return one matching endpoint

- Properties define search criteria for endpoint selection
- Name
  - Port type name
  - Typically matches the Interface on the Reference for the dynamic callout
- Namespace
  - Port type namespace
- Version
  - A freeform string used to represent the version of the port type
  - Matched with version field for port type in WebSphere Service Registry and Repository
  - There is no equivalent in WebSphere Integration Developer
- Registry Name
  - Identifies the registry against which to do the lookup
  - Registries are administratively defined in the server runtime at the cell level
  - Leaving this blank results in use of the registry designated as the default
- Match Policy
  - Return one matching endpoint – arbitrarily select one returned endpoint to place in the SMO
  - Return all matching endpoints – place all returned endpoints in the SMO

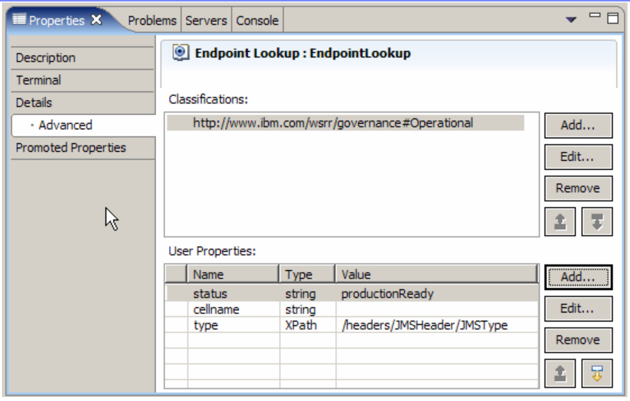
The definition of search criteria for an endpoint lookup starts with the specification of the Port Type, which defines the interface the service endpoint is to support. The two values Name and Namespace define the definition of the interface. These normally are the same as the interface specified on Mediation Flow component Reference that is a associated with the dynamic Callout node in the flow. The Version is a freeform string which is meant to identify the version of the port type. It is matched with a version specification in WebSphere Service Registry and Repository. There is no equivalent concept of a version specification for interfaces in WebSphere Integration Developer.

The next property is the Registry name, identifying the registry against which to do the lookup. The WebSphere Enterprise Service Bus and WebSphere Process Server manage registry references administratively, and this is the name the registry is known by in the server. The server designates one registry as the default, and that registry is used if this property is left blank. More details about how the server manages registry references is provided later in this presentation.

The Match Policy can be set to one or all. If set to one, a single endpoint matching the selection criteria is arbitrarily chosen and its information placed into the SMO. This includes setting the target address in the SMOHeader. If set to all, all the endpoints matching the selection criteria are returned but the target address in the SMOHeader is not updated.



Properties (cont.)



Properties | Problems | Servers | Console

Description
Terminal
Details
· Advanced
Promoted Properties

Endpoint Lookup : EndpointLookup

Classifications:

http://www.ibm.com/wsrr/governance#Operational

Add... Edit... Remove

User Properties:

Name	Type	Value
status	string	productionReady
cellname	string	
type	XPath	/headers/JMSHeader/JMSType

Add... Edit... Remove

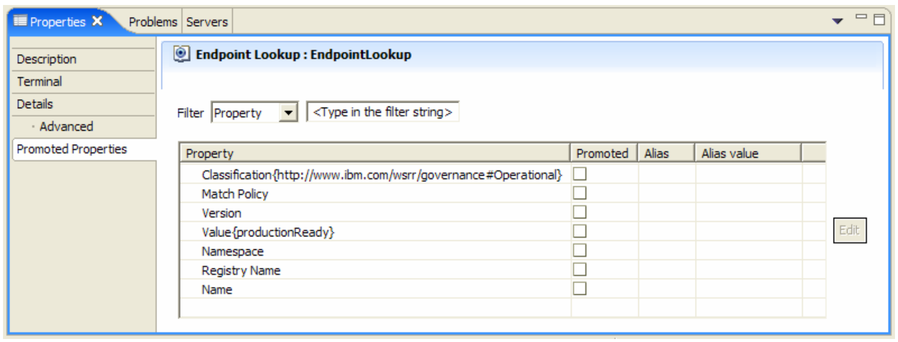
- Classifications
  - List of classifications associated with the selected endpoints
  - Each classification is specified as a URI as defined by the OWL classification system
- User Properties
  - List of user defined properties associated with the selected endpoints
  - String with value – selected endpoint must have specified property with specified value
  - String without value – selected endpoint must have specified property with any value
  - XPath – selected endpoint must have specified property with the value that was extracted from the leaf node element in the SMO defined by the XPath expression

The endpoint lookup primitive has an Advanced Details panel, as is shown here. It contains the Classifications property and the User Properties property.

The Classifications property is a list of classifications that should be associated with the selected endpoint. A classification is specified as a URI defines by the OWL classification system.

The User Properties property is a table of name value pairs that should be associated with the selected endpoint. The Name column contains the name of a user property. If the Type column contains XPath, then the Value column contains an XPath expression identifying an SMO element that contains the value for the user property. If the Type column contains string, then the Value column contains either the value for that user property or blank. When both Name and Value are present, the selected endpoint must have the specified user property with the specified value. When Value is left blank, then the selected endpoint must have the specified user property but it can have any value.

Promoted properties



Endpoint Lookup : EndpointLookup

Filter Property <Type in the filter string>

Property	Promoted	Alias	Alias value
Classification{http://www.ibm.com/wsrr/governance#Operational}	☐		
Match Policy	☐		
Version	☐		
Value{productionReady}	☐		
Namespace	☐		
Registry Name	☐		
Name	☐		

- Promotable
  - Classification (individual rows)
  - Match Policy
  - Version
  - User Properties (the Value column)
  - Namespace
  - Registry Name
  - Name

All of the properties for the endpoint lookup are promotable.

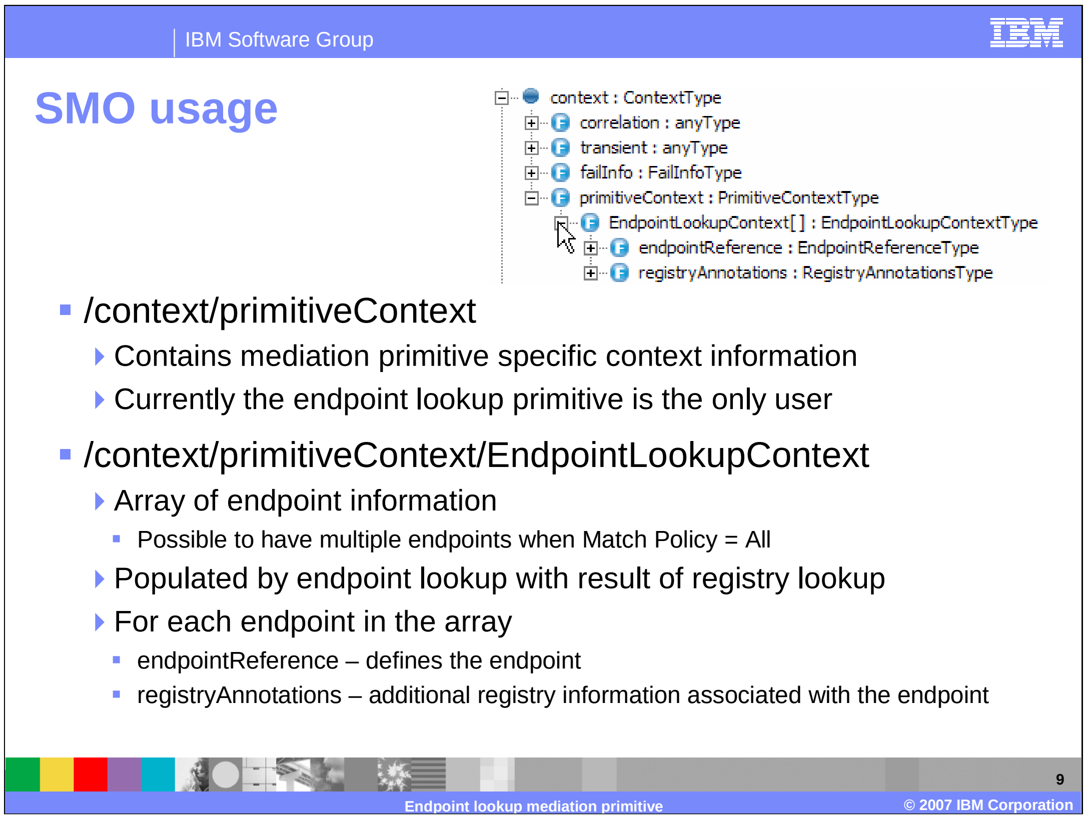
The Classification property is a table with a single column. Individual rows in the table can be promoted.

The User Properties table designates the Value column as promotable, allowing values for individual rows to be promoted.

Promoting the Registry name would allow administrators to dynamically manage which registry is being used.

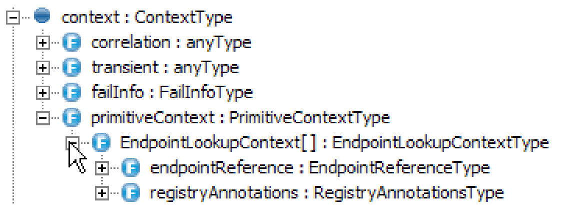
The remaining properties are all promotable, but promoting them and changing their values at runtime would have to be done with great care. Changing the Port Type by changing the Name, Namespace or Version could result in a failure if the newly selected endpoints have an interface which is incompatible with the definition of the flow.

Changing the Match Policy would typically imply that the mediation flow logic would need to be changed as well. Therefore, for most circumstances, this does not appear to be something that should be dynamically changed.



IBM Software Group

SMO usage



context : ContextType
correlation : anyType
transient : anyType
failInfo : FailInfoType
primitiveContext : PrimitiveContextType
EndpointLookupContext[ ] : EndpointLookupContextType
endpointReference : EndpointReferenceType
registryAnnotations : RegistryAnnotationsType

- /context/primitiveContext
  - Contains mediation primitive specific context information
  - Currently the endpoint lookup primitive is the only user
- /context/primitiveContext/EndpointLookupContext
  - Array of endpoint information
    - Possible to have multiple endpoints when Match Policy = All
  - Populated by endpoint lookup with result of registry lookup
  - For each endpoint in the array
    - endpointReference – defines the endpoint
    - registryAnnotations – additional registry information associated with the endpoint

9

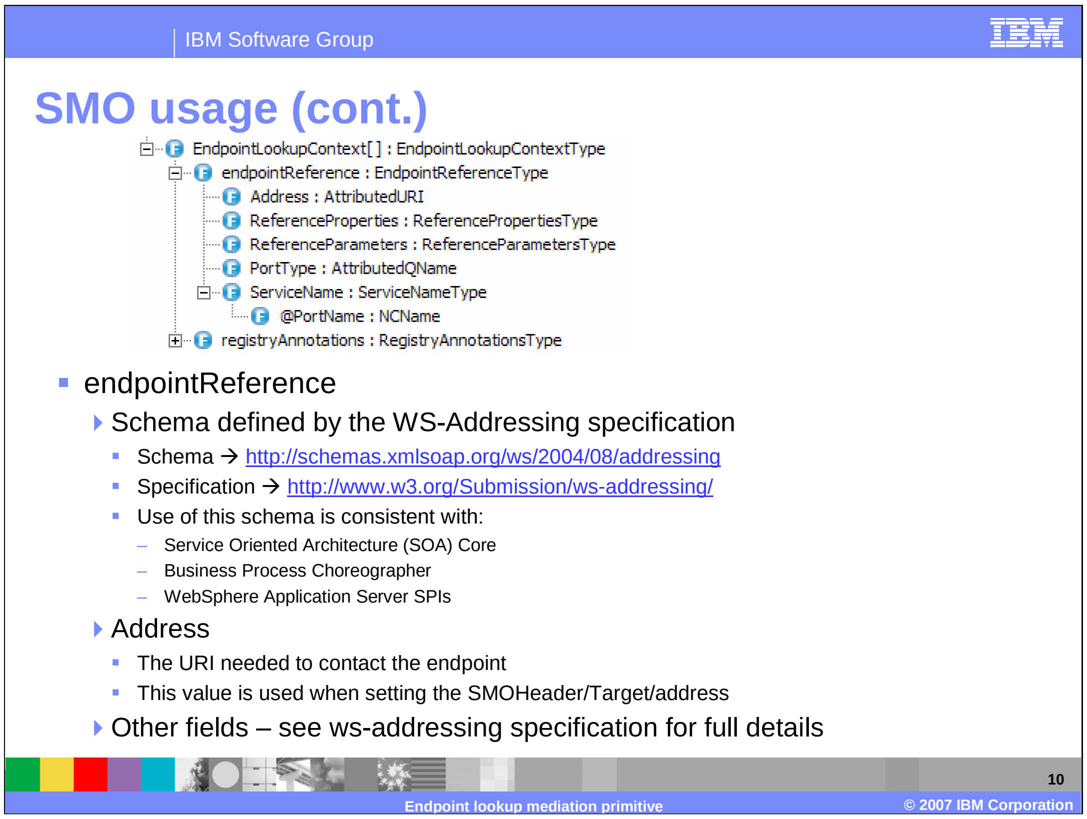
Endpoint lookup mediation primitive

© 2007 IBM Corporation

The next few slides look at the SMO content that is specific to the endpoint lookup primitive.

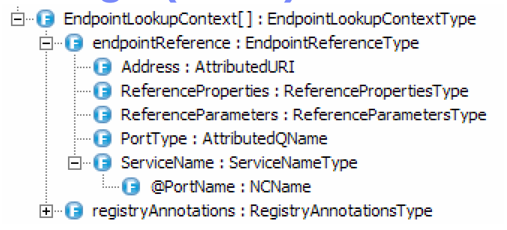
Looking at the screen capture, you see that there is a primitiveContext contained within the context section of the SMO. Its purpose it to allow mediation primitive types to define specific usage of the SMO for their type. Currently, only the endpoint lookup primitive type makes use of this context.

Looking again at the screen capture, you see that the primitiveContext contains an EndpointLookupContext which is an array of endpoint information. This array is populated by the endpoint lookup primitive with the results of a registry lookup. Each endpoint in the array has endpointReference data defining the endpoint and registyAnnotations data defining additional information about the endpoint from the registry. These are both covered in more detail on the following slides.



IBM Software Group

SMO usage (cont.)



```
EndpointLookupContext[ ] : EndpointLookupContextType
  endpointReference : EndpointReferenceType
    Address : AttributedURI
    ReferenceProperties : ReferencePropertiesType
    ReferenceParameters : ReferenceParametersType
    PortType : AttributedQName
    ServiceName : ServiceNameType
      @PortName : NCName
  registryAnnotations : RegistryAnnotationsType
```

- endpointReference
  - Schema defined by the WS-Addressing specification
    - Schema → http://schemas.xmlsoap.org/ws/2004/08/addressing
    - Specification → http://www.w3.org/Submission/ws-addressing/
    - Use of this schema is consistent with:
      - Service Oriented Architecture (SOA) Core
      - Business Process Choreographer
      - WebSphere Application Server SPIs
  - Address
    - The URI needed to contact the endpoint
    - This value is used when setting the SMOHeader/Target/address
  - Other fields – see ws-addressing specification for full details

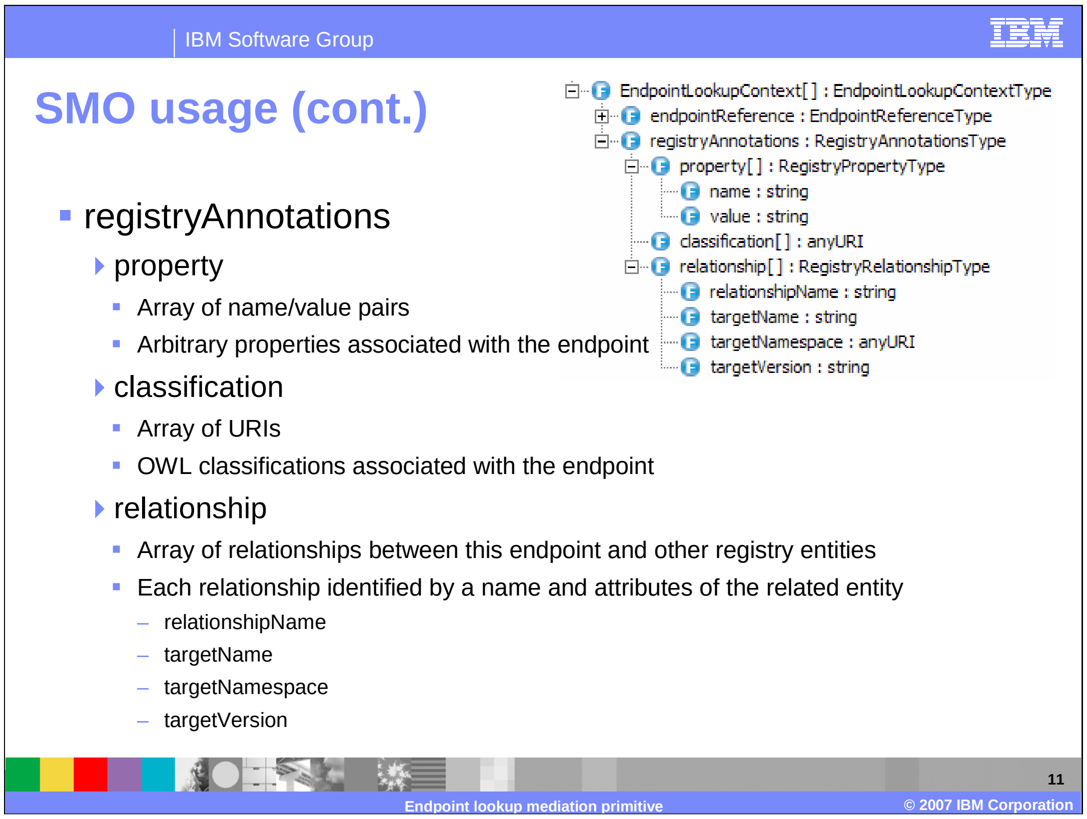
10

Endpoint lookup mediation primitive

© 2007 IBM Corporation

The schema for the endpointReference is defined by the WS-Addressing specification as defined by the World Wide Web Consortium (W3C). Looking at the slide, you see the URLs identifying where to find this schema definition and specification. Using this schema provides consistency with the Service Oriented Architecture Core, Business Process Choreographer and WebSphere Application Server SPIs.

This presentation does not attempt to describe this schema. The key element from the schema to understand is the Address element. This contains the URI that is needed to contact the service endpoint. It is this value that would be placed into the target address element of the SMOHeader for use by a dynamic callout.



IBM Software Group

SMO usage (cont.)

```
EndpointLookupContext[ ] : EndpointLookupContextType
  endpointReference : EndpointReferenceType
  registryAnnotations : RegistryAnnotationsType
    property[ ] : RegistryPropertyType
      name : string
      value : string
    classification[ ] : anyURI
    relationship[ ] : RegistryRelationshipType
      relationshipName : string
      targetName : string
      targetNamespace : anyURI
      targetVersion : string
```

- registryAnnotations
  - property
    - Array of name/value pairs
    - Arbitrary properties associated with the endpoint
  - classification
    - Array of URIs
    - OWL classifications associated with the endpoint
  - relationship
    - Array of relationships between this endpoint and other registry entities
    - Each relationship identified by a name and attributes of the related entity
      - relationshipName
      - targetName
      - targetNamespace
      - targetVersion

11

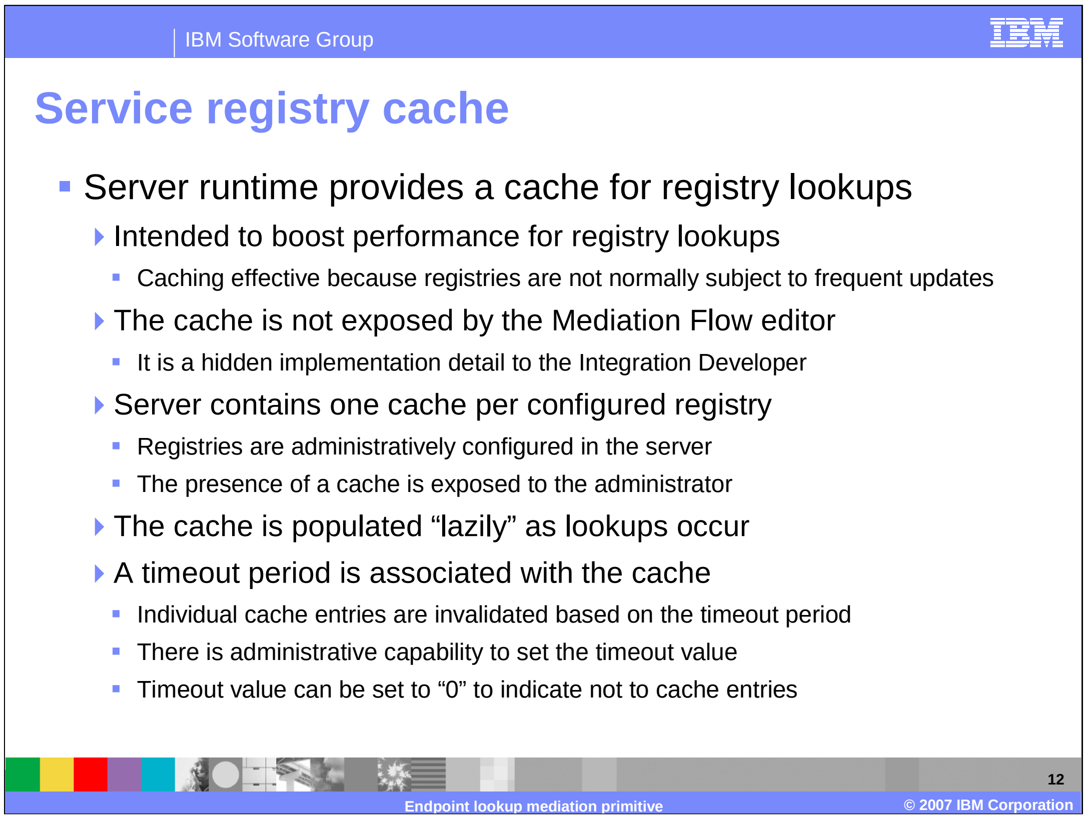
Endpoint lookup mediation primitive © 2007 IBM Corporation

This slide examines the registryAnnotations, which is composed of three arrays.

The first is the property array that contains the name value pairs for the user properties associated with the endpoint.

The next is the classification array, containing the URIs of the OWL classifications associated with the endpoint.

Finally, there is the relationship array. It contains information about the relationship between this endpoint and other entities in the registry. Each relationship is defined by a relationship name and the name, namespace and version of the target entity.



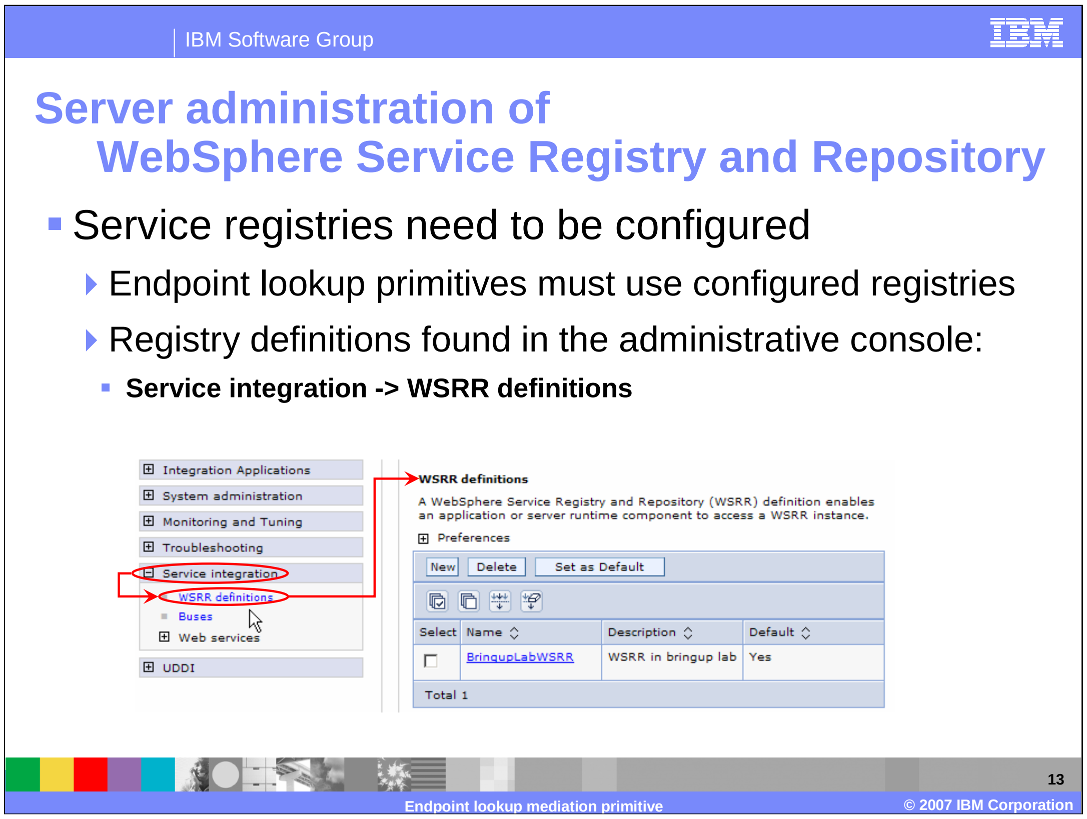
Service registry cache

- Server runtime provides a cache for registry lookups
  - Intended to boost performance for registry lookups
    - Caching effective because registries are not normally subject to frequent updates
  - The cache is not exposed by the Mediation Flow editor
    - It is a hidden implementation detail to the Integration Developer
  - Server contains one cache per configured registry
    - Registries are administratively configured in the server
    - The presence of a cache is exposed to the administrator
  - The cache is populated "lazily" as lookups occur
  - A timeout period is associated with the cache
    - Individual cache entries are invalidated based on the timeout period
    - There is administrative capability to set the timeout value
    - Timeout value can be set to "0" to indicate not to cache entries

Performing frequent lookups to the WebSphere Service Registry and Repository can be expensive in terms of performance. Registries do not normally have frequent updates, so caching is a viable way to address this. Therefore, WebSphere Enterprise Service Bus and WebSphere Process Server provide a registry cache which is intended to boost the overall performance of registry lookups by reducing the number of calls made to the registry.

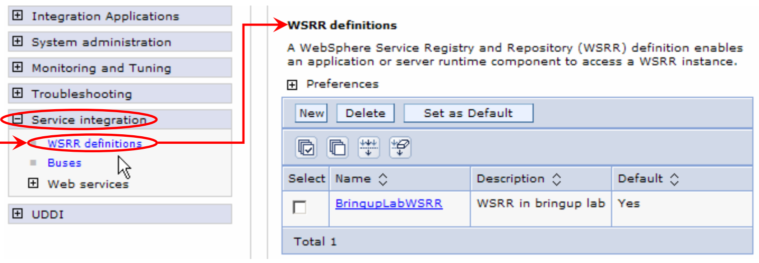
The endpoint lookup primitive and the Mediation Flow editor do not expose the registry cache. Therefore, from an integration developers perspective, the cache is hidden.

The cache is exposed to the server administrator who is responsible for configuring the registries. There is one registry cache per registry configured with the server. Population of the cache occurs in a lazy fashion, with entries being added as lookups occur. The administrator can assign a timeout period that is applied to cache entries, causing entries to become invalid once the are older than the timeout period. If a timeout period is set to zero, the effective result is that caching does not take place.



Server administration of WebSphere Service Registry and Repository

- Service registries need to be configured
  - Endpoint lookup primitives must use configured registries
  - Registry definitions found in the administrative console:
    - **Service integration -> WSRR definitions**



WSRR definitions

A WebSphere Service Registry and Repository (WSRR) definition enables an application or server runtime component to access a WSRR instance.

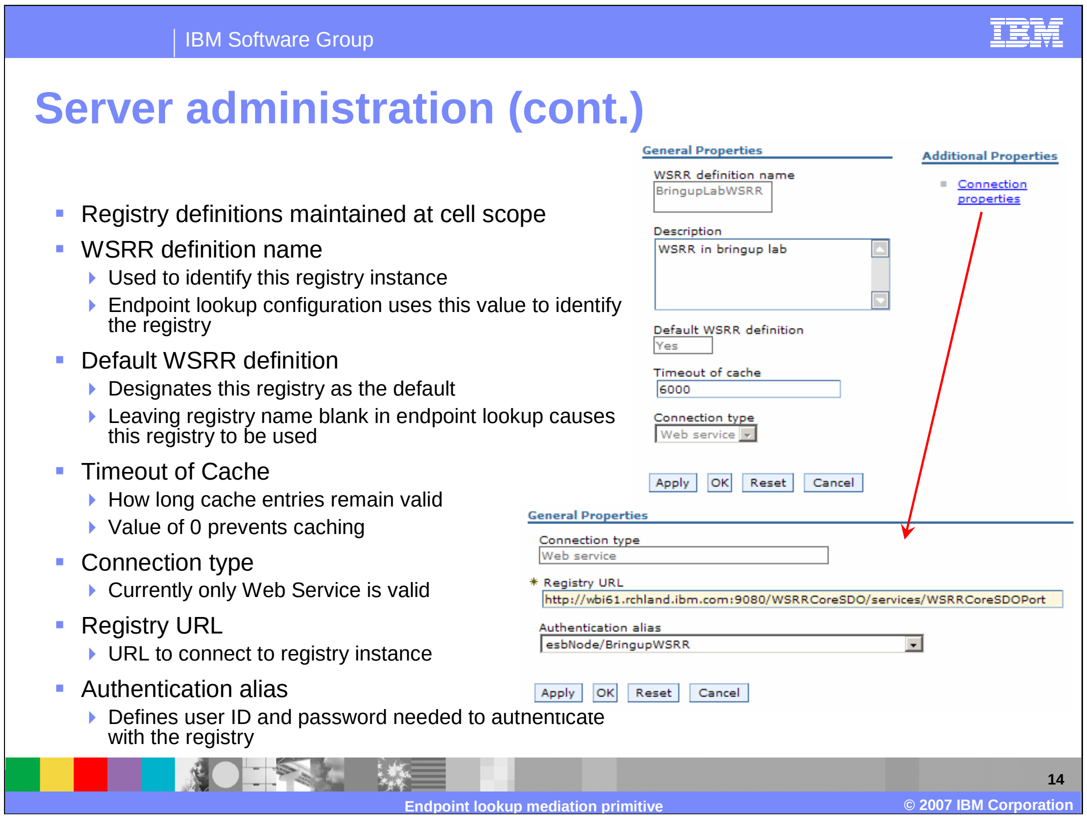
Select	Name	Description	Default
☐	BringupLabWSRR	WSRR in bringup lab	Yes

Total 1

The next couple of slides show how the WebSphere Service Registry and Repository registries are administered in WebSphere Enterprise Service Bus and WebSphere Process Server.

Note that in order for an endpoint lookup primitive to use a registry, it must be administratively defined as described here.

This slide shows the navigation in the Administrative Console. On the left panel, open Service integration and then select WSRR definitions. This opens a panel containing a list of the configured registries. Clicking on one of the registries causes it to open in a registry properties panel, as shown on the next slide.



Server administration (cont.)

- Registry definitions maintained at cell scope
- WSRR definition name
  - Used to identify this registry instance
  - Endpoint lookup configuration uses this value to identify the registry
- Default WSRR definition
  - Designates this registry as the default
  - Leaving registry name blank in endpoint lookup causes this registry to be used
- Timeout of Cache
  - How long cache entries remain valid
  - Value of 0 prevents caching
- Connection type
  - Currently only Web Service is valid
- Registry URL
  - URL to connect to registry instance
- Authentication alias
  - Defines user ID and password needed to authenticate with the registry

General Properties

WSRR definition name: BringupLabWSRR

Description: WSRR in bringup lab

Default WSRR definition: Yes

Timeout of cache: 6000

Connection type: Web service

Additional Properties

- Connection properties

General Properties

Connection type: Web service

* Registry URL: http://wbi61.rchland.ibm.com:9080/WSRRCoreSDO/services/WSRRCoreSDOPort

Authentication alias: esbNode/BringupWSRR

This slide shows the properties that are used to configure a WebSphere Service Registry and Repository. Within a WebSphere cell, these definitions are maintained only at a cell scope, and cannot be specified at the node or server scope.

The first field is the 'WSRR definition name' field; this property is a name used to identify this registry instance. This is the value that you specify in the endpoint lookup primitive to identify the registry to use for the lookup.

The Description property provides a text comment describing this registry.

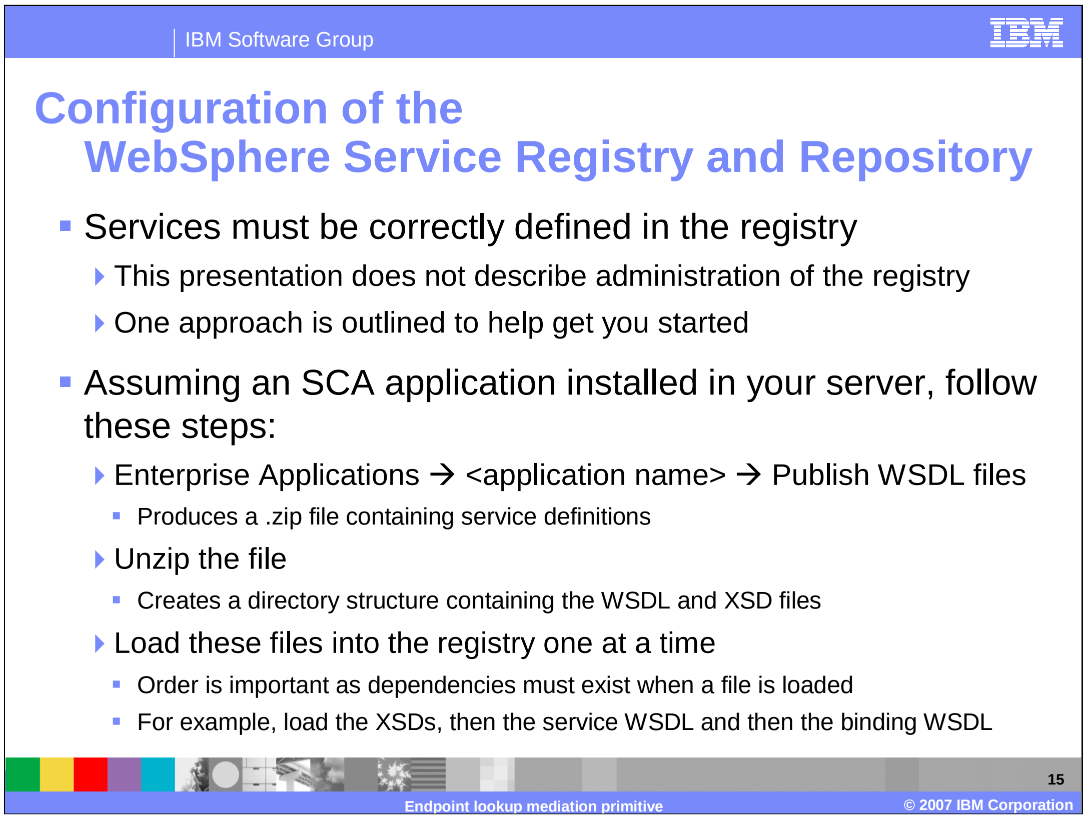
The 'Default WSRR definition' field, if set to yes, designates this registry instance as the default registry. The default registry is the one that is used when the registry name property in the endpoint lookup is left blank.

The Timeout of cache property specifies how long cached entries remain valid. A value of zero prevents any caching of entries for this registry instance.

The Connection type defines what type of protocol is used to connect with the registry. Currently, Web service is the only valid connection type.

The Registry URL property provides the specific URL needed to connect to this registry instance.

The Authentication alias property identifies an authentication alias containing the user ID and password needed to authenticate with the registry.


IBM Software Group

Configuration of the WebSphere Service Registry and Repository

- Services must be correctly defined in the registry
  - This presentation does not describe administration of the registry
  - One approach is outlined to help get you started
- Assuming an SCA application installed in your server, follow these steps:
  - Enterprise Applications → <application name> → Publish WSDL files
    - Produces a .zip file containing service definitions
  - Unzip the file
    - Creates a directory structure containing the WSDL and XSD files
  - Load these files into the registry one at a time
    - Order is important as dependencies must exist when a file is loaded
    - For example, load the XSDs, then the service WSDL and then the binding WSDL

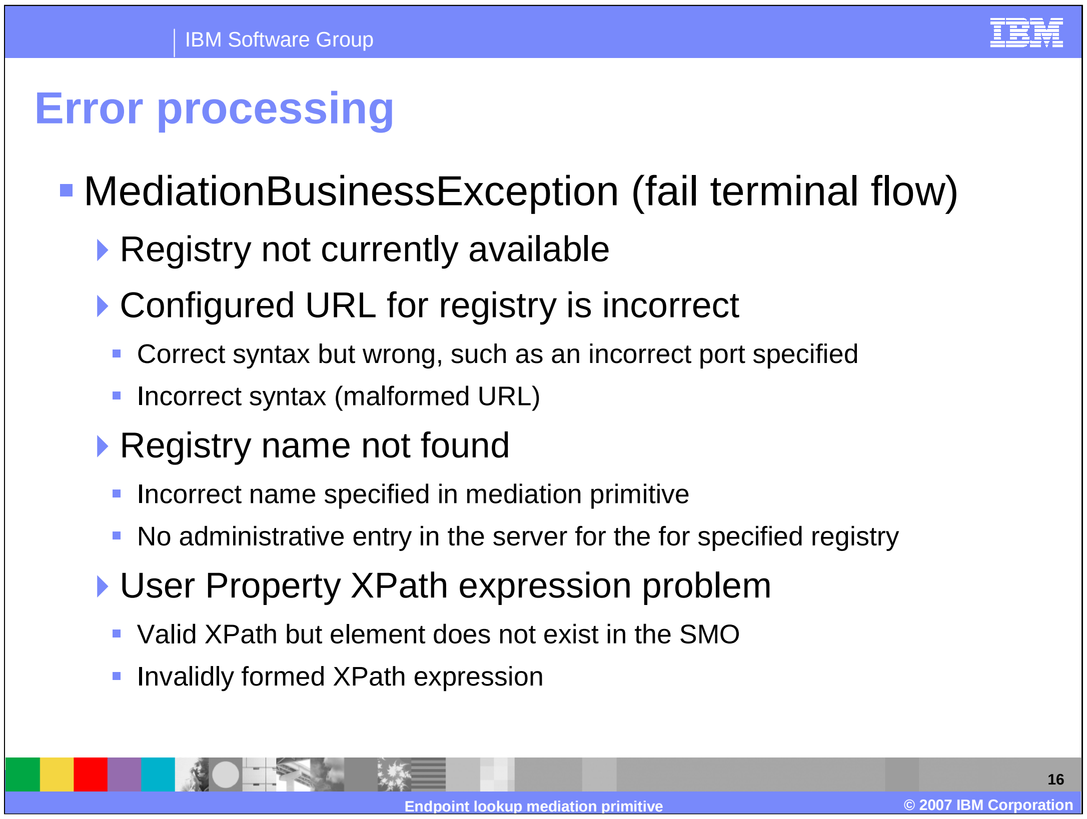
15

Endpoint lookup mediation primitive © 2007 IBM Corporation

Describing how to configure and administer the WebSphere Service Registry and Repository is beyond the scope of this presentation. However, it is important that the services be correctly defined in the registry if the endpoint lookup requests are to result in matches. This slide outlines one approach to registering your services.

The assumption is that you have an SCA application installed in WebSphere Enterprise Service Bus or WebSphere Process Server. To register the services of the SCA application in the registry, these are the basic steps to follow.

First, using the Administrative Console, navigate to Enterprise Applications, SCA application name and then select Publish WSDL files. This produces a .zip file that contains the service definitions. You should then unzip the file, which creates a directory structure containing the WSDL and XSD files needed to describe the services. Using the WebSphere Service Registry and Repository, load these files one at a time. The order in which they are loaded is significant, as dependencies must be met when a file is first loaded. Therefore, you need to do things like load the XSD files that define the business objects first, then load the WSDL files that define the interfaces and finally load the WSDL files defining the bindings.



Error processing

- MediationBusinessException (fail terminal flow)
  - Registry not currently available
  - Configured URL for registry is incorrect
    - Correct syntax but wrong, such as an incorrect port specified
    - Incorrect syntax (malformed URL)
  - Registry name not found
    - Incorrect name specified in mediation primitive
    - No administrative entry in the server for the for specified registry
  - User Property XPath expression problem
    - Valid XPath but element does not exist in the SMO
    - Invalidly formed XPath expression

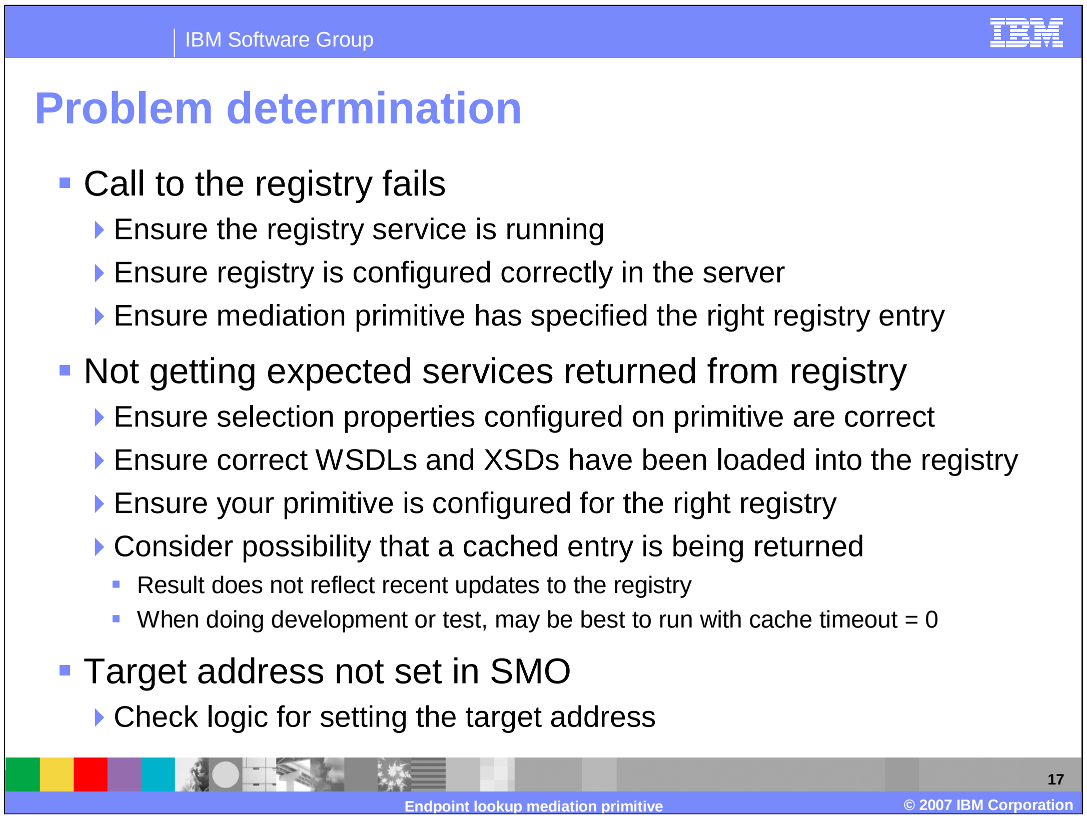
There are several conditions that would cause the MediationBusinessException to occur. When any of these happen, the mediation flow continues through the fail terminal if it is wired, otherwise the exception is re-thrown and the mediation flow is ended.

One issue is that the registry specified is not currently available and therefore cannot be contacted with the lookup request.

Another possible reason would be if the URL for the registry had been incorrectly specified in the administrative definition of the registry. This could either be a URL with a good syntax but a mistake such as an incorrect host or port specification, or it could also be for a malformed URL.

The MediationBusinessException can also occur if the administrative definition for the registry specified in the endpoint lookup properties cannot be found. This could occur if the administrative name for the registry was misspelled, or if the registry was never administratively defined.

Another reason for this exception would be if the User Properties table contained an XPath expression which could not be resolved. This could occur if the XPath expression was incorrect or if this instance of the SMO did not happen to have the element defined by the XPath.



Problem determination

- Call to the registry fails
  - Ensure the registry service is running
  - Ensure registry is configured correctly in the server
  - Ensure mediation primitive has specified the right registry entry
- Not getting expected services returned from registry
  - Ensure selection properties configured on primitive are correct
  - Ensure correct WSDLs and XSDs have been loaded into the registry
  - Ensure your primitive is configured for the right registry
  - Consider possibility that a cached entry is being returned
    - Result does not reflect recent updates to the registry
    - When doing development or test, may be best to run with cache timeout = 0
- Target address not set in SMO
  - Check logic for setting the target address

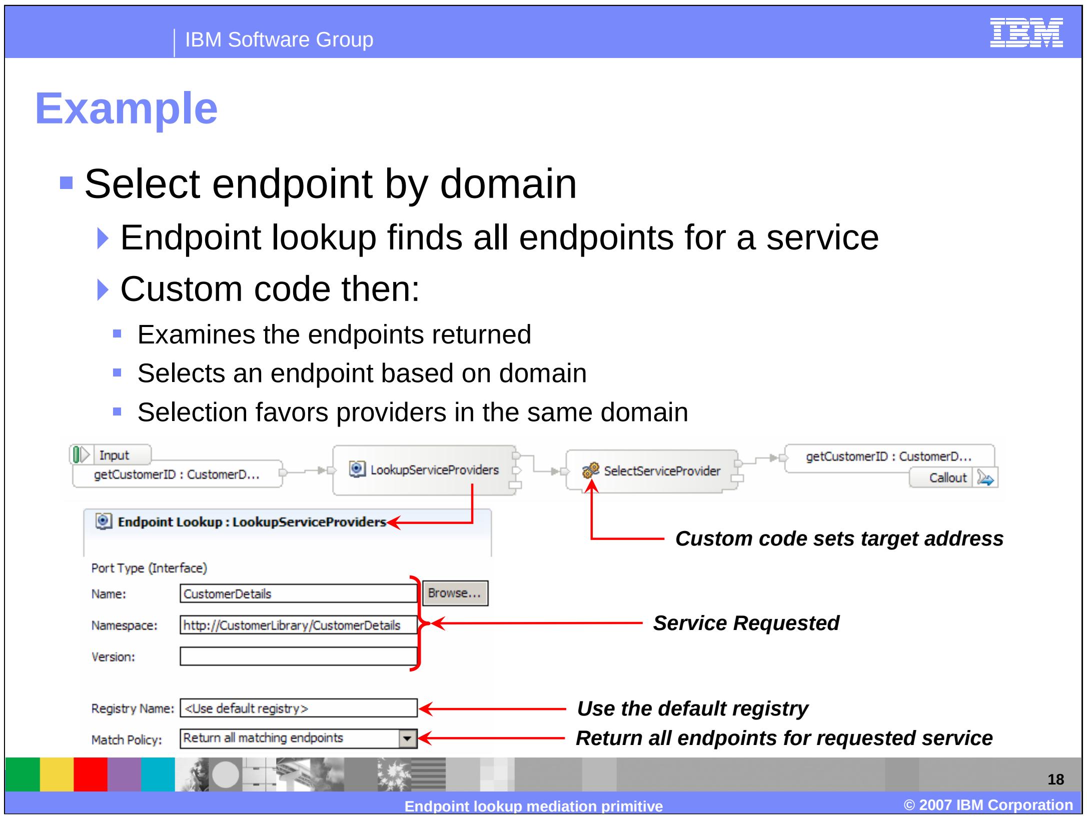
Because of the interaction between the endpoint lookup primitive, the server administration of registries and the call to the registry, there are several things that could go wrong. This slide gives you some things to look for when your endpoint lookup primitive fails at runtime.

The first set of issues revolve around the call to the registry failing. If this is the case, ensure that the registry is running and that the configuration of the registry in the server is correct. Also check that the mediation primitive has specified the correct administrative name for the registry to be used.

The next set of issues revolve around the result of a lookup not returning the expected services. The first thing to check would be the configuration of the endpoint lookup primitive to ensure that the various search criteria properties are correctly specified. If so, then make sure the correct XSD and WSDL files have been loaded into the registry. And finally, make sure that the endpoint lookup is configured to use the right registry.

If there have been any registry updates, consider the possibility that your endpoint lookup is being resolved through the registry cache rather than a call to the registry. This is not likely to be an issue in a production environment, but is definitely an issue during development and test. As a best practice, use a cache timeout of zero to prevent caching during the development phase and possibly also during the test phase.

If the dynamic callout is failing because the target address is not set in the SMO, there are a couple of things you can check. If the endpoint lookup is configured to have a Match Policy of one, check to see why no endpoint was returned from the lookup. If the endpoint lookup is configured to have a Match Policy of all and one or more endpoints was returned, check the mediation flow logic following the endpoint lookup to determine why the target address was not set.



Example

- Select endpoint by domain
  - Endpoint lookup finds all endpoints for a service
  - Custom code then:
    - Examines the endpoints returned
    - Selects an endpoint based on domain
    - Selection favors providers in the same domain

Endpoint Lookup : LookupServiceProviders

Port Type (Interface)

Name:	CustomerDetails
Namespace:	http://CustomerLibrary/CustomerDetails
Version:	
Registry Name:	<Use default registry>
Match Policy:	Return all matching endpoints

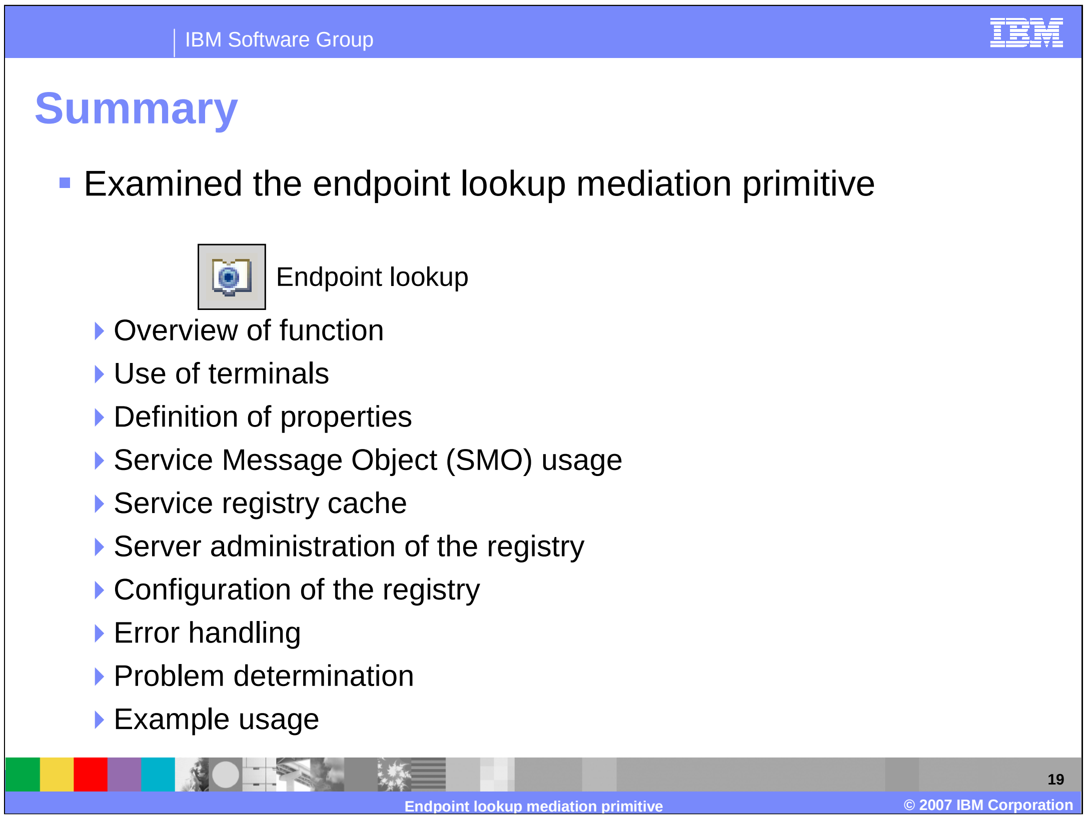
Custom code sets target address

Service Requested

Use the default registry

Return all endpoints for requested service

In this example, the selection of the endpoint is optimized to use a service provider in the same domain as the mediation. In the screen capture, you can see the mediation flow and the properties for the endpoint lookup. Notice that the lookup uses the default registry configured in the server and the match policy results in all endpoints for this service being returned. Following the endpoint lookup is a custom mediation primitive. It examines the address URIs of the returned endpoints looking for one that is in the same domain. If found, it places that into the target address element of the SMOHeader. If none are in the same domain, one endpoint is arbitrarily picked.



Summary

- Examined the endpoint lookup mediation primitive

  Endpoint lookup

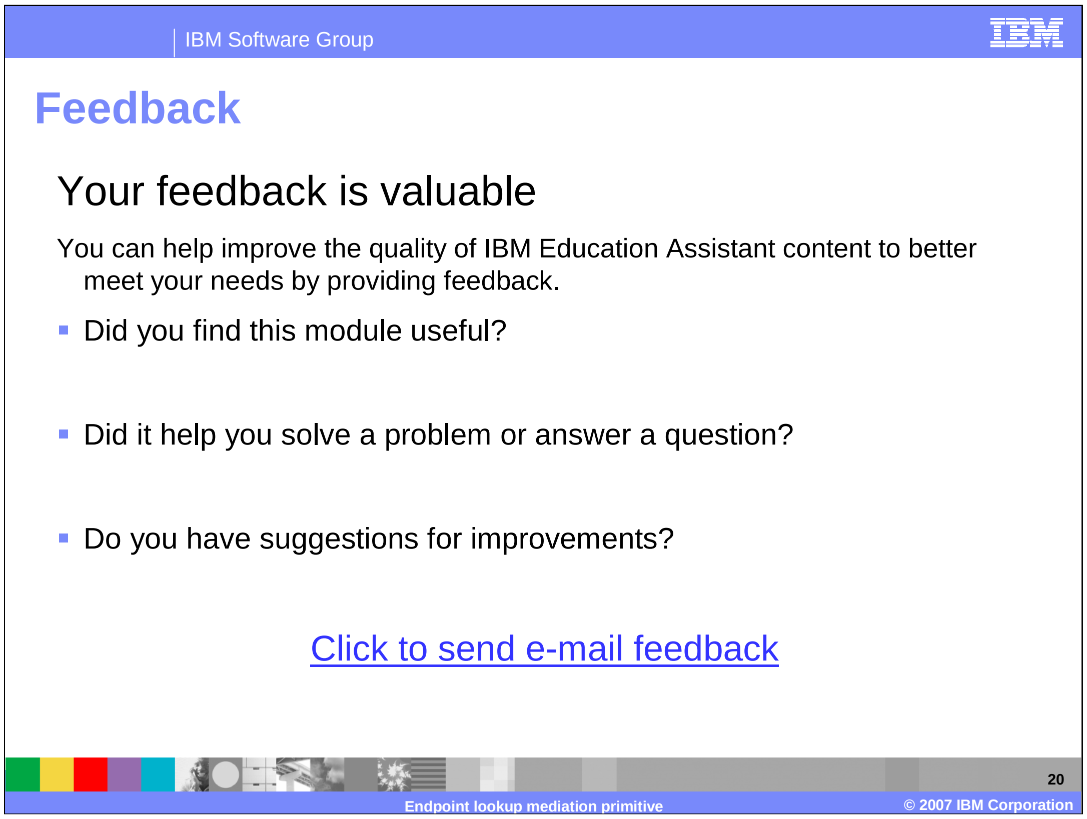
  - Overview of function
  - Use of terminals
  - Definition of properties
  - Service Message Object (SMO) usage
  - Service registry cache
  - Server administration of the registry
  - Configuration of the registry
  - Error handling
  - Problem determination
  - Example usage

In summary, this presentation introduced you to an overview of the function provided by the endpoint lookup primitive, along with information about the primitive’s use of terminals and its properties.

Since this primitive has a special relationship to the Service Message Object elements in the SMO context, the schema for this was examined.

In order to understand the endpoint lookup primitive, it is necessary to understand more than the behaviour of the primitive itself and how it fits into a mediation flow. The endpoint lookup primitive interfaces with the WebSphere Service Registry and Repository. Use of the registry is enabled through capabilities provided by the WebSphere Enterprise Service Bus and the WebSphere Process Server. You learned about the service registry cache and the administration of registries that is provided by the server. You were also introduced to an approach for registry configuration.

The presentation then returned to looking specifically at the primitive, covering error handling, problem determination and an example of its usage.



IBM Software Group

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback

20

Endpoint lookup mediation primitive

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers