P08/22-43

Shopping Cart

Analysis & Design

Author:John Smith

Version:1.7
Status:Draft
Publication:23/05/2013

Modeliosoft

21, avenue Victor Hugo, 75016 Paris

Table of Contents

1 Introduction

This project contains the model for the Online Shopping Cart system, including all the models used to specify and realize the IT system.
This model addresses the development of an online shopping cart system, that allows suppliers to sell their products directly online to customers. It is provided as a UML example covering the analysis of the business context and processes, and the design and deployment of a possible implementation. The goal is to show the power UML brings to analysing and designing a robust system that corresponds to the initial requirements.
This model can be used to create demos of the Modelio tool. It is documented in depth, in order to produce pertinent documents using Modelio DocumentPublisher. Some examples of documents are also included in this example, in the "Documents" package.

2 Use Cases

2.1 Actors

Actor	Description
Client	Person purchasing products online
	The client, also known as customer, is the person that logs onto the shopping cart online system to purchase products of his/her choice.
Administrator	Person responsible for the system.
	The administrator is the person in charge of managing and administering the system. He also assumes the role of supervisor in the sense that he enforces the "Terms of use" of the site, and has the right to revoke clients privileges by deleting their account.
Supplier	The person selling his products through the online shopping cart system.
	The supplier is the person or company providing the goods on sale in the online shopping cart system. It is his responsibility to populate the product catalog, and to update any information relative to the products on the site.
Bank	The supplier's bank
	The bank gets involved to debit the client's account and credit the supplier's account during a purchase.
System	Dummy actor representing the system.
	This actor is a dummy actor used to represent the system in interactions diagrams for the use cases.
Netsurfer	
Visitor	

Table 1 Table of Actors

2.2 Use Cases

Use-Cases	Description
Manage Shopping Cart	This use case describes how the user can browse the catalog, view his/her cart, add to or remove items from the cart, edit the quantities or empty his/her shopping cart. All scenarios result in the persistance of the cart items, so the user can retrieve his/her cart between sessions.
Check out	This use case describes how the client finalizes his/her purchases by checking out from the shopping cart. Payment is processed in this use case.
File Complaint	This use case describes how the client can file a complaint if the goods received do not correspond to the order or are damaged.
Search For Products	Aucune
Consult Orders	
Manage Client Account	This use case describes the operations the user can access to create, modify or detele his/her account. Some of these operations are also accessible by the system administrator whose role it is to enforce the "Terms of Use" of the online system.
	This use case describes the operations the user can access to create, modify or detele his/her account. Some of these operations are also

Use-Cases	Description
	accessible by the system administrator whose role it is to enforce the "Terms of Use" of the online system.
Create A Client Account	
Manage Product Catalog	This use case is used by the supplier to manage products in the catalog. The supplier can add, remove or modify products in the catalog. The system will insure that the orders placed before the actual publication of the updated catalog are honoured at the old conditions.
Recall Product	The supplier can place a recall on defective products, soas to encourage people to return their goods for replacement or refund.
Advertise Product	This use case describes how a product is advertized by placing a announcement on the first page of the online shopping cart system.
Ship Order	This use case describes how the supplier sends the goods ordered to the client.
Handle Complaint	This use case describes how the supplier handles a complaint received by a client concerning the goods delivered to him.

Table 2 Table of Use Cases

2.2.1 Use Case "Actors"

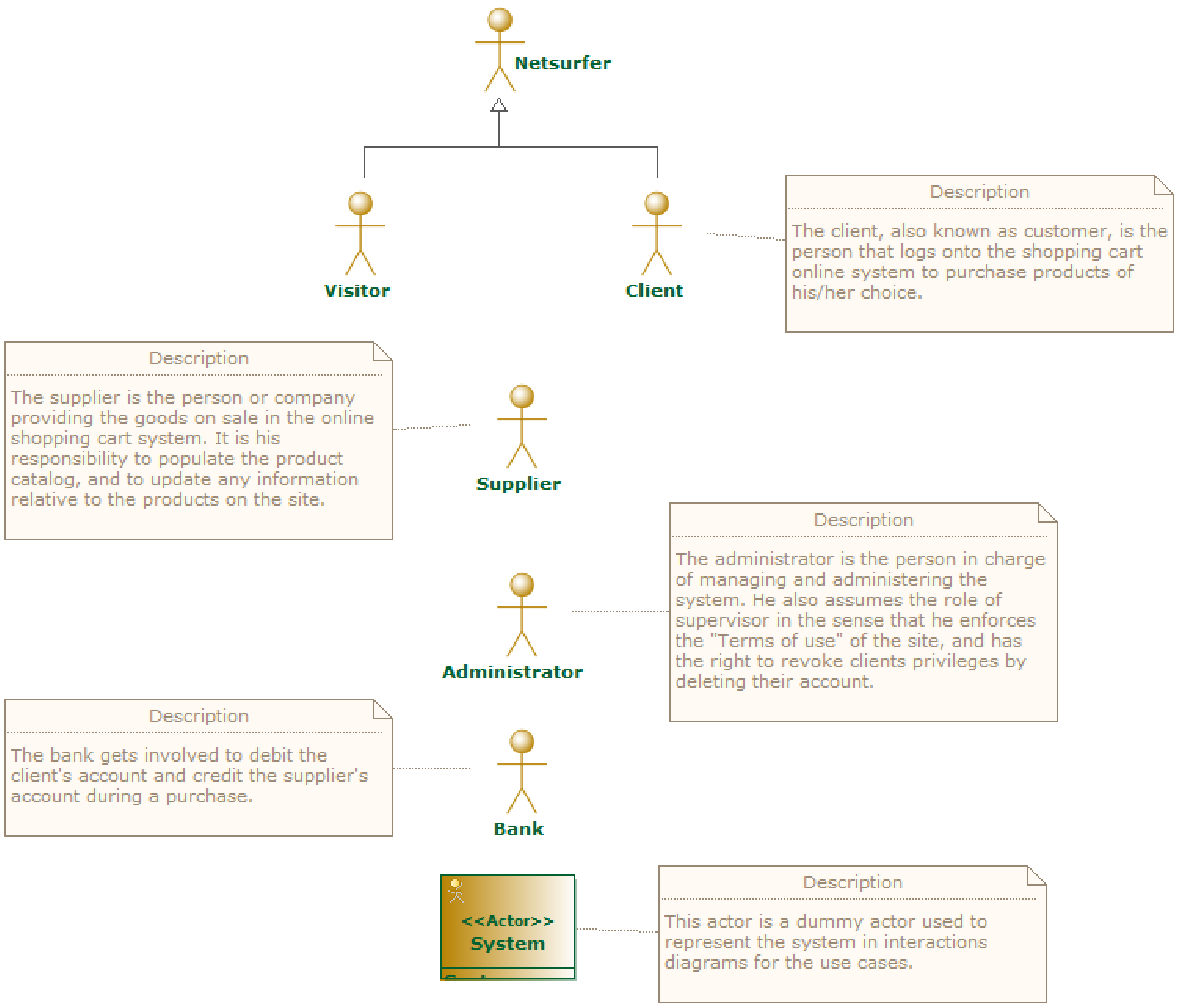
Netsurfer
Visitor
Client
Description
The client, also known as customer, is the person that logs onto the shopping cart online system to purchase products of his/her choice.
Description
The supplier is the person or company providing the goods on sale in the online shopping cart system. It is his responsibility to populate the product catalog, and to update any information relative to the products on the site.
Supplier
Administrator
Description
The administrator is the person in charge of managing and administering the system. He also assumes the role of supervisor in the sense that he enforces the "Terms of use" of the site, and has the right to revoke clients privileges by deleting their account.
Description
The bank gets involved to debit the client's account and credit the supplier's account during a purchase.
Bank
<<Actor>>
System
Description
This actor is a dummy actor used to represent the system in interactions diagrams for the use cases.

Figure 1 : Actors

2.2.2 Use Case "Online Shopping usecase diagram"

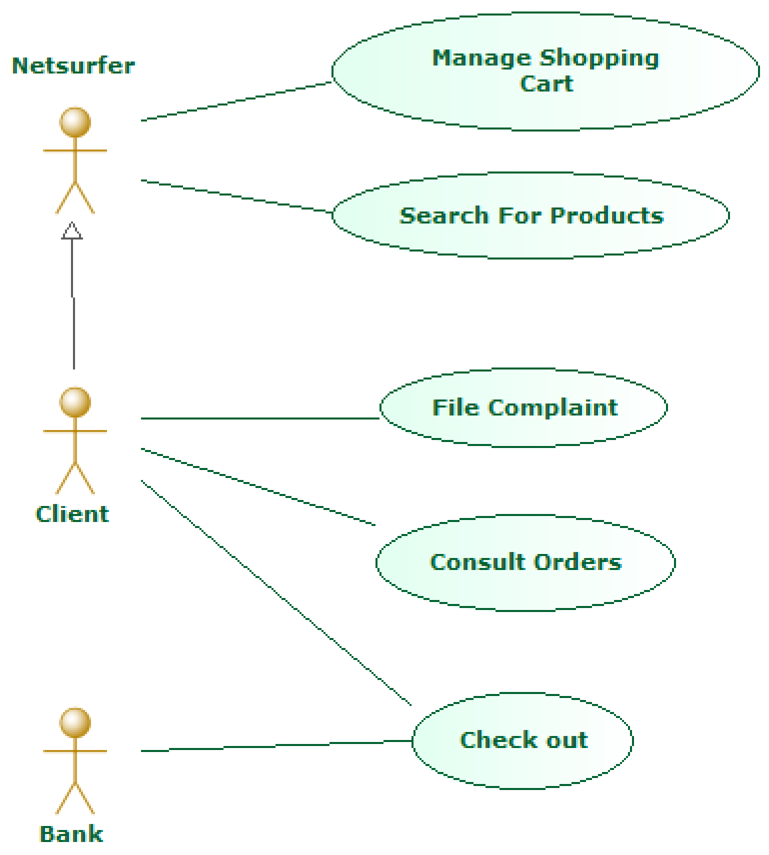

Figure 2 : Online Shopping usecase diagram

2.2.3 Use Case "Account Management usecase diagram"

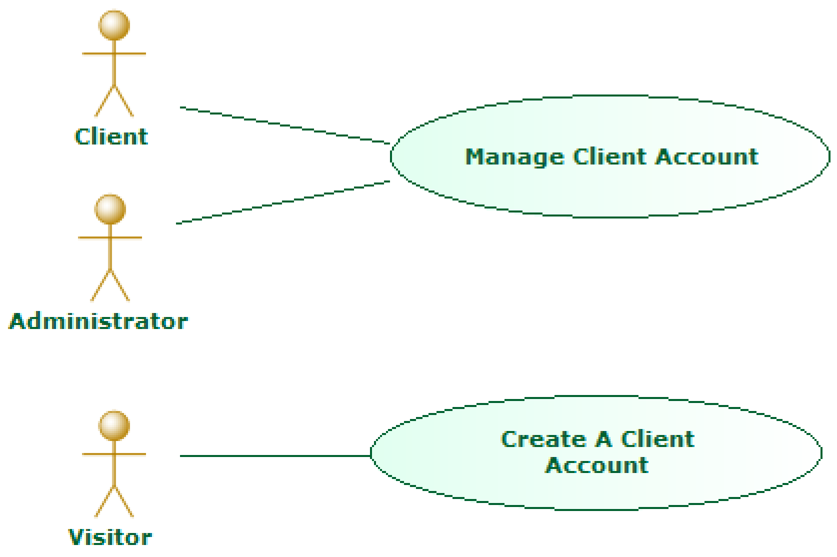

Figure 3 : Account Management usecase diagram

2.2.4 Use Case "Catalog Management usecase diagram"

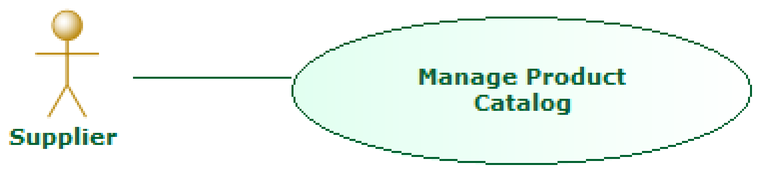

Figure 4 : Catalog Management usecase diagram

2.2.5 Use Case "Product Management usecase diagram"

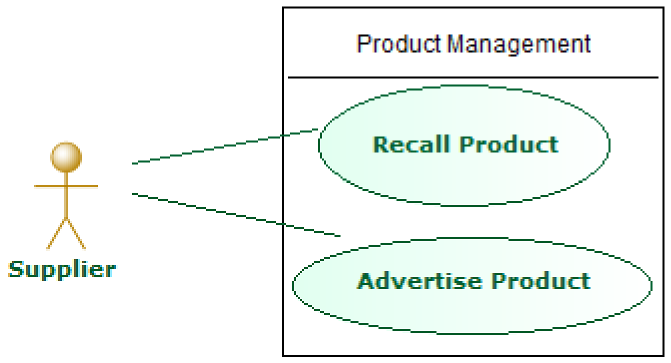

Figure 5 : Product Management usecase diagram

2.2.6 Use Case "Order Management usecase diagram"

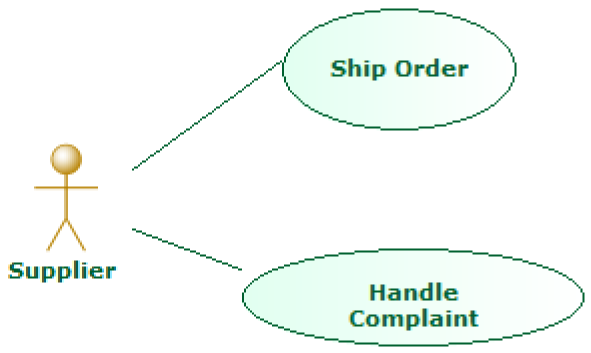

Figure 6 : Order Management usecase diagram

3 Package Index

ShoppingCart

Analysis	Analysis models for the Online Shopping Cart system
Actors	Persons or entities external to the system interacting with the system.
Processes	Main processes describing the online shopping system core business.
Use Cases	Use case for interacting with the shopping cart system.
Online Shopping	Use cases related to the client's shopping activities
Account Management	Use cases related to the management of accounts: creation, deletion...
Catalog Management	Use cases related to the management of the product catalog.
Product Management	Use cases related to the management of products for sale online.
Order Management	Supplier order management use cases.
Domains	
onlineStore	This package contains all the classes directly related to the online store.
client	This package contains all the classes related to the client.
products	This package contains the classes related to the products.
bank	In this package, the classes related to the bank are located.
types	Non business-related base types.
Design	This package contains the models explaining the way the system is designed.
Logical Architecture	The Online Shopping Cart logical architecture.
Presentation	User interface components layer.
Business	Business layer and its components
Exchange Types	Exchange types included in the component service contract.
Implementation	Internal part of the component used to implement its functionalities.
Entity	Persistence layer components.
Deployment	Deployment model for the Online Shopping Cart System.

4 Package "ShoppingCart"

This project contains the model for the Online Shopping Cart system, including all the models used to specify and realize the IT system.
This model addresses the development of an online shopping cart system, that allows suppliers to sell their products directly online to customers. It is provided as a UML example covering the analysis of the business context and processes, and the design and deployment of a possible implementation.
The goal is to show the power UML brings to analysing and designing a robust system that corresponds to the initial requirements.
This model can be used to create demos of the Modelio tool. It is documented in depth, in order to produce pertinent documents using Modelio DocumentPublisher. Some examples of documents are also included in this example, in the "Documents" package.

Name	Summary
Analysis	Analysis models for the Online Shopping Cart system
Design	This package contains the models explaining the way the system is designed.
Deployment	Deployment model for the Online Shopping Cart System.

Table 3 Owned Packages of Package "ShoppingCart"

Name	Summary
AnalystProperties	

Table 4 Owned Classes of Package "ShoppingCart"

4.1 Class "AnalystProperties"

from Package ShoppingCart

Stereotypes: Model Component

5 Package "Analysis"

from Package ShoppingCart

This package contains the models used to analyse the business and its context in order to properly scope what the system must do.

Name	Summary
Actors	Persons or entities external to the system interacting with the system.
Processes	Main processes describing the online shopping system core business.
Use Cases	Use case for interacting with the shopping cart system.
Domains	

Table 5 Owned Packages of Package "Analysis"

6 Package "Actors"

from Package ShoppingCart.Analysis

This package contains all the persons, systems or protocols that interact with the system during its life.

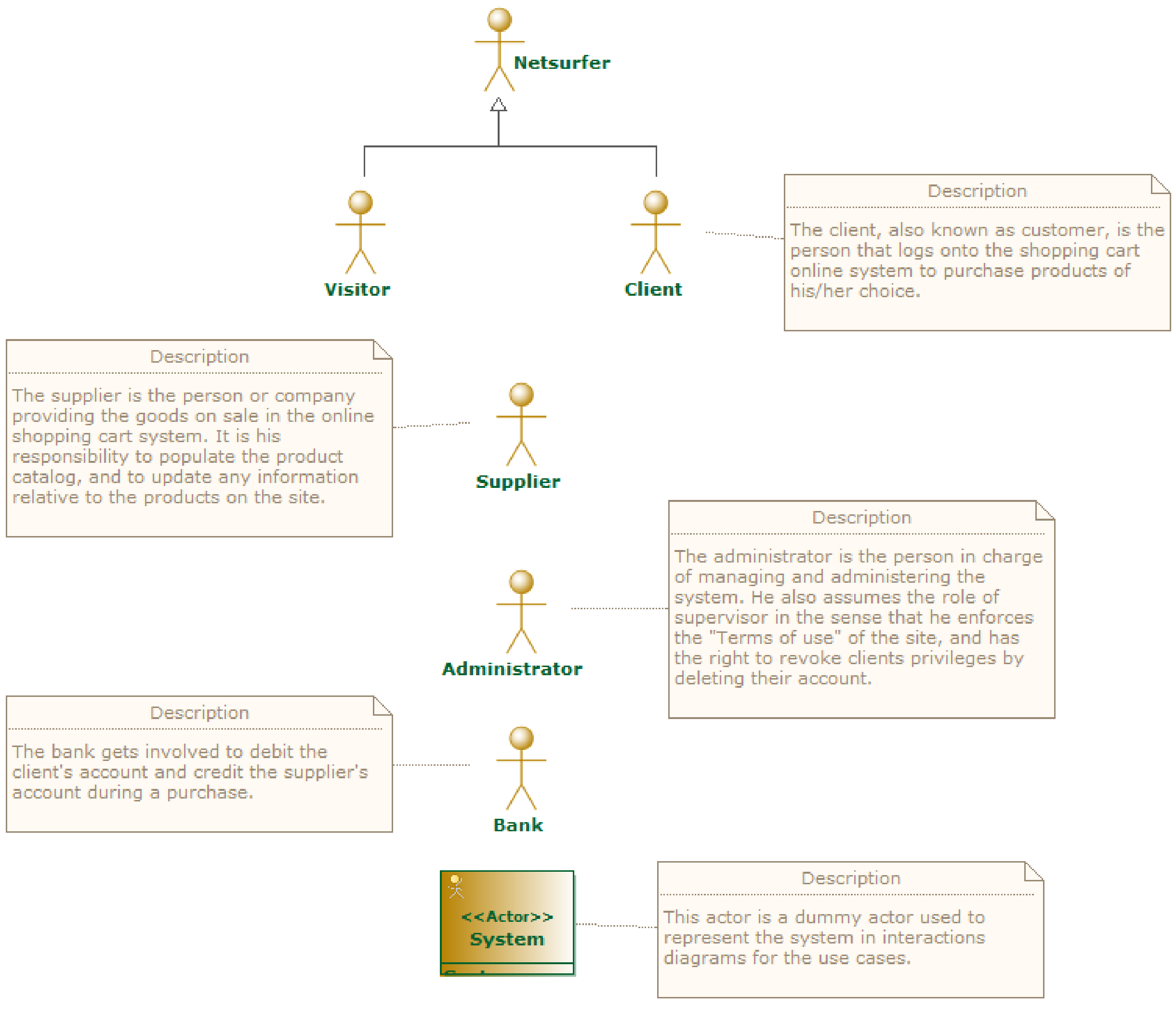
Netsurfer
Visitor
Client
Description
The client, also known as customer, is the person that logs onto the shopping cart online system to purchase products of his/her choice.
Description
The supplier is the person or company providing the goods on sale in the online shopping cart system. It is his responsibility to populate the product catalog, and to update any information relative to the products on the site.
Supplier
Description
The administrator is the person in charge of managing and administering the system. He also assumes the role of supervisor in the sense that he enforces the "Terms of use" of the site, and has the right to revoke clients privileges by deleting their account.
Administrator
Description
The bank gets involved to debit the client's account and credit the supplier's account during a purchase.
Bank
<<Actor>>
System
Description
This actor is a dummy actor used to represent the system in interactions diagrams for the use cases.

Figure 7 Actors

7 Package "Processes"

from Package ShoppingCart.Analysis

This package contains the analysis of the three main business processes the online shopping system must implement. They are described using UML activity diagrams.

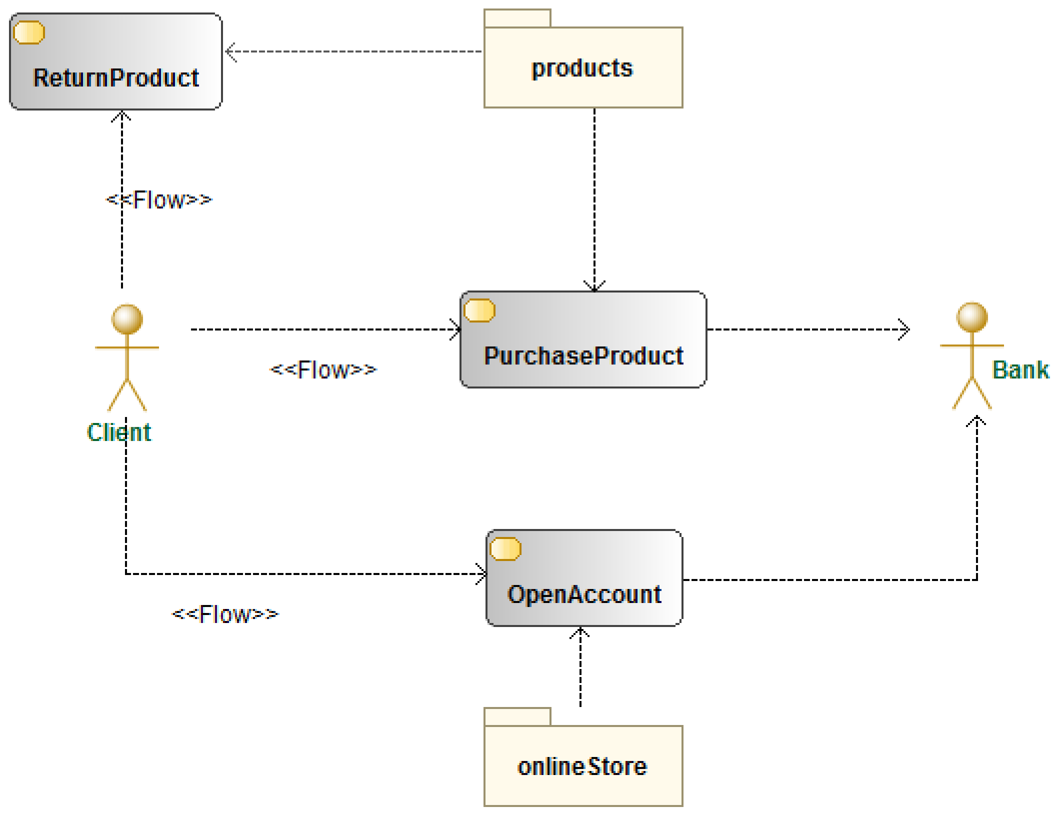

Figure 8 Processes Overview

8 Package "Use Cases"

from Package ShoppingCart.Analysis

This package contains a description of all the use cases that can be played by external actors to use the system.

Name	Summary
Online Shopping	Use cases related to the client's shopping activities
Account Management	Use cases related to the management of accounts: creation, deletion...
Catalog Management	Use cases related to the management of the product catalog.
Product Management	Use cases related to the management of products for sale online.
Order Management	Supplier order management use cases.

Table 6 Owned Packages of Package "Use Cases"

9 Package "Online Shopping"

from Package ShoppingCart.Analysis.Use Cases

Here we find the use cases related to the shopping activities. These are the main interactions between the system and the client during buying sessions.

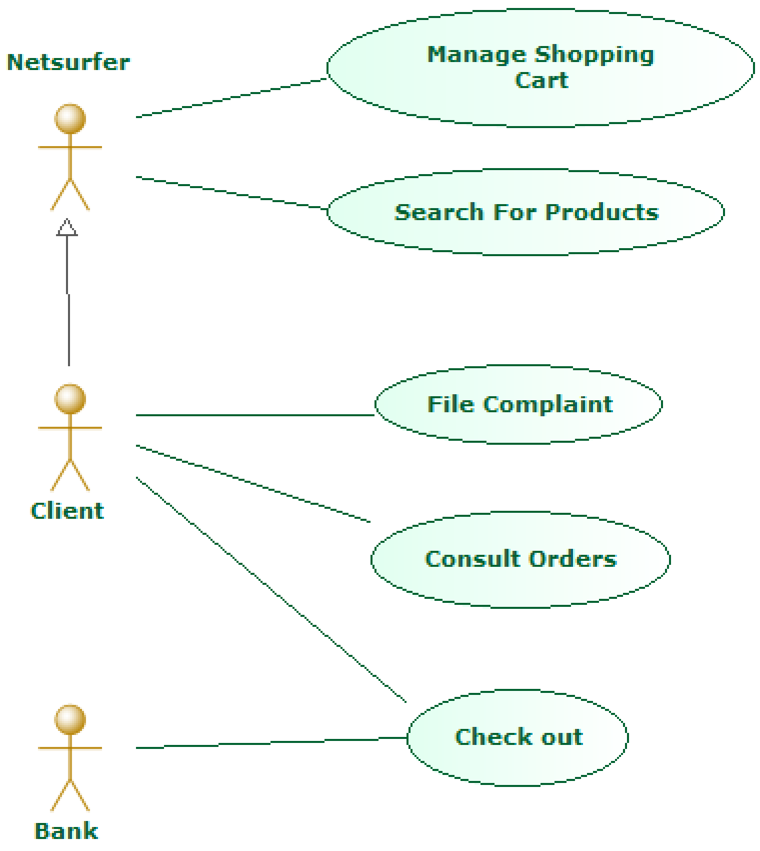

Figure 9 Online Shopping usecase diagram

10Package "Account Management"

from Package ShoppingCart.Analysis.Use Cases

This package contains the use cases related to the management of client's accounts. All the interactions between the system and the client that relate to accounts are described here.

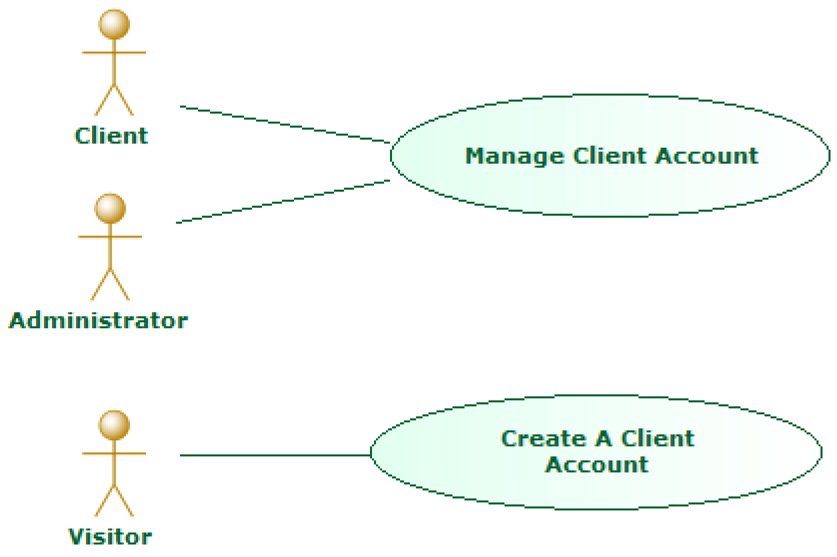

Figure 10 Account Management usecase diagram

11Package "Catalog Management"

from Package ShoppingCart.Analysis.Use Cases

The use cases contained in this package describe the interactions between users and system to manage the product catalog.

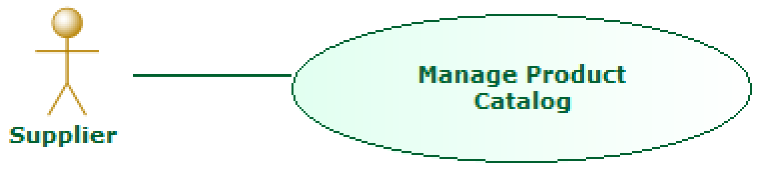

Figure 11 Catalog Management usecase diagram

12 Package "Product Management"

from Package ShoppingCart.Analysis.Use Cases

The use cases contained in this package describe how suppliers interact with the system to update product descriptions or decide on advertizing or product recalls.

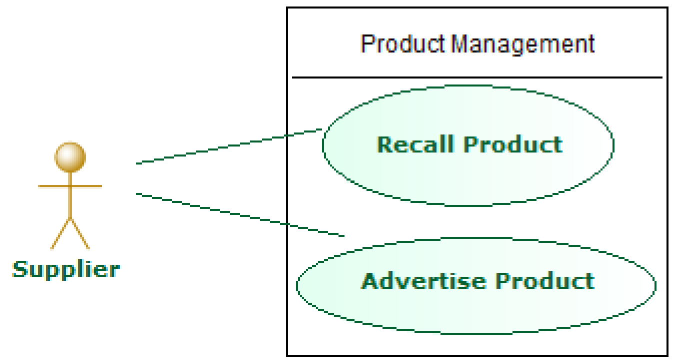

Figure 12 Product Management usecase diagram

13Package "Order Management"

from Package ShoppingCart.Analysis.Use Cases

This package contains the use cases related to the management of the orders. Complete descriptions of the expected steps to carry out in order to manage orders are provided.

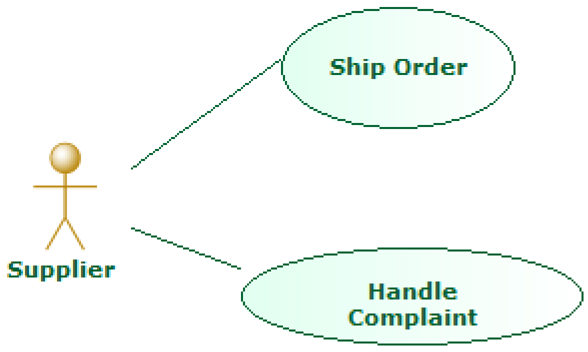

Figure 13 Order Management usecase diagram

14Package "Domains"

from Package ShoppingCart.Analysis

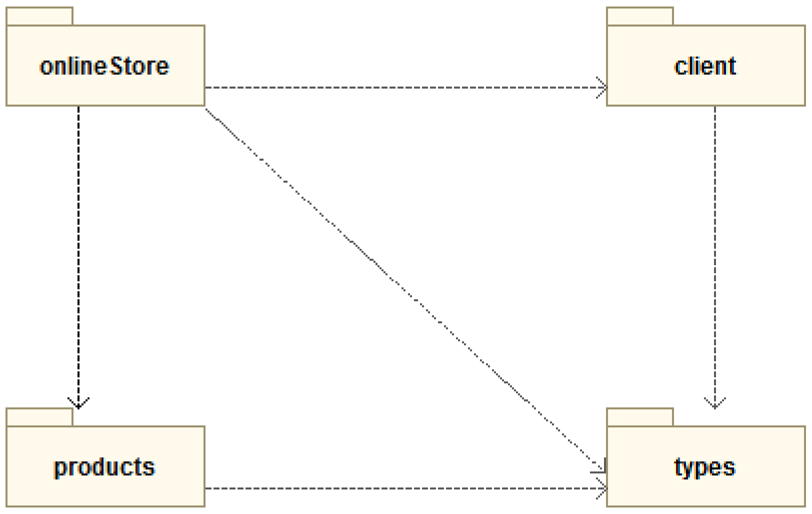

Figure 14 Domains class diagram

Name	Summary
onlineStore	This package contains all the classes directly related to the online store.
client	This package contains all the classes related to the client.
products	This package contains the classes related to the products.
bank	In this package, the classes related to the bank are located.
types	Non business-related base types.

Table 7 Owned Packages of Package "Domains"

15Package "onlineStore"

from Package ShoppingCart.Analysis.Domains

This package contains all the classes related to the online store and the shopping cart. These are the main classes of the online shopping system, supporting the product cart.

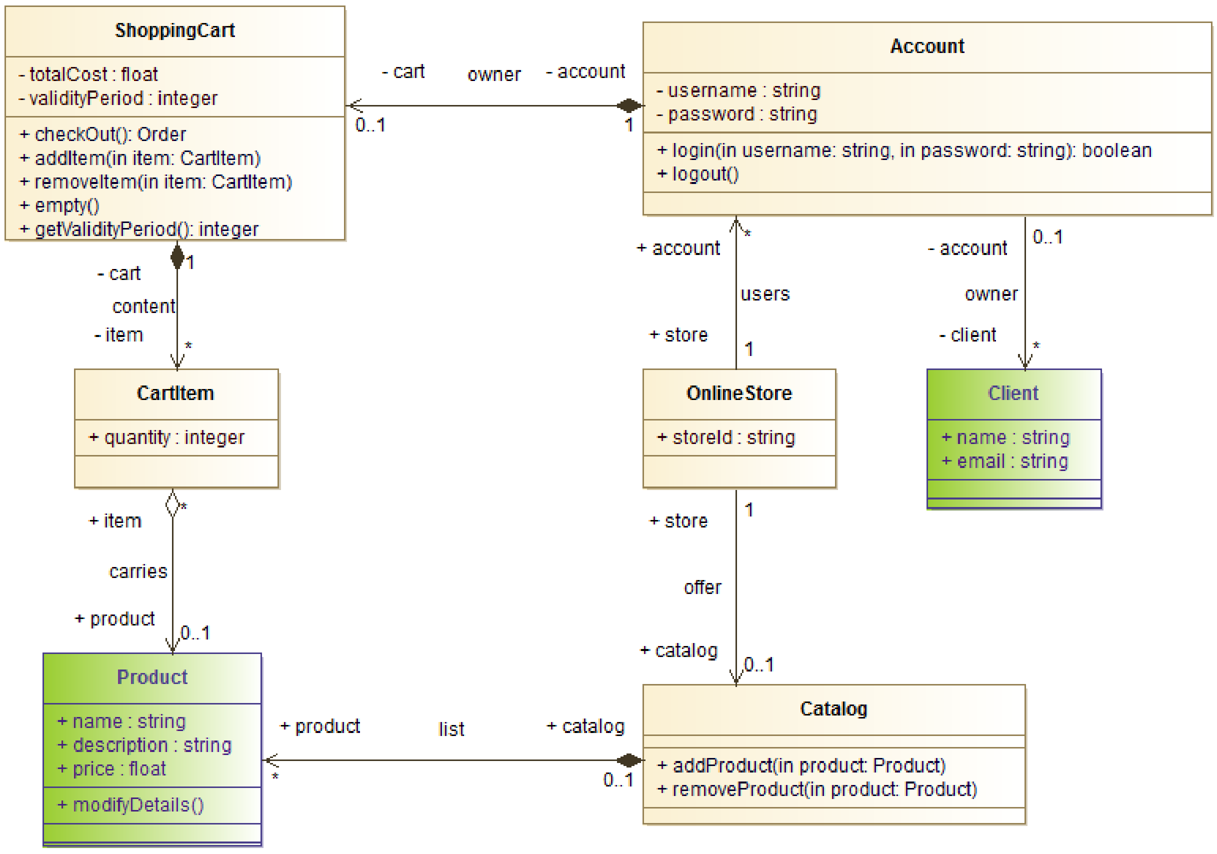

Figure 15 OnlineStore

Name	Summary
Account	The account held by the customer.
ShoppingCart	The container in which customers can add product for purchase.
CartItem	One entry in the shopping cart, containing a reference to a product and a quantity
Catalog	The reference listing containing all the products on offer for sale.
OnlineStore	The main class, representing the online store.

Table 8 Owned Classes of Package "onlineStore"

15.1 Class "Account"

from Package ShoppingCart.Analysis.Domains.onlineStore

The class Account represents the account held by a customer. it allows a customer to login to the system to carry out purchases. In addition to a reference to a Client, it contains the credentials for the customer.

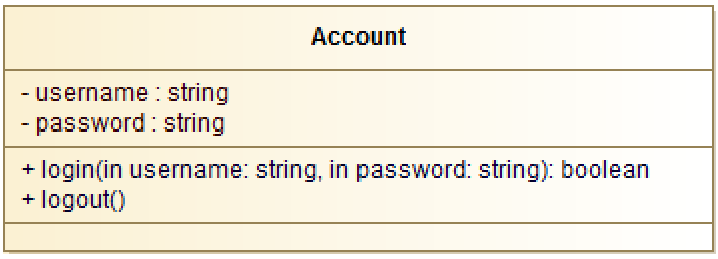

Figure 16 Account

State Machine "Account States"

This staqte machine describes the different states an account can be in. It is important to note that an account needs to go through a certain number of states before being activated.

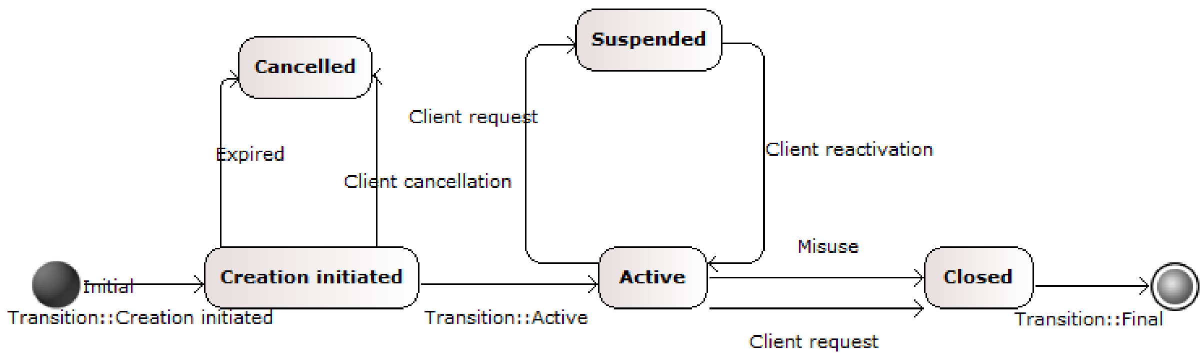

Figure 17 Account state diagram

Name	Description
boolean login (IN **username string,**IN **password string)**	Method invoked by the system in response to the client hitting the login button. this emthod verifies the credentials provided by the client at login time. A boolean return value indicates success or failure.
logout ()	This method is called in response to the client pressing the logout button. It makes sure the session is terminated, preventing any user from accessing the reserved pages.

Table 9 Operations of Class "Account"

Name	Description
username : [1..1] string	This is the unique identifier for each customer. The username is used during all exchanges with the customer via the online shopping cart site, to provide the users with anonymity.
password : [1..1] string	The password the customer must provide in order to authenticate one self on the shopping cart site. The password is encrytped by the system for security and confidentiality.

Table 10 Attributes of Class "Account"

Name	Description
users->store : [1..1] OnlineStore	This association shows that the OnlineStore class is in relation with 0 or many accounts, the accounts of the users currently active in the system.
owner->client : [0..*]	Each client has one and only one account. A client cannot exist if he does

Name	Description
Client	not have an account. An account cannot exist if it is not attached to a client.
owner->cart : [0..1] ShoppingCart	An account is associated to a single shopping cart. It is composition to clearly state that a shopping cart cannot exist without the associated account.

Table 11 Associations of Class "Account"

15.2 Class "ShoppingCart"

from Package ShoppingCart.Analysis.Domains.onlineStore

This is the container holding the products the customer wants to purchase. It persists its content to permanent storage until the customer proceeds to the check out. This means that items added to the shopping cart can be retrieved between sessions by the customer. Each customer has one cart that is restored every time the customer logs in.

Name	Description
Order checkOut ()	This method is called when the client decides to purchase the content of his/her shopping cart.
addItem (IN **item CartItem)**	This method is called when the client wants to add an item to his/her shopping cart.
removeItem (IN **item CartItem)**	This method is used to remove a specific item from the shopping cart. It is necessary to provide the id of the item to be removed.
empty ()	Instead of calling removeItems for all items in the cart one by one, this method enables the system to remove all items at once.
integer getValidityPeriod ()	
float getTotalCost ()	

Table 12 Operations of Class "ShoppingCart"

Name	Description
validityPeriod : [1..1] integer	The validity period is expressed in number of days for which the shopping cart can contain items without a check out operation.
totalCost : [1..1] float	This is the total cost for the shopping cart at a given time. Although this field is calculated, it is directly persisted on the ShoppingCart class to avoid calculation delays during queries.

Table 13 Attributes of Class "ShoppingCart"

Name	Description
owner->account : [1..1] Account	An account is associated to a single shopping cart. It is composition to clearly state that a shopping cart cannot exist without the associated account.
content->item : [0..*] CartItem	This association illustrates the relation between cart items and carts. Carts contain items. This relation is a composition to express the idea that items cannot exist without a cart, at anytime.

Table 14 Associations of Class "ShoppingCart"

15.3 Class "CartItem"

from Package ShoppingCart.Analysis.Domains.onlineStore

A cart item is an entry in the shopping cart describing the product desired and the quantity required. In addition to a reference to a product, the cart item manages the calculation for the total price of the item. This is usually equal to the product unit price multiplied by the quantity.

Name	Description
getTotalCost ()	Since the value returned by this method is calculated on the basis of the product unit cost and the quantity, it is not persisted in a class variable. This method interrogates the Product class to obtain the unit cost.

Table 15 Operations of Class "CartItem"

Name	Description
quantity : [1..1] integer	The quantity of products in this item. It can also be that the cart contains several items referencing the same product. But for large quantities, it is useful to be able to specify it in the CartItem class.

Table 16 Attributes of Class "CartItem"

Name	Description
content->cart : [1..1] ShoppingCart	This association illustrates the relation between cart items and carts. Carts contain items. This relation is a composition to express the idea that items cannot exist without a cart, at anytime.
carries->product : [0..1] Product	Each CartItem contains a reference to the product associated with the item. This association expresses this relation, in order to be able to reference the product from the item.

Table 17 Associations of Class "CartItem"

15.4 Class "Catalog"

from Package ShoppingCart.Analysis.Domains.onlineStore

The catalog is the reference containing all the products on sale online. It contains a complete description of each product, so the client can be fully informed about the product before purchasing it. The catalog organizes products in categories to ease browsing for the customer.

Name	Description
addProduct (IN **product Product)**	This method is used to add new products to the catalog. It is called as a result of a supplier publishing a new product for sale. It takes the new product to add as argument.
removeProduct (IN **product Product)**	This method enables a supplier or administrator to remove a product from the catalog. The argument is the product to remove.

Table 18 Operations of Class "Catalog"

Name	Description
offer->store : [1..1] OnlineStore	This association has been added to allow the OnlineStore class to manipulate catalogs as required. An online store can have several catalogs, and for example decide to publish one at a given time, and a different one at a later stage, hence the "many" cardinality.
list->product :	This relation illustrates the fact that catalogs have products in them. This

Name	Description
[0..*] Product	association enables the catalog class to maintain a reference on products listed in the catalog, in order to interrogate the Product class for product descriptions and details.

Table 19 Associations of Class "Catalog"

15.5 Class "OnlineStore"

from Package ShoppingCart.Analysis.Domains.onlineStore

This class is an abstraction for the online store. It allows for all the other classes to be connected together in a structured way, and provides the glue that permit other entities to be combined.

Name	Description
storeId : [1..1] string	This varibale uniquely identifies the OnlineStore instance. This might become useful for a system that would support distinct stores on the same site for example.

Table 20 Attributes of Class "OnlineStore"

Name	Description
offer->catalog : [0..1] Catalog	This association has been added to allow the OnlineStore class to manipulate catalogs as required. An online store can have several catalogs, and for example decide to publish one at a given time, and a different one at a later stage, hence the "many" cardinality.
users->account : [0..*] Account	This association shows that the OnlineStore class is in relation with 0 or many accounts, the accounts of the users currently active in the system.

Table 21 Associations of Class "OnlineStore"

16Package "client"

from Package ShoppingCart.Analysis.Domains

This package contains all the classes related to the client.

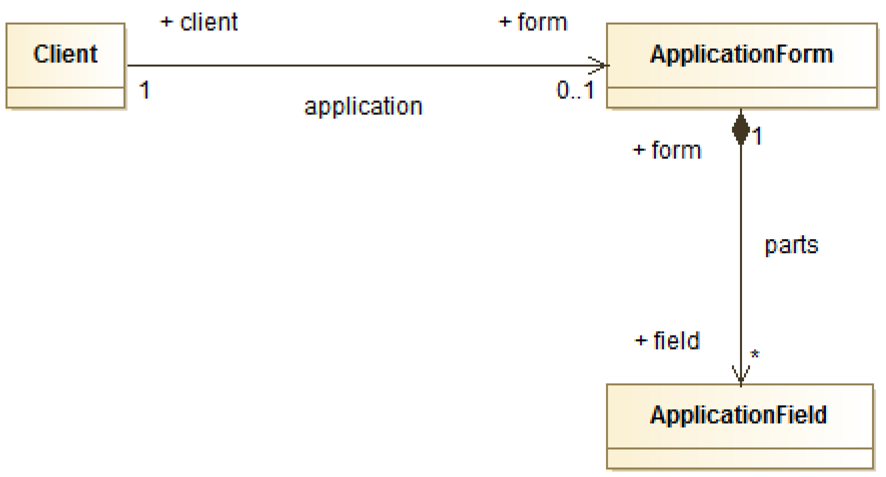

Figure 18 Client

Name	Summary
Client	The Client class represents the client or customer.
ApplicationForm	Form to fill to open a new account
ApplicationField	Entry in an application form.

Table 22 Owned Classes of Package "client"

16.1 Class "Client"

from Package ShoppingCart.Analysis.Domains.client

The Client class represents the client or customer. Its main role is to carry all the data attached to the client, like its name and address.

Name	Description
name : [1..1] string	This attribute holds the full name of the client, in accordance with the Home Office registry.
email : [1..1] string	This is the email address used to communicate with the customer in all circumstances.
address : [0..1] Address	

Table 23 Attributes of Class "Client"

Name	Description
owner->account : [0..1] Account	This association represents the relation between the Account class and the Client class. Each client has one and only one account.
application->form : [0..1] ApplicationForm	Each client has zero or one application form. This is the initial request for an account creation.

Table 24 Associations of Class "Client"

16.2 Class "ApplicationForm"

from Package ShoppingCart.Analysis.Domains.client

The ApplicationForm class regroups all the fields that need to be populated by the customer when applying for an account with the store.

Name	Description
id : [1..1] string	The unique identifier for the application form.

Table 25 Attributes of Class "ApplicationForm"

Name	Description
application->client : [1..1] Client	Each client has zero or one application form. This is the initial request for an account creation.
parts->field : [0..*] ApplicationField	An application form is made up of a certain number of application fields. Depending on the nature of the application, the number of fields may vary. The composition indicates the strong influence on the application field's lifecycle.

Table 26 Associations of Class "ApplicationForm"

16.3 Class "ApplicationField"

from Package ShoppingCart.Analysis.Domains.client

An application field is a simple question/answer pair of text corresponding to all the details required by the system to register a client. This class allows for free form applications to be created.

Name	Description
question : [1..1] string	The question asked to the client while filling the form. For simple details like the name or the address, this variable will have a simple text value like "Name:" or "Address:".
answer : [1..1] string	This variable is used to store the answer provided by the client to the corresponding question, for later processing by the component or actor concerned.

Table 27 Attributes of Class "ApplicationField"

Name	Description
parts->form : [1..1] ApplicationForm	An application form is made up of a certain number of application fields. Depending on the nature of the application, the number of fields may vary. The composition indicates the strong influence on the application field's lifecycle.

Table 28 Associations of Class "ApplicationField"

17Package "products"

*from Package ShoppingCart.Analysis.*Domains

The classes included in this package relate to the products on sale in the online shopping cart. All concepts related to the suppliers are modelled in this package.

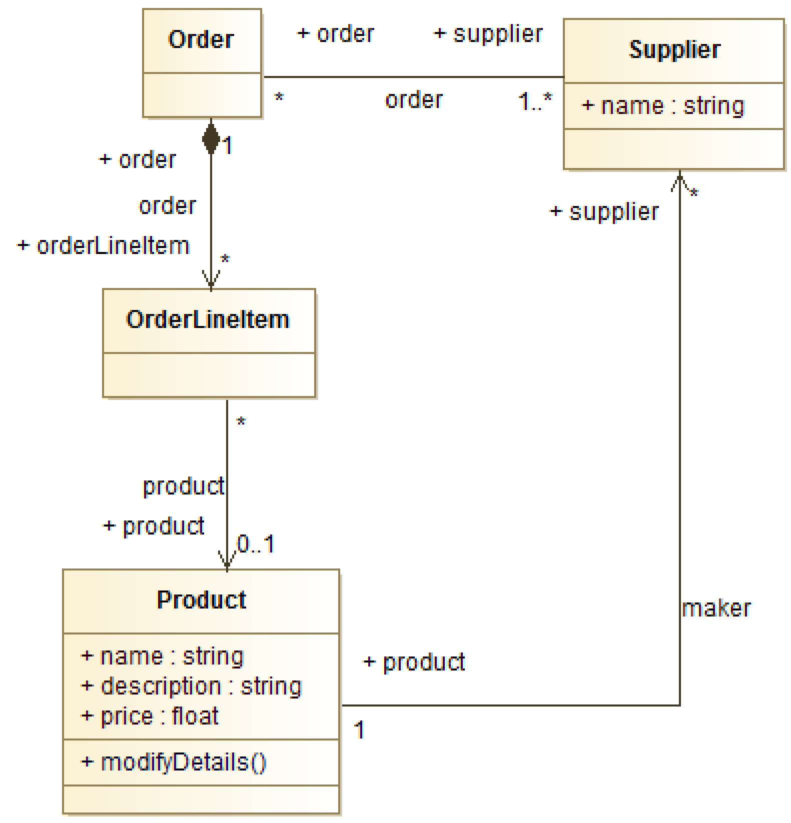

Figure 19 Products

Name	Summary
Product	Class repesenting the differents products suppliers offer for sale.
Supplier	Person or organisation providing goods for sale to the system.
Order	Document received by the supplier as a result of a client purchasing products online.
OrderLineItem	Represents one line in an order document.

Table 29 Owned Classes of Package "products"

17.1 Class "Product"

*from Package ShoppingCart.Analysis.Domains.*products

Class repesenting the differents products suppliers offer for sale. The complete product's details are included to allow for a comprehensive description to be provided to the buyer.

Name	Description
modifyDetails	Throuhg this method, client can change the details for the product. This method

Name	Description
()	does not provide a mean to modify the product's name, which is invariable. In the case the product changes name, a new Product instance must be created.

Table 30 Operations of Class "Product"

Name	Description
name : [1..1] string	This variable holds the name of the product. It is impossible to change the name of the product. The only access to the variables of this class is via the method modifyDetails(), which does not write this variable.
description : [1..1] string	Complete text provided by the supplier, and describing the product in details.
price : [1..1] float	The current price for the product. The price can be modified, providing the impact on customers is controlled. In no cases, the price can be changed once the client has pressed the "Check Out" button.

Table 31 Attributes of Class "Product"

Name	Description
list->catalog : [0..1] Catalog	This relation illustrates the fact that catalogs have products in them. This association enables the catalog class to maintain a reference on products listed in the catalog, in order to interrogate the Product class for product descriptions and details.
product-> : [0..*] OrderLineItem	In the same manner a shopping cart item has a reference to a product, an OrderLineItem has also a reference to a product.
carries->item : [0..*] CartItem	Each CartItem contains a reference to the product associated with the item. This association expresses this relation, in order to be able to reference the product from the item.
maker->supplier : [0..*] Supplier	Association modeling the relationship between products and supplier. In this model, a supplier does not have products, it's the product that has one and only one supplier. The same product may not have several supplier.

Table 32 Associations of Class "Product"

17.2 Class "Supplier"

from Package ShoppingCart.Analysis.Domains.products

Person or organisation providing goods for sale to the system. The system itself is onyl responsible for the selling, the supplier taking also responsibility for shipping and receiving products.

Name	Description
name : [1..1] string	The name of the supplier. this must be equal to the social name of the organisation or the Home Office record in the case of an individual.

Table 33 Attributes of Class "Supplier"

Name	Description
maker->product : [1..1] Product	Association modeling the relationship between products and supplier. In this model, a supplier does not have products, it's the product that has one and only one supplier. The same product may not have several supplier.
order->order : [0..*] Order	The supplier holds a certain number of orders at any one time, representing the orders still to be fulfilled.

Table 34 Associations of Class "Supplier"

17.3 Class "Order"

from Package ShoppingCart.Analysis.Domains.[products]

Document received by the supplier as a result of a client purchasing products online. The supplier uses orders to track the purchase requests originating from the online shopping system.

State Machine "State machine"

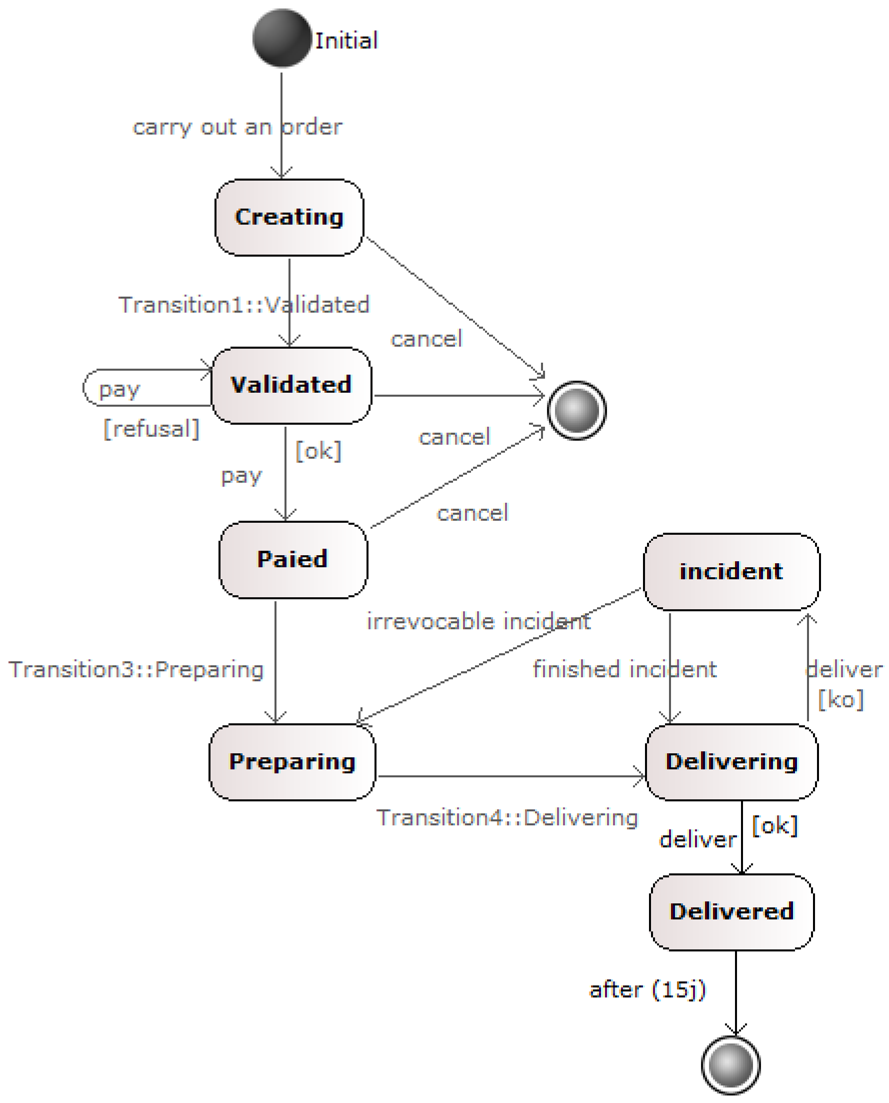

Figure 20 Order state diagram

Name	Description
order->orderLineItem : [0..*] [OrderLineItem]	Famous pattern of object-oriented modeling, this association between Order and OrderLineItem is a composition to show that items can only exist if belonging to an order.
order->supplier : [1..*] [Supplier]	This association connects the Supplier to its orders. Orders are received by the supplier originated from the system, as a result of a client purchasing a product online.

Table 35 Associations of Class "Order"

17.4 Class "OrderLineItem"

from Package ShoppingCart.Analysis.Domains.products

Represents one line in an order document. In the reality of the Online Shopping system, the OrderLineItem instances are in a one-to-one relation to the shopping cart items.

Name	Description
id : [1..1] string	Unique identifier for the OrderLineItem.

Table 36 Attributes of Class "OrderLineItem"

Name	Description
order->order : [1..1] Order	Famous pattern of object-oriented modeling, this association between Order and OrderLineItem is a composition to show that items can only exist if belonging to an order.
product->product : [0..1] Product	In the same manner a shopping cart item has a reference to a product, an OrderLineItem has also a reference to a product.

Table 37 Associations of Class "OrderLineItem"

18Package "bank"

from Package ShoppingCart.Analysis.Domains

This package contains all the classes in relation to the bank and payments. Since the bank system is separate from the online shopping cart, the classes contained in this package represent the interface to the bank payment system, and in particular the exchange data types used to converse with the bank.

Name	Summary
Payment	This class represents a payment.

Table 38 Owned Classes of Package "bank"

18.1 Class "Payment"

from Package ShoppingCart.Analysis.Domains.bank

This class represents a payment. it carries all the details necessary to the actual financial transaction and serves as reference for historisation.

Name	Description
sourceAccount : [1..1] string	The account of the buyer to be debited.
targetAccount : [1..1] string	The account of the seller to be credited.
amount : [1..1] float	The amount to be transferred between the accounts of the seller and the buyer.
date : [0..1] date	

Table 39 Attributes of Class "Payment"

19Package "types"

from Package ShoppingCart.Analysis.Domains

This package contains the definition of non-primitive base data types useful to give a type to domain class attributes.

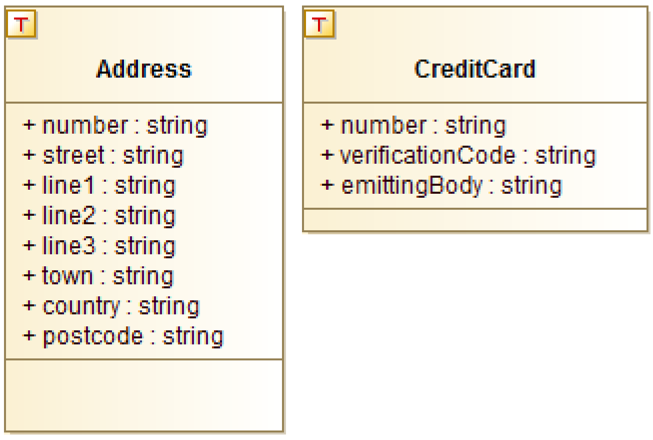

Figure 21 Base Types

Name	Summary
Address	Datatype for an address
CreditCard	Datatype for a credit card.

Table 40 Owned DataTypes of Package "types"

19.1 DataType "Address"

from Package ShoppingCart.Analysis.Domains.types

Datatype for an address. The fields of this datatype have been designed to support addresses in arbitrary countries.

Name	Description
number : [1..1] string	the street number. This field accepts alphanumerical characters (10a, 12c...)
street : [1..1] string	The name and type of street.
line1 : [1..1] string	Additonal line of text to fully describe the address.
line2 : [1..1] string	Additonal line of text to fully describe the address.
line3 : [1..1] string	Additonal line of text to fully describe the address.
town : [1..1] string	The name of the town
country : [1..1] string	The name of the country
postcode : [1..1] string	The postcode or zipcode depending on the country.

Table 41 Attributes of DataType "Address"

19.2 DataType "CreditCard"

from Package ShoppingCart.Analysis.Domains.types

Datatype for a credit card. This type contains the 16-digit card number uniquely identifying the credit card.

Name	Description
number : [1..1] string	The 16-digit number uniquely identifying the credit card. In order to support alphanumerical characters in the future, the type of this variable is "string" rather than an integer.
verificationCode : [1..1] string	The three-digit code indicated on the back of the card, and used to verify authenticity. In order to support alphanumerical characters in the future, this variable is typed by "string".
emittingBody : [1..1] string	The organisation that emitted the card, usually a financial body.

Table 42 Attributes of DataType "CreditCard"

20Package "Design"

from Package ShoppingCart

This section describes the logical architecture chosen to implement the online shopping cart system. It is based on a component approach in accordance with the SCA architecture. Components are organized in several layers according to their level of stability. The layer "Entity" contains the most stable components, while the layer "Presentation" contains the least stable ones. Components use other components to provide the required functionalities. A component can only make use of the interfaces of components belonging to a layer of greater stability, and this to insure the highest level of reusability possible.

Name	Summary
Logical Architecture	The Online Shopping Cart logical architecture.

Table 43 Owned Packages of Package "Design"

21Package "Logical Architecture"

from Package ShoppingCart.Design

The logical architecture contained in this package correspond to the technological choices made by the architects to implement the shopping cart system.

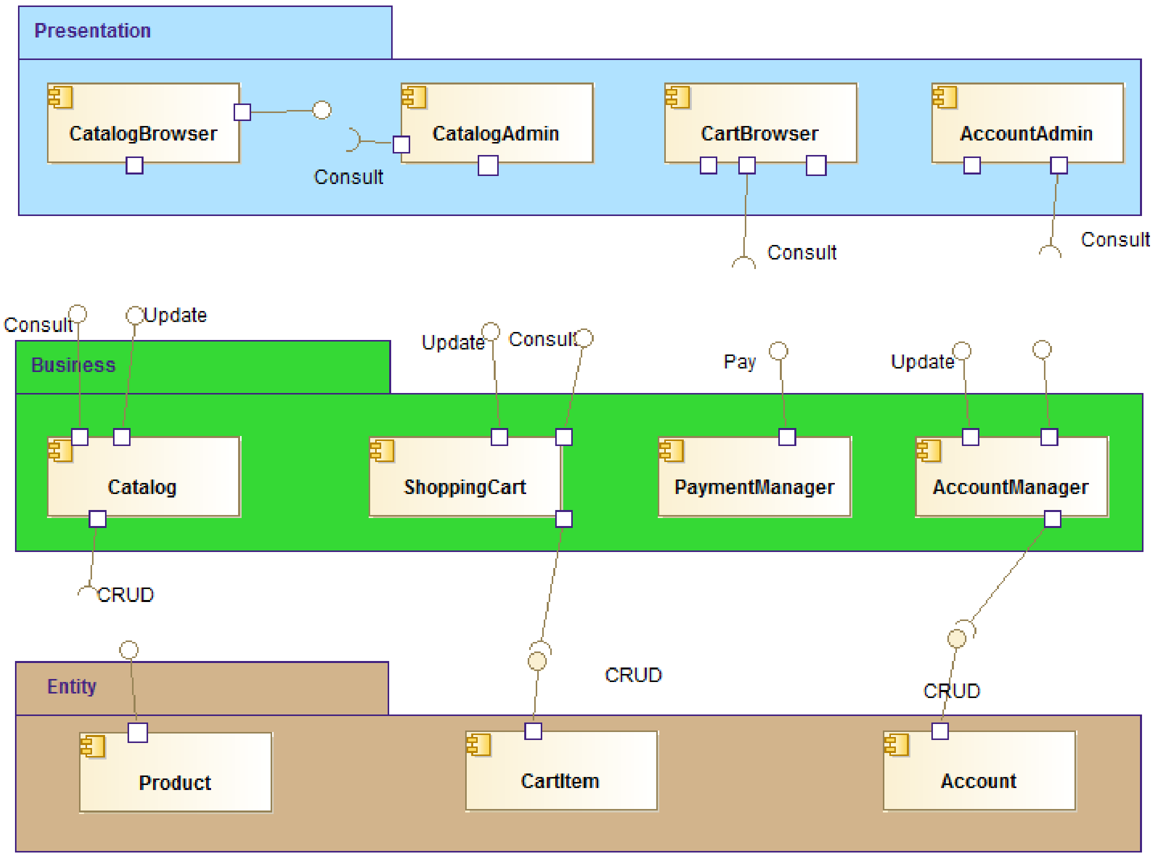

Figure 22 Logical Architecture

Name	Summary
Presentation	User interface components layer.
Business	Business layer and its components
Entity	Persistence layer components.

Table 44 Owned Packages of Package "Logical Architecture"

22Package "Presentation"

from Package ShoppingCart.Design.Logical Architecture

Regroups the components in charge of providing a user interface to the online shopping cart system.

Name	Summary
CatalogBrowser	Interface component allowing users to browse the catalog.
CartBrowser	Interface component providing the functionalities to browse the user cart.
CatalogAdmin	This component provids the administration interface for the catalog.
AccountAdmin	This component provides the Online Shopping Cart system administration interface.

Table 45 Owned Classes of Package "Presentation"

22.1 Class "CatalogBrowser"

from Package ShoppingCart.Design.Logical Architecture.Presentation

This user-interface component is in charge of providing all the widgets and functionalities to browse the product catalog. In order to provide a good ergonomy, the component implements pagination and ordering functionalities on the list of products.

Name	Requires	Provides

Table 46 Ports of Component "CatalogBrowser"

22.2 Class "CartBrowser"

from Package ShoppingCart.Design.Logical Architecture.Presentation

This GUI component allows users to browse their shopping cart. In addition to all the functions required to visualize the items, this component provides the ability to add and remove items, and presents the check out button.

Name	Requires	Provides
		Interface Consult

Table 47 Ports of Component "CartBrowser"

22.3 Class "CatalogAdmin"

from Package ShoppingCart.Design.Logical Architecture.Presentation

This component provides the graphical interface for administering the catalog. It reuses the CatalogBrowser component, and adds the functionalities required to add and remove products from the catalog. Its access is reserved for the suppliers and makers of products.

Name	Requires	Provides
		Interface Consult

Table 48 Ports of Component "CatalogAdmin"

22.4 Class "AccountAdmin"

from Package ShoppingCart.Design.Logical Architecture.Presentation

This component comprises all the functions to administer the user accounts. The system administrator controls access to the site by granting accounts to users. It is also possible to revoke accounts for users not respecting the conditions of use of the site.

Name	Requires	Provides
		Interface Consult

Table 49 Ports of Component "AccountAdmin"

23Package "Business"

from Package ShoppingCart.Design.Logical Architecture

In this package, the components in charge of implementing the business behaviour of the system are described.

Name	Summary
Catalog	This component manages the system catalog.
ShoppingCart	This component provides the shopping cart management functions.
PaymentManager	Component implementing the payment related operations.
AccountManager	User accounts management component.

Table 50 Owned Classes of Package "Business"

23.1 Class "Catalog"

from Package ShoppingCart.Design.Logical Architecture.Business

This component provides all the functions related to the catalog. It bears all of the query operations that can be carried out on the catalog, including set operations returning lists of catalog items based on filtered queries.

Name	Summary
Consult	
Update	

Table 51 Owned Interfaces of Class "Catalog"

Name	Requires	Provides
	Interface Consult	
	Interface Update	
		Interface CRUD

Table 52 Ports of Component "Catalog"

23.2 Class "ShoppingCart"

from Package ShoppingCart.Design.Logical Architecture.Business

This component implements the Shopping Cart. It contains all the elements required to completely describe the component's internal and external parts. Its role is to manage the shopping cart and to provide business-level validation for the cart content.

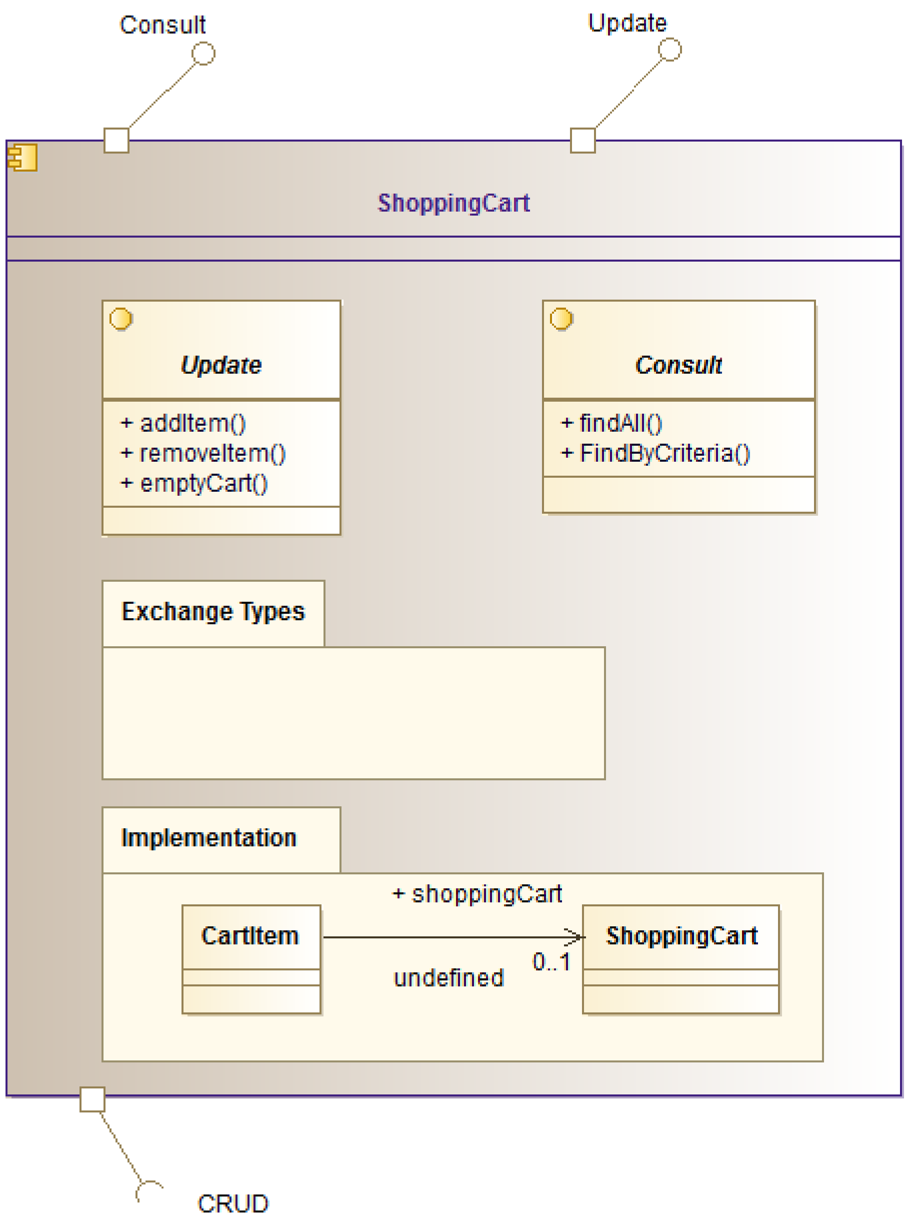

Figure 23 ShoppingCart Component View

Name	Summary
Update	
Consult	

Table 53 Owned Interfaces of Class "ShoppingCart"

Name	Requires	Provides
	Interface Update	
	Interface Consult	
		Interface CRUD

Table 54 Ports of Component "ShoppingCart"

23.3 Class "PaymentManager"

from Package ShoppingCart.Design.Logical Architecture.Business

This component provides all the operations related to payments. It represents the integration point with the external system pertaining to the bank that handles actual payments.

Name	Summary
Pay	

Table 55 Owned Interfaces of Class "PaymentManager"

Name	Requires	Provides
	Interface Pay	

Table 56 Ports of Component "PaymentManager"

23.4 Class "AccountManager"

from Package ShoppingCart.Design.Logical Architecture.Business

This component provides the interfaces that allow for the management of user accounts. It includes the authorisation checks necessary to make sure only the allowed customers can log into the system to carry out purchases.

Name	Summary
Update	
Consult	

Table 57 Owned Interfaces of Class "AccountManager"

Name	Requires	Provides
	Interface Update	
		Interface CRUD

Table 58 Ports of Component "AccountManager"

24Package "Exchange Types"

from Component ShoppingCart.Design.Logical Architecture.Business.ShoppingCart

This package contains the description of the exchange types included in the service contract for the ShoppingCart component. The exchange types are deduced from the domain analysis, and are the only data types accepted and produced by the component.

Name	Summary
ItemsList	
ProductDescription	

Table 59 Owned Classes of Package "Exchange Types"

24.1 Class "ItemsList"

from Package ShoppingCart.Design.Logical Architecture.Business.ShoppingCart.Exchange Types

24.2 Class "ProductDescription"

from Package ShoppingCart.Design.Logical Architecture.Business.ShoppingCart.Exchange Types

25Package "Implementation"

from Component ShoppingCart.Design.Logical Architecture.Business.ShoppingCart

This package contains the elements used to implement the functionalities provided by the component. It is the internal part of the component, that can change without affecting clients of the component services.

Name	Summary
CartItem	
ShoppingCart	
Account	

Table 60 Owned Classes of Package "Implementation"

25.1 Class "CartItem"

from Package ShoppingCart.Design.Logical Architecture.Business.ShoppingCart.Implementation

Name	Description
undefined->shoppingCart : [0..1] ShoppingCart	

Table 61 Associations of Class "CartItem"

25.2 Class "ShoppingCart"

from Package ShoppingCart.Design.Logical Architecture.Business.ShoppingCart.Implementation

Name	Description
undefined-> : [..] CartItem	

Table 62 Associations of Class "ShoppingCart"

25.3 Class "Account"

from Package ShoppingCart.Design.Logical Architecture.Business.ShoppingCart.Implementation

26Package "Entity"

from Package ShoppingCart.Design.Logical Architecture

This package represents the most stable layer of the logicla architecture, and contains the components in charge of persisting the business entities required by the system.

Name	Summary
Product	Product persistence component.
CartItem	CartItem persistence component.
Account	Account persistence component.

Table 63 Owned Classes of Package "Entity"

26.1 Class "Product"

from Package ShoppingCart.Design.Logical Architecture.Entity

This is a low-level component that is responsible for implementing the persistence for Product instances. It contains the mapping between the object instances and the physical persistence relational database.

Name	Summary
CRUD	

Table 64 Owned Interfaces of Class "Product"

Name	Requires	Provides

Table 65 Ports of Component "Product"

26.2 Class "CartItem"

from Package ShoppingCart.Design.Logical Architecture.Entity

This is a low-level component that is responsible for implementing the persistence for CartItem instances. It contains the mapping between the object instances and the physical persistence relational database.

Name	Summary
CRUD	

Table 66 Owned Interfaces of Class "CartItem"

Name	Requires	Provides

Table 67 Ports of Component "CartItem"

26.3 Class "Account"

*from Package ShoppingCart.Design.Logical Architecture.*Entity

This is a low-level component that is responsible for implementing the persistence for Account instances. It contains the mapping between the object instances and the physical persistence relational database.

Name	Summary
CRUD	

Table 68 Owned Interfaces of Class "Account"

Name	Requires	Provides

Table 69 Ports of Component "Account"

27Package "Deployment"

from Package ShoppingCart

This package contains the deployment model for the online shopping cart system. It currently contains a simple deployment scenario that will be used to measure performances and charges.

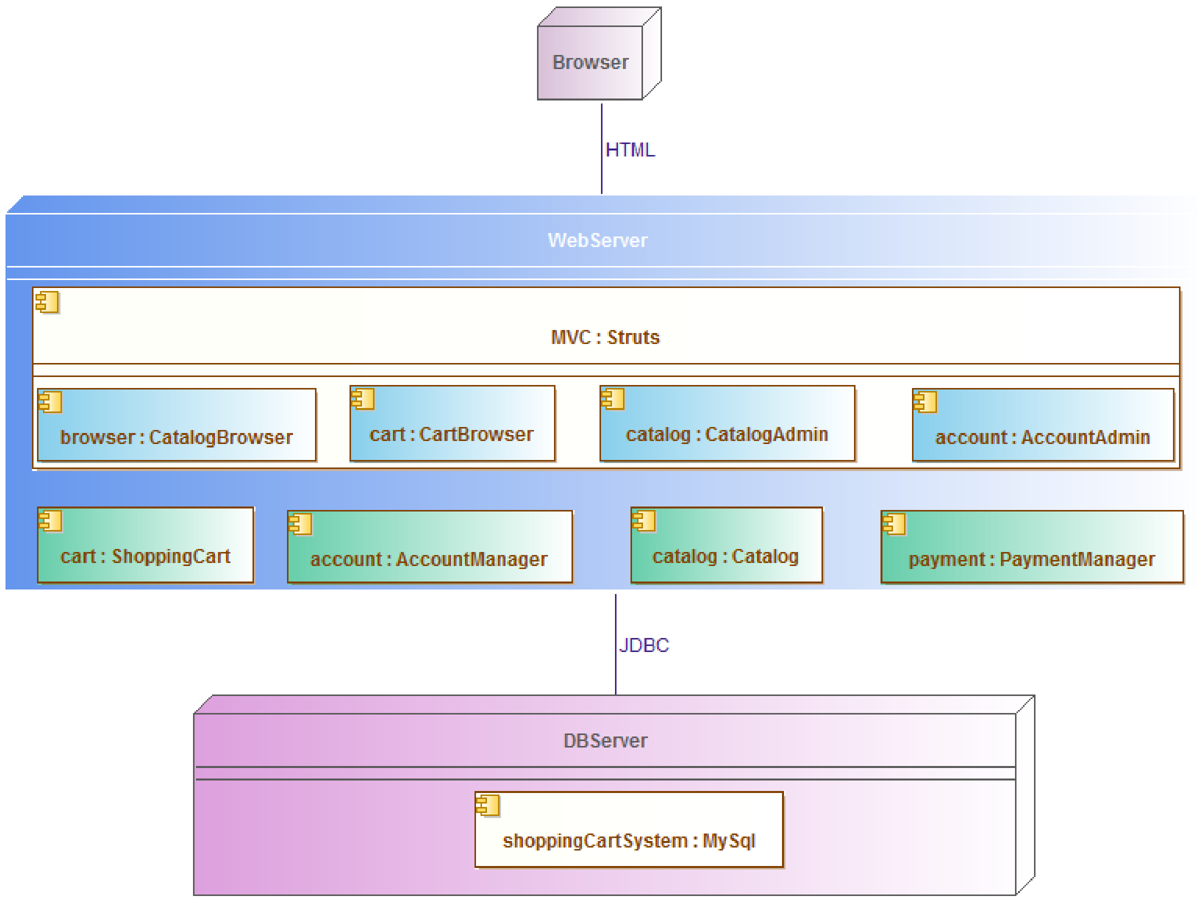

Figure 24 Deployment

Name	Summary
MySql	Chosen database server.
Struts	Graphical framework component.

Table 70 Owned Classes of Package "Deployment"

27.1 Class "MySql"

*from Package ShoppingCart.*Deployment

This component represents the database chosen for persistence. In this case, we choose MySQL as the relational database, for its ease-of-use combined with extensive functionalities.

27.2 Class "Struts"

from Package ShoppingCart.Deployment

This component represents the chosen graphical framework to implement the actual web pages displayed to the user by the system.