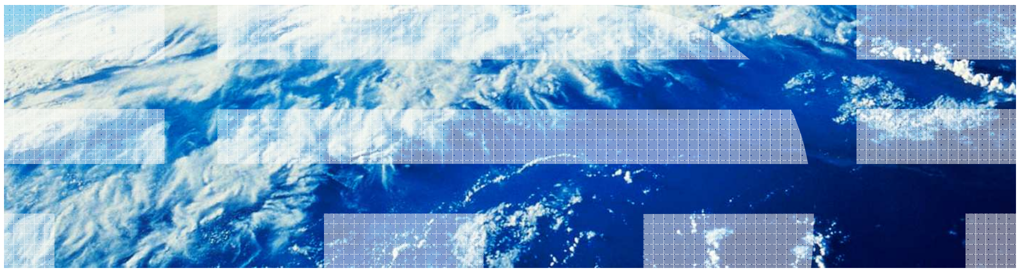
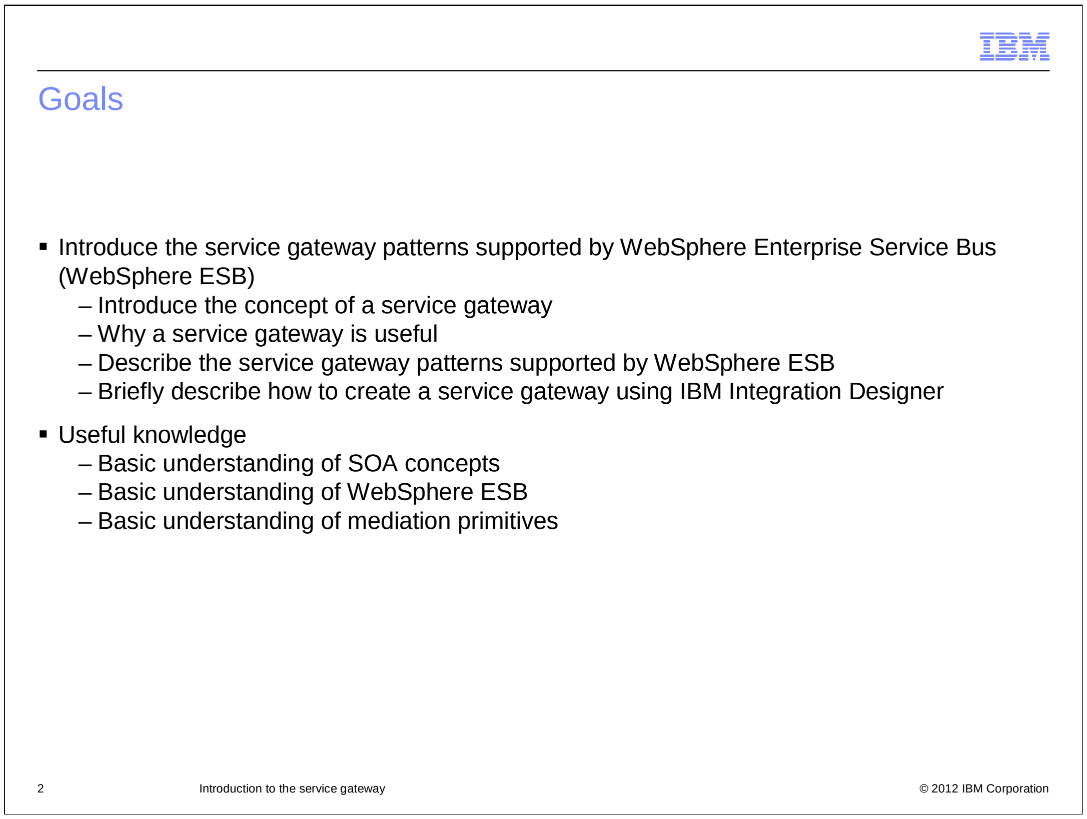
# WebSphere Enterprise Service Bus

## Introduction to the service gateway



This presentation provides an overview of the service gateway patterns supported by the WebSphere Enterprise Service Bus, also known as WebSphere ESB, product.

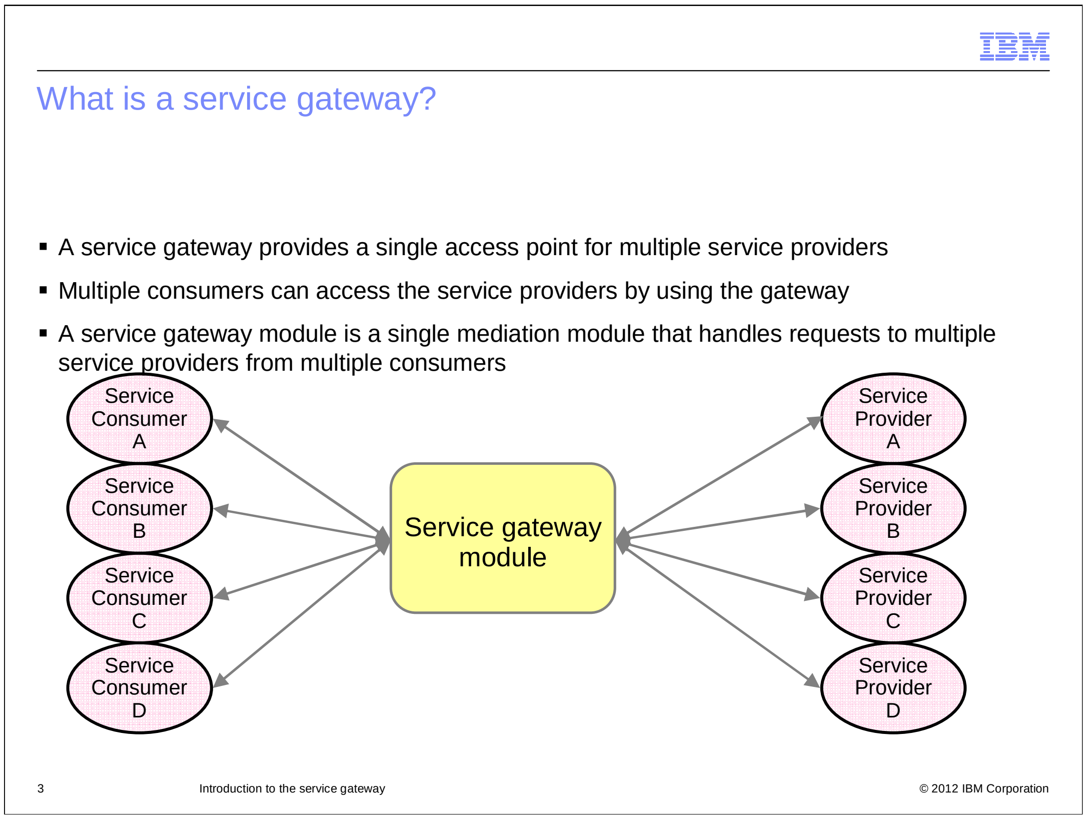


## Goals

- Introduce the service gateway patterns supported by WebSphere Enterprise Service Bus (WebSphere ESB)
  – Introduce the concept of a service gateway
  – Why a service gateway is useful
  – Describe the service gateway patterns supported by WebSphere ESB
  – Briefly describe how to create a service gateway using IBM Integration Designer
- Useful knowledge
  – Basic understanding of SOA concepts
  – Basic understanding of WebSphere ESB
  – Basic understanding of mediation primitives

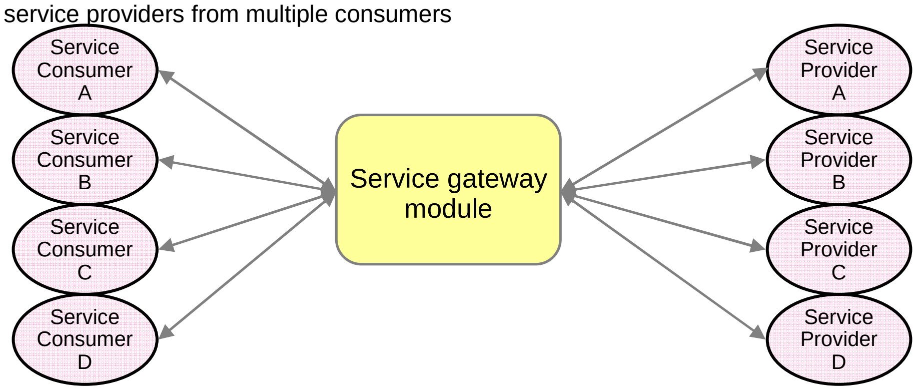
2 Introduction to the service gateway © 2012 IBM Corporation

The goal of this presentation is to introduce you to the service gateway patterns supported by the WebSphere ESB product. You will learn the basic concepts of a service gateway and why it is useful. Then you will learn about the three service gateway patterns supported by the WebSphere ESB product. Finally you will learn how to create a service gateway using IBM Integration Designer. This presentation assumes you have a basic understanding of Service Oriented Architecture concepts and of the WebSphere ESB product and that you understand what a mediation primitive is.

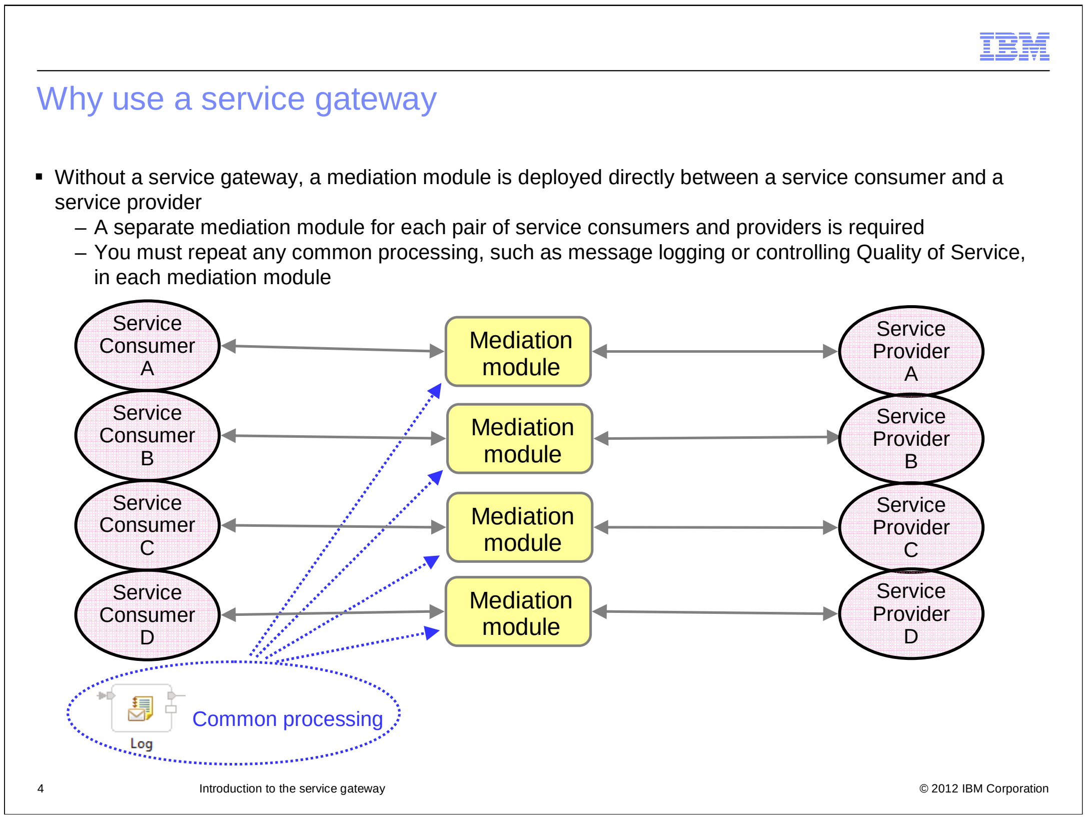


## What is a service gateway?

- A service gateway provides a single access point for multiple service providers
- Multiple consumers can access the service providers by using the gateway
- A service gateway module is a single mediation module that handles requests to multiple service providers from multiple consumers

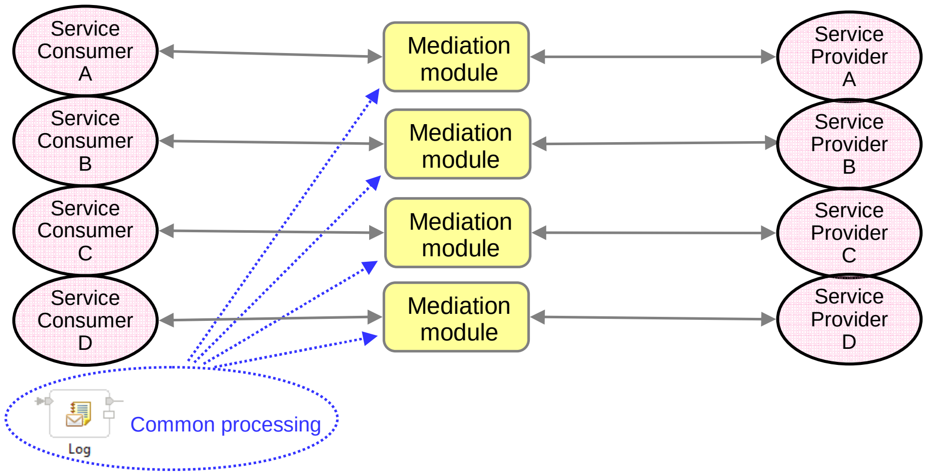


A service gateway provides a single point of access to multiple service providers. Multiple service consumers can access the service providers by using the gateway and a many-to-many relationship can exist between the service consumers and providers. A service gateway module is a single mediation module that implements a service gateway, and provides access to multiple service providers from multiple consumers.



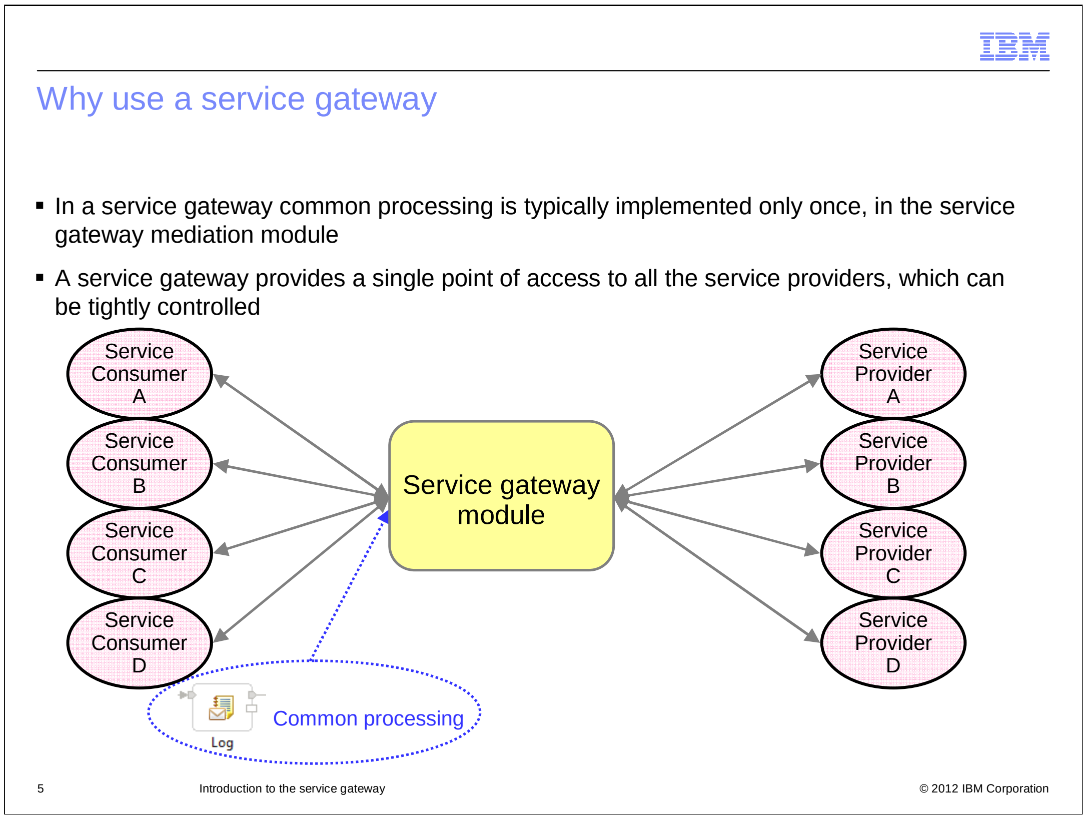
## Why use a service gateway

- Without a service gateway, a mediation module is deployed directly between a service consumer and a service provider
  - A separate mediation module for each pair of service consumers and providers is required
  - You must repeat any common processing, such as message logging or controlling Quality of Service, in each mediation module



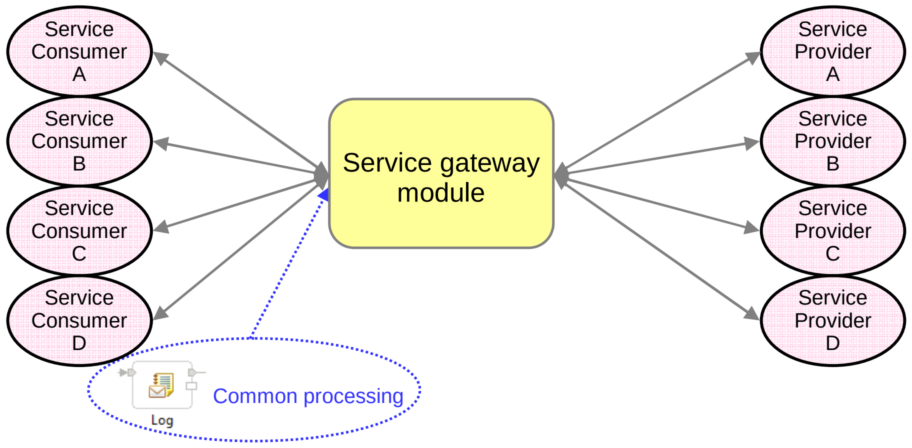
4 Introduction to the service gateway © 2012 IBM Corporation

Here is a typical configuration when a service gateway is not used. In this case, a mediation module is typically deployed directly between a service consumer and a service provider. This means that you need a separate mediation module for each pair of service consumers and service providers. You must repeat any common processing that is required, such as logging the message, controlling Quality of Service, or applying security constraints, in each of the mediation modules.



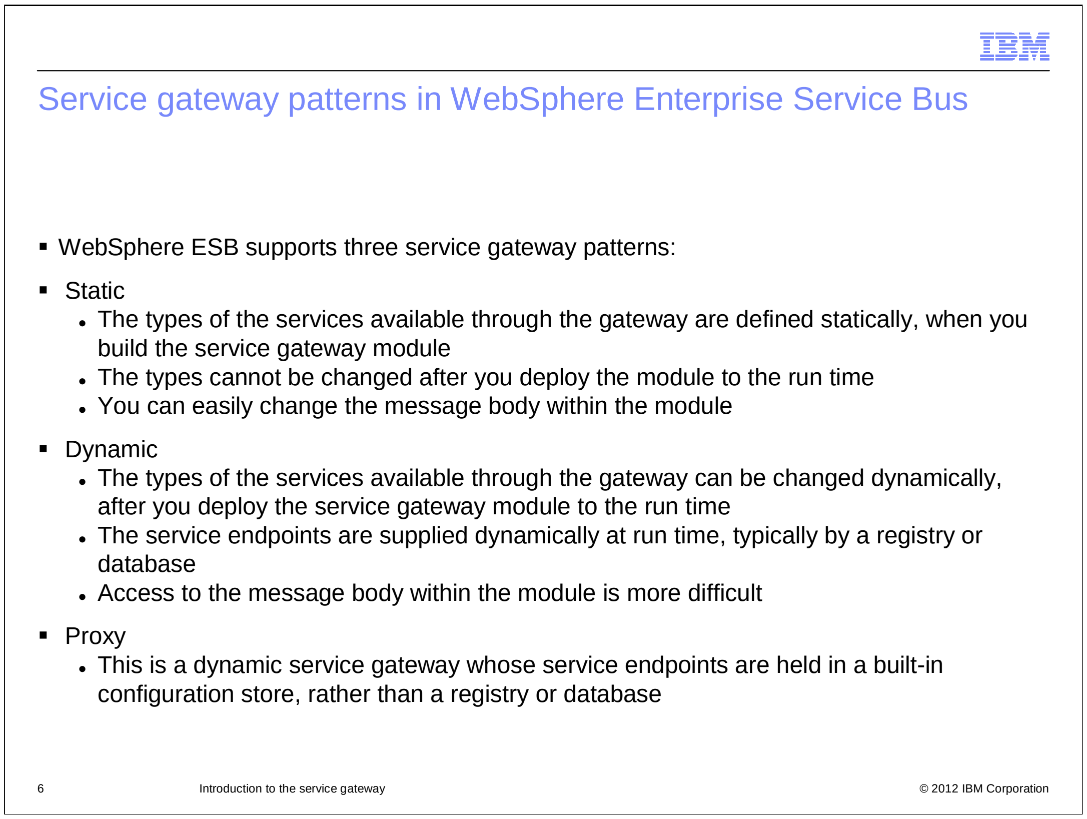
## Why use a service gateway

- In a service gateway common processing is typically implemented only once, in the service gateway mediation module
- A service gateway provides a single point of access to all the service providers, which can be tightly controlled



5 Introduction to the service gateway © 2012 IBM Corporation

This slide shows a configuration that uses a service gateway. When a service gateway is used, you need to implement any common processing only once, in the service gateway mediation module. The mediation flow in the service gateway mediation module is used by all of the service consumers. The service gateway provides a single point of access to all of the service providers, which can be tightly controlled.



## Service gateway patterns in WebSphere Enterprise Service Bus

- WebSphere ESB supports three service gateway patterns:
- Static
  - The types of the services available through the gateway are defined statically, when you build the service gateway module
  - The types cannot be changed after you deploy the module to the run time
  - You can easily change the message body within the module
- Dynamic
  - The types of the services available through the gateway can be changed dynamically, after you deploy the service gateway module to the run time
  - The service endpoints are supplied dynamically at run time, typically by a registry or database
  - Access to the message body within the module is more difficult
- Proxy
  - This is a dynamic service gateway whose service endpoints are held in a built-in configuration store, rather than a registry or database

6 Introduction to the service gateway © 2012 IBM Corporation

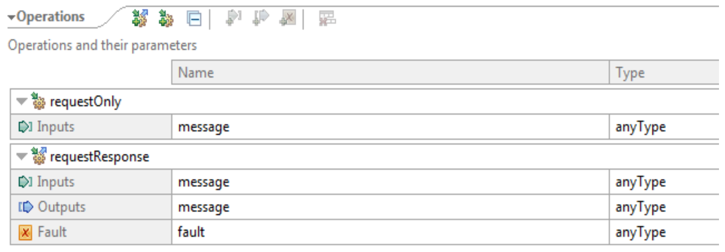
Three patterns of service gateway are supported by WebSphere ESB. The first pattern is called “static” because the types of the services available through the gateway are defined statically when you build the service gateway module. These types, which are defined by the service provider interfaces, cannot be changed after you have deployed the module to the run time. Because the types are known at development time and at run time, you can access and change the message body within the mediation module.

The second type of service gateway pattern is called “dynamic” because the types of the services available through the gateway can be changed dynamically, after you have deployed the module to the run time. The endpoint addresses of the service providers are supplied at run time, typically by a registry or database. Because the types of the messages are not normally known, it is more difficult to access the message body within the mediation module.

The third type of service gateway pattern is called a “proxy” gateway. The proxy gateway is a dynamic service gateway where the endpoint addresses of the service providers are held in a configuration store that is built-in to WebSphere ESB, rather than in an external registry or database.

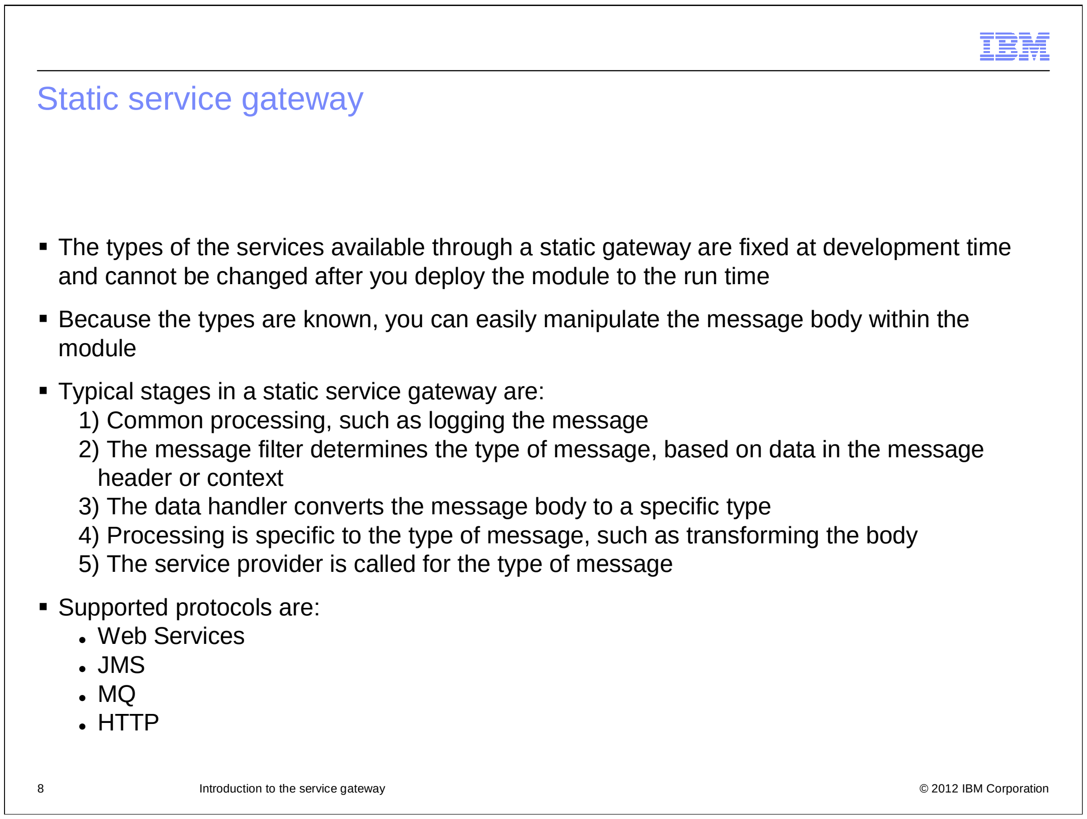
## Service gateway interface

- Messages that enter a service gateway can be of many different types
- The service gateway interface is designed to handle any type of message. It must be used when creating a service gateway
- The interface defines a *requestOnly* (one-way) operation and a *requestResponse* (two-way) operation, as shown in this example from the interface editor in IBM Integration Designer:



|  | Name | Type |
|---|---|---|
| requestOnly |  |  |
| Inputs | message | anyType |
| requestResponse |  |  |
| Inputs | message | anyType |
| Outputs | message | anyType |
| Fault | fault | anyType |

This slide describes the service gateway interface. The messages entering a service gateway can be for many different service providers and can be of many different types. WebSphere ESB provides a service gateway interface that is designed to handle any type of message. You must use this interface when creating a service gateway. The interface defines both a “request Only” operation, which is one-way, and a “requestResponse” operation, which is two-way. The example on this slide shows the service gateway interface displayed in the interface editor of the IBM Integration Designer product.



## Static service gateway

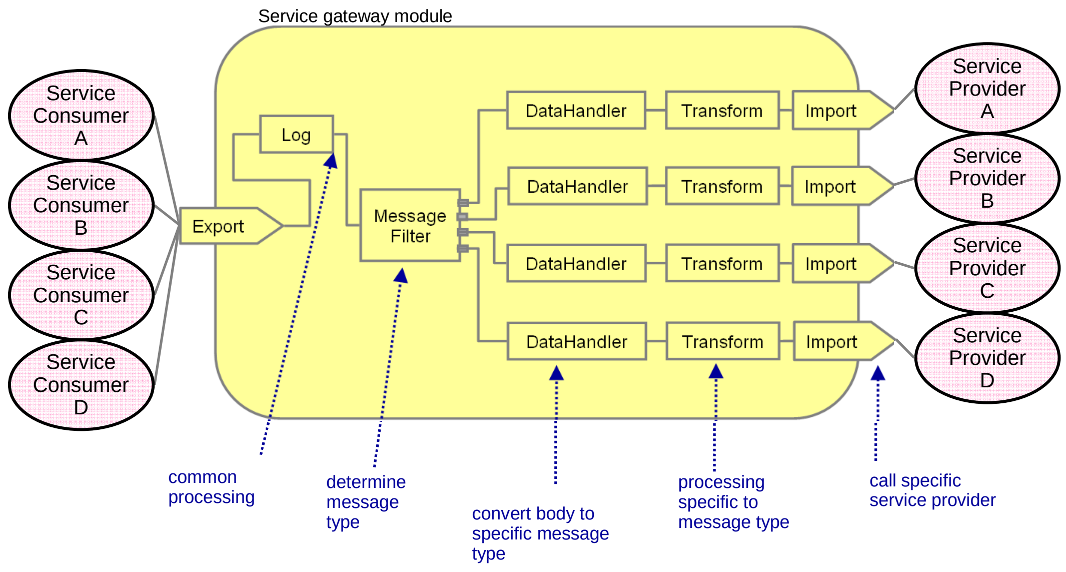
- The types of the services available through a static gateway are fixed at development time and cannot be changed after you deploy the module to the run time
- Because the types are known, you can easily manipulate the message body within the module
- Typical stages in a static service gateway are:
  1) Common processing, such as logging the message
  2) The message filter determines the type of message, based on data in the message header or context
  3) The data handler converts the message body to a specific type
  4) Processing is specific to the type of message, such as transforming the body
  5) The service provider is called for the type of message
- Supported protocols are:
  - Web Services
  - JMS
  - MQ
  - HTTP

8 Introduction to the service gateway © 2012 IBM Corporation

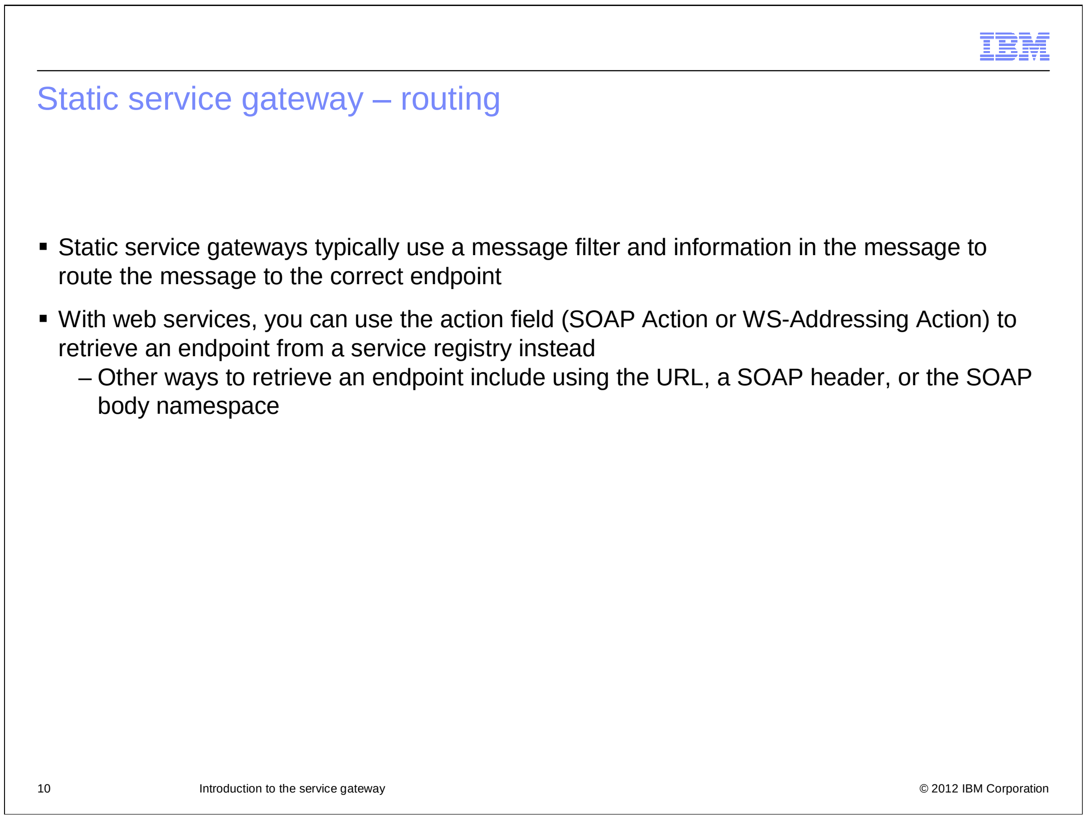
The types of the services available through a static gateway are fixed at development time and you cannot change them after the module has been deployed to the run time. Because the types of the messages are known, it is straightforward to manipulate the message body within the gateway module. In the first stage of a static service gateway module you typically perform any common processing that is required for all the messages, before the messages are split into specific types. Common processing might include logging the message, applying security constraints, or configuring quality of service. In the next stage you split the messages into their specific types, typically with a message filter mediation primitive acting on some data in the message context or header. Following this, you use a data handler mediation primitive to convert the message body from the “anyType” used in the service gateway interface to the specific body type defined for the message type. You can now perform any processing that is specific to the message type. Finally, you pass the message to the service provider through a specific Import.

The static service gateway supports the web services, JMS, MQ and HTTP protocols.

## Diagram of a typical static service gateway



Shown here is a typical static service gateway. The gateway in this diagram handles four types of messages, one for each of the service providers. There are also four service consumers. The number of service consumers does not have to be the same as the number of service providers. For example, multiple consumers could use the same service provider. The four service consumers all send messages to the service gateway export, which is the entry point into the service gateway module. The export uses the service gateway interface. The messages are all logged by the “Log” mediation primitive. The “Message Filter” mediation primitive routes each message to one of its output terminals, depending on the type of the message. Following the message filter there are four alternate paths in the mediation flow, one for each of the message types. Each of these paths has a “DataHandler” mediation primitive that converts the message body into the specific type defined for the particular message type. Following the data handler is a “Transform” mediation primitive that can manipulate the message body. After the transformation, the message passes to an import and the service provider is called.



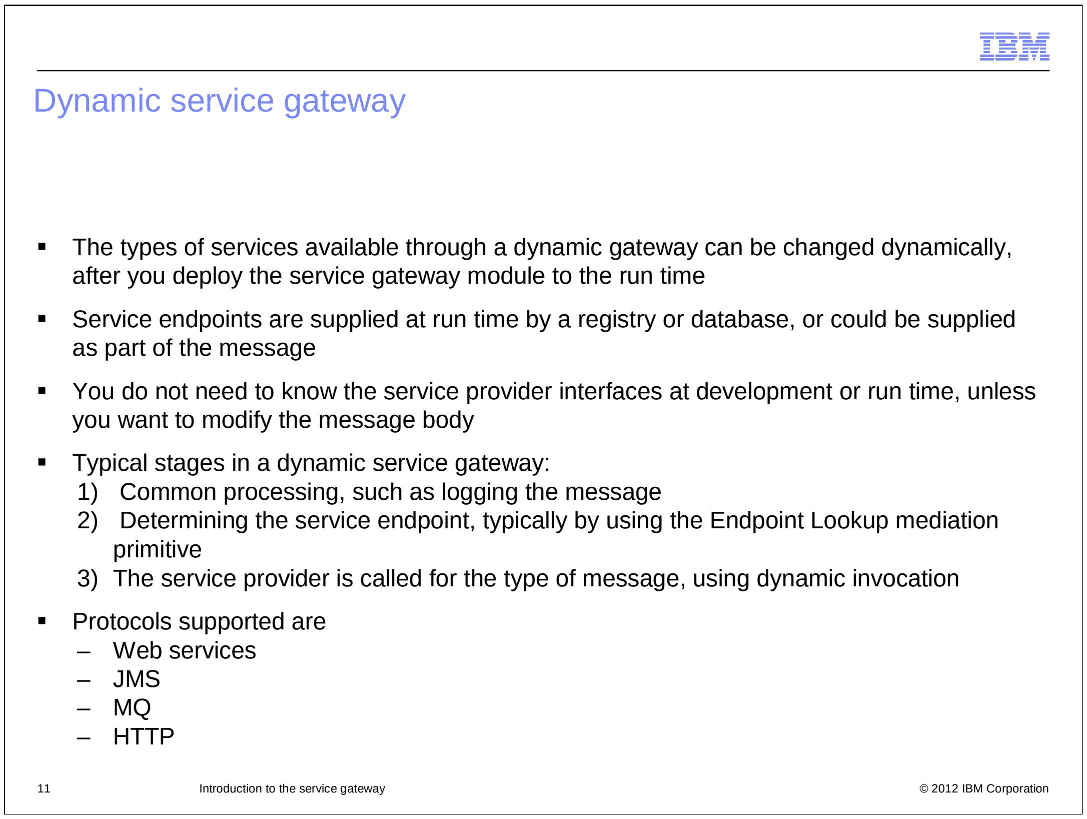
## Static service gateway – routing

- Static service gateways typically use a message filter and information in the message to route the message to the correct endpoint
- With web services, you can use the action field (SOAP Action or WS-Addressing Action) to retrieve an endpoint from a service registry instead
  - Other ways to retrieve an endpoint include using the URL, a SOAP header, or the SOAP body namespace

10 Introduction to the service gateway © 2012 IBM Corporation

Generally, in a static service gateway, routing is performed using a message filter mediation primitive operating on information in the message header or context. The message filter routes the message to one of its output terminals. Each output terminal is wired to an import for the associated service provider, as shown in the diagram on the previous slide.

If you use the web services protocol, you can use the action field, either the SOAP Action field or the Web Services Addressing Action field, for routing. You can use a gateway endpoint lookup mediation primitive to retrieve the service endpoint from a service registry, based on the value of the action field.



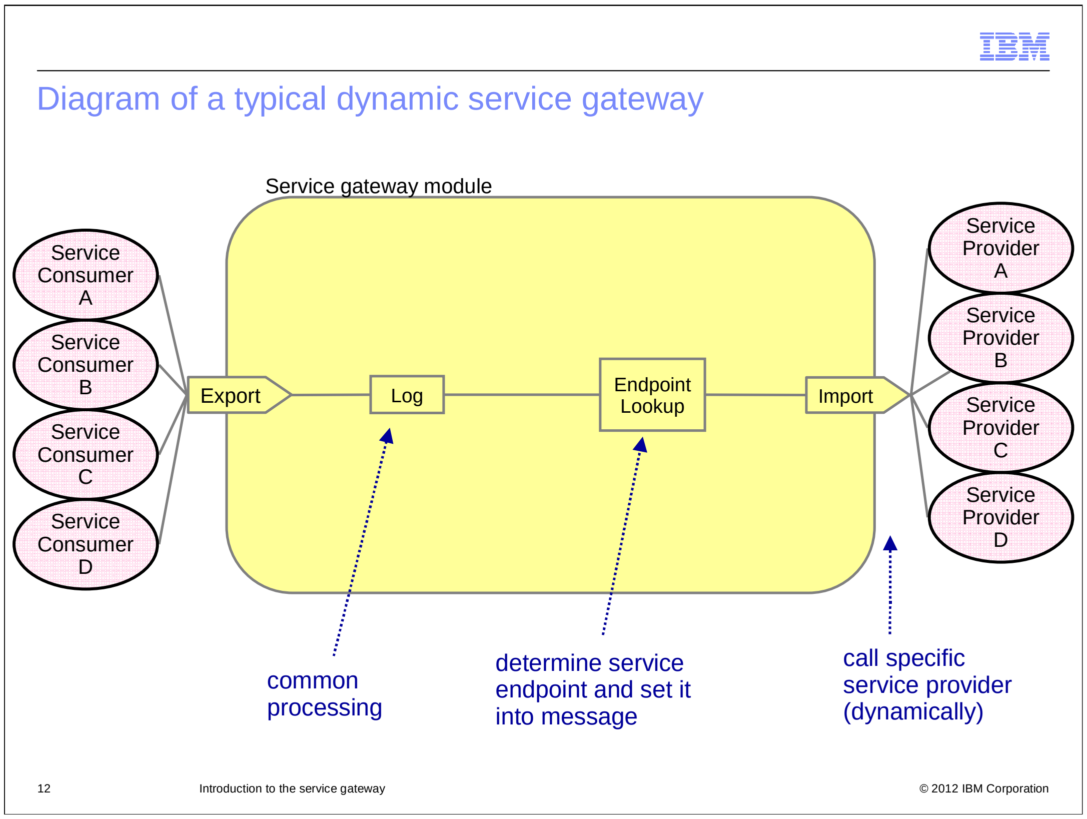
## Dynamic service gateway

- The types of services available through a dynamic gateway can be changed dynamically, after you deploy the service gateway module to the run time
- Service endpoints are supplied at run time by a registry or database, or could be supplied as part of the message
- You do not need to know the service provider interfaces at development or run time, unless you want to modify the message body
- Typical stages in a dynamic service gateway:
  1) Common processing, such as logging the message
  2) Determining the service endpoint, typically by using the Endpoint Lookup mediation primitive
  3) The service provider is called for the type of message, using dynamic invocation
- Protocols supported are
  - Web services
  - JMS
  - MQ
  - HTTP

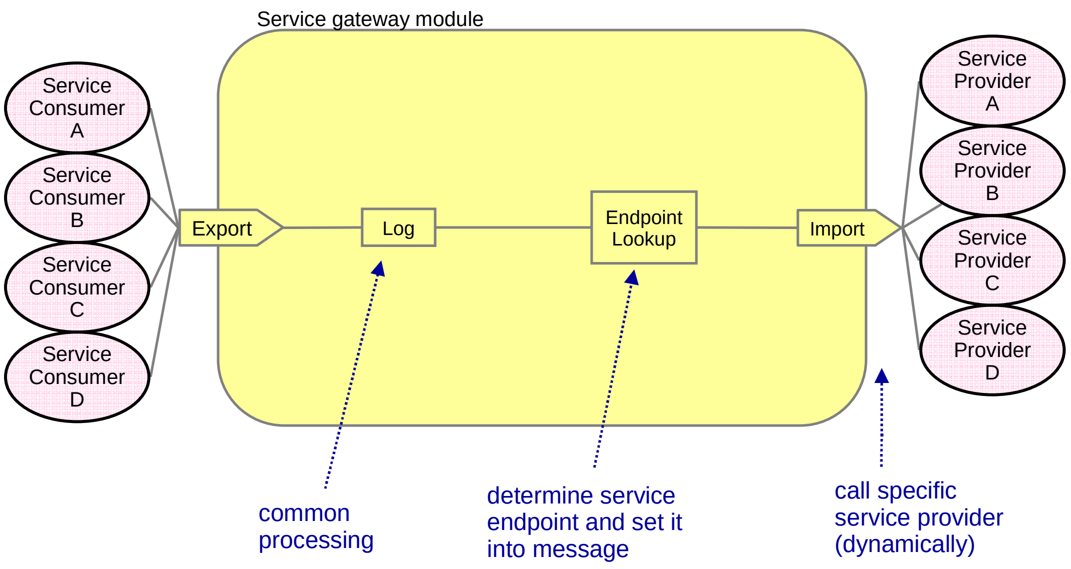
11 Introduction to the service gateway © 2012 IBM Corporation

The types of the services available through a dynamic service gateway can be changed dynamically, after you have deployed the service gateway module to the run time. The endpoint address of the service provider for a message is supplied at run time by a registry or database, or it could be carried in the message itself. You do not need to know the service provider interfaces at development or run time, unless you want to access or modify the message body, in which case you must supply the WSDL file for the message. In the first stage of a dynamic service gateway module you can perform any common processing that is required for all the messages. The next stage is to determine the service endpoint by using an endpoint lookup mediation primitive to retrieve that endpoint from a registry or database. Finally, the service endpoint is used with dynamic invocation to call the service provider.

The dynamic service gateway supports the web services, JMS, MQ, and HTTP protocols.



## Diagram of a typical dynamic service gateway

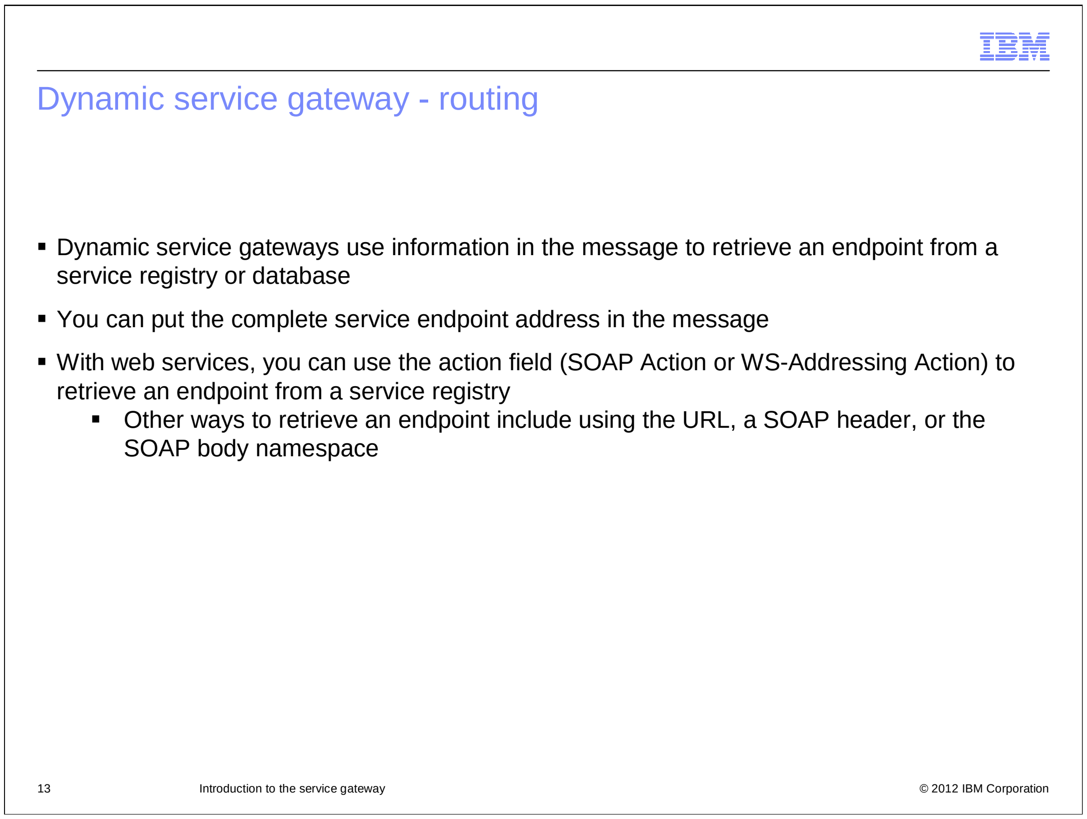


12 Introduction to the service gateway © 2012 IBM Corporation

The diagram on this slide shows a typical dynamic service gateway. Everything outside of the service gateway module is the same as for the static service gateway shown on slide 9. The gateway handles four types of messages, one for each of the service providers. There are four service consumers. The number of service consumers does not have to be the same as the number of service providers. The four service consumers all send messages to the service gateway export, which is the entry point into the service gateway module, and this uses the service gateway interface.

The first stage of the dynamic service gateway module is a “Log” mediation primitive, as it was in the static service gateway module. This is followed by the “Endpoint Lookup” mediation primitive, which determines the endpoint address of the service provider for this message by looking it up in a registry or database. The Endpoint Lookup mediation primitive writes the endpoint address into the message context. The import uses the endpoint address in the message context to dynamically invoke the service provider.

In a dynamic service gateway module all messages follow the same path. The path does not split for each message type as it does in the static gateway. You can put any common processing anywhere along this path, so the “Log” mediation primitive can work just as well if it comes after the “Endpoint Lookup” mediation primitive.



## Dynamic service gateway - routing

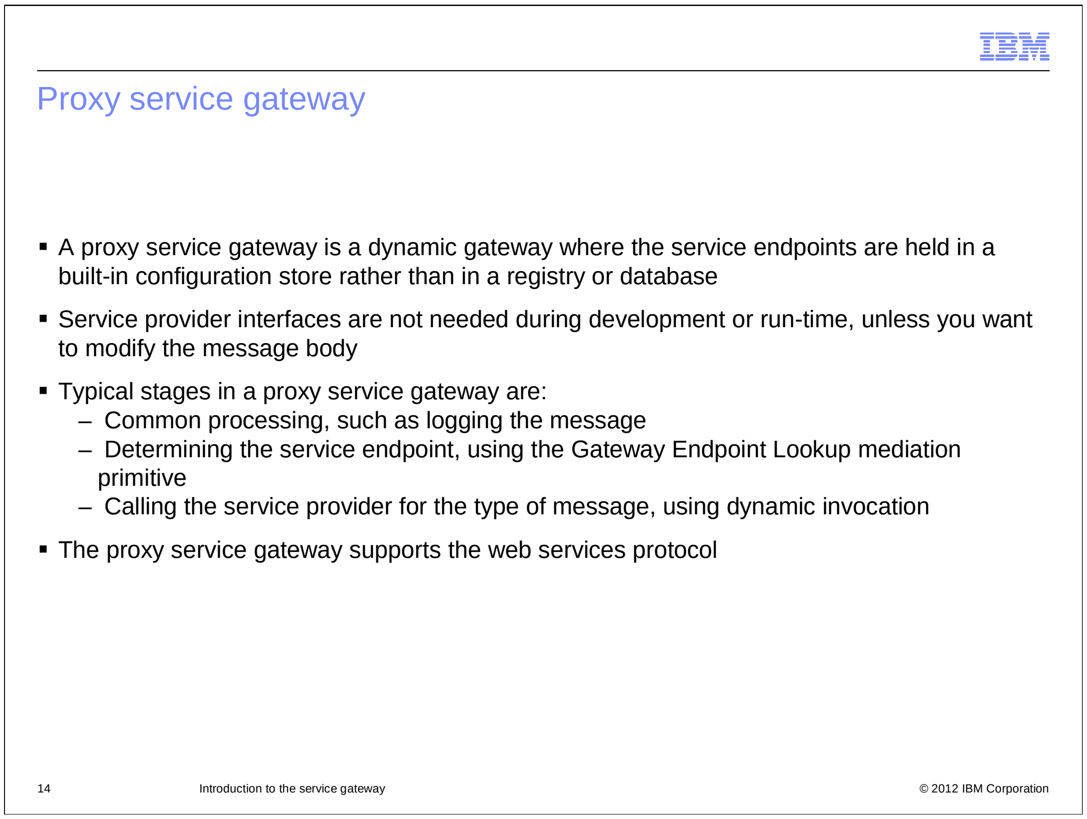
- Dynamic service gateways use information in the message to retrieve an endpoint from a service registry or database
- You can put the complete service endpoint address in the message
- With web services, you can use the action field (SOAP Action or WS-Addressing Action) to retrieve an endpoint from a service registry
  - Other ways to retrieve an endpoint include using the URL, a SOAP header, or the SOAP body namespace

13 Introduction to the service gateway © 2012 IBM Corporation

This slide describes how messages are routed in a dynamic service gateway. Routing is done using an endpoint lookup mediation primitive to retrieve the endpoint from a registry or database. The endpoint lookup mediation primitive stores the endpoint address in the context area of the message and the import retrieves the address from the context and dynamically invokes the service provider.

Instead of retrieving the address from a registry or database, you can supply the address in the message itself.

If you use the web services protocol, you can also use the action field, either the SOAP Action field or the Web Services Addressing Action field, for routing. You can use a gateway endpoint lookup mediation primitive to retrieve the service endpoint from a service registry, based on the value of the action field.

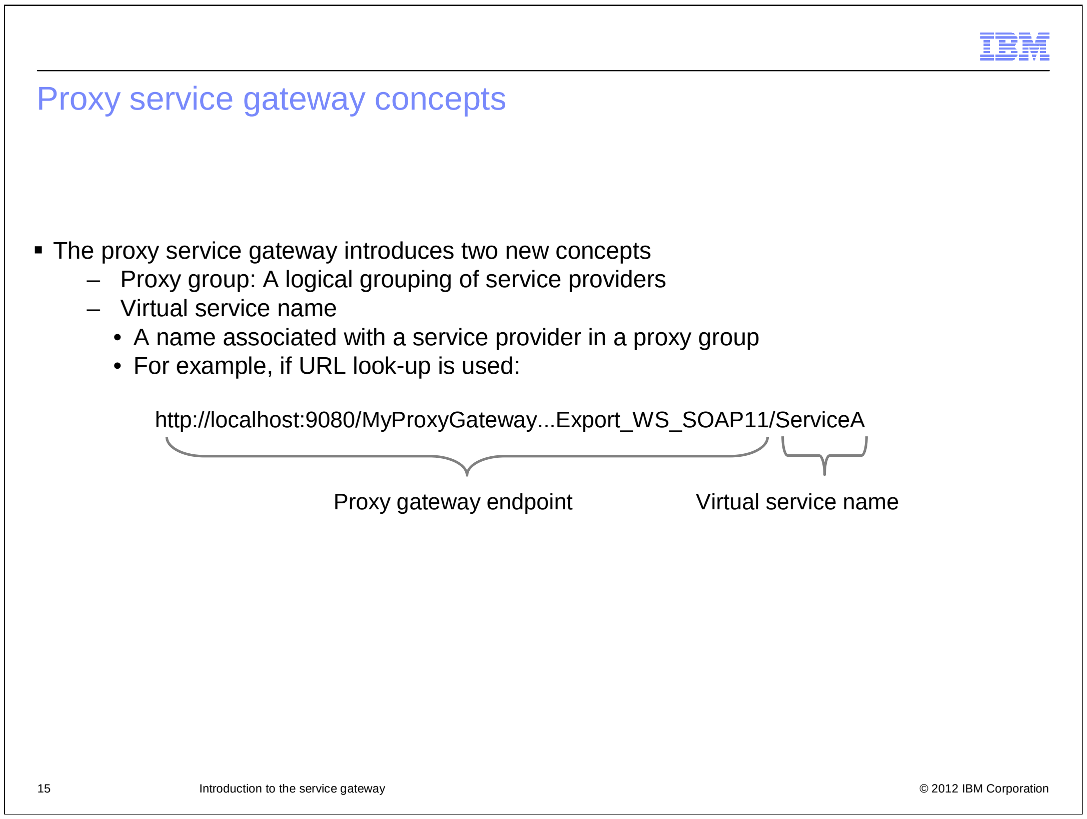


## Proxy service gateway

- A proxy service gateway is a dynamic gateway where the service endpoints are held in a built-in configuration store rather than in a registry or database
- Service provider interfaces are not needed during development or run-time, unless you want to modify the message body
- Typical stages in a proxy service gateway are:
  – Common processing, such as logging the message
  – Determining the service endpoint, using the Gateway Endpoint Lookup mediation primitive
  – Calling the service provider for the type of message, using dynamic invocation
- The proxy service gateway supports the web services protocol

14 Introduction to the service gateway © 2012 IBM Corporation

This slide provides more information about the proxy service gateway. A proxy service gateway is a dynamic gateway where the service endpoint addresses are held in a built-in configuration store, rather than an external registry or database. You do not need to know the service provider interfaces at development or run time, unless you want to access or modify the message body, in which case you must supply the WSDL file for the message. In the first stage of a proxy service gateway module you perform any common processing that is required for all the messages. The next stage is to determine the service endpoint by using a gateway endpoint lookup mediation primitive to retrieve it from the built-in configuration store. Finally, the service endpoint is used with dynamic invocation to call the service provider. The proxy service gateway supports the web services protocol.



## Proxy service gateway concepts

- The proxy service gateway introduces two new concepts
  - Proxy group: A logical grouping of service providers
  - Virtual service name
    - A name associated with a service provider in a proxy group
    - For example, if URL look-up is used:

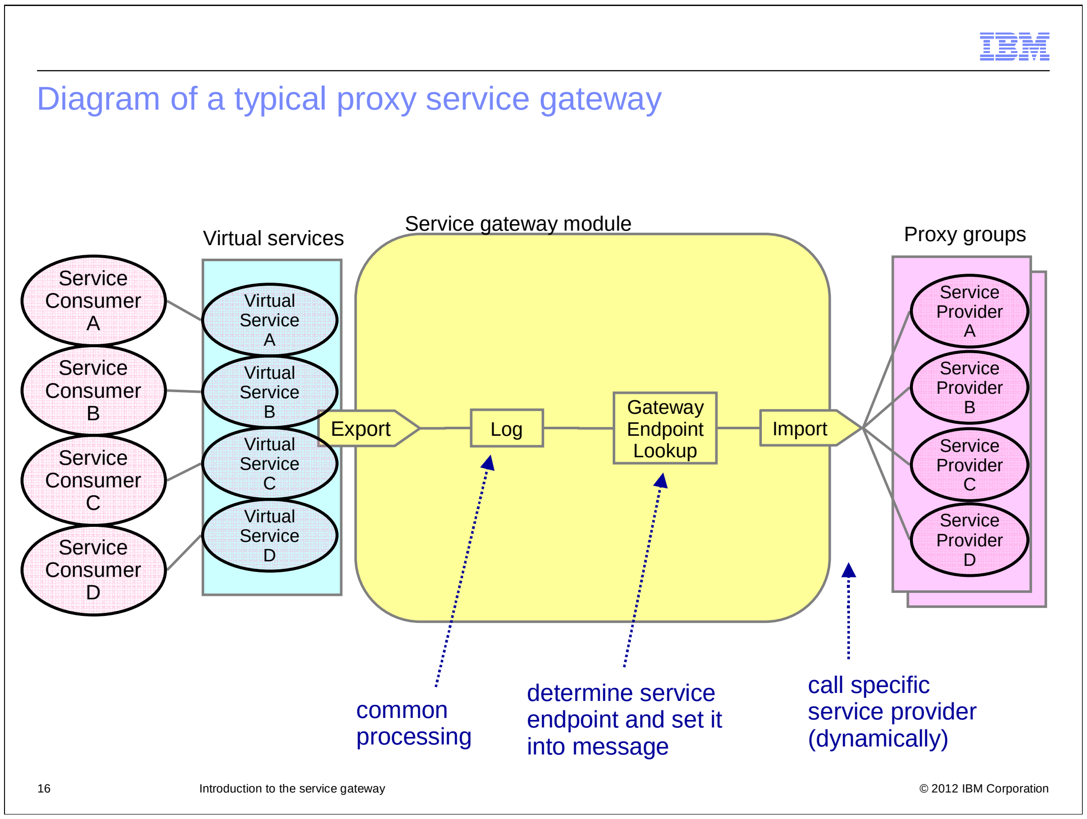
$$\underbrace{\text{http://localhost:9080/MyProxyGateway...Export\_WS\_SOAP11}}_{\text{Proxy gateway endpoint}}/\underbrace{\text{ServiceA}}_{\text{Virtual service name}}$$

15 Introduction to the service gateway © 2012 IBM Corporation

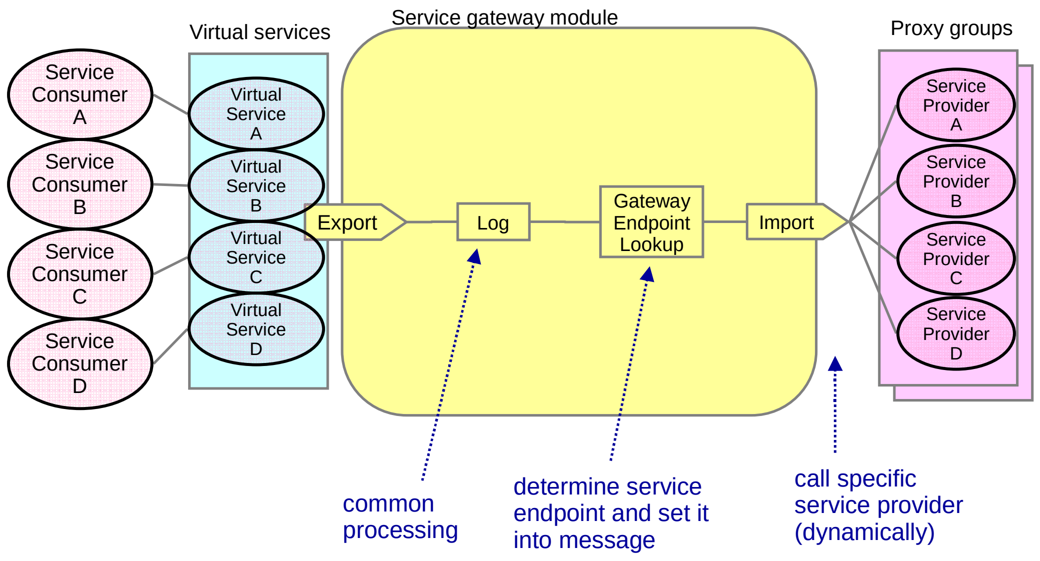
There are two new concepts introduced by the proxy service gateway.

The first concept is the “proxy group”. The service providers for a proxy gateway are divided into logical groups which are called “proxy groups”. This is designed to make the configuration of the service providers more manageable. If there is a small number of service providers, you may only need one proxy group.

The second concept is the “virtual service name”. This is a name associated with a service provider in a proxy group. It is used to look up the service provider endpoint in the built in configuration store. The virtual service name can be carried either in the URL or in the content of the message. The example on the slide shows the virtual service name in the URL of the message.



## Diagram of a typical proxy service gateway



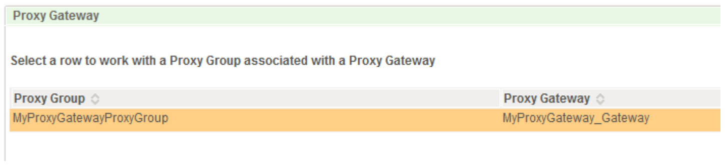
This is a diagram of a typical proxy service gateway. There are four service consumers and four virtual services; each virtual service represents one of the service providers in the proxy group. Each consumer sends messages to a virtual service. The number of consumers does not have to be the same as the number of virtual services; for example multiple consumers can use the same virtual service. The messages enter the proxy gateway through the export and are logged by the “Log” mediation primitive. The “Gateway Endpoint Lookup” mediation primitive uses the virtual service name from the message to query the built-in configuration store and retrieve the service endpoint address for the message, it writes this into the context area of the message. The import uses the service endpoint address to dynamically invoke the service provider.

All messages follow the same path in the proxy service gateway module, so the “Log” mediation primitive can work just as well if it came after the “Endpoint Lookup” mediation primitive.

## Proxy service gateway - configuring virtual services

- Virtual services are defined and configured using the Proxy Gateway widget in Business Space
- The Proxy Gateway widget discovers the proxy groups defined for each proxy gateway on the server

  In this example, there is one proxy group in one proxy gateway



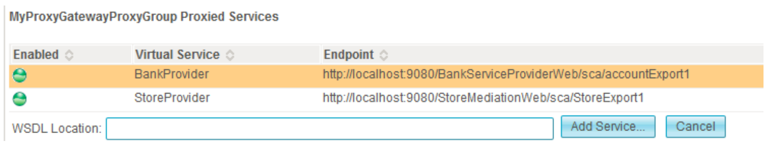
Proxy Gateway

Select a row to work with a Proxy Group associated with a Proxy Gateway

| Proxy Group | Proxy Gateway |
|---|---|
| MyProxyGatewayProxyGroup | MyProxyGateway_Gateway |

- You can use the Proxy Gateway widget to define a virtual service for a proxy group and associate it with an endpoint. You can give the virtual service any name

  In this example, there are two virtual services defined for the proxy group

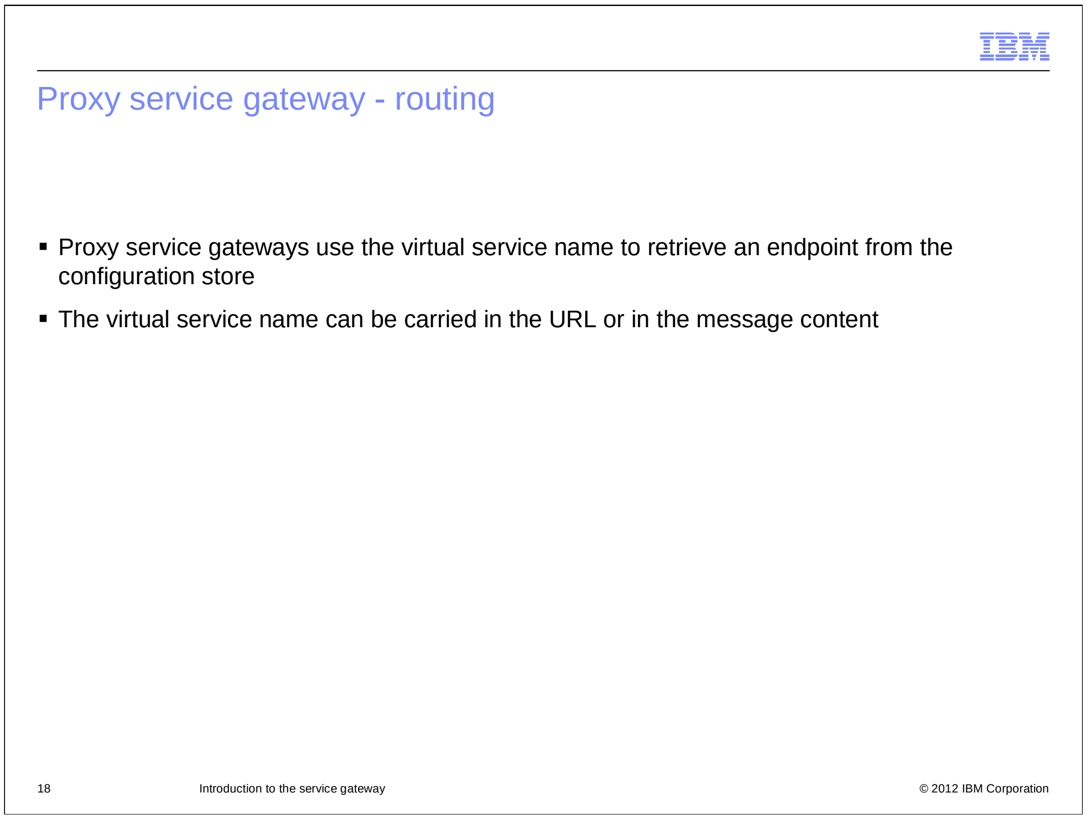


MyProxyGatewayProxyGroup Proxied Services

| Enabled | Virtual Service | Endpoint |
|---|---|---|
| ● | BankProvider | http://localhost:9080/BankServiceProviderWeb/sca/accountExport1 |
| ● | StoreProvider | http://localhost:9080/StoreMediationWeb/sca/StoreExport1 |

WSDL Location:

This slide gives you an introduction to defining and configuring virtual services using the Proxy Gateway widget in the Business Space associated with your run time server. You can get more details on this from links four and five on slide 20. The Proxy Gateway widget automatically discovers all the proxy groups defined for each proxy gateway on the server. The example in the screen capture of the Proxy Gateway widget shows one proxy group called “MyProxyGatewayProxyGroup” defined for the proxy gateway “MyProxyGateway_Gateway”. You can expand the proxy group in the Proxy Gateway widget and add a new virtual service for a service provider, or change or delete an existing virtual service. The Proxy Gateway widget assigns a default name to a new virtual service but you can change the name. The second screen capture shows the “MyProxyGatewayProxyGroup” expanded in the widget, and shows two virtual services, called “BankProvider” and “StoreProvider”.

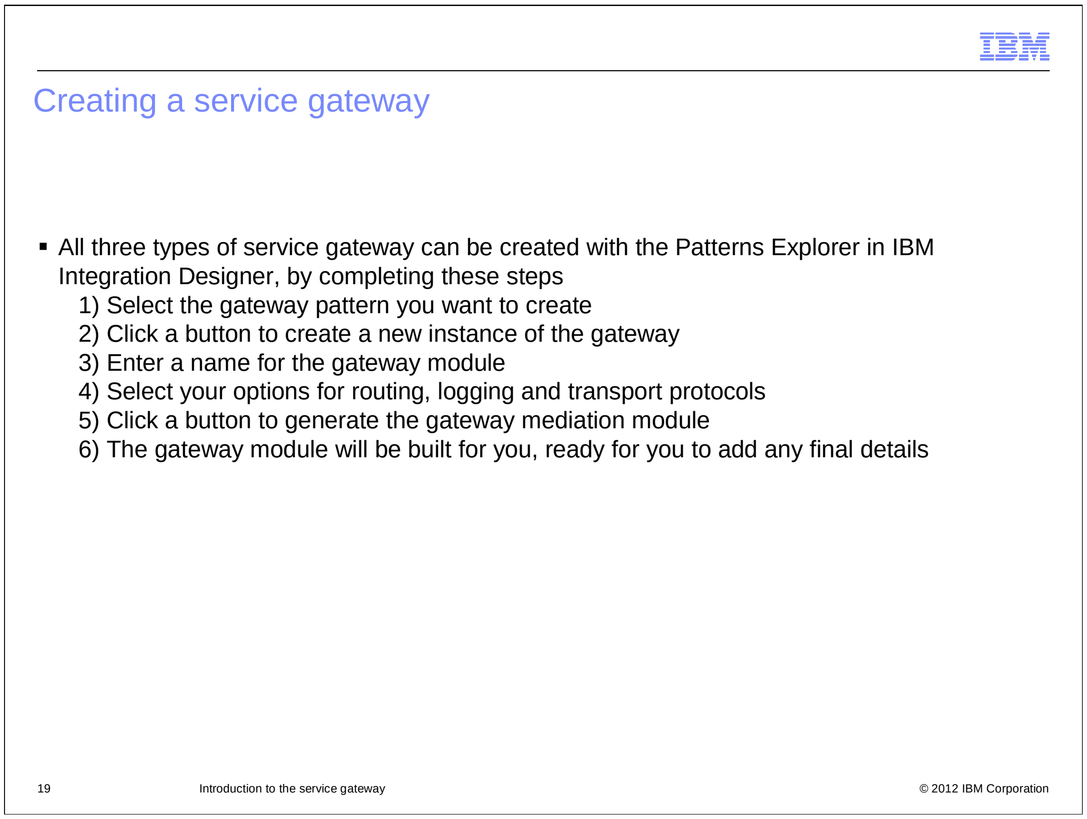


## Proxy service gateway - routing

- Proxy service gateways use the virtual service name to retrieve an endpoint from the configuration store
- The virtual service name can be carried in the URL or in the message content

18 Introduction to the service gateway © 2012 IBM Corporation

This slide describes how messages are routed in a proxy service gateway. A proxy service gateway contains a gateway endpoint look up mediation primitive; this uses the virtual service name to retrieve the service endpoint from the built-in configuration store. The virtual service name can be carried in either the URL or the message content. To carry it in the URL, you add the virtual service name to the end of the proxy gateway endpoint address, as shown on slide 15. If you carry the virtual service name in the message, you must configure the gateway endpoint lookup mediation primitive with an XPath expression that identifies where it is held in the message.

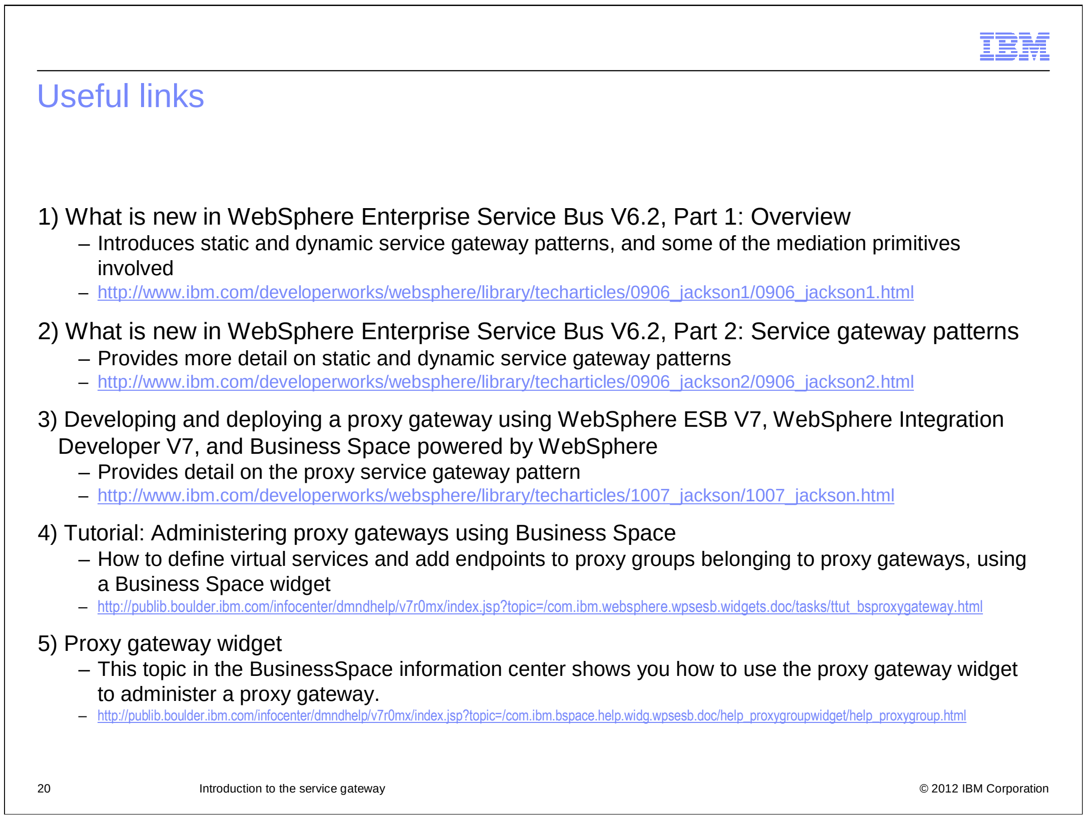


## Creating a service gateway

- All three types of service gateway can be created with the Patterns Explorer in IBM Integration Designer, by completing these steps
  1) Select the gateway pattern you want to create
  2) Click a button to create a new instance of the gateway
  3) Enter a name for the gateway module
  4) Select your options for routing, logging and transport protocols
  5) Click a button to generate the gateway mediation module
  6) The gateway module will be built for you, ready for you to add any final details

19 Introduction to the service gateway © 2012 IBM Corporation

You can create a service gateway using the Patterns Explorer in the IBM Integration Designer product. The first step is to select the gateway pattern - static, dynamic, or proxy - that you want to create. Then you click a button to create a new instance of the gateway pattern. You are prompted to enter a name for the gateway module, after which you can select the options you want to use for routing, logging, and transport protocol. Finally, you click a button to generate the gateway mediation module. A basic gateway module is built for you, which you can then configure and extend as needed. You can get a more detailed description of creating a proxy service gateway from link three on slide 20.



## Useful links

1) What is new in WebSphere Enterprise Service Bus V6.2, Part 1: Overview
   - Introduces static and dynamic service gateway patterns, and some of the mediation primitives involved
   - http://www.ibm.com/developerworks/websphere/library/techarticles/0906_jackson1/0906_jackson1.html

2) What is new in WebSphere Enterprise Service Bus V6.2, Part 2: Service gateway patterns
   - Provides more detail on static and dynamic service gateway patterns
   - http://www.ibm.com/developerworks/websphere/library/techarticles/0906_jackson2/0906_jackson2.html

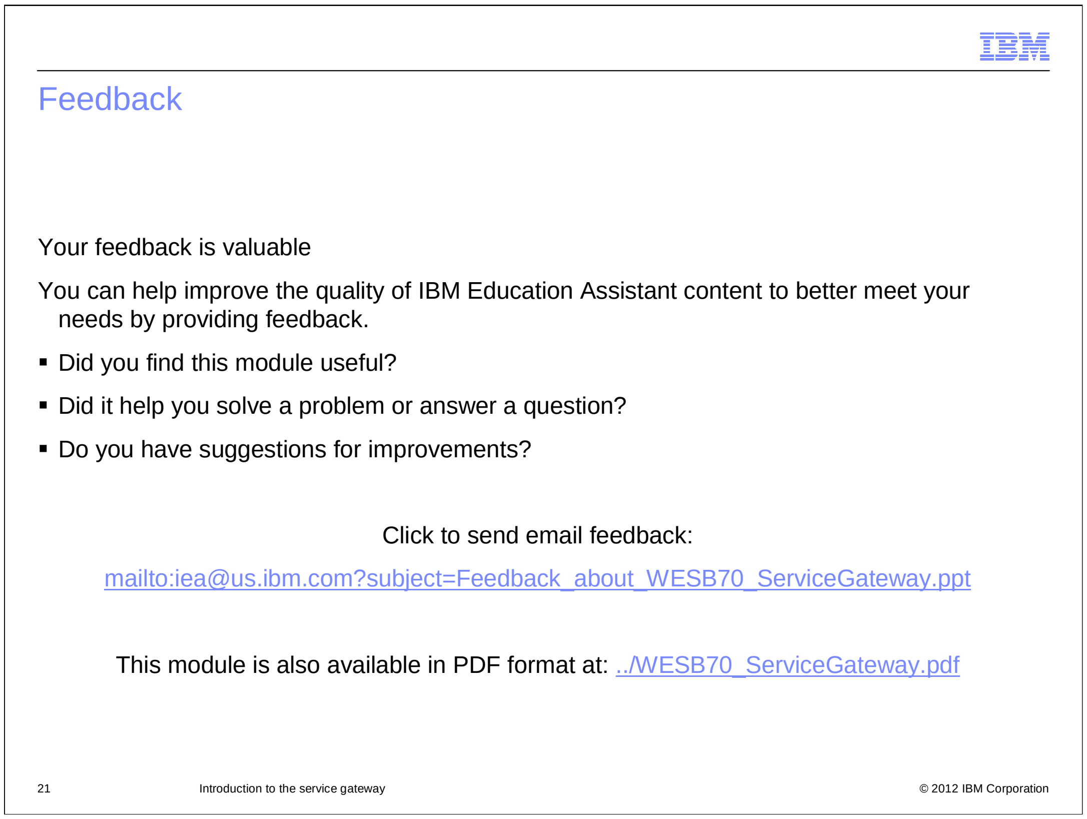
3) Developing and deploying a proxy gateway using WebSphere ESB V7, WebSphere Integration Developer V7, and Business Space powered by WebSphere
   - Provides detail on the proxy service gateway pattern
   - http://www.ibm.com/developerworks/websphere/library/techarticles/1007_jackson/1007_jackson.html

4) Tutorial: Administering proxy gateways using Business Space
   - How to define virtual services and add endpoints to proxy groups belonging to proxy gateways, using a Business Space widget
   - http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.websphere.wpsesb.widgets.doc/tasks/ttut_bsproxygateway.html

5) Proxy gateway widget
   - This topic in the BusinessSpace information center shows you how to use the proxy gateway widget to administer a proxy gateway.
   - http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.bspace.help.widg.wpsesb.doc/help_proxygroupwidget/help_proxygroup.html

20 Introduction to the service gateway © 2012 IBM Corporation

The links on this slide bring you to more information to help you understand, build, and configure service gateways. The first three links are to developerWorks articles. The first developerWorks article introduces the static and dynamic gateway patterns and some of the mediation primitives that they use. The second article provides you with much more detail on static and dynamic gateway patterns. The third article provides a lot of information on the proxy gateway pattern, including an introduction to the concepts, how to build a proxy gateway using the Patterns Explorer, and how to configure it using the Proxy Gateway widget in Business Space. The last two links are to the information center. The first of these, link number 4, is to a tutorial on administering proxy gateways. The second of these, link number 5, is to the Business Space information center and provides detailed information on how to use the Proxy Gateway widget.



## 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 email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WESB70_ServiceGateway.ppt

This module is also available in PDF format at: ../WESB70_ServiceGateway.pdf

21 Introduction to the service gateway © 2012 IBM Corporation

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

## Trademarks, disclaimer, and copyright information