IBM Software Group

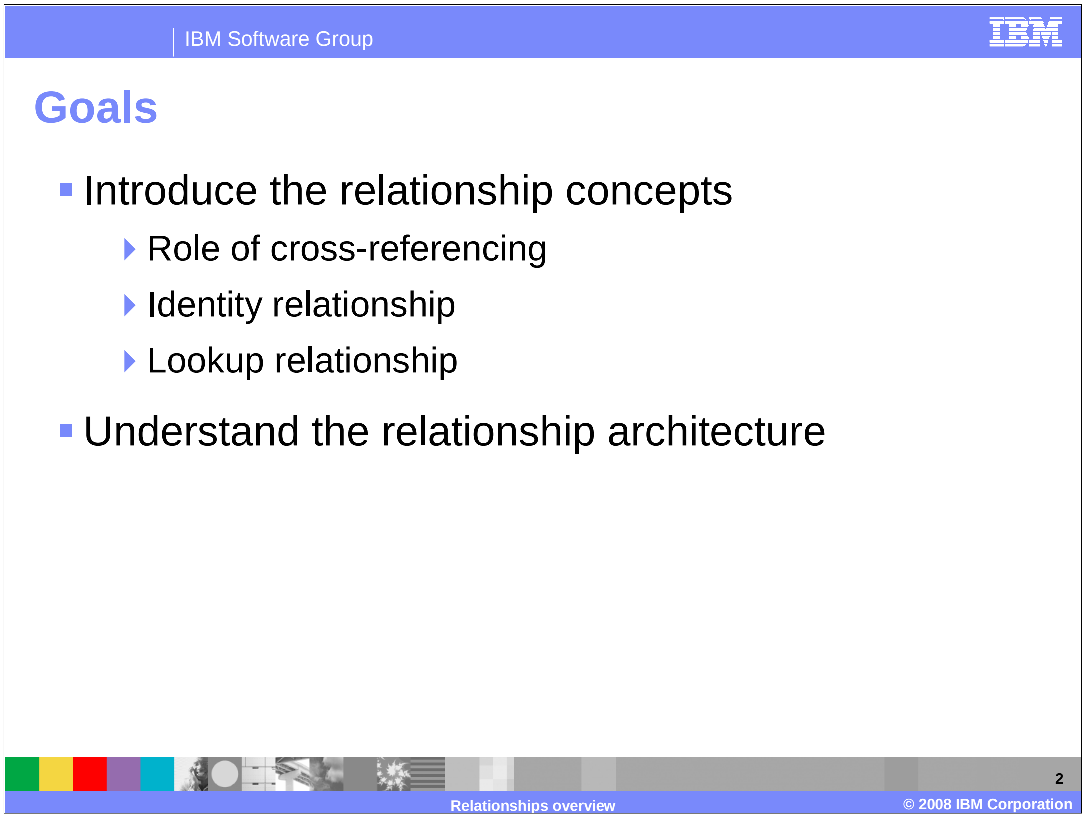
WebSphere® Process Server V6.1 WebSphere Integration Developer V6.1

Relationships overview

business on demand.

Updated April 18, 2008

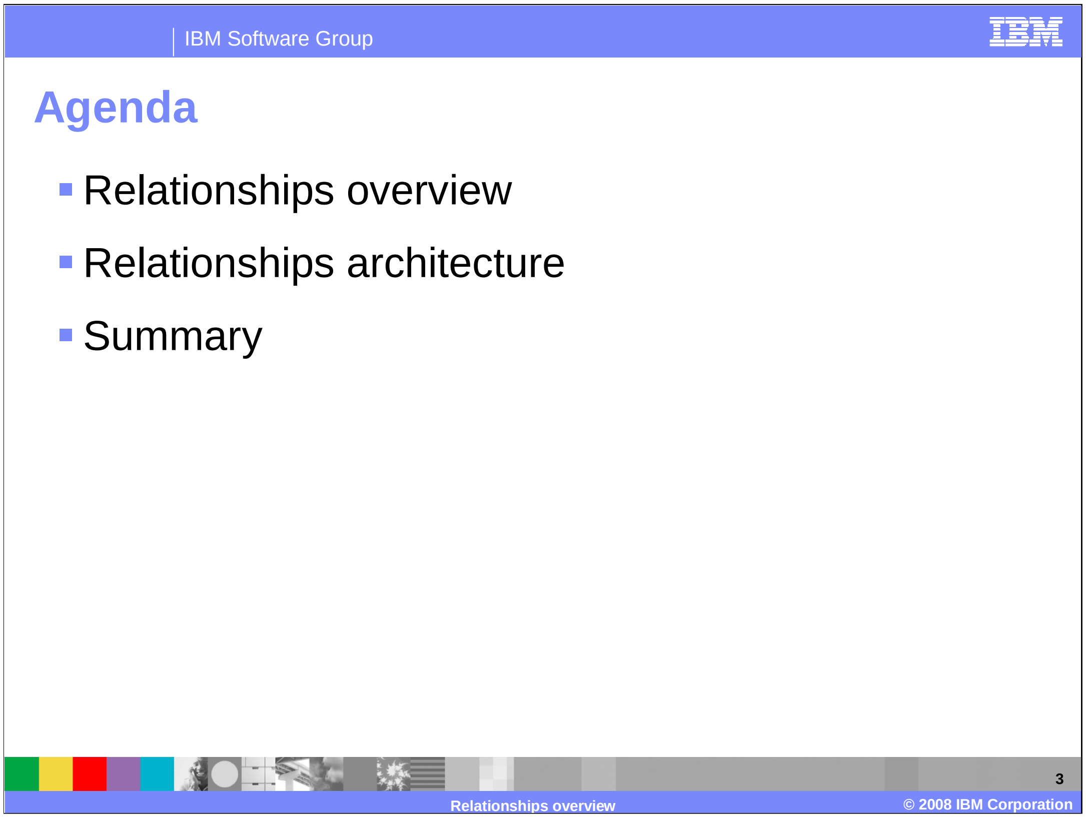
This presentation will provide an overview of relationships and the relationship service of WebSphere Process Server and WebSphere Integration Developer V6.1.



Goals

- Introduce the relationship concepts
  - Role of cross-referencing
  - Identity relationship
  - Lookup relationship
- Understand the relationship architecture

The goals for this presentation are to introduce the relationship concepts, including the role of cross-referencing, identity relationships, and lookup relationships, and provide an understanding of the relationship service architecture.



IBM Software Group

Agenda

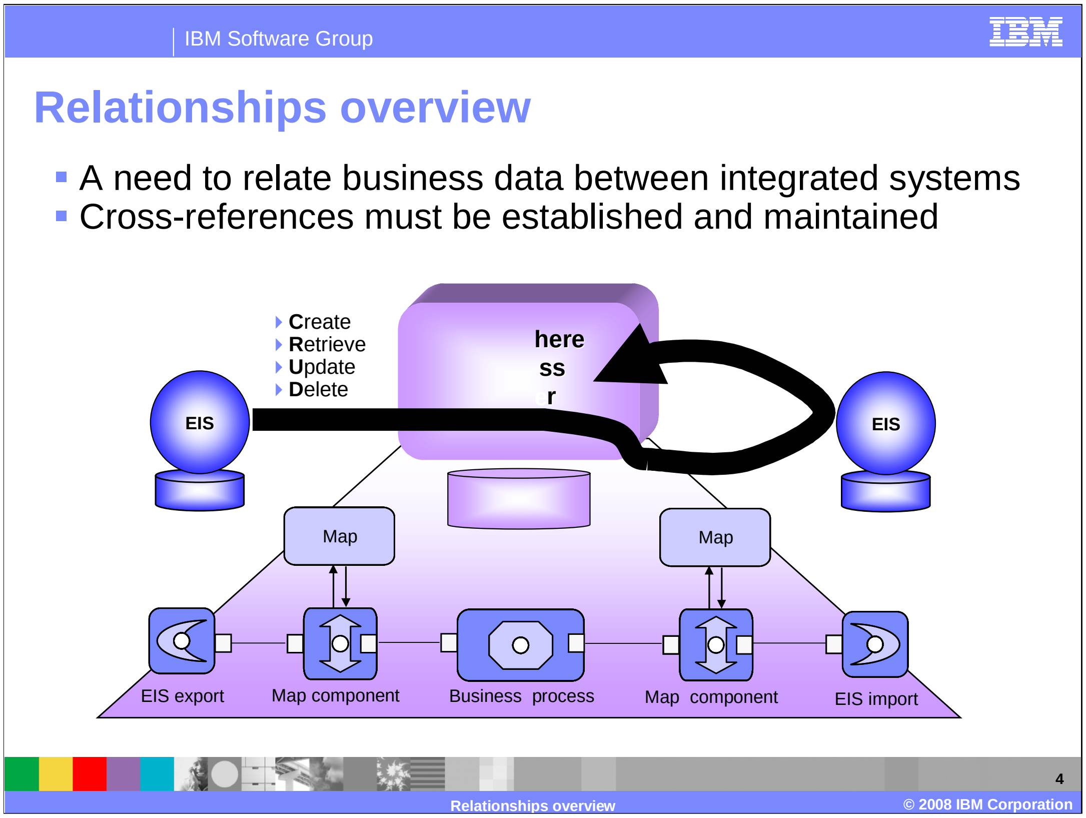
- Relationships overview
- Relationships architecture
- Summary

3

Relationships overview

© 2008 IBM Corporation

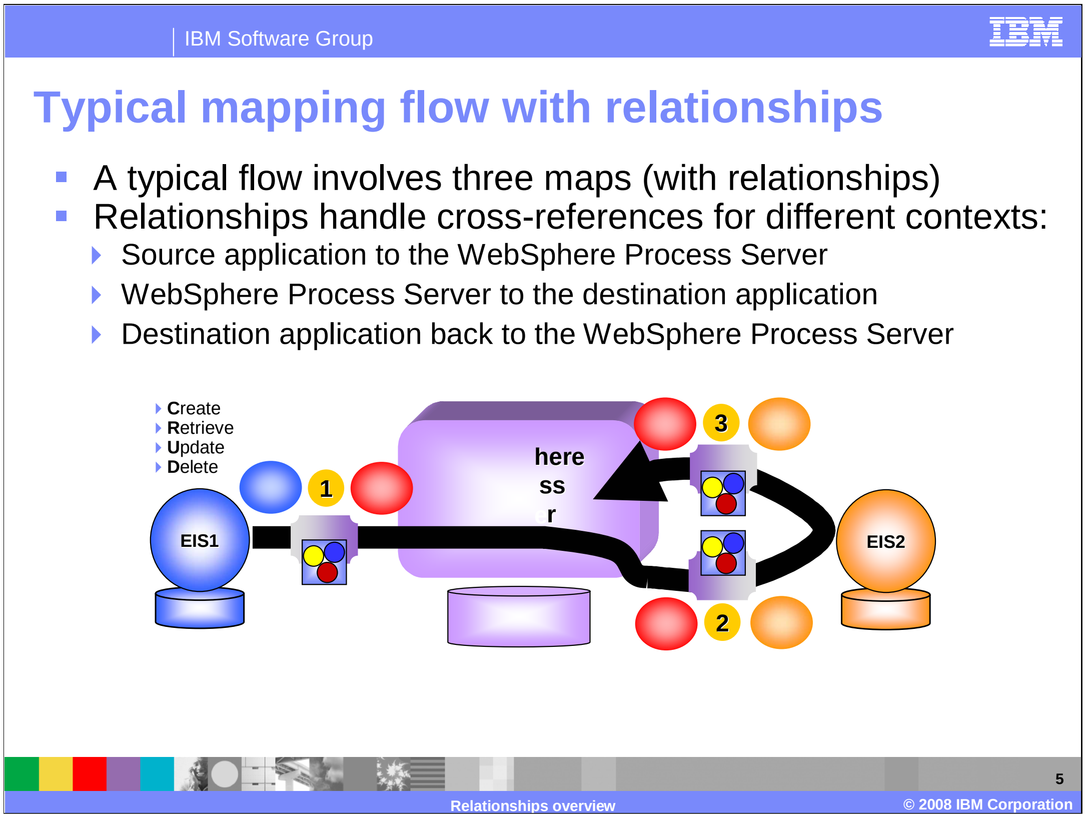
This section will provide an overview of relationships.



Relationships overview

- A need to relate business data between integrated systems
- Cross-references must be established and maintained

There is a fundamental need in integration scenarios to provide a mechanism that enables creation and maintenance of cross-reference information relating together various business data entities integrated across the enterprise systems. In this graphic, the integration environment consists of two enterprise information Systems (EIS) with WebSphere Process Server in between as the integration broker. The requirement is to integrate business data across the system with the need to "relate" the business data for future reference. For example, you may create a new customer record in your front desk system that gets integrated into your backend accounting system, and, in the future, updates occur to this customer record. You need to integrate these changes and reflect them in the backend accounting customer records that actually correspond to the customer that was initially integrated across the system. WebSphere Process Server relationships, along with the relationship service, provide for this kind of functionality.

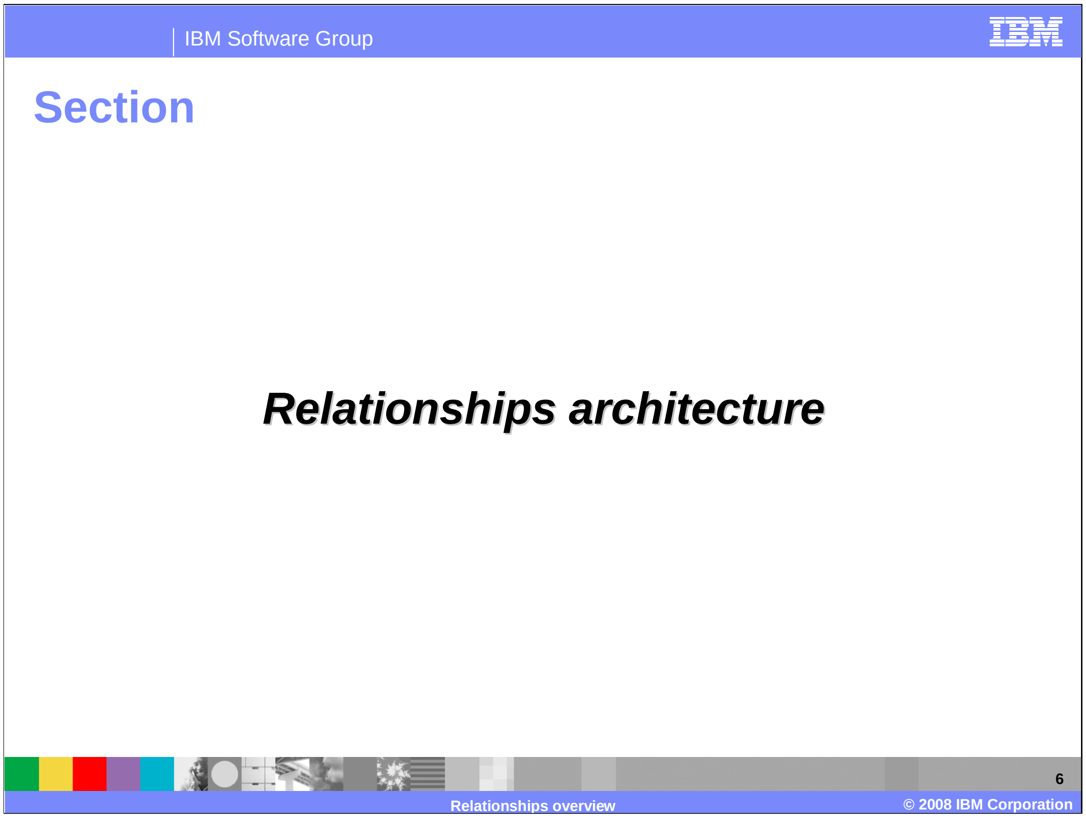


Typical mapping flow with relationships

- A typical flow involves three maps (with relationships)
- Relationships handle cross-references for different contexts:
  - Source application to the WebSphere Process Server
  - WebSphere Process Server to the destination application
  - Destination application back to the WebSphere Process Server

- Create
- Retrieve
- Update
- Delete

Here you see a typical mapping scenario with relationships added. A typical flow involves at least three maps. In this example, the application specific business entity from the source application is transformed into a generic business entity and a relationship is invoked from within a map. A relationship is one of the mapping transformation rules. The generic business entity is transformed into an application specific business entity for the destination application and another relationship is potentially invoked, depending on the verb. The application specific business entity is transformed back into a generic business entity to complete the flow, and another relationship is potentially invoked, again depending on the verb. A fourth map would be needed if the result of the business integration processing was expected back in the source application.



IBM Software Group

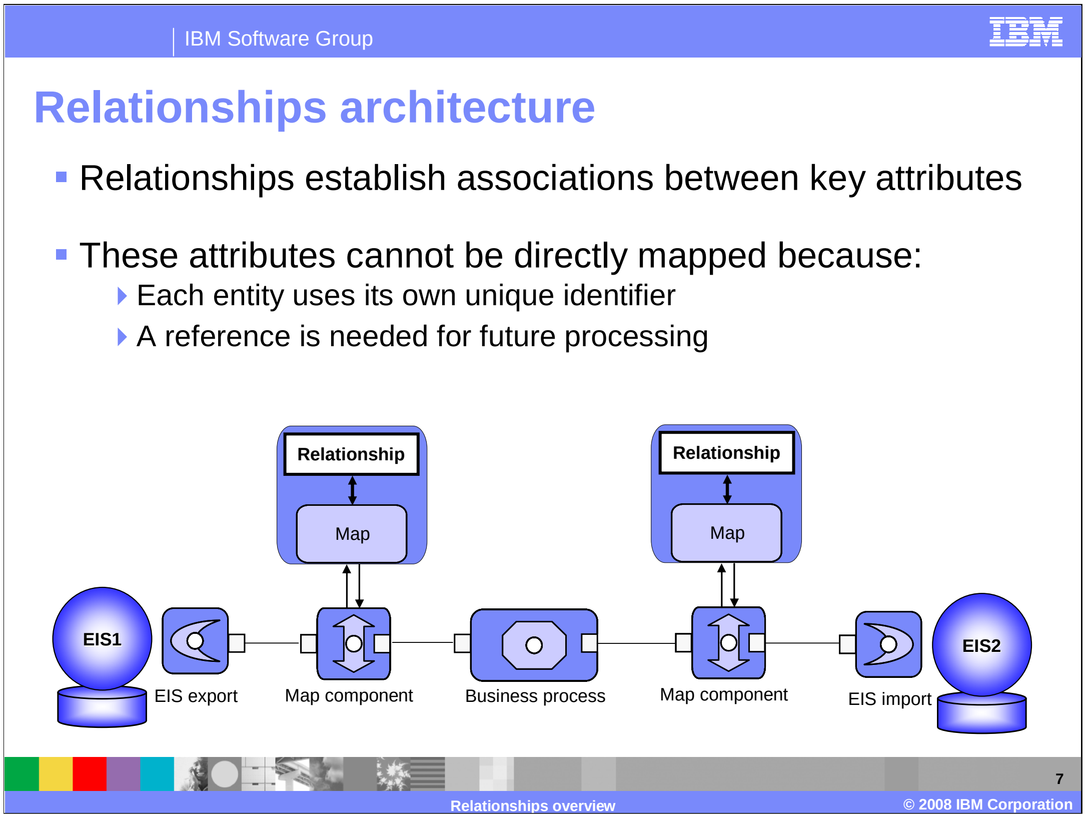
Section

Relationships architecture

6

Relationships overview

This section will discuss the relationships architecture.



IBM Software Group

Relationships architecture

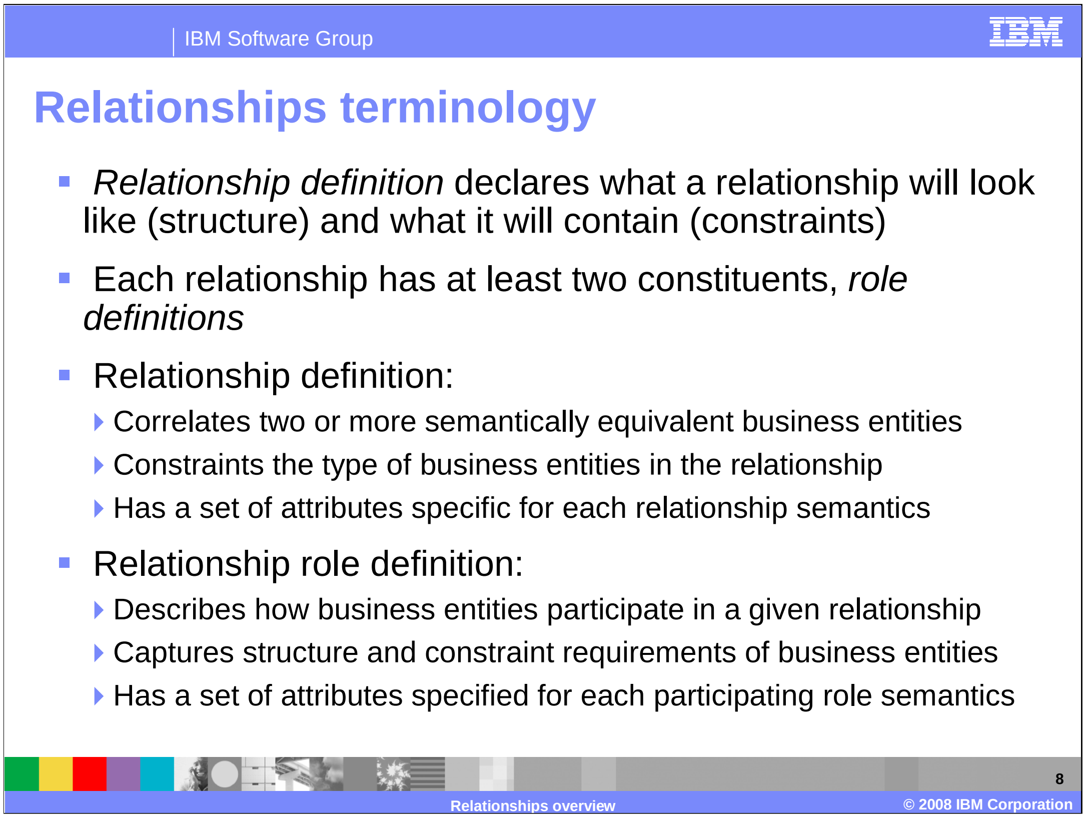
- Relationships establish associations between key attributes
- These attributes cannot be directly mapped because:
  - Each entity uses its own unique identifier
  - A reference is needed for future processing

Relationships overview

7

© 2008 IBM Corporation

Relationships are used to establish associations between attributes of multiple related business entities. Relationship transformation is one of the transformation rules available within maps. In the chart you see that the relationship is invoked from within the map itself. The relationship establishes an association between key attributes on the source business entity and the target business entity within the map. These key attributes cannot be directly mapped because each entity uses its own unique identifier. Each EIS application maintains its own mechanism for generating unique values for those attributes. For example, the source EIS maintains a different primary key for the business entity and the target EIS also maintains a different primary key for the semantically equivalent business entity. The requirement exists to establish a cross reference of these equivalent business entities for any subsequent future processing and being able to look up the target entity that has already been integrated across the solution.



IBM Software Group

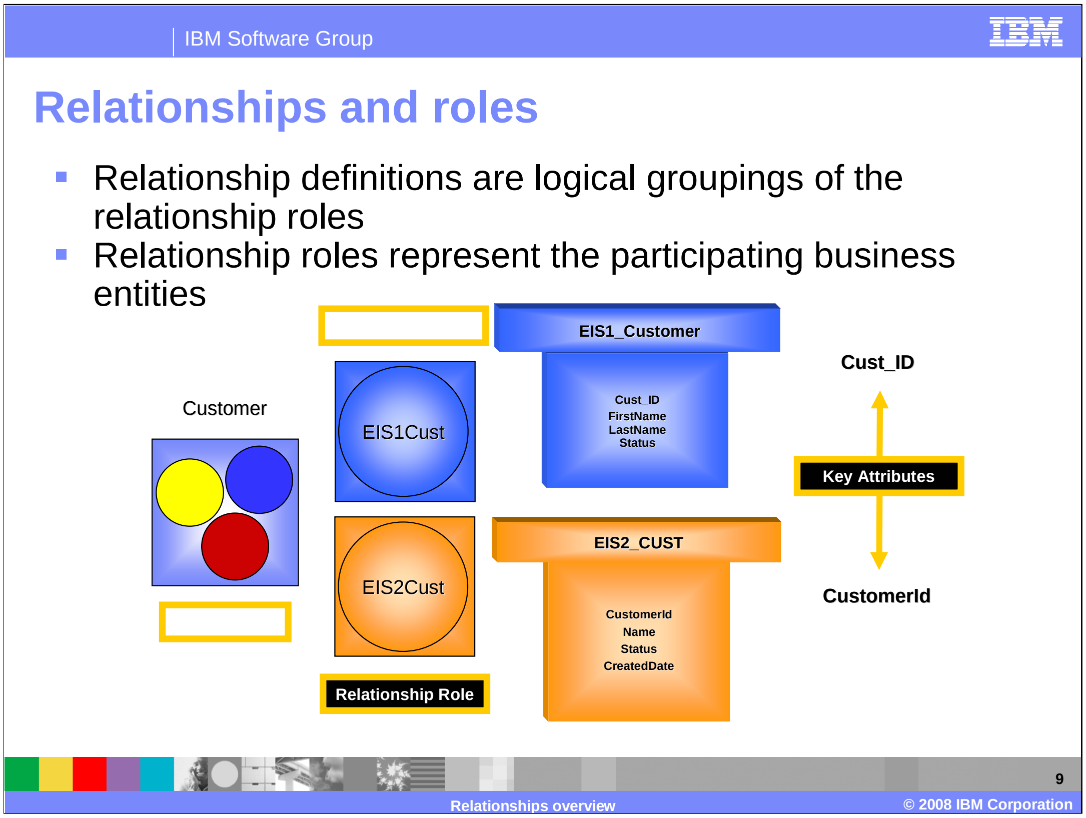
Relationships terminology

- *Relationship definition* declares what a relationship will look like (structure) and what it will contain (constraints)
- Each relationship has at least two constituents, *role definitions*
- Relationship definition:
  - Correlates two or more semantically equivalent business entities
  - Constraints the type of business entities in the relationship
  - Has a set of attributes specific for each relationship semantics
- Relationship role definition:
  - Describes how business entities participate in a given relationship
  - Captures structure and constraint requirements of business entities
  - Has a set of attributes specified for each participating role semantics

8

Relationships overview

The relationship definition is a template that declares what a relationship will look like, including its structure and what it will contain and which key attributes will constitute the cross reference. Each relationship has at least two constituents also known as role definitions. These role definitions correlate two or more semantically equivalent business entities. The role definition represents a particular business entity on each side. A role definition for an application specific customer represents that application specific business entity within the relationship definition. It constrains the type of business entities because you associate the role with a particular business object. A relationship role definition describes how business entities participate in a particular relationship. It captures the structure and the key attribute, or attributes, if that entity is using a compound key.   The relationship role definition has a set of attributes for those roles that you can configure.



IBM Software Group

Relationships and roles

- Relationship definitions are logical groupings of the relationship roles
- Relationship roles represent the participating business entities

Customer

EIS1Cust

EIS2Cust

Relationship Role

EIS1_Customer

Cust_ID
FirstName
LastName
Status

EIS2_CUST

CustomerId
Name
Status
CreatedDate

Cust_ID

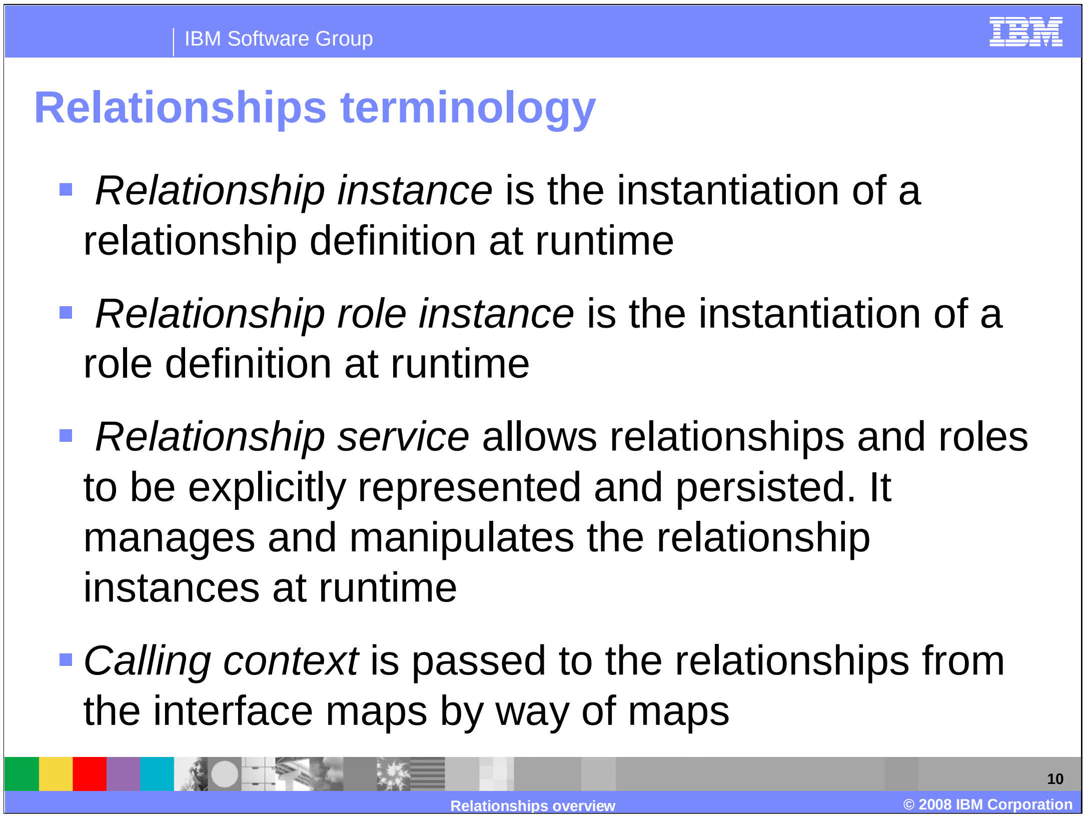
Key Attributes

CustomerId

9

Relationships overview

Relationship definitions are logical groupings of two or more relationship roles. Relationship roles represent the participating business entities. On the left of this graphic, you see a customer relationship which contains two relationship roles, EIS1Cust and EIS2Cust, which are roles representing customer entities from the two EIS systems being cross referenced. These two customer entities each have their primary keys stored in different attributes, Cust_ID and CustomerId, as seen at the right of the graphic.



IBM Software Group

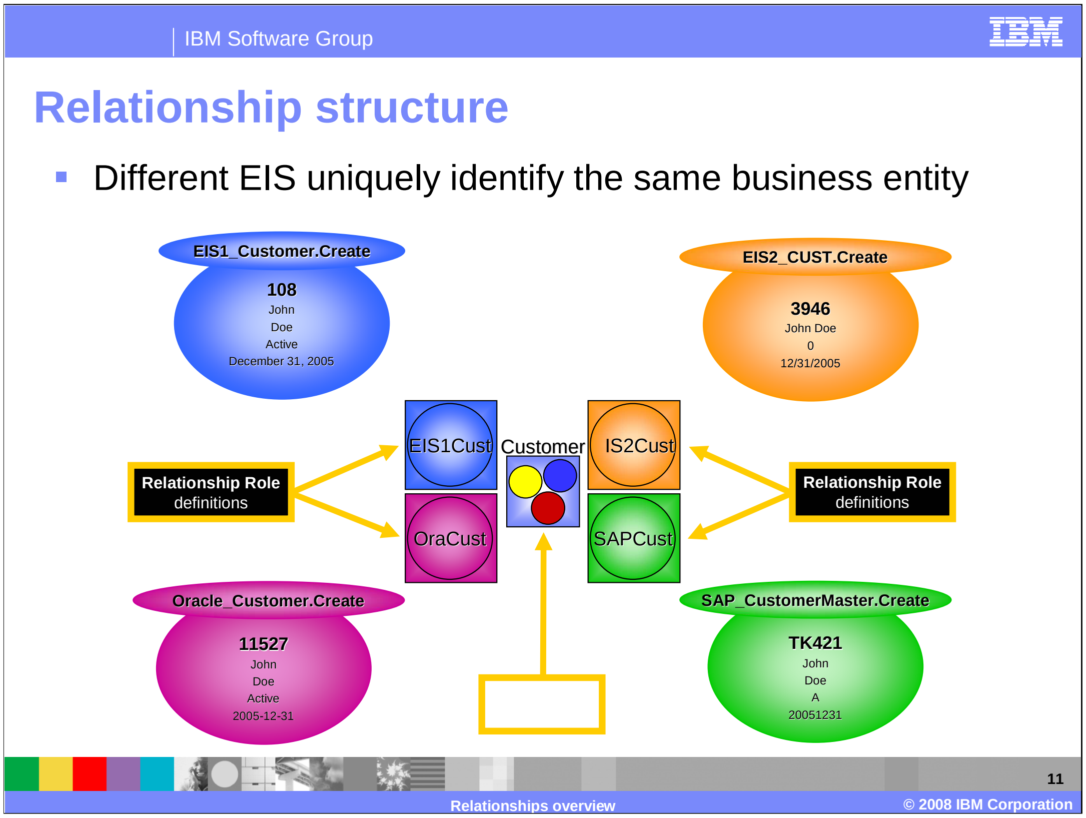
Relationships terminology

- *Relationship instance* is the instantiation of a relationship definition at runtime
- *Relationship role instance* is the instantiation of a role definition at runtime
- *Relationship service* allows relationships and roles to be explicitly represented and persisted. It manages and manipulates the relationship instances at runtime
- *Calling context* is passed to the relationships from the interface maps by way of maps

10

Relationships overview

The relationship instance is the instantiation of a relationship definition at runtime. Similarly, the relationship role instance is the instantiation of a relationship role definition at runtime. The relationship service allows relationships and roles to be explicitly represented and persisted. It manages and manipulates the relationship instances at runtime using information received from the calling context. The calling context is received by the relationship service from the map. It is used by the relationship service to determine in which part of the request flow is the relationship being invoked. There are four calling contexts, event delivery, service call request, service call response, and service call failure. The calling context will be discussed in more detail in a few slides.



Relationship structure

- Different EIS uniquely identify the same business entity

EIS1_Customer.Create

108
John
Doe
Active
December 31, 2005

EIS2_CUST.Create

3946
John Doe
0
12/31/2005

Relationship Role definitions

EIS1Cust

Customer

IS2Cust

OraCust

SAPCust

Relationship Role definitions

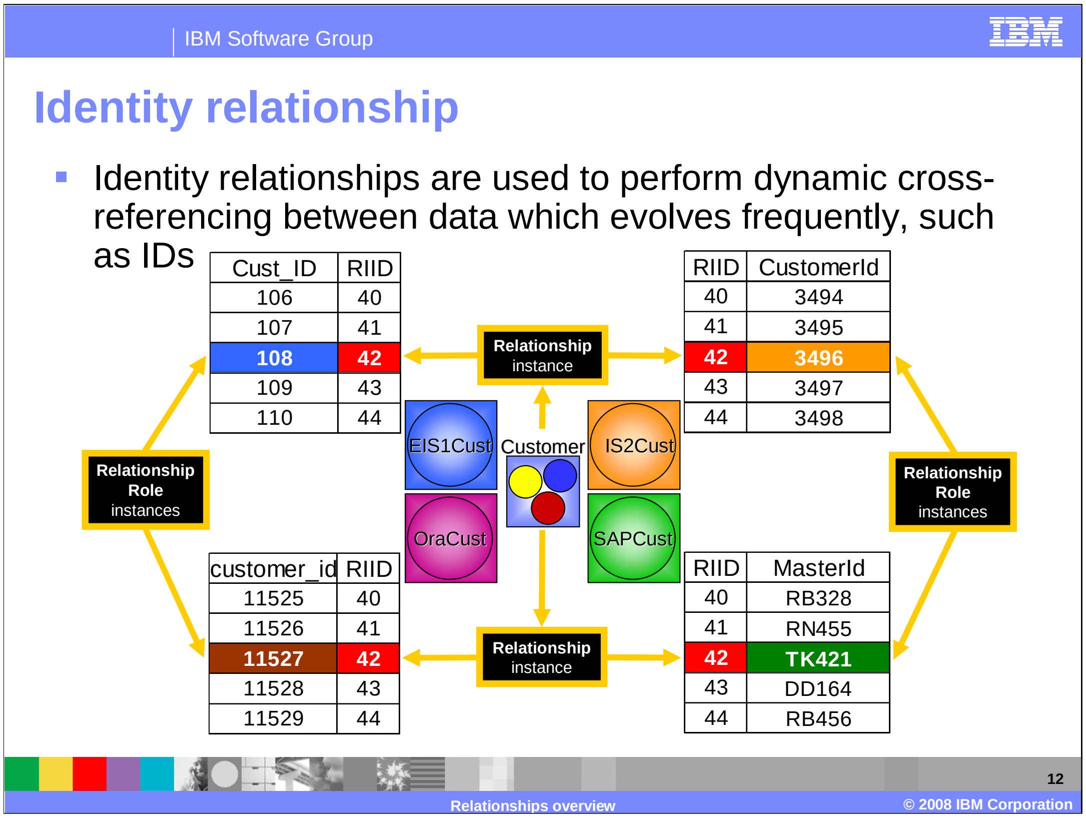
Oracle_Customer.Create

11527
John
Doe
Active
2005-12-31

SAP_CustomerMaster.Create

TK421
John
Doe
A
20051231

Here is an example of what the relationship instance and relationship roles may look like. You have the customer relationship definition in the middle of the graphic which contains four different relationship roles for different participating business entities from different participating systems.   On the outside of the graphic, you see the actual runtime customer data entities that vary for the different systems.



IBM Software Group

IBM

Identity relationship

- Identity relationships are used to perform dynamic cross-referencing between data which evolves frequently, such as IDs

Cust_ID	RIID
106	40
107	41
108	42
109	43
110	44

RIID	CustomerId
40	3494
41	3495
42	3496
43	3497
44	3498

customer_id	RIID
11525	40
11526	41
11527	42
11528	43
11529	44

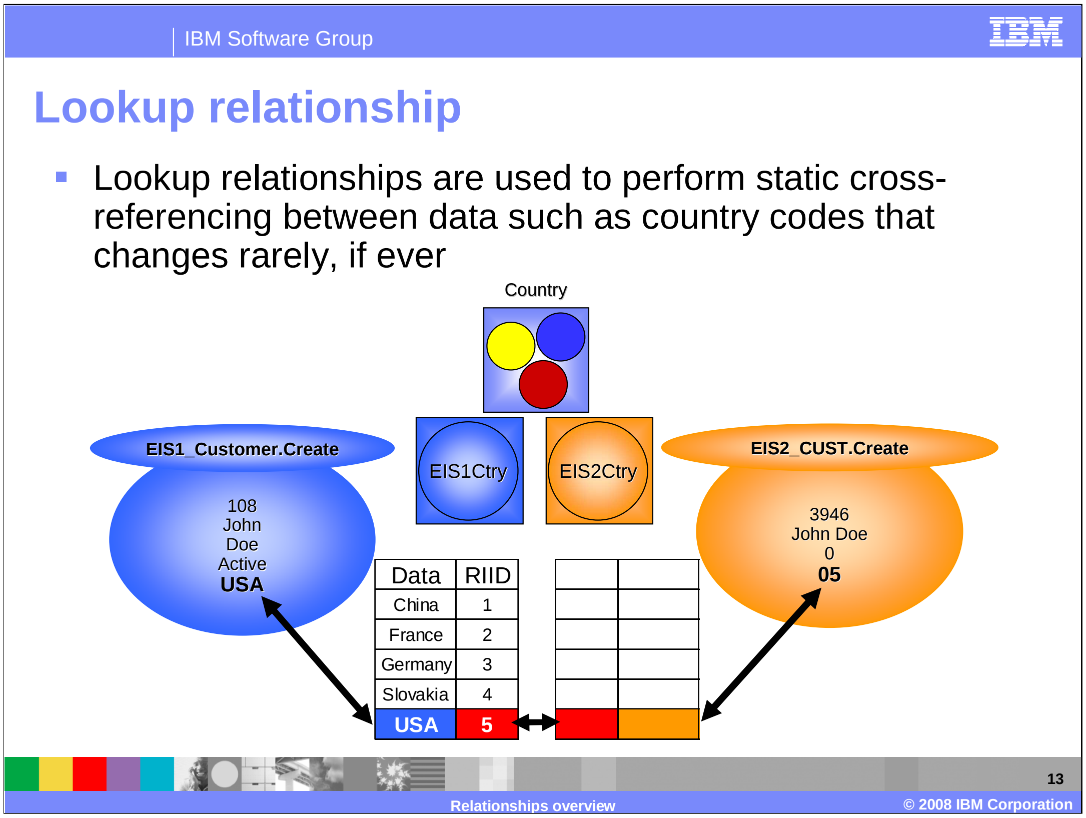
RIID	MasterId
40	RB328
41	RN455
42	TK421
43	DD164
44	RB456

12

Relationships overview

© 2008 IBM Corporation

An identity relationship is used to perform dynamic cross referencing between data entities which evolve frequently during the course of business. This slide depicts a scenario you could see in an identity relationship instance at runtime. In this example, there are four business applications running on systems external to the WebSphere Process Server. A customer entity coming in from one system where it is represented with a primary key of 108, is being mapped to a generic customer representation which has been assigned a relationship instance ID of 42. This generic customer entity is then being mapped to a specific format used by a different system and the entity is assigned the primary key value as determined by the application on that particular system.

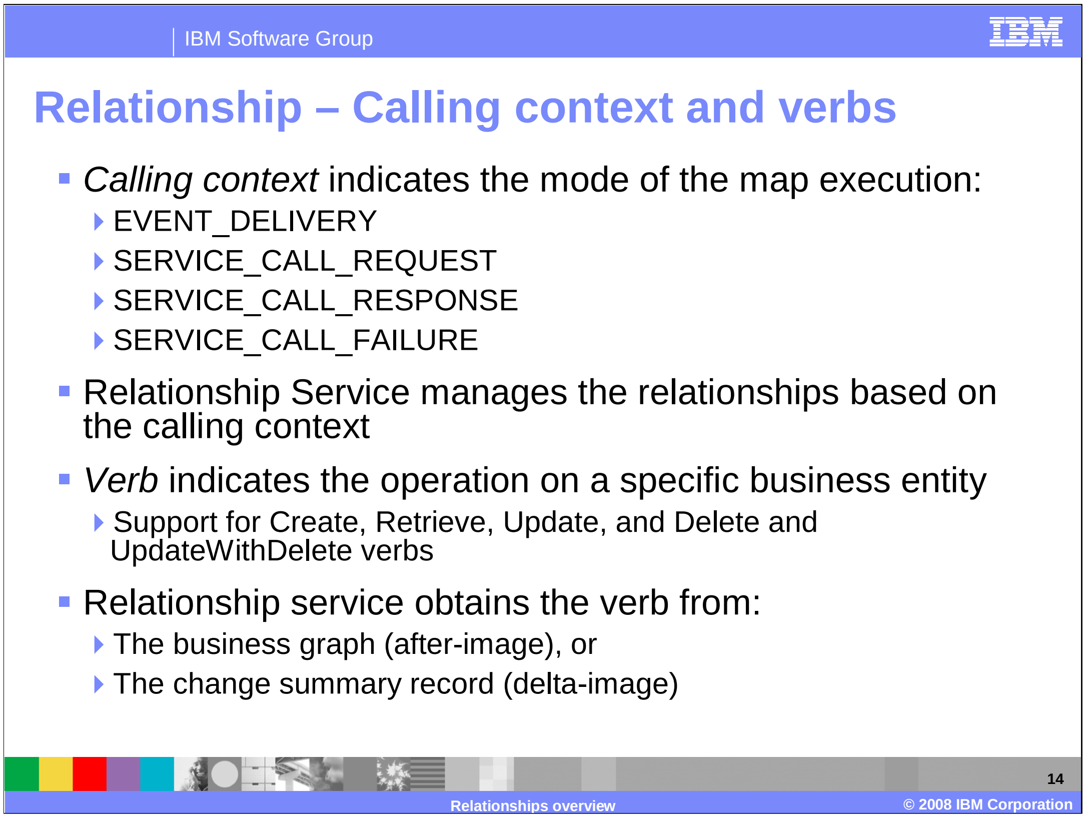


Lookup relationship

- Lookup relationships are used to perform static cross-referencing between data such as country codes that changes rarely, if ever

Data	RIID
China	1
France	2
Germany	3
Slovakia	4
USA	5

A second type of relationship is a lookup relationship. This type of relationship is used to perform a static cross referencing between data that does not change frequently, for example, country codes or zip codes. In this example, one system stores country names in the specific format of spelling out the country name, while another system stores country names in the specific format of a numerical country name representation. There is a need to maintain a cross reference between these two different representations of the same business value. The set of static references can be predefined in a persistent data store and then used by a lookup relationship at runtime. When a customer entity comes in to the integration application from the EIS1 system, the relationship service will look up the relationship instance ID and use it when the generic customer entity is being transformed to the EIS2 format.



Relationship – Calling context and verbs

- *Calling context* indicates the mode of the map execution:
  - EVENT_DELIVERY
  - SERVICE_CALL_REQUEST
  - SERVICE_CALL_RESPONSE
  - SERVICE_CALL_FAILURE
- Relationship Service manages the relationships based on the calling context
- *Verb* indicates the operation on a specific business entity
  - Support for Create, Retrieve, Update, and Delete and UpdateWithDelete verbs
- Relationship service obtains the verb from:
  - The business graph (after-image), or
  - The change summary record (delta-image)

As previously stated, the relationship service maintains the relationships based on the calling context. There are four calling contexts that are passed to the relationship service from the map that receives the calling context from the interface map component. The four calling contexts and their definitions are:

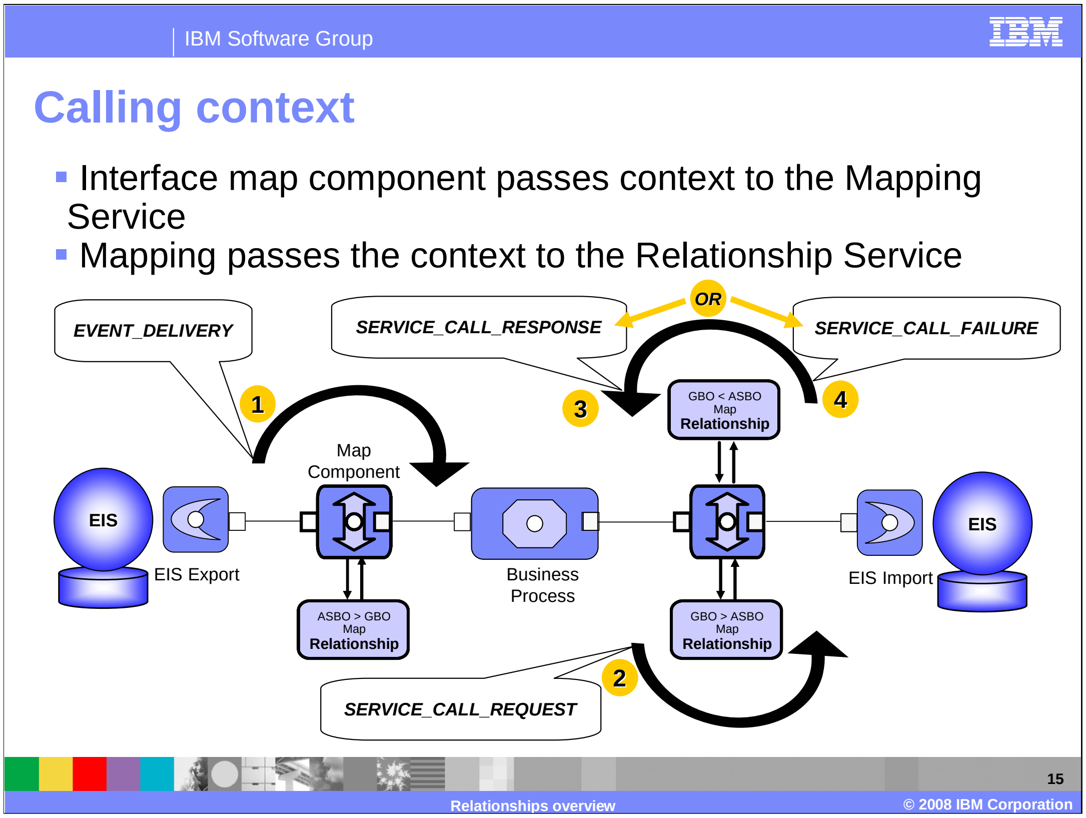
Event delivery is a one way invocation, for example, the map (and the relationship) is invoked on the request path with no response expected;

Service call request is a request invocation where the map (and the relationship) is invoked on the request path and the response is expected;

Service call response is a response invocation where the map (and the relationship) is invoked on the response path with a completion status;

Service call failure is a response invocation where the map (and the relationship) is invoked on the response path with a failure status.

The relationship service manages the implemented relationships based on the received calling context and based on the verb in the business graph that indicates the operation that has or is being performed on a business entity. The verbs include create, retrieve, update, delete, and UpdatewithDelete.



IBM Software Group

Calling context

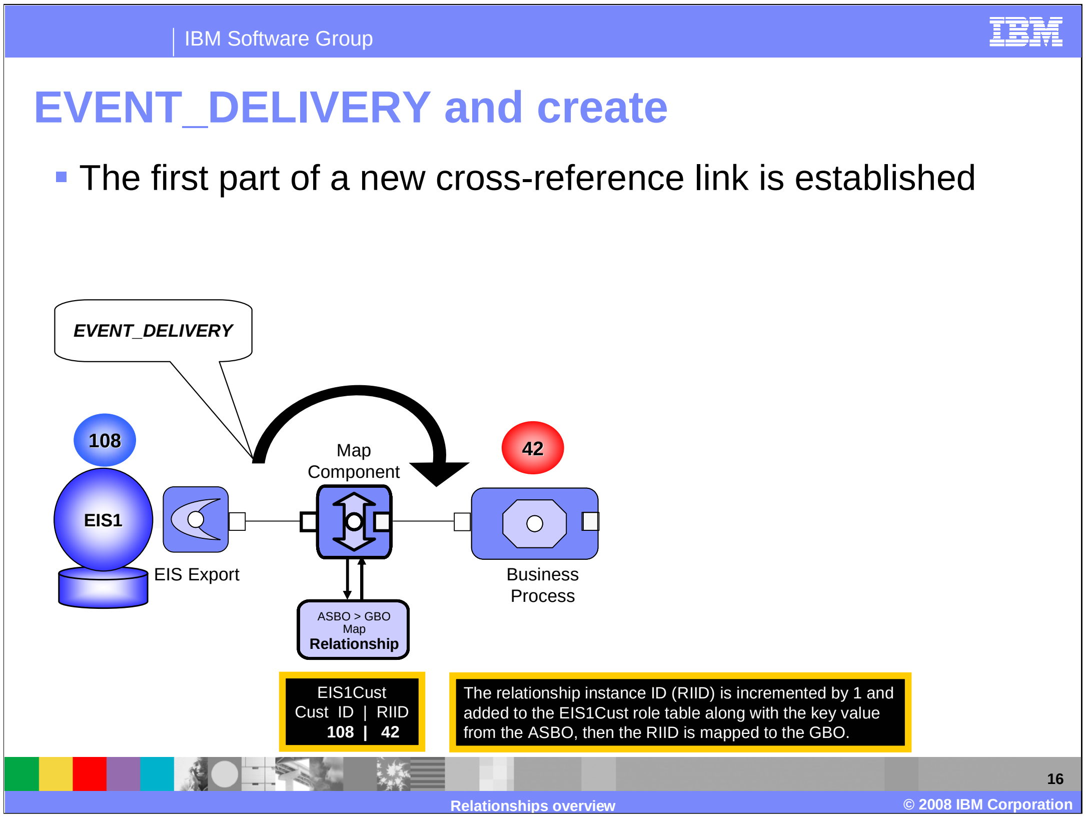
- Interface map component passes context to the Mapping Service
- Mapping passes the context to the Relationship Service

15

Relationships overview

© 2008 IBM Corporation

Maps implementing relationships pass the calling context to the relationship service to initially establish references between business data entities across the integration environment and to maintain already established references between related business data entities. Over the next several slides, you will take a closer look at the four calling contexts and how the calling context and the verb are used together to determine the behavior of the relationship service.



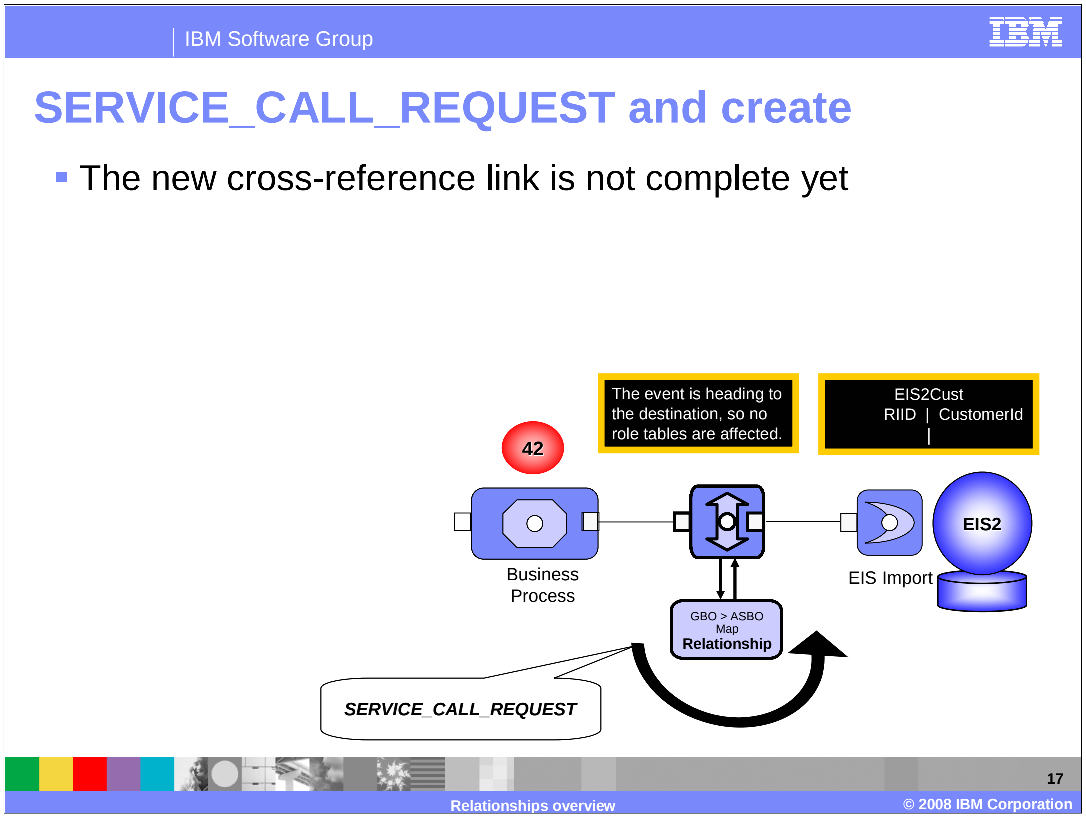
EVENT_DELIVERY and create

- The first part of a new cross-reference link is established

EIS1Cust	
Cust ID	RIID
108	**42**

The relationship instance ID (RIID) is incremented by 1 and added to the EIS1Cust role table along with the key value from the ASBO, then the RIID is mapped to the GBO.

In this scenario, a new business entity has been created in the source application and is being integrated across the solution. The calling context is event delivery which indicates a new event coming in and the verb in this example is create. At this point, the first part of a new cross reference link between the application specific entity and the generic representation of that entity is established. The mapping transformation invokes the implemented relationship. If there is an existing instance found for the input entity, then the retrieved instance ID is set in the key attribute property of the generic business entity. If no instance exists, then the relationship service generates a new unique relationship instance ID for the generic business entity. The relationship service creates a new entity in the role table and maps the unique ID of 108 from the application specific entity to the unique relationship instance ID of 42 for the generic business entity.



IBM Software Group

SERVICE_CALL_REQUEST and create

- The new cross-reference link is not complete yet

The event is heading to the destination, so no role tables are affected.

EIS2Cust
RIID | CustomerId
|

42

Business Process

GBO > ASBO
Map
Relationship

EIS Import

EIS2

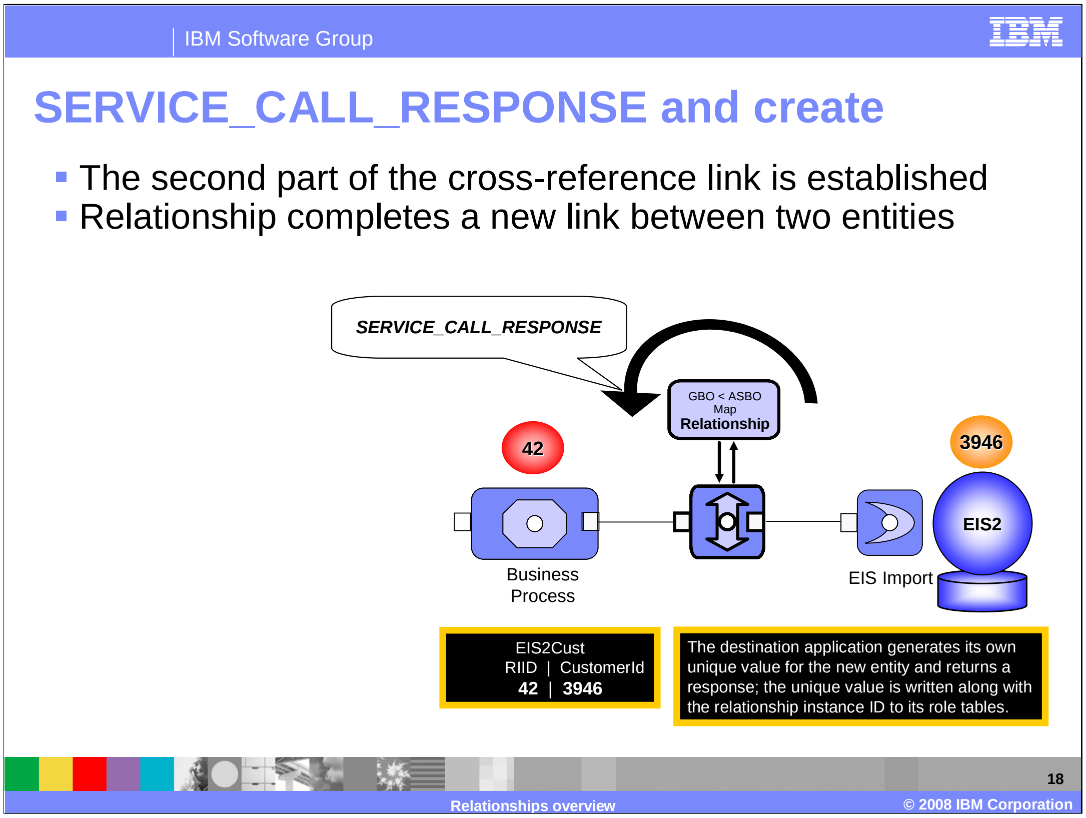
SERVICE_CALL_REQUEST

17

Relationships overview

© 2008 IBM Corporation

Continuing with this example, here you see the calling context of service call request with the verb create as the generic business entity is sent as a create request to the target destination system. The mapping transformation invokes the implemented relationship. The relationship service does not know a new unique ID for the destination business entity as it has not yet been created. Therefore, the relationship service does not create a new entry in the role table at this time.



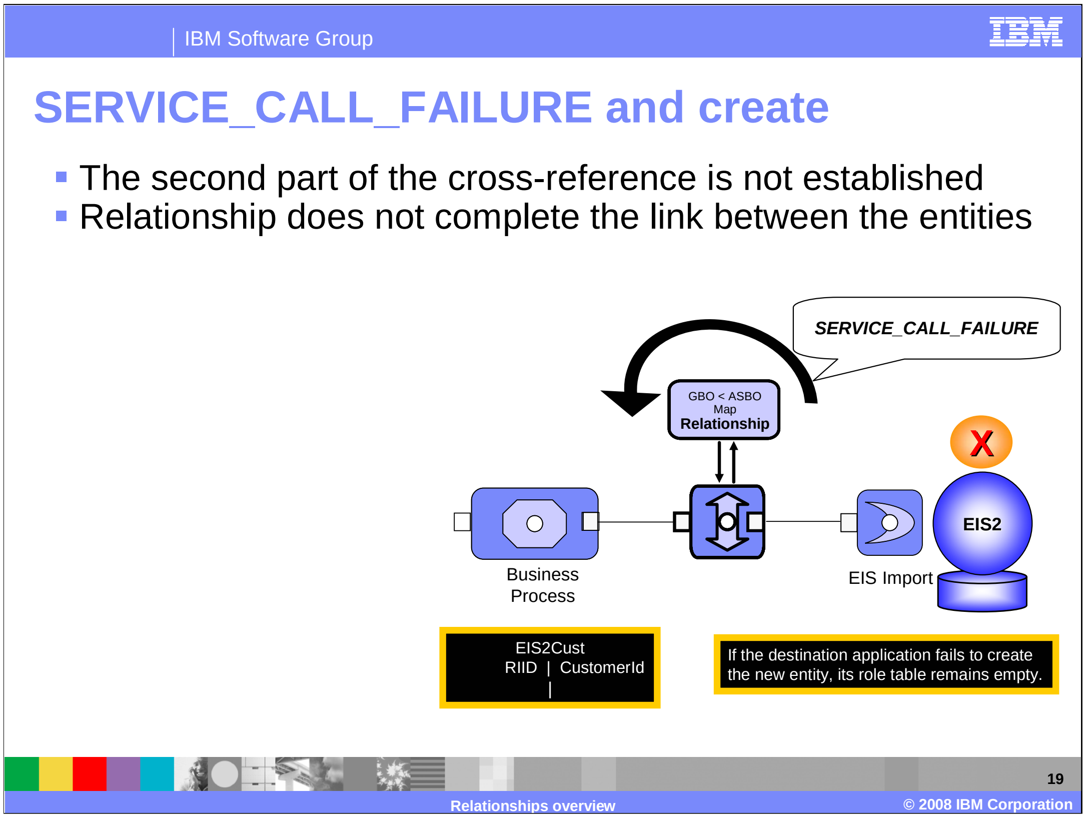
SERVICE_CALL_RESPONSE and create

- The second part of the cross-reference link is established
- Relationship completes a new link between two entities

EIS2Cust
RIID | CustomerId
42 | 3946

The destination application generates its own unique value for the new entity and returns a response; the unique value is written along with the relationship instance ID to its role tables.

Continuing with this example, here you see that the target destination application has successfully created the business entity in its system and a response containing the application specific unique ID is being returned to the integration solution. In this case, the calling context is of service call response with the verb Create. The mapping transformation invokes the implemented relationship. The relationship service receives a new unique ID for the destination business entity. The relationship services creates a new entry in the role table, completing the second part of the cross reference link, as it now has the necessary primary key information to map the newly created application specific business entity with its CustomerId of 3946 to the generic representation of that business entity with its relationship instance ID of 42.



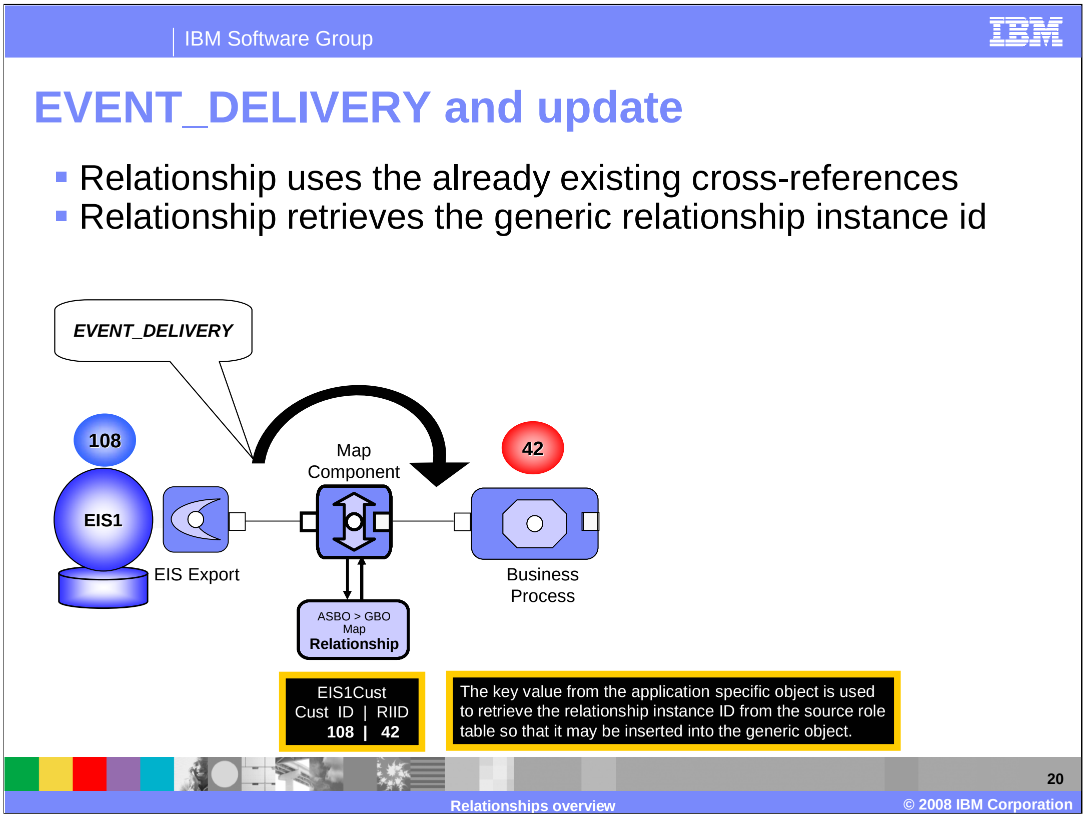
SERVICE_CALL_FAILURE and create

- The second part of the cross-reference is not established
- Relationship does not complete the link between the entities

EIS2Cust
RIID | CustomerId
|

If the destination application fails to create the new entity, its role table remains empty.

If the target destination application fails to create a new business entity, the response contains the calling context of Service Call Failure with the verb Create. Again, the mapping transformation invokes the implemented relationship; however, the relationship service does not receive a new unique ID for the destination business entity as there isn’t one. The relationship services do not create a new entry in the role table.



EVENT_DELIVERY and update

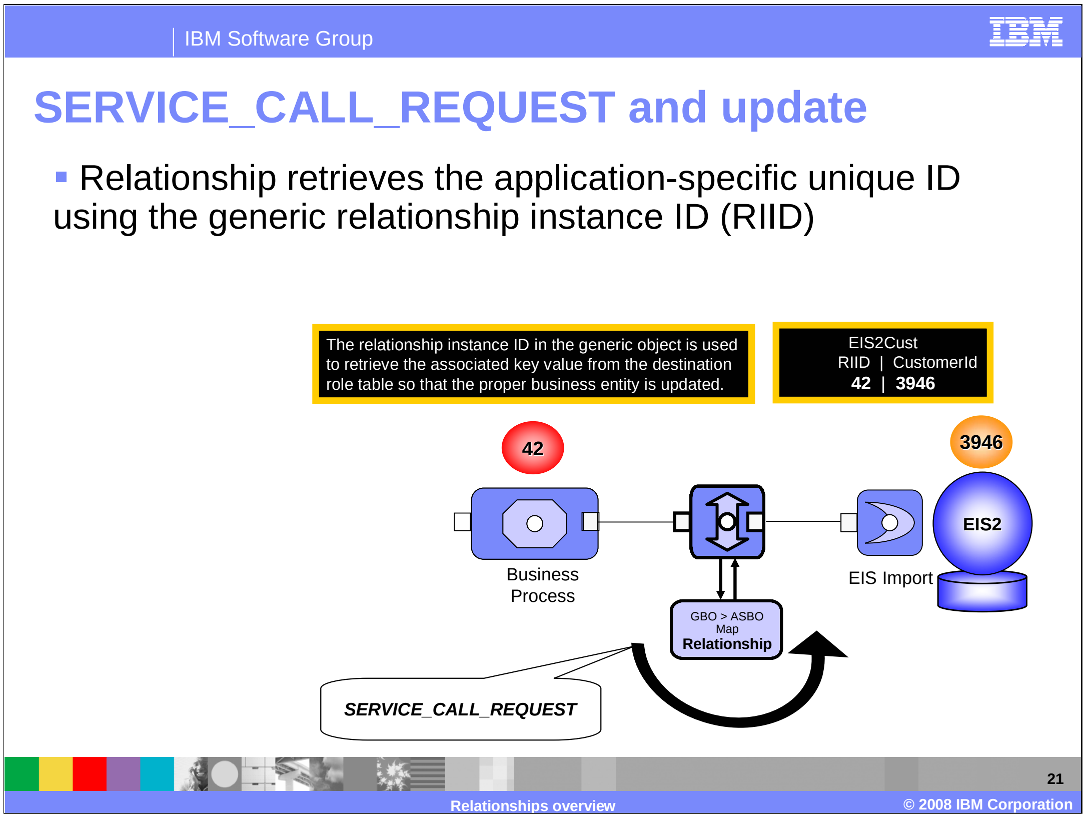
- Relationship uses the already existing cross-references
- Relationship retrieves the generic relationship instance id

EIS1Cust

Cust ID	RIID
108	**42**

The key value from the application specific object is used to retrieve the relationship instance ID from the source role table so that it may be inserted into the generic object.

Continuing with this example, you will look at what happens with the different calling contexts when the verb is Update. A new event comes in to the system with a calling context of Event Delivery and a verb of Update, indicating there have been changes made to the specific business entity. These changes need to be integrated and synchronized across to the other systems participating in the relationship. The mapping transformation invokes the implemented relationship. The relationship service uses the Cust_ID primary key of 108 to look up and retrieve the ID of 42 for the generic business entity from the role table. The relationship service maps the unique ID to the generic business entity. If no generic business entity is found for the input, then a DataNotFoundException is thrown to indicate that an expected entry could not be found.



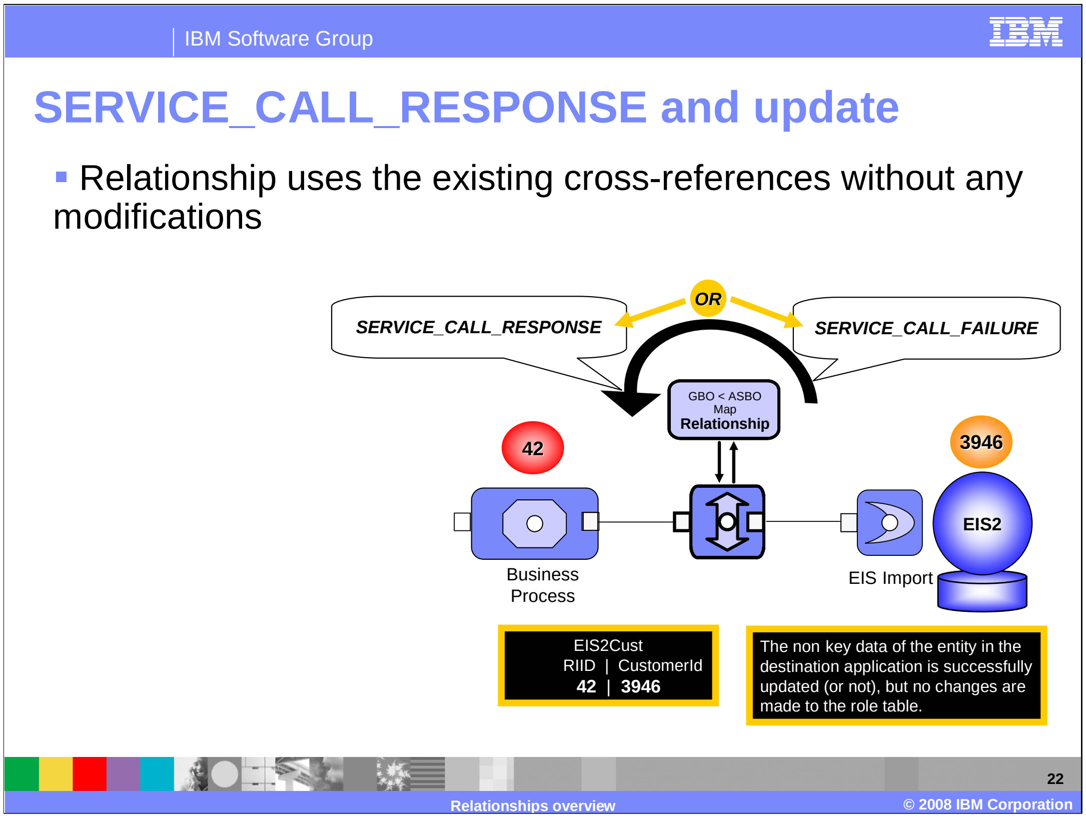
SERVICE_CALL_REQUEST and update

- Relationship retrieves the application-specific unique ID using the generic relationship instance ID (RIID)

The relationship instance ID in the generic object is used to retrieve the associated key value from the destination role table so that the proper business entity is updated.

EIS2Cust	
RIID	CustomerId
42	**3946**

The generic business entity is sent through the system as a request to the target destination application with the calling context of Service Call Request and a verb of Update. The mapping transformation invokes the implemented relationship. In this example, the relationship service uses the relationship instance ID of 42 in the generic business entity to look up the primary key of 3946 of the associated application specific business entity, and it retrieves the unique ID for the destination business entity. The relationship service does not modify the role table since the cross-reference already exists. If no entry exists for the input generic business entity, then a DataNotFoundException is thrown to indicate that an expected entry could not be fount.



SERVICE_CALL_RESPONSE and update

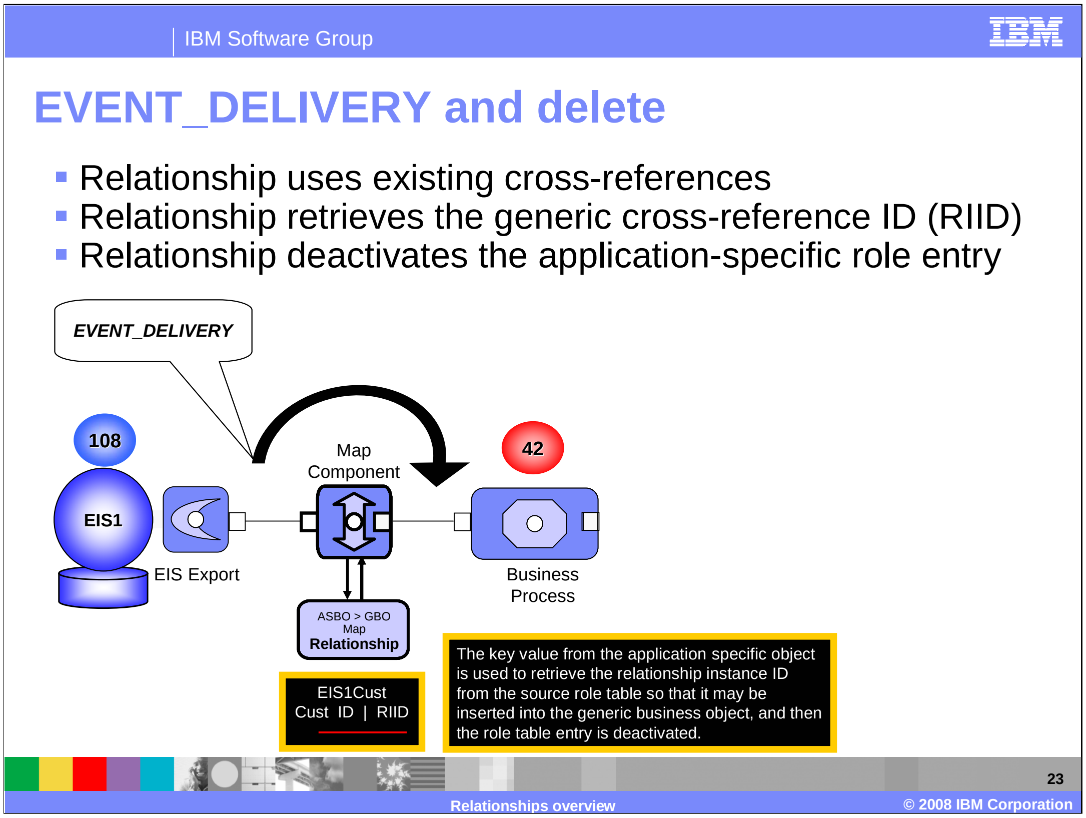
- Relationship uses the existing cross-references without any modifications

EIS2Cust
RIID | CustomerId
42 | 3946

The non key data of the entity in the destination application is successfully updated (or not), but no changes are made to the role table.

The destination application updates the existing business entity and returns the response. The calling context in this case is Service Call Response and the verb is Update. The mapping transformation invokes the implemented relationship. The relationship service does not modify the role table since the cross-reference already exists. If there is no entry found for the input business entity, then a DataNotFoundException is thrown to indicate that an expected participant could not be found.

If the calling context is Service Call Failure, the original Input business object is used to populate the output business object. No retrieval or modification of entries is done.



IBM Software Group

EVENT_DELIVERY and delete

- Relationship uses existing cross-references
- Relationship retrieves the generic cross-reference ID (RIID)
- Relationship deactivates the application-specific role entry

EIS1Cust
Cust ID | RIID

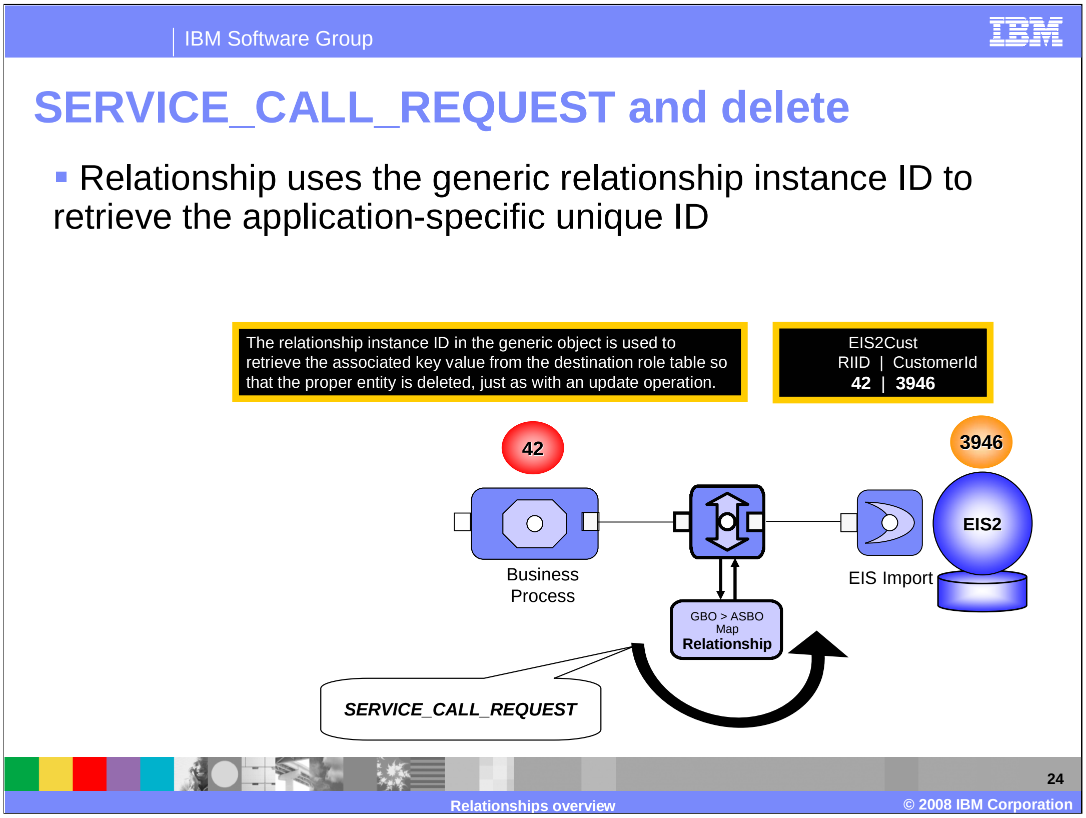
The key value from the application specific object is used to retrieve the relationship instance ID from the source role table so that it may be inserted into the generic business object, and then the role table entry is deactivated.

23

Relationships overview

© 2008 IBM Corporation

Continuing with this example, you will look at what happens with the different calling contexts when the verb is Delete. In this case, an existing business entity has been deleted in the source application. The calling context is Event Delivery and the verb is Delete. The mapping transformation invokes the implemented relationship. The relationship service retrieves the ID for the generic business entity from the role table. The relationship service changes the logical state of the cross-reference entry in the role table to one of “inactive”. This is some times called a logical or soft delete. If no entry exists for the relationship, then a DataNotFoundException is thrown.



IBM Software Group

SERVICE_CALL_REQUEST and delete

- Relationship uses the generic relationship instance ID to retrieve the application-specific unique ID

The relationship instance ID in the generic object is used to retrieve the associated key value from the destination role table so that the proper entity is deleted, just as with an update operation.

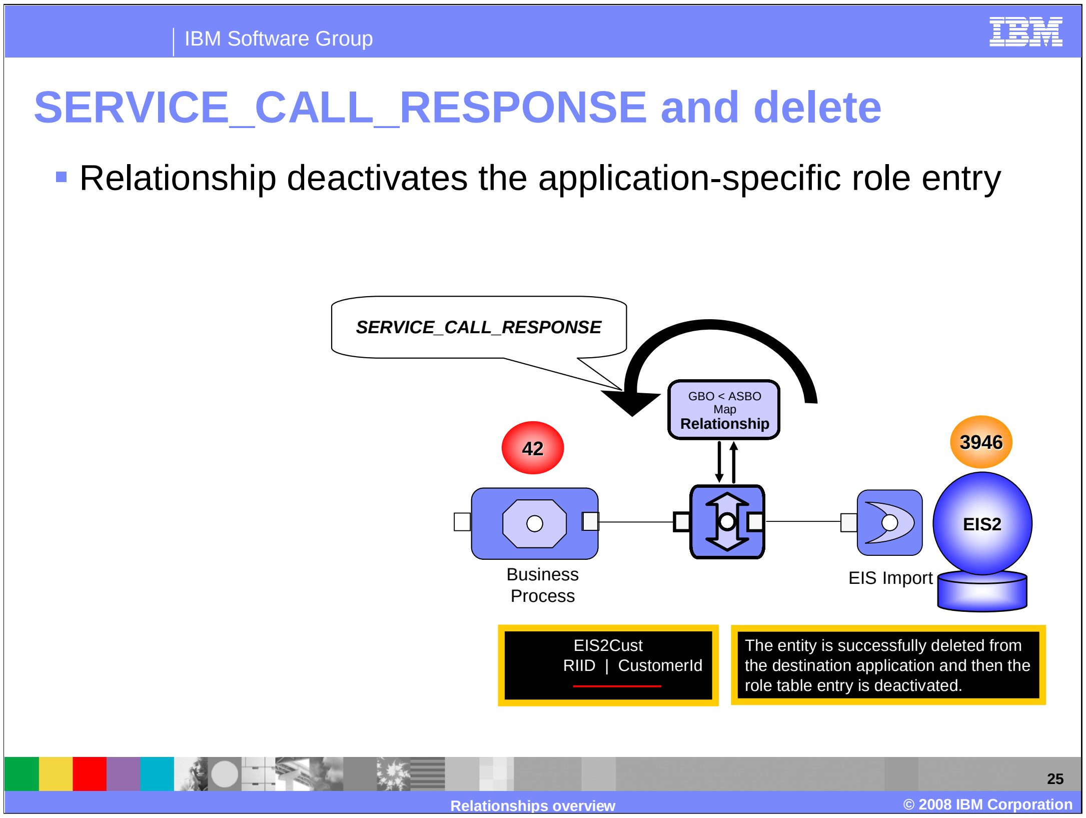
EIS2Cust

RIID	CustomerId
42	**3946**

24

Relationships overview

The generic business entity is sent through the system as a request to the target destination application with the calling context of Service Call Request and a verb of Delete. The mapping transformation invokes the implemented relationship. In this example, the relationship service uses the relationship instance ID of 42 in the generic business entity to look up the primary key of 3946 of the associated application specific business entity, and it retrieves the unique ID for the destination business entity. The relationship role table remains unaffected until the destination entity is successfully deleted. If no entry is found for input business entity, then a DataNotFoundException is thrown.



IBM Software Group

SERVICE_CALL_RESPONSE and delete

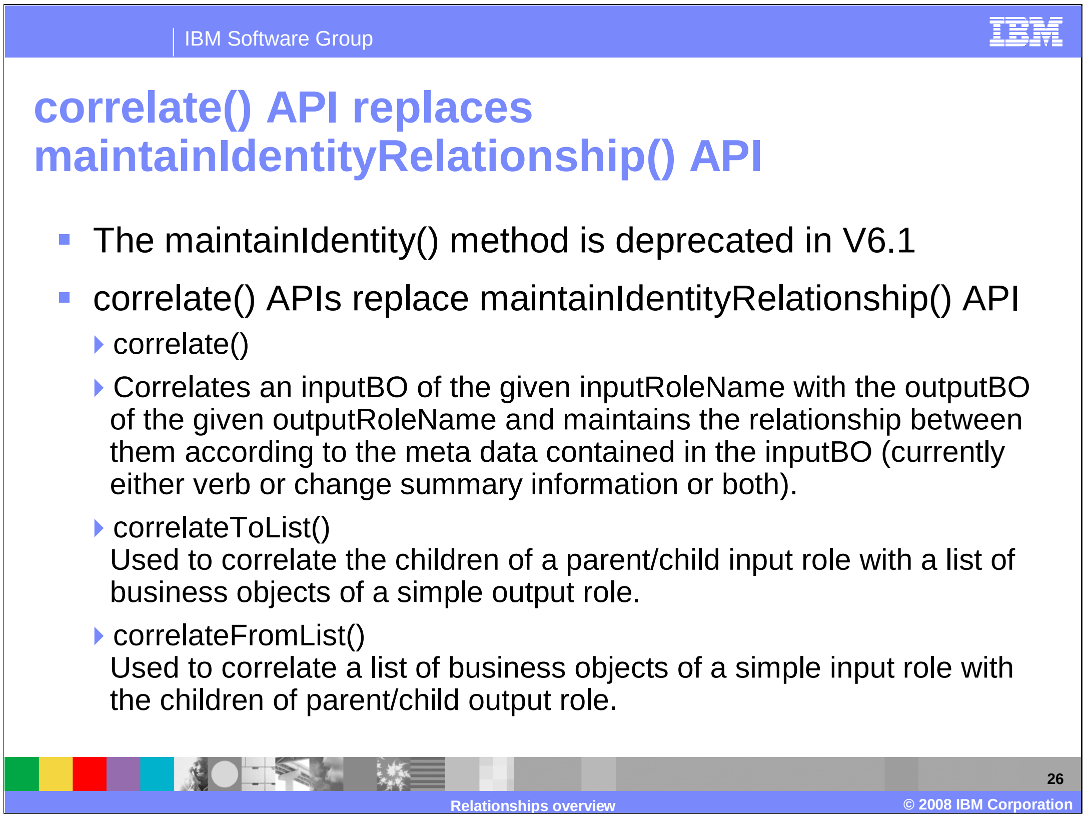
- Relationship deactivates the application-specific role entry

The entity is successfully deleted from the destination application and then the role table entry is deactivated.

25

Relationships overview

Upon successful deletion of the entity in the destination system, a response is returned to the integration solution. The calling context is Service Call Response and the verb is Delete. The mapping transformation invokes the implemented relationship. The relationship changes the logical state of the cross-reference entry in the role table to one of “inactive” by performing a soft or logical delete. In the case where the calling context is a Service Call Failure, the relationship service does not change the state of the cross-reference entity in the role table, as the business entity was not deleted in the destination system.



IBM Software Group

correlate() API replaces maintainIdentityRelationship() API

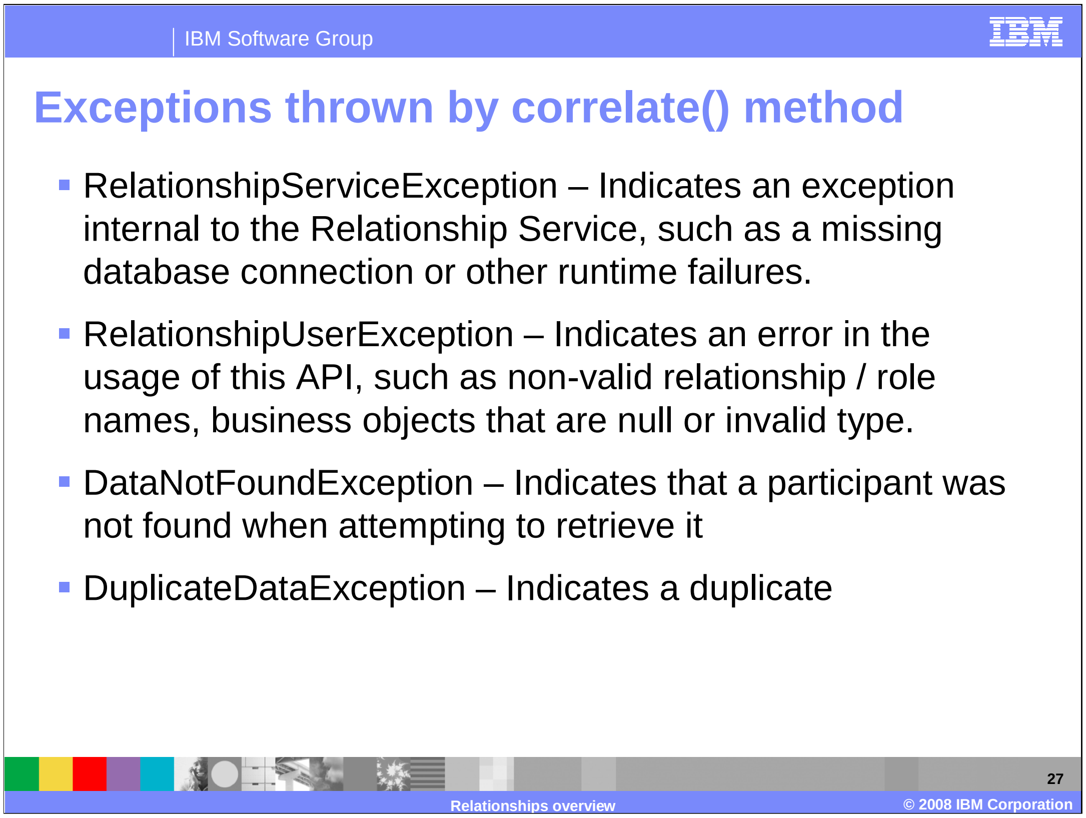
- The maintainIdentity() method is deprecated in V6.1
- correlate() APIs replace maintainIdentityRelationship() API
  - correlate()
  - Correlates an inputBO of the given inputRoleName with the outputBO of the given outputRoleName and maintains the relationship between them according to the meta data contained in the inputBO (currently either verb or change summary information or both).
  - correlateToList()
    Used to correlate the children of a parent/child input role with a list of business objects of a simple output role.
  - correlateFromList()
    Used to correlate a list of business objects of a simple input role with the children of parent/child output role.

26

Relationships overview

© 2008 IBM Corporation

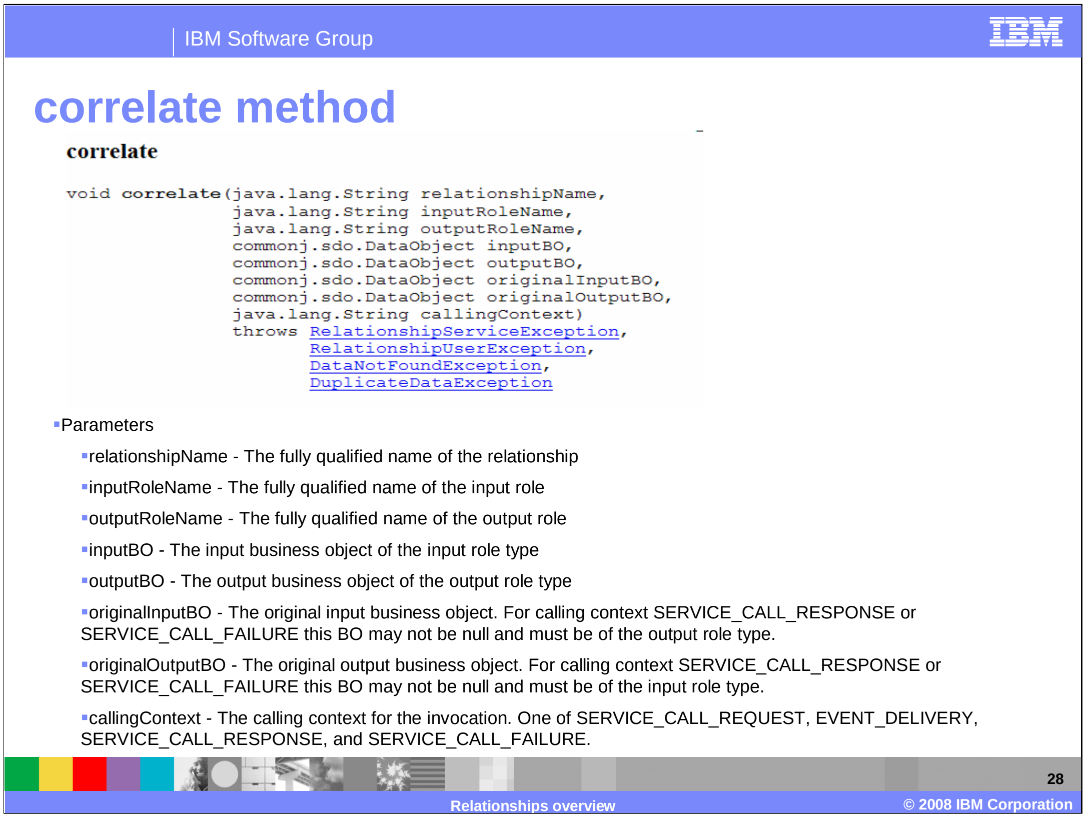
In version 6.1, the maintainIdentityRelationship method is deprecated and replaced with the more robust correlate() methods. The relationship service uses these correlate methods to manage and maintain the relationships. The calling context and verb scenarios you just reviewed are all handled by the relationship service through the use of the application programming interface.



Exceptions thrown by correlate() method

- RelationshipServiceException – Indicates an exception internal to the Relationship Service, such as a missing database connection or other runtime failures.
- RelationshipUserException – Indicates an error in the usage of this API, such as non-valid relationship / role names, business objects that are null or invalid type.
- DataNotFoundException – Indicates that a participant was not found when attempting to retrieve it
- DuplicateDataException – Indicates a duplicate

The correlate() method can throw these exceptions; RelationshipServiceException (which indicates a problem with the relationship service itself); RelationshipUserException (which indicates a user error in the use of the API); DataNotFoundException (which indicates a participant was not found when trying to retrieve it); and DuplicateDataException, indicating duplicated data.



correlate method

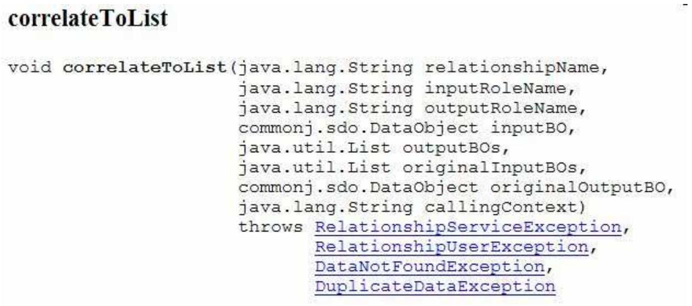
correlate

```
void correlate(java.lang.String relationshipName,
               java.lang.String inputRoleName,
               java.lang.String outputRoleName,
               commonj.sdo.DataObject inputBO,
               commonj.sdo.DataObject outputBO,
               commonj.sdo.DataObject originalInputBO,
               commonj.sdo.DataObject originalOutputBO,
               java.lang.String callingContext)
               throws RelationshipServiceException,
                      RelationshipUserException,
                      DataNotFoundException,
                      DuplicateDataException
```

- Parameters
  - relationshipName - The fully qualified name of the relationship
  - inputRoleName - The fully qualified name of the input role
  - outputRoleName - The fully qualified name of the output role
  - inputBO - The input business object of the input role type
  - outputBO - The output business object of the output role type
  - originalInputBO - The original input business object. For calling context SERVICE_CALL_RESPONSE or SERVICE_CALL_FAILURE this BO may not be null and must be of the output role type.
  - originalOutputBO - The original output business object. For calling context SERVICE_CALL_RESPONSE or SERVICE_CALL_FAILURE this BO may not be null and must be of the input role type.
  - callingContext - The calling context for the invocation. One of SERVICE_CALL_REQUEST, EVENT_DELIVERY, SERVICE_CALL_RESPONSE, and SERVICE_CALL_FAILURE.

Here you see the signature of the correlate method.

correlateToList method



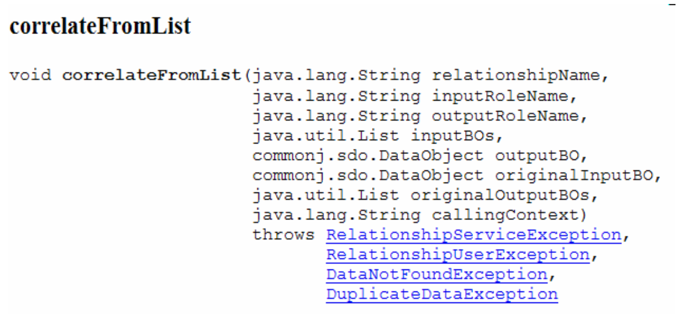
correlateToList

```
void correlateToList(java.lang.String relationshipName,
                     java.lang.String inputRoleName,
                     java.lang.String outputRoleName,
                     commonj.sdo.DataObject inputBO,
                     java.util.List outputBOs,
                     java.util.List originalInputBOs,
                     commonj.sdo.DataObject originalOutputBO,
                     java.lang.String callingContext)
                     throws RelationshipServiceException,
                            RelationshipUserException,
                            DataNotFoundException,
                            DuplicateDataException
```

- Parameters
  - relationshipName - The fully qualified name of the relationship
  - inputRoleName - The fully qualified name of the input role
  - outputRoleName - The fully qualified name of the output role
  - inputBO - The input business object of the input role type
  - outputBOs – A list of output business objects of the output role type
  - originalInputBOs – A list of original input business objects. For calling context SERVICE_CALL_RESPONSE or SERVICE_CALL_FAILURE this list may not be null and must be of the output role type.
  - originalOutputBO - The original output business object. For calling context SERVICE_CALL_RESPONSE or SERVICE_CALL_FAILURE this BO may not be null and must be of the input role type.
  - callingContext - The calling context for the invocation. One of SERVICE_CALL_REQUEST, EVENT_DELIVERY, SERVICE_CALL_RESPONSE, and SERVICE_CALL_FAILURE.

Here you see the signature of the correlateToList method. This method is used to correlate the children of a parent/child input role with a list of business objects of a simple output role.

correlateFromList method

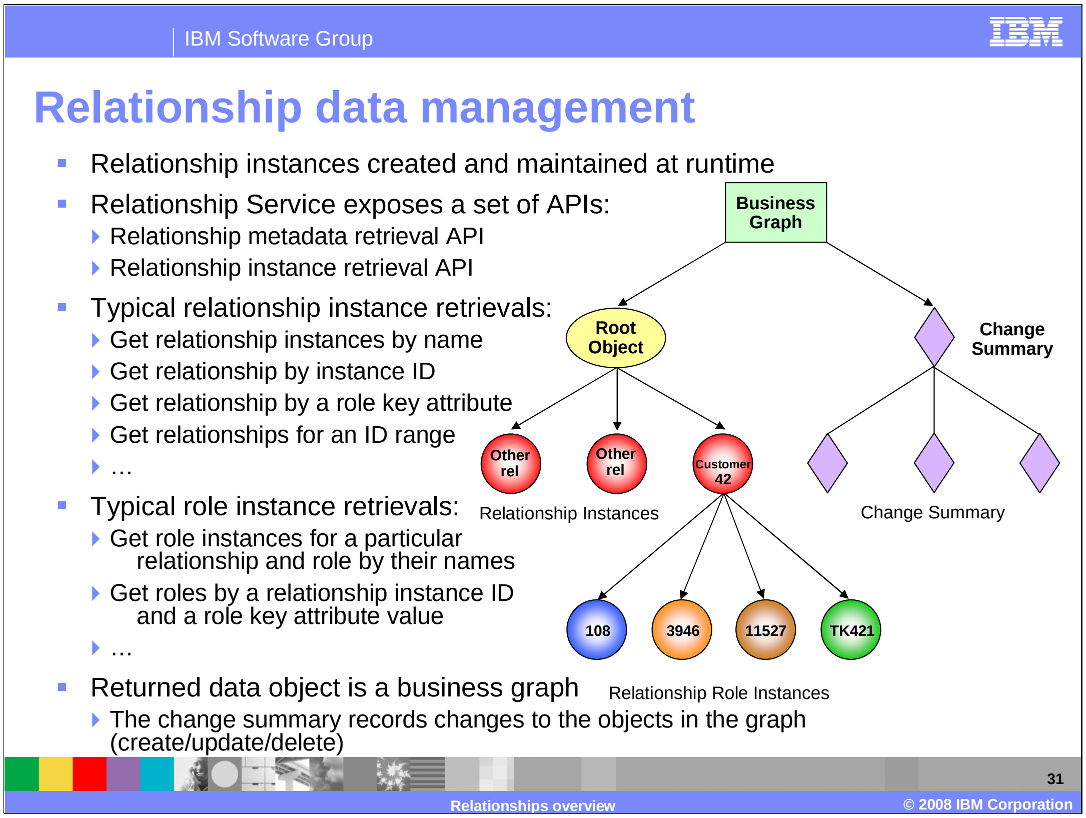


correlateFromList

```
void correlateFromList(java.lang.String relationshipName,
                       java.lang.String inputRoleName,
                       java.lang.String outputRoleName,
                       java.util.List inputBOs,
                       commonj.sdo.DataObject outputBO,
                       commonj.sdo.DataObject originalInputBO,
                       java.util.List originalOutputBOs,
                       java.lang.String callingContext)
                       throws RelationshipServiceException,
                              RelationshipUserException,
                              DataNotFoundException,
                              DuplicateDataException
```

- Parameters
  - relationshipName - The fully qualified name of the relationship
  - inputRoleName - The fully qualified name of the input role
  - outputRoleName - The fully qualified name of the output role
  - inputBO - The input business object of the input role type
  - outputBO - The output business object of the output role type
  - originalInputBO - The original input business object. For calling context SERVICE_CALL_RESPONSE or SERVICE_CALL_FAILURE this BO may not be null and must be of the output role type.
  - originalOutputBO - The original output business object. For calling context SERVICE_CALL_RESPONSE or SERVICE_CALL_FAILURE this BO may not be null and must be of the input role type.
  - callingContext - The calling context for the invocation. One of SERVICE_CALL_REQUEST, EVENT_DELIVERY, SERVICE_CALL_RESPONSE, and SERVICE_CALL_FAILURE.

Here you see the signature for the correlateFromList method. This method is used to correlate a list of business objects of a simple input role with the children of parent/child output role.



IBM Software Group

Relationship data management

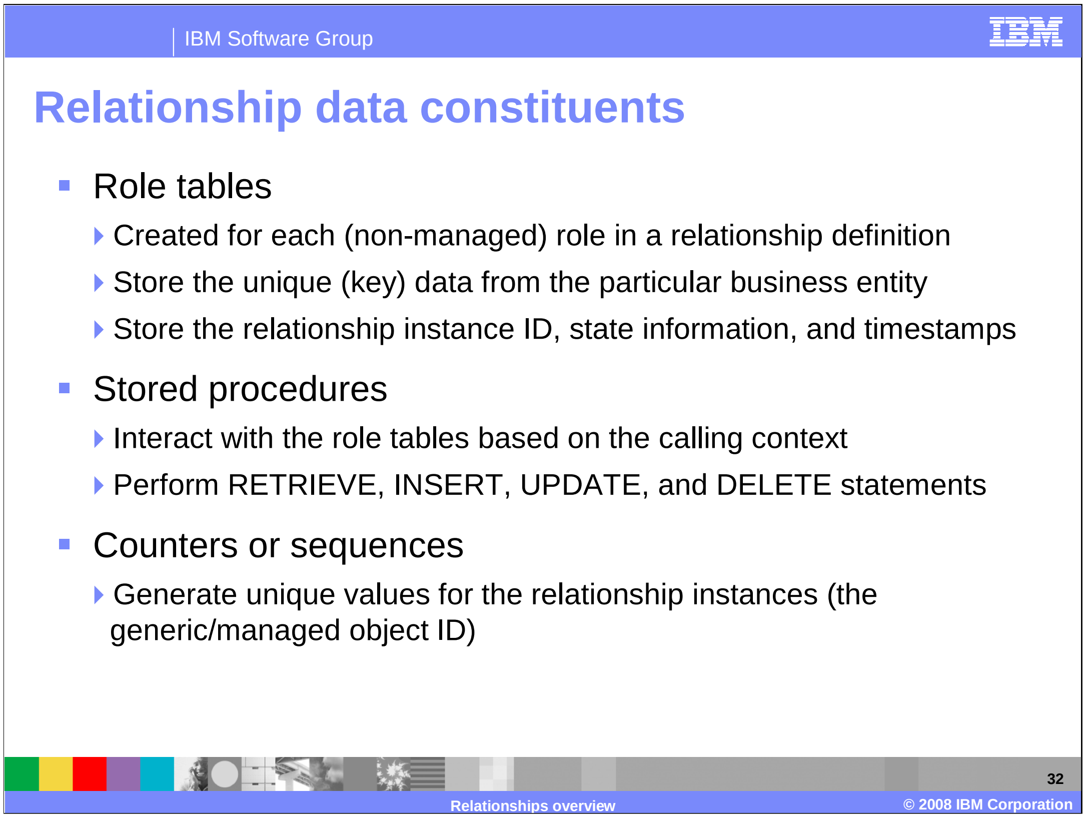
- Relationship instances created and maintained at runtime
- Relationship Service exposes a set of APIs:
  - Relationship metadata retrieval API
  - Relationship instance retrieval API
- Typical relationship instance retrievals:
  - Get relationship instances by name
  - Get relationship by instance ID
  - Get relationship by a role key attribute
  - Get relationships for an ID range
  - …
- Typical role instance retrievals:
  - Get role instances for a particular relationship and role by their names
  - Get roles by a relationship instance ID and a role key attribute value
  - …
- Returned data object is a business graph
  - The change summary records changes to the objects in the graph (create/update/delete)

31

Relationships overview

© 2008 IBM Corporation

Relationship instances are created and maintained at runtime. The set of correlate() APIs that were just discussed are used to retrieve metadata and instance data. Listed here are a few of the typical data retrievals for relationship instances and role instances. The returned data object is a business graph.



IBM Software Group

Relationship data constituents

- Role tables
  - Created for each (non-managed) role in a relationship definition
  - Store the unique (key) data from the particular business entity
  - Store the relationship instance ID, state information, and timestamps
- Stored procedures
  - Interact with the role tables based on the calling context
  - Perform RETRIEVE, INSERT, UPDATE, and DELETE statements
- Counters or sequences
  - Generate unique values for the relationship instances (the generic/managed object ID)

32

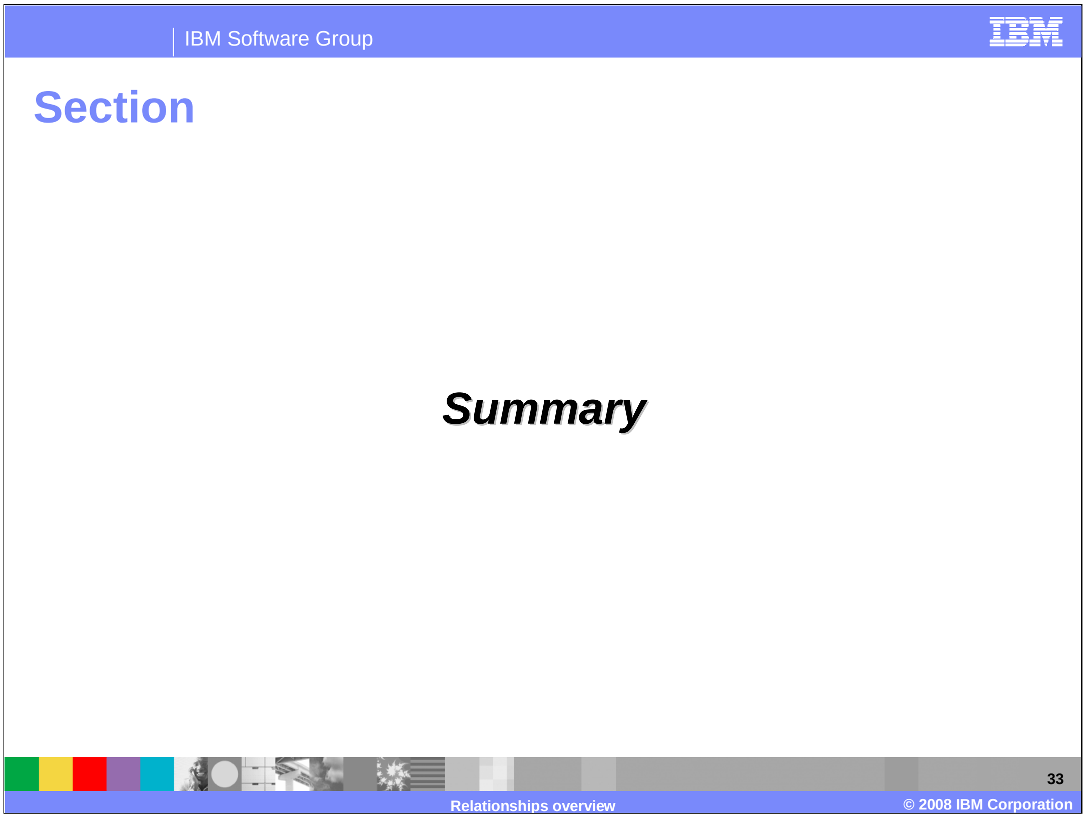
Relationships overview

© 2008 IBM Corporation

The relationship service performs all of the functions described using role tables and stored procedures. The relationship information is stored in role tables. The role tables are created for each non-managed role in a relationship definition. A Non-managed role represents the application specific business entity. All the application specific business entity roles are non-managed roles because the primary keys for those entities are provided by the applications themselves. The role tables store the unique keys from these application specific entities and the common reference which is the relationship instance ID.

A set of stored procedures is responsible for updating role tables in the persistent store database. Stored procedures are automatically generated when a map containing a relationship is deployed.

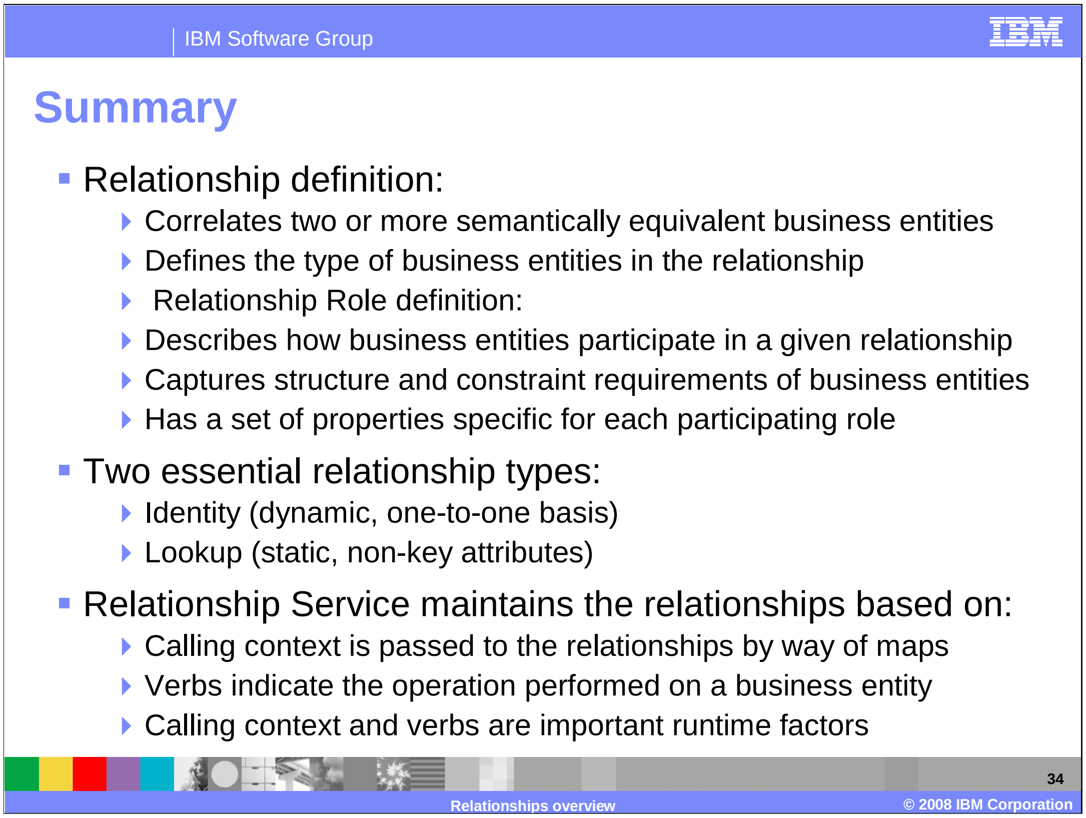
For managed roles, a sequence or counter column is used to generate the unique values for the relationship instance ID. The database used to generate the unique ID determines whether a sequence or counter is used to create the generic representation that exists within the WebSphere Process Server.



Section

Summary

This section will provide a summary of this presentation.



IBM Software Group

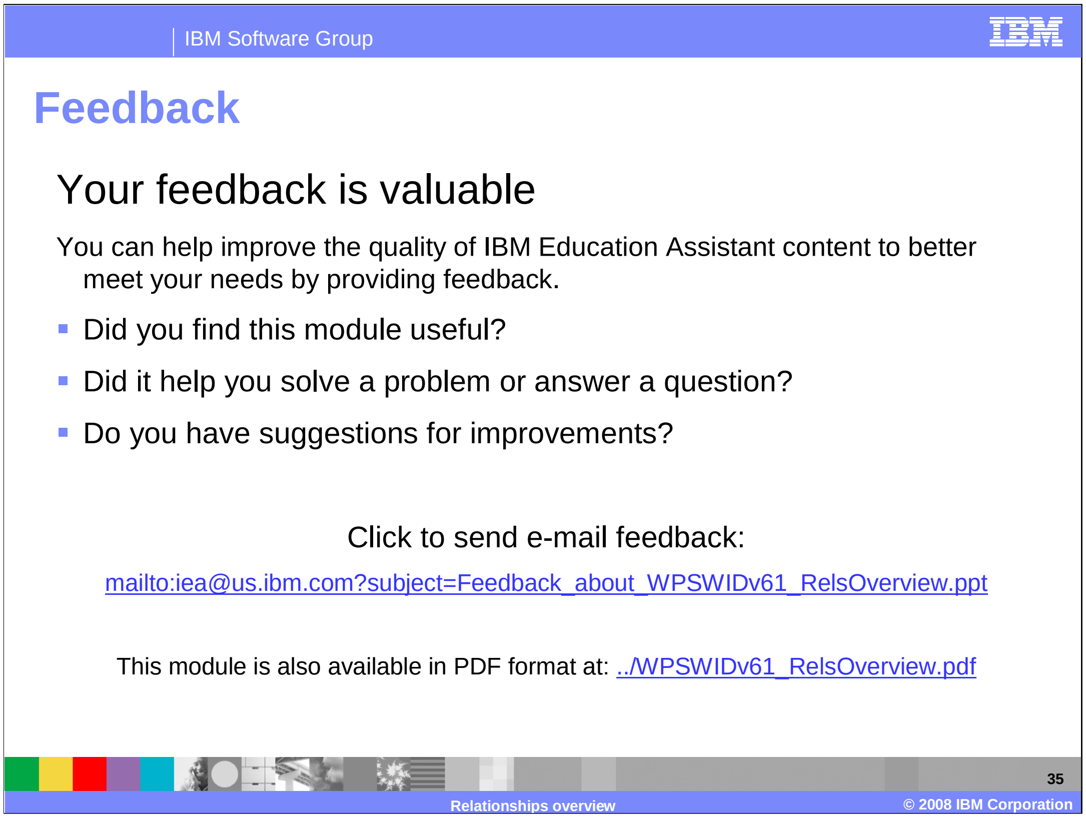
Summary

- Relationship definition:
  - Correlates two or more semantically equivalent business entities
  - Defines the type of business entities in the relationship
  - Relationship Role definition:
  - Describes how business entities participate in a given relationship
  - Captures structure and constraint requirements of business entities
  - Has a set of properties specific for each participating role
- Two essential relationship types:
  - Identity (dynamic, one-to-one basis)
  - Lookup (static, non-key attributes)
- Relationship Service maintains the relationships based on:
  - Calling context is passed to the relationships by way of maps
  - Verbs indicate the operation performed on a business entity
  - Calling context and verbs are important runtime factors

34

Relationships overview

To summarize, relationship definitions correlate two or more semantically equivalent business entities. They define the type of business entities that will be correlated in the relationship. Each definition contains two or more roles. There are two types of relationships, identity relationships which provide for data that changes frequently; and lookup relationships which provide for static data that rarely changes. The relationship service uses the calling context and the verb to determine what kind of relationship operation needs to be performed.



Feedback

Your feedback is valuable

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

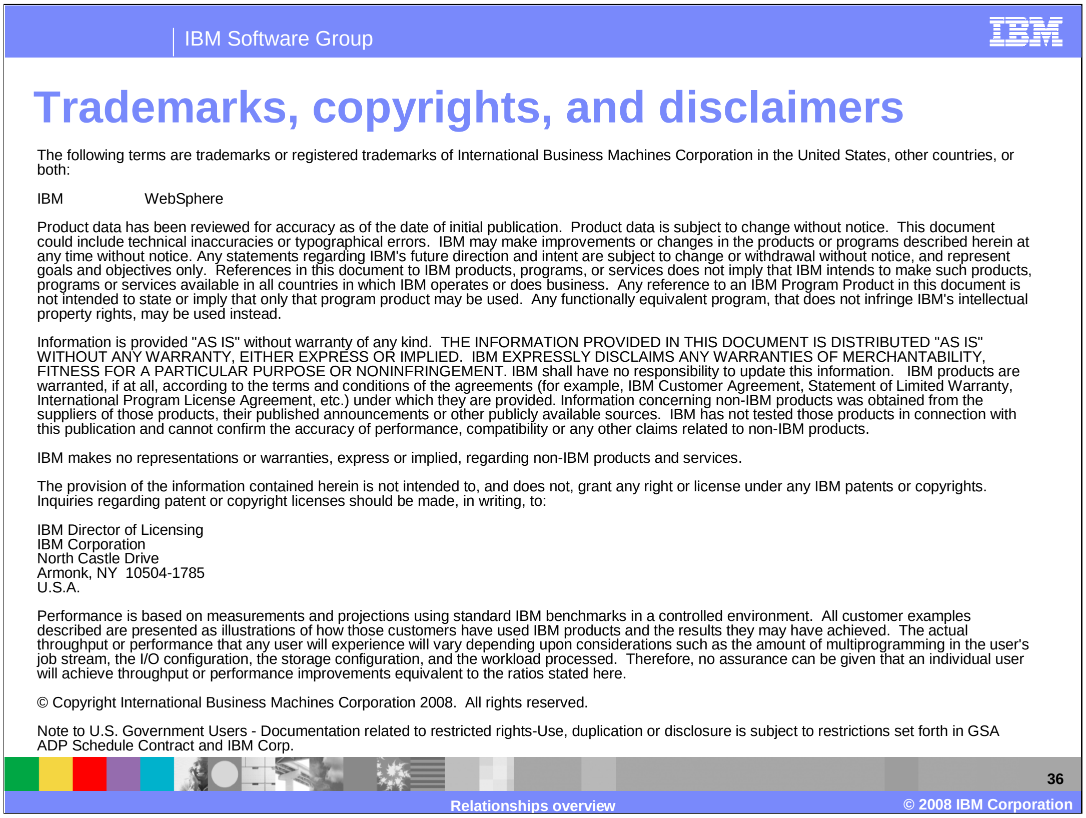
- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WPSWIDv61_RelsOverview.ppt

This module is also available in PDF format at: ../WPSWIDv61_RelsOverview.pdf

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



Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	WebSphere

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.