Introduction to Purchase Flow

Presenter's Name
Presenter's Title

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.
The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Oracle Training Materials – Usage Agreement

Use of this Site ("Site") or Materials constitutes agreement with the following terms and conditions:

1. Oracle Corporation ("Oracle") is pleased to allow its business partner ("Partner") to download and copy the information, documents, and the online training courses (collectively, "Materials") found on this Site. The use of the Materials is restricted to the non-commercial, internal training of the Partner's employees only. The Materials may not be used for training, promotion, or sales to customers or other partners or third parties.

2. All the Materials are trademarks of Oracle and are proprietary information of Oracle. Partner or other third party at no time has any right to resell, redistribute or create derivative works from the Materials.

3. Oracle disclaims any warranties or representations as to the accuracy or completeness of any Materials. Materials are provided "as is" without warranty of any kind, either express or implied, including without limitation warranties of merchantability, fitness for a particular purpose, and non-infringement.

4. Under no circumstances shall Oracle or the Oracle Authorized Delivery Partner be liable for any loss, damage, liability or expense incurred or suffered which is claimed to have resulted from use of this Site of Materials. As a condition of use of the Materials, Partner agrees to indemnify Oracle from and against any and all actions, claims, losses, damages, liabilities and expenses (including reasonable attorneys' fees) arising out of Partner's use of the Materials.

5. Reference materials including but not limited to those identified in the Boot Camp manifest can not be redistributed in any format without Oracle written consent.

Agenda

- Purchase Process Subsystems
- Commerce Object Model

Learning Objectives

At the end of this lesson you should be able to:

- Understand the interaction between various subsystems in the purchase process
- Learn about the various commerce objects that make up the Order
- Use the relationships objects to represent dependencies between commerce objects

Section 1:
Purchase Process Subsystems

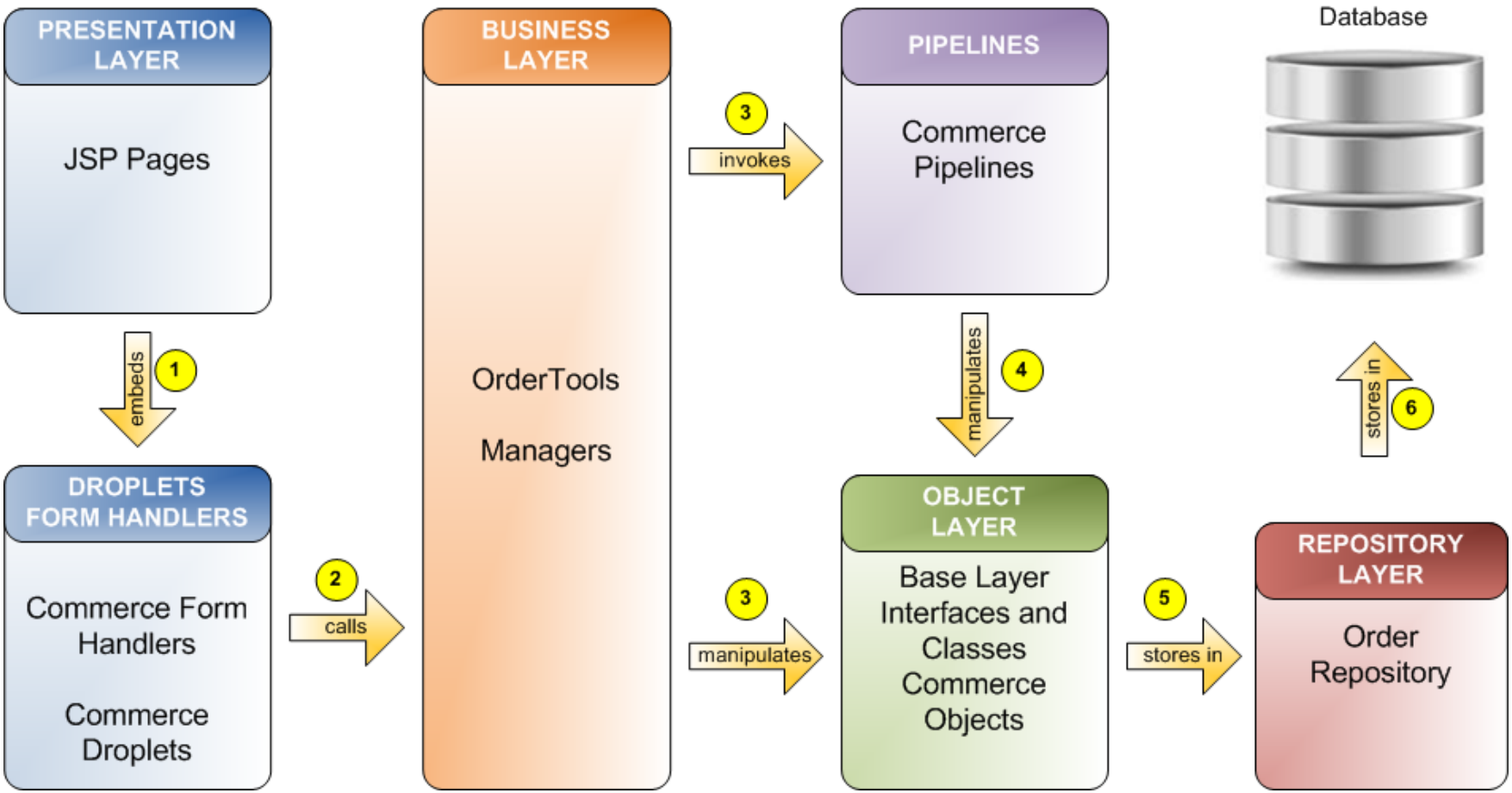
The Purchase Process Subsystems

- The purchase process represents a number of subsystems that facilitate the purchase and checkout of an order by the customer.
- The purchase process is made of the following subsystems:
  - Base commerce classes and interfaces,
  - Commerce form handlers and droplets,
  - Business layer classes,
  - Pipelines,
  - Order repository.
- These subsystems interact with each other to collect information, validate it, and place an order.

Interactions Between Subsystems (1)

- JSP pages embed commerce form handlers and droplets.
- The droplets call the manager classes.
- The manager classes manipulate the commerce objects and also invoke the commerce pipelines.
- The pipeline processors manipulate the commerce objects as well.
- Commerce objects leverage the order repository to store the order information.

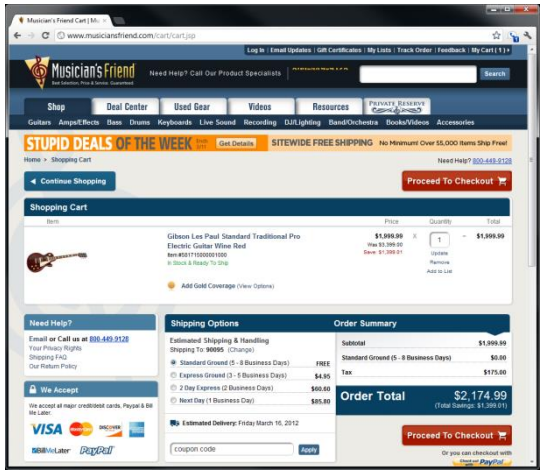
Interactions Between Subsystems (2)



Example of Purchase Process

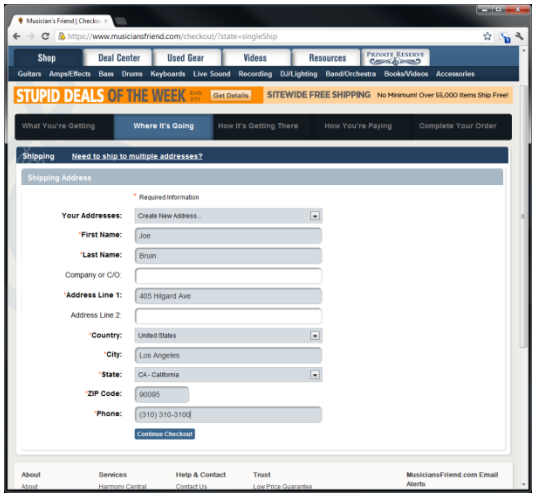
- Tommy Trojan decides to purchase guitar picks from musiciansfriend.com.
- He places guitar picks in the cart.
- He proceeds to specify where the items should be shipped (shipping address).
- He then picks from available shipping method options.
- Tommy specifies the payment method and payment address to pay for the order.
- He then proceeds to the order confirmation page.

Ordering Steps



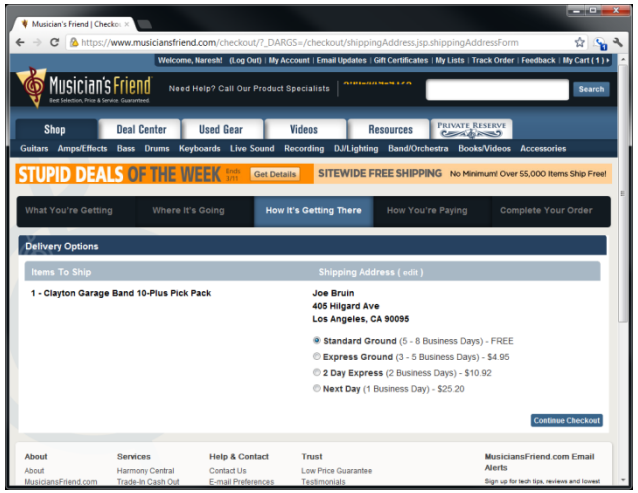
Cart Page

1



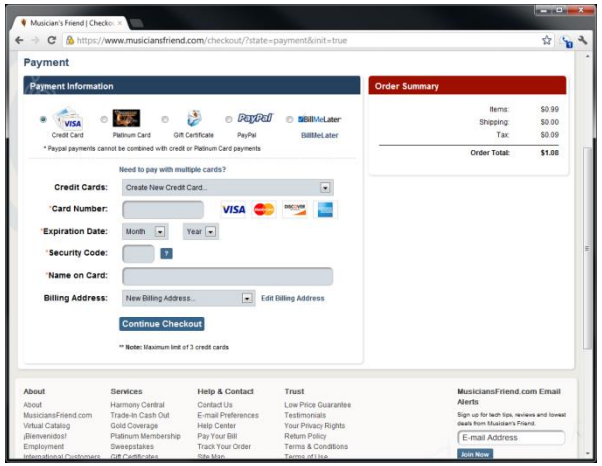
Shipping Address

2



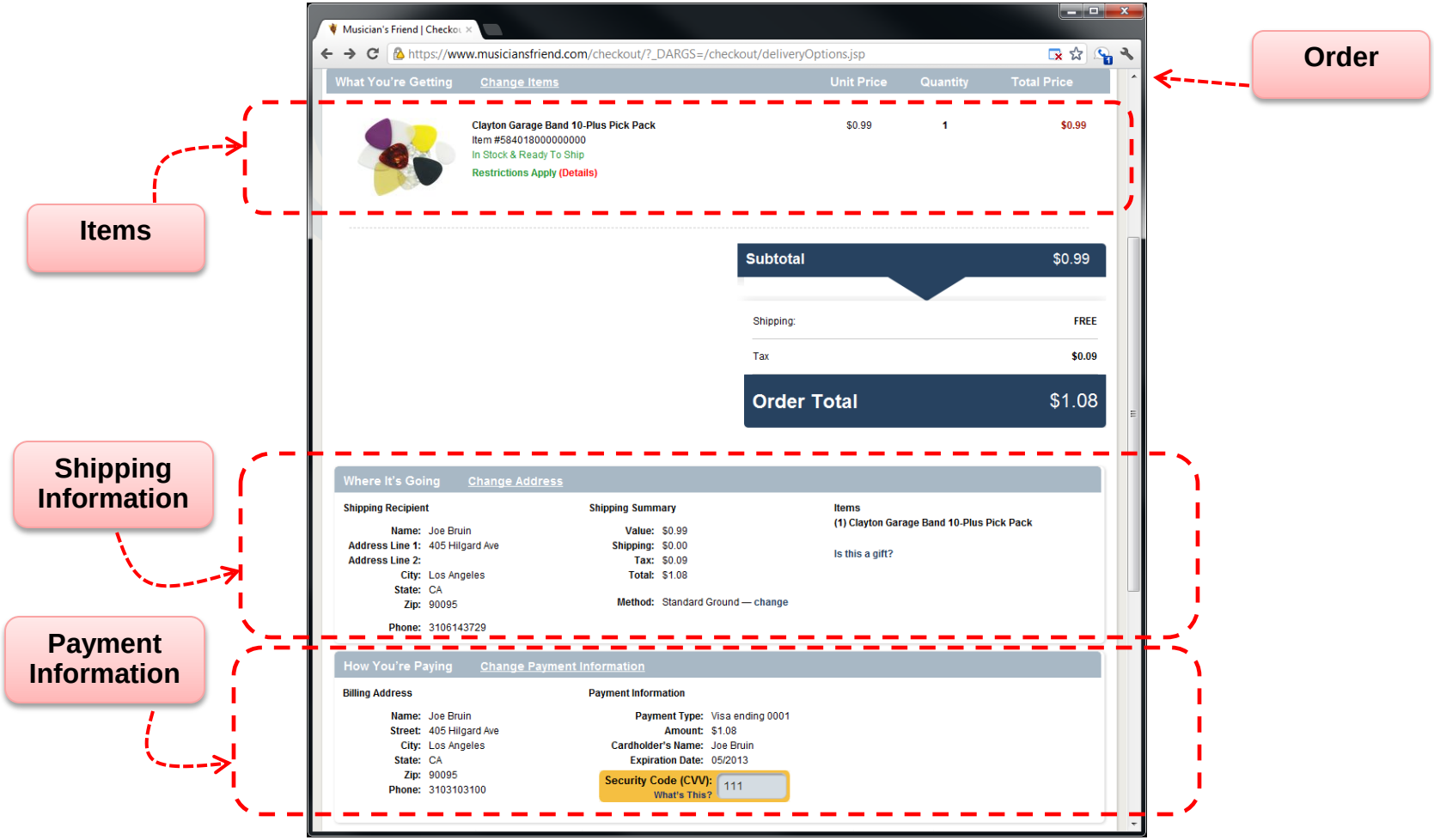
Shipping Method

3



Payment Method

Order Confirmation Page



Purchase Process Subsystem

- During the purchase process that Tommy went through, several form handlers and droplets were invoked including the cart modifier form handler, shipping form handler, etc.
- These form handlers invoke several managers such as OrderManager, ShippingManager, etc.
- These managers worked with the order object.
- They then invoke the pipeline processors such as processOrder chain and validateShippingInfo chain, etc.
- The order object eventually stored the information to the order repository in the order repository item which got written to the dcs_order table in the database.

Section 1
Check Your Understanding

What are key pieces of information captured by an order confirmation page?

Answer:

The items that are sold, the cost, shipping and billing info.

Section 1 ✔
Check Your Understanding

What purchase process subsystems work with the commerce objects?

Answer:

The pipeline components and business layer classes.

Section 1 ✔
Check Your Understanding

Which one purchase process subsystem deals directly with the order repository?

Answer:

The commerce objects.

Section 1
Check Your Understanding

What is the purchase process subsystem that manages user interaction from the JSP?

Answer:

Form handlers and droplets.

Section 1
Check Your Understanding

Name a few form handlers used in the purchase process.

Answer:

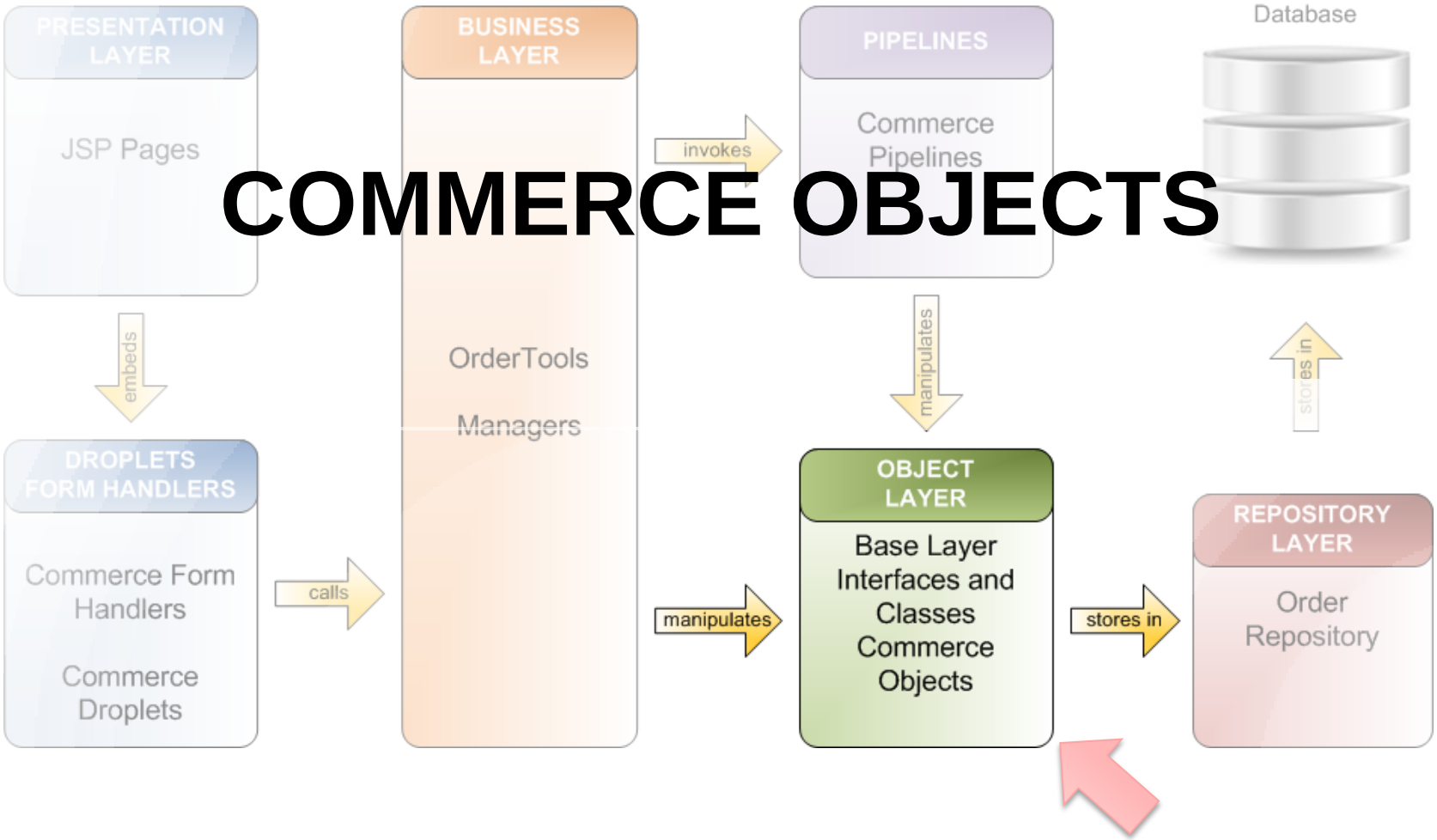
Cart modifier form handler, shipping form handler.

Summary

- The purchase process represents a number of sub systems that facilitate the purchase and checkout of an order by the customer.
- The **commerce form handler**s are embedded on the JSP page and manage user interaction.
- They interact with the **business layer classes**.
- The business layer classes invoke **pipelines**.
- Both the business layer and pipelines manipulate the **commerce objects**.
- The commerce objects store the information using the **order repository,** which gets saved to a database.

Section 2:
Commerce Object Model

COMMERCE OBJECTS



Elements of an Order

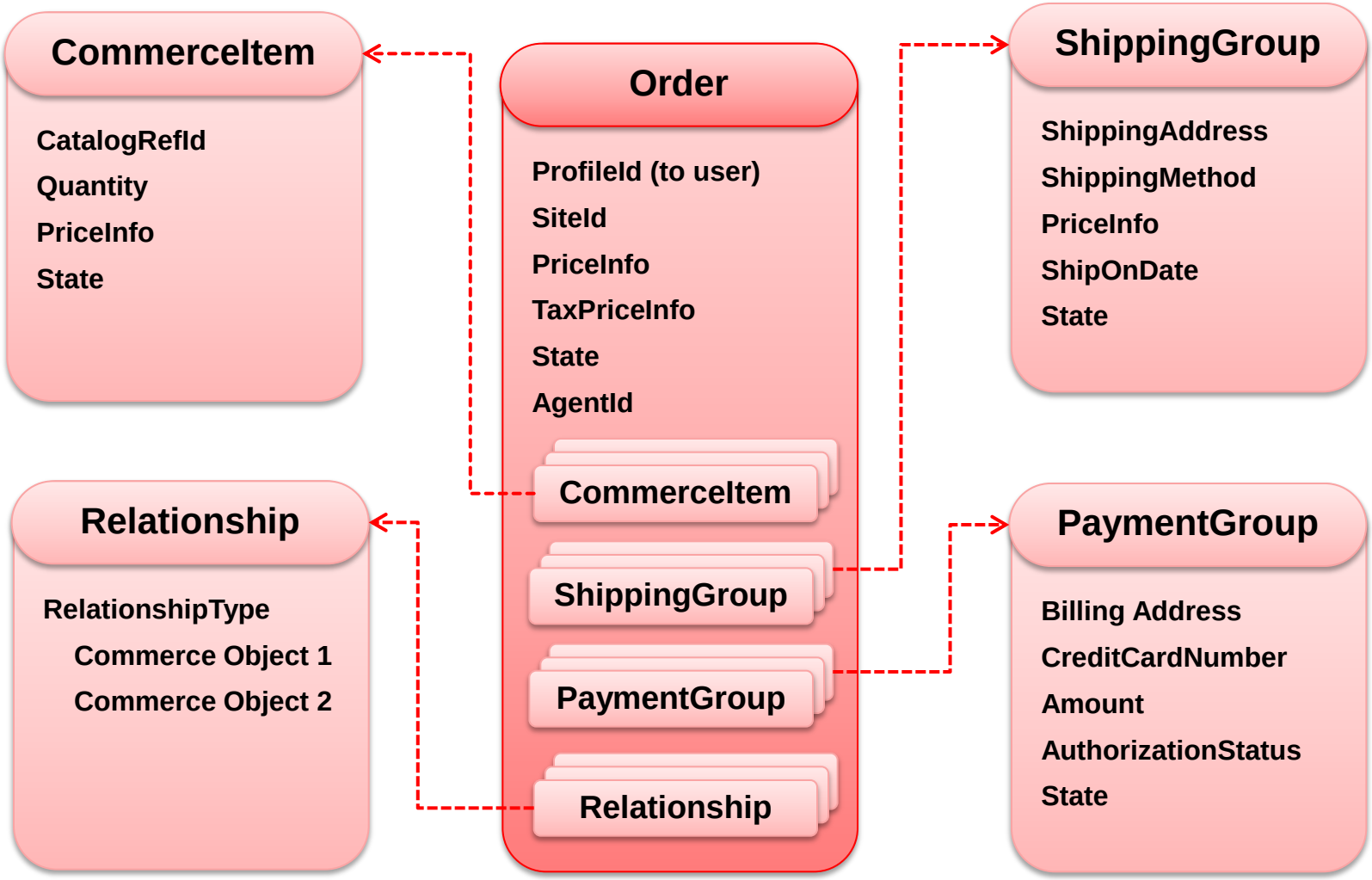
- To accurately capture information contained in a commerce order, an order object should answer the following questions:
  - What are the items being purchased?
  - What is the shipping information such as shipping method, items in that shipment, and shipping address?
  - What is the payment information such as payment method, items that this method pays for?
  - What is the cost of the items, shipping, tax, and entire order?

- The order object needs to contain objects to answer the questions above.

The Order Interface (1)

- The **order** interface is central to the entire purchase process.
- It acts as a container to all the data associated with that order, such as:
  - **CommerceItem** represents the information about a product and SKU that is being purchased.
  - **ShippingGroup** represents the information about the delivery of a collection of CommerceItem objects.
  - **PaymentGroup** represents the payment method and information for items, shipping, tax, and the entire order.
- In addition, it holds associations between the different commerce objects called **relationships**.

The Order Interface (2)



Implementation of Commerce Interfaces

- **OrderImpl** is an implementation of the order interface.
- **CommerceItemImpl** is an implementation of CommerceItem interface.
- **ShippingGroupImpl** implements ShippingGroup. Further extensions to this class add:
  - **HardgoodShippingGroup** adds address, carrier, etc.
  - **ElectronicShippingGroup** adds delivery instructions such as email.
- **PaymentGroupImpl** implements PaymentGroup.
  - **CreditCard** adds credit card info.
  - **GiftCertificate** adds gift certificate info used to pay for the order.
  - **StoreCredit** adds store credit if used to pay for the order.
- Each class has B2B extensions that add B2B features and cost center support.

HandlingInstruction

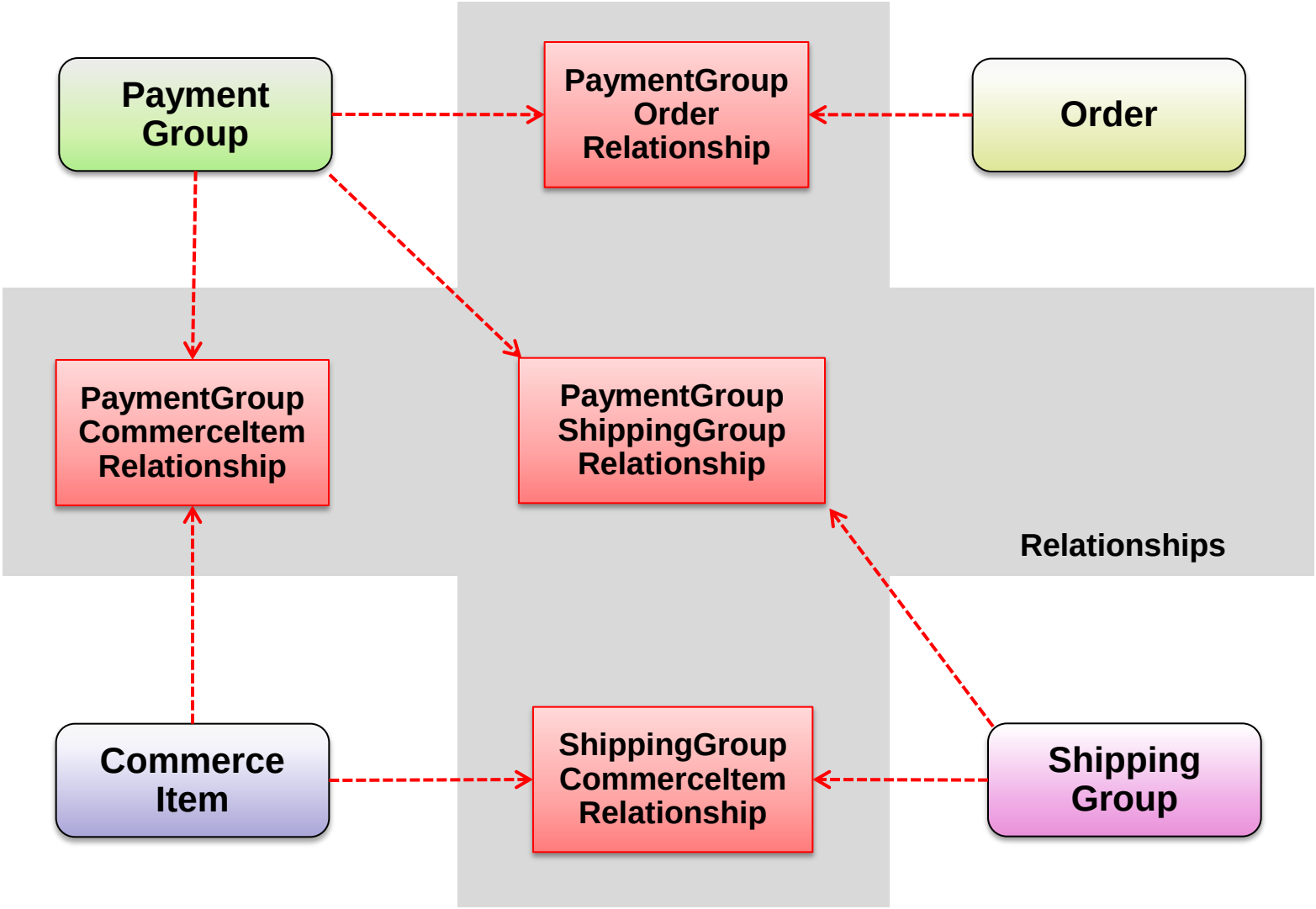
- **HandlingInstruction** interface describes special handling for a CommerceItem within a given ShippingGroup.
- **HandlingInstructionImpl** implements the interface.
- It contains ShippingGroup ID, CommerceItem ID, and quantity to indicate which CommerceItems in the ShippingGroup require special handling.
- **GiftListHandlingInstruction** extends HandlingInstructionImpl. It maintains data about which CommerceItems in the order were added from a gift list.
- HandlingInstruction objects can be found in the ShippingGroup.
- An example of special handling instructions is gift wrapping.

RELATIONSHIPS

Implementation of Relationship

- Relationship objects have associated type which determine who is participating in the relationship and how to interpret it.
- There are four relationship objects:
  - ShippingGroupCommerceItemRelationship,
  - PaymentGroupOrderRelationship,
  - PaymentGroupCommerceItemRelationship,
  - PaymentGroupShippingGroupRelationship.
- Each of the above objects have multiple **types of relationships** associated with them.
- In ATG commerce, developers never add a relationship to an order. It is done implicitly by calling the related business layer classes such as commerce object managers.

Relationship Objects



Relationship Objects with Properties

ShippingGroup CommerceItem Relationship

- Amount
- Quantity
- Range
- RelationshipType
- State
- Commerce Item
- Shipping Group

PaymentGroup CommerceItem Relationship

- Amount
- Quantity
- Range
- RelationshipType
- Commerce Item
- Payment Group

PaymentGroup Order Relationship

- Amount
- RelationshipType
- Order
- Payment Group

PaymentGroup ShippingGroup Relationship

- Amount
- RelationshipType
- Shipping Group
- Payment Group

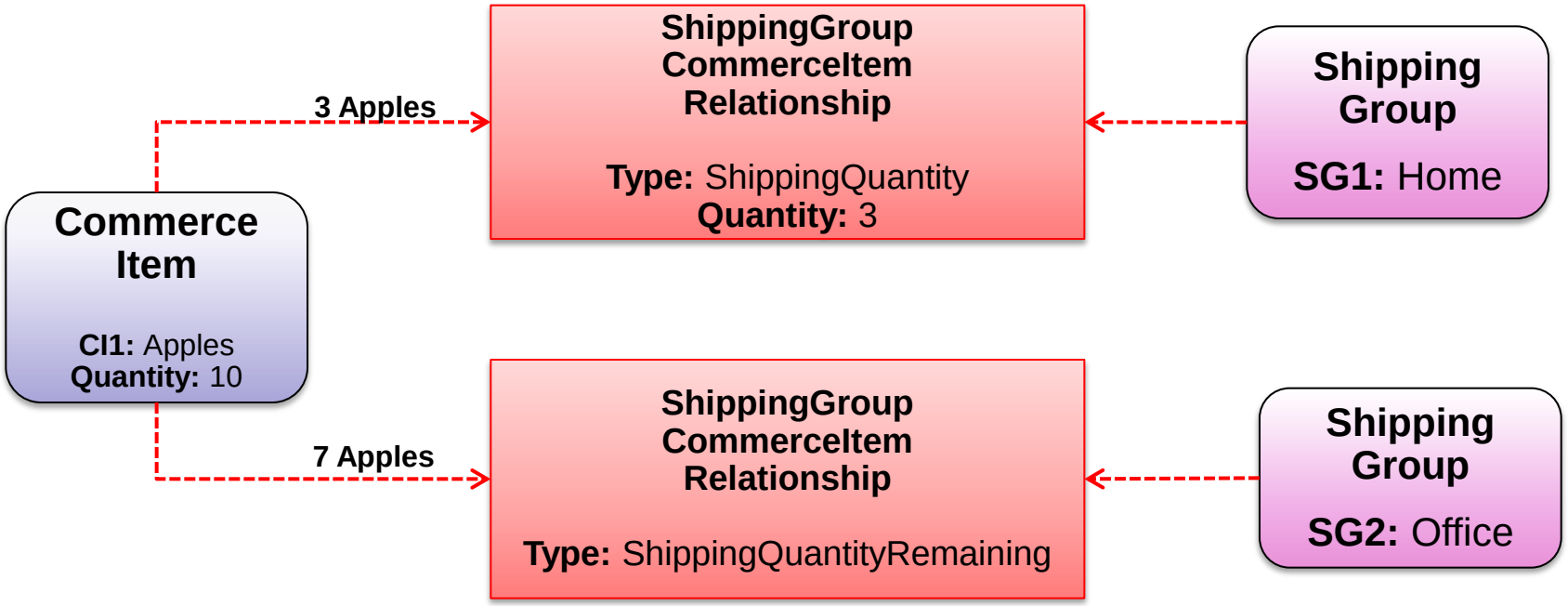
ShippingGroup CommerceItem Relationship

- ShippingGroupCommerceItemRelationship represents the relationship between CommerceItems and ShippingGroups indicating the specific quantity of items that will be shipped using this particular group.
- The two types of this relationship are:
  - **ShippingQuantity** indicates that the specific quantity of the item will be shipped using the information in the ShippingGroup. Quantity to be shipped is stored in the relationship.
  - **ShippingQuantityRemaining** indicates that the remaining quantity of the item unaccounted for by other ShippingGroupCommerceItemRelationship objects will be shipped using information in the attached Shipping Group.
- A commerce item can have only one relationship of type ShippingQuantityRemaining.

Usage of Shipping Relationship (1)

- A customer places an order for CommerceItem apple (CI1) with quantity 10. He wants 3 to ship to home (SG1) and the remaining to be shipped to office (SG2).
- Two relationships would be created between CommerceItem (CI1) and ShippingGroups SG1 and SG2:
  - One of type ShippingQuantity between CI1 and home (SG1) with quantity to ship is 3.
  - Another of type ShippingQuantityRemaining between CI1 and office (SG2).

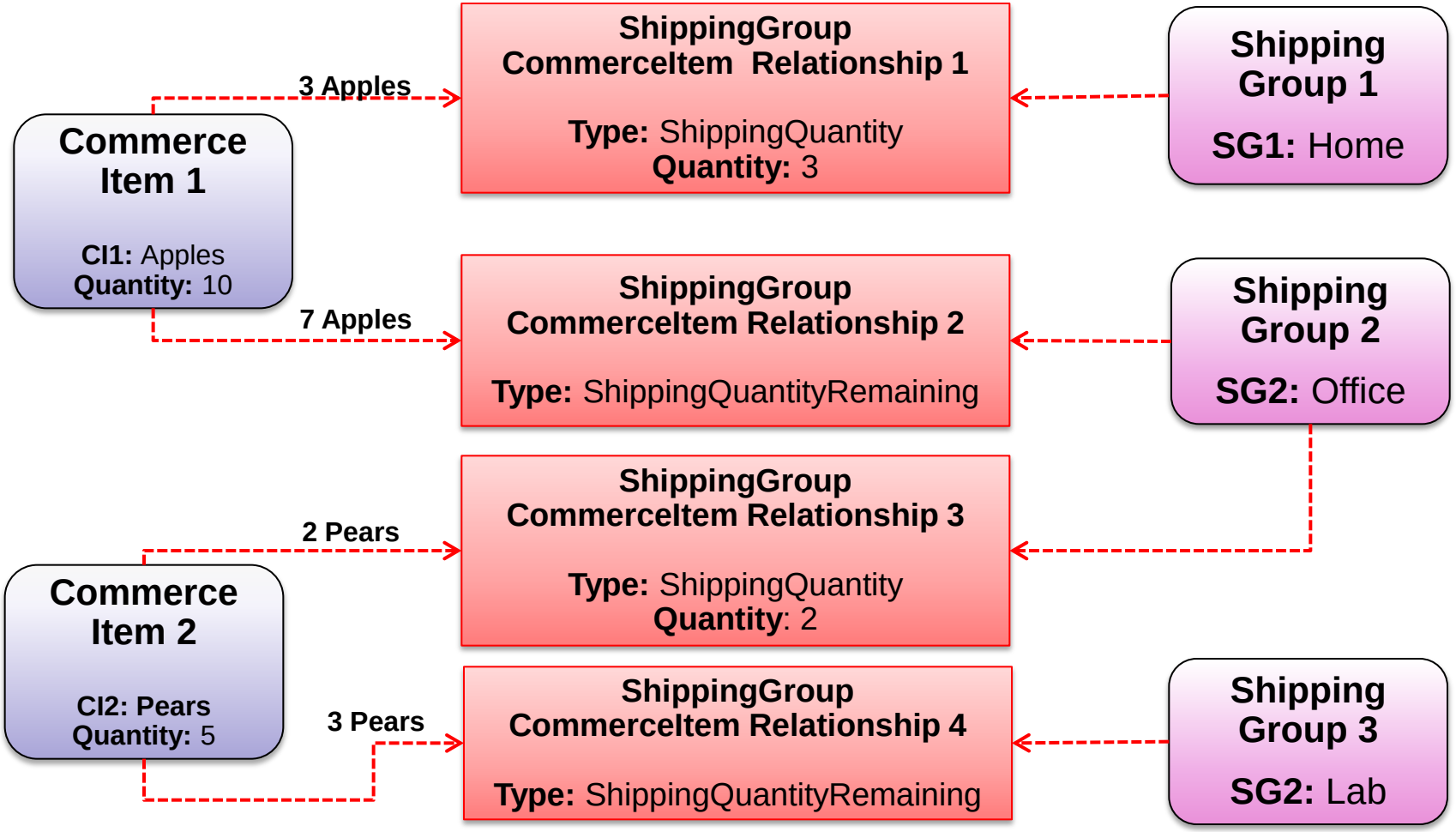
Usage of Shipping Relationship (2)



- If while ordering, the customer increases the quantity, the ShippingQuantityRemaining object type would dynamically adjust to include the additional items.

Usage of Shipping Relationship (3)

- The following complicated relationships are also possible:



PaymentGroupOrderRelationship Object

- Represents a relationship between an order and a payment group indicating how the order will be paid.
- There are two ways to split the cost of the order across payment groups:
  - Split the entire order by using PaymentGroupOrderRelationship.
  - Assign different types of costs (item vs. tax) to separate PaymentGroups.
- Consequently there are four relationship types for PaymentGroupOrderRelationships.

PaymentGroupOrderRelationship Types

- **OrderAmount** relationship type indicates that a specific amount of the total cost of the order will be paid using the information in the PaymentGroup.
- **OrderAmountRemaining** relationship type indicates that the remaining cost of the order unaccounted for by other PaymentGroupOrderRelationship objects will be paid using the information in the PaymentGroup.
- **TaxAmount** relationship type indicates that a specific amount of the tax charged for the order will be paid using the information in the PaymentGroup.
- **TaxAmountRemaining** relationship type indicates that a tax cost unaccounted for by other PaymentGroupRelationship objects will be paid using the information in the referenced payment group.

PaymentGroup CommerceItem Relationship Object

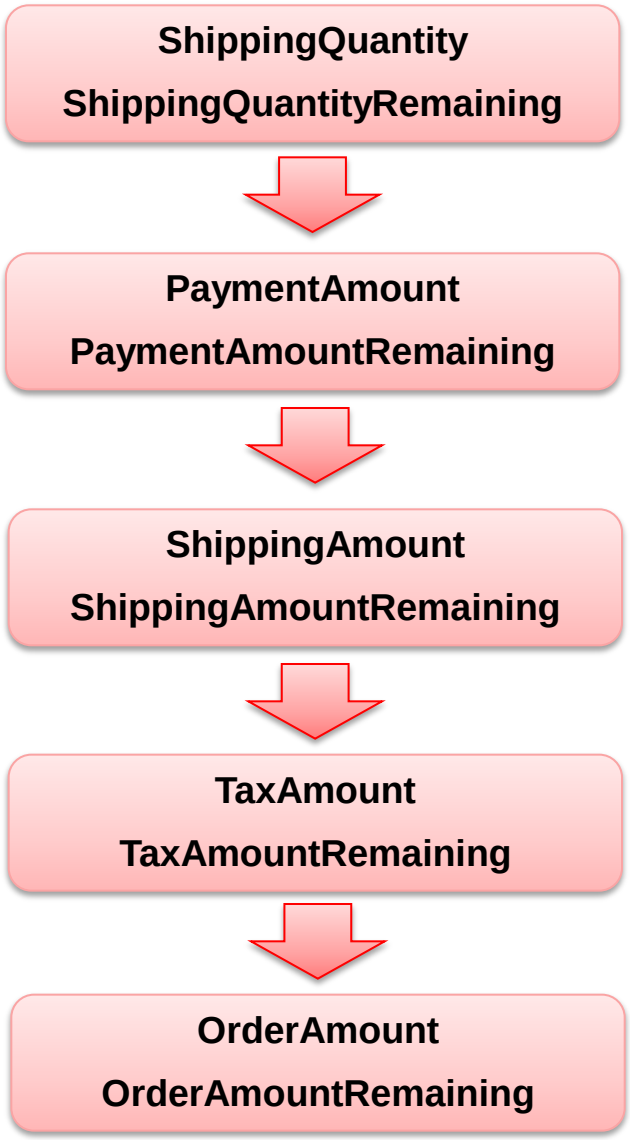
- Represents a relationship between a CommerceItem and a PaymentGroup.
- It is used for split payment for a single CommerceItem between multiple payment groups.
- There are two relationship types:
  - **PaymentAmount** indicates that a specific amount of the item's cost will be paid using the information in the PaymentGroup.
  - **PaymentAmountRemaining** indicates that any remaining payment amount unaccounted for by other PaymentGroupCommerceItemRelationship objects will be paid using the information in the referenced PaymentGroup.
- A CommerceItem can have only one relationship of type PaymentAmoutRemaining.

PaymentGroup ShippingGroup Relationship Object

- Represents a relationship between a ShippingGroup and a PaymentGroup.
- This type of relationship is used to assign shipping costs in a ShippingGroup to PaymentGroups.
- There are two relationship types:
  - **ShippingAmount**: This relationship type indicates that a specific amount of the shipping cost will be paid using the information in the PaymentGroup.
  - **ShippingAmountRemaining**: This relationship type indicates that any remaining shipping cost amount unaccounted for by other PaymentGroupShippingGroupRelationship objects will be paid using the information in the PaymentGroup.

Relationship Priority

- Priority determines the order in which relationships are processed or fulfilled during order processing.
- A relationship's priority is determined by its relationship type.
- If relationship type is the same, the priority is determined by the order in which the relationships were created.



Section 2 ✔
Check Your Understanding

How would gift wrapping be representing in the commerce objects?

Answer:

HandlingInstruction interface.

Section 2
Check Your Understanding

What is the type ShippingQuantityRemaining in the ShippingGroup CommerceItem relationship?

Answer:

ShippingQuantityRemaining indicates that the remaining quantity of the item unaccounted for by other ShippingGroupCommerceItemRelationship objects will be shipped using information in the attached shipping group.

Section 2 ✔
Check Your Understanding

What are the types of PaymentGroupOrderRelationship?

Answer:

Four: OrderAmount, OrderAmountRemaining, TaxAmount, TaxAmountRemaining.

Section 2
Check Your Understanding

What does the PaymentGroup ShippingGroup relationship signify?

Answer:

This type of relationship is used to assign shipping costs in a ShippingGroup to PaymentGroups.

Section 2 ✔
Check Your Understanding

What is the PaymentGroupCommerceItem relationship used for?

Answer:

It is used for split payment for a single CommerceItem between multiple payment groups.

Summary

- The **order** interface is central to the entire purchase process.
- **CommerceItem** represents the information about a product and SKU that is being purchased.
- **ShippingGroup** represents the information about the delivery of a collection of CommerceItem objects.
- **PaymentGroup** represents the payment method and information for items, shipping, tax, and the entire order.
- **Relationship** objects have associated type which determine who is participating in the relationship and how to interpret it.

Q&A

ORACLE IS THE INFORMATION COMPANY

ORACLE®