Shop Domain Model

Version Latest

Table of Contents

Introduction

This document describes Shop domain using concepts defined by Domain-Driven Design (DDD) methodology. It was generated directly from source code and can be considered as a close summary of what is actually implemented. A description of followed conventions is given below.

The document is split in two parts:

1. the description of the different modules and their components,
2. the ubiquitous language, presented in the form of a glossary. Each entry is composed of a name, the module (if relevant), the type of component and a short description.

1. Modules

Each module has its own section, each containing sub-sections for each aggregate, service and domain process in the module.

Each module section starts with the description of the module and an undirected graph. The nodes represent the aggregates of the module and the edges represent the links between those aggregates. A link between two aggregates means that one aggregates holds a reference to the other in its attributes.

2. Aggregates

Each aggregate section starts with the description of the aggregate and an undirected graph. Each node of the graph represents a component (an entity or a value object) part of the aggregate. The edges represent links between the components of the aggregate.

Follows the description of the Value Objects and Entities part of the aggregate and represented in the aggregate graph.

The aggregate section ends with a directed graph showing how current aggregate is connected to other aggregates, modules or external systems in terms of consumption and transmission of events.

Current aggregate is represented by a box with bold borders, other aggregates are represented by boxes with a thin border. Dashed boxes represent other modules or external systems issuing or consuming events. Elliptic nodes represent the events.

An edge going from a box to an ellipse means that the component represented by the box issues the event represented by the ellipse. An edge going from an ellipse to a box means that the component represented by the box consumes the event represented by the ellipse.

3. Domain Processes

Each domain process section starts with the description of the process and a directed graph.

Each node of the graph represents a message listener (ellipses) or other modules or external systems (boxes) producing and/or sending messages.

Each directed edge represents a message being produced by source node and consumed by destination node.

The section ends with the list of message listeners involved in the process and their description. The naming convention for the message listeners is

```
Component.listenerName(Event)
```

where

```
Component
```

is the name of the enclosing component (an Aggregate Root, a Factory, etc.),

```
listenerName
```

is the name of the listener inside of the component and

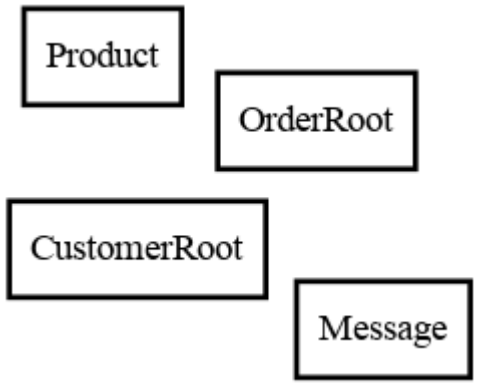
```
Event
```

is the name of the consumed event.

Shop Domain

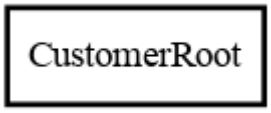
1. Shop

Models an online shop where Customers may buy Products by placing Orders. Customers receive Message giving them an update about the handling of their Orders.

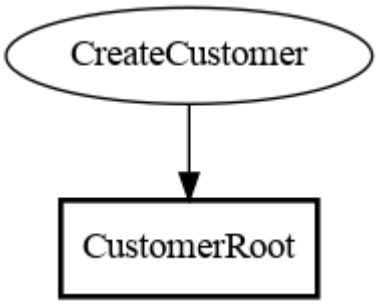


1.1. Aggregates

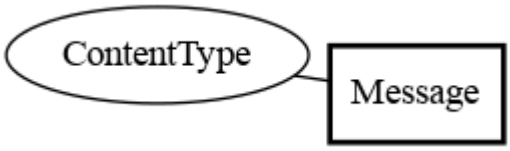
1.1.1. CustomerRoot



1.1.1.1. Events



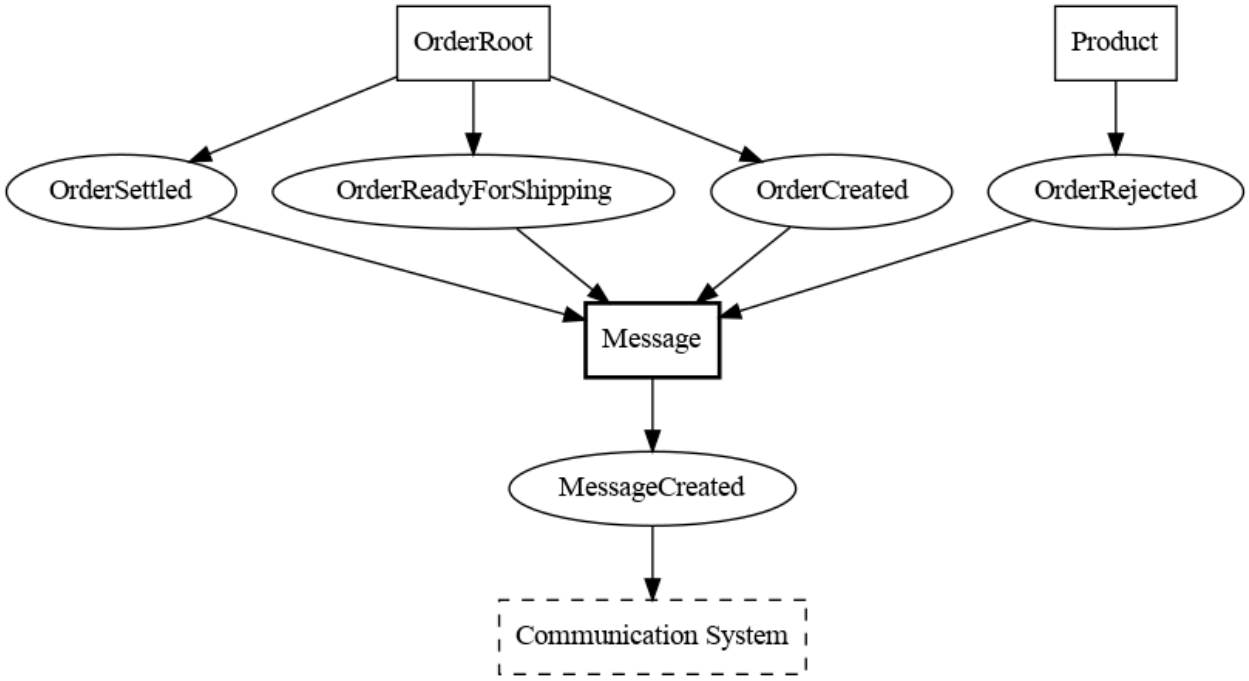
1.1.2. Message



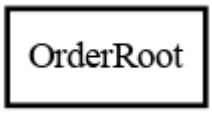
1.1.2.1. Value Objects

- **ContentType:**

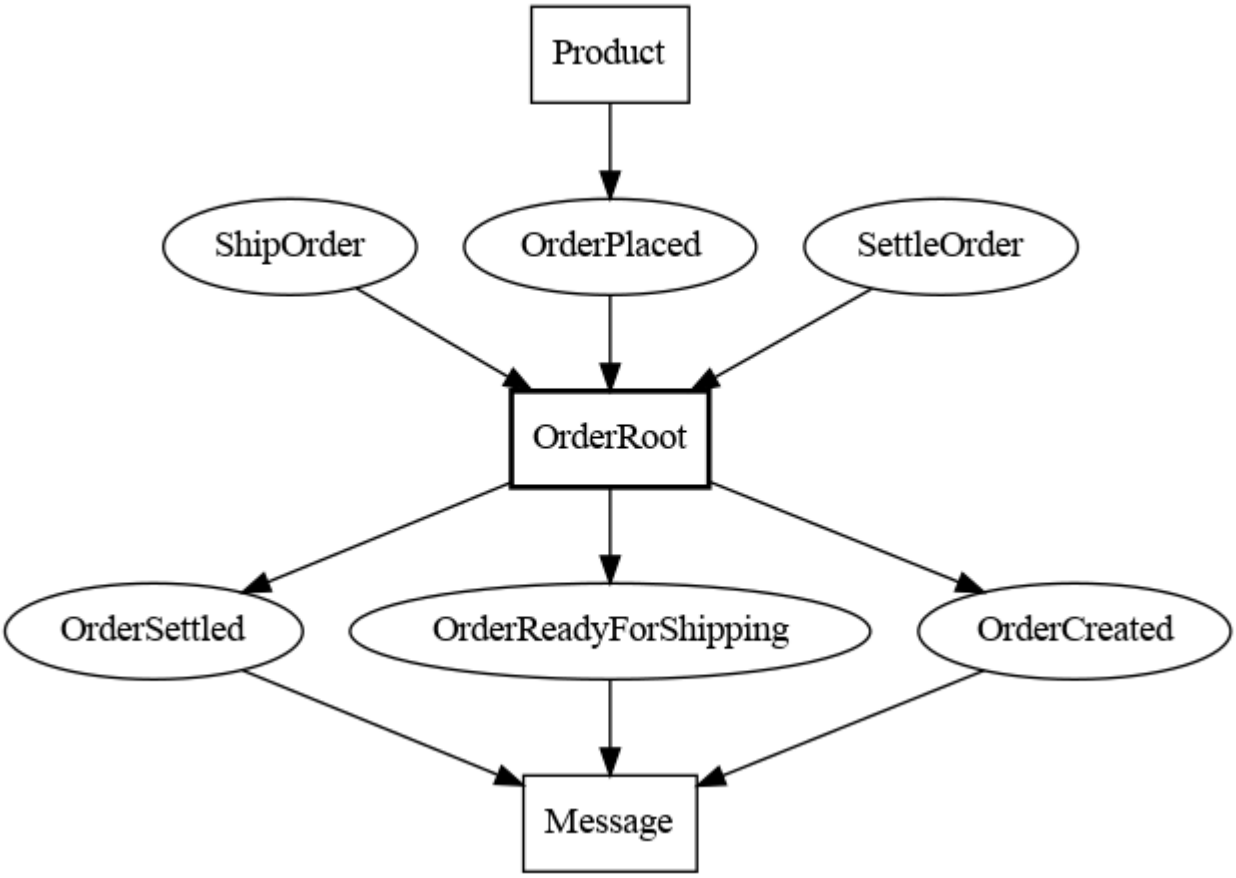
1.1.2.2. Events



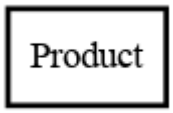
1.1.3. OrderRoot



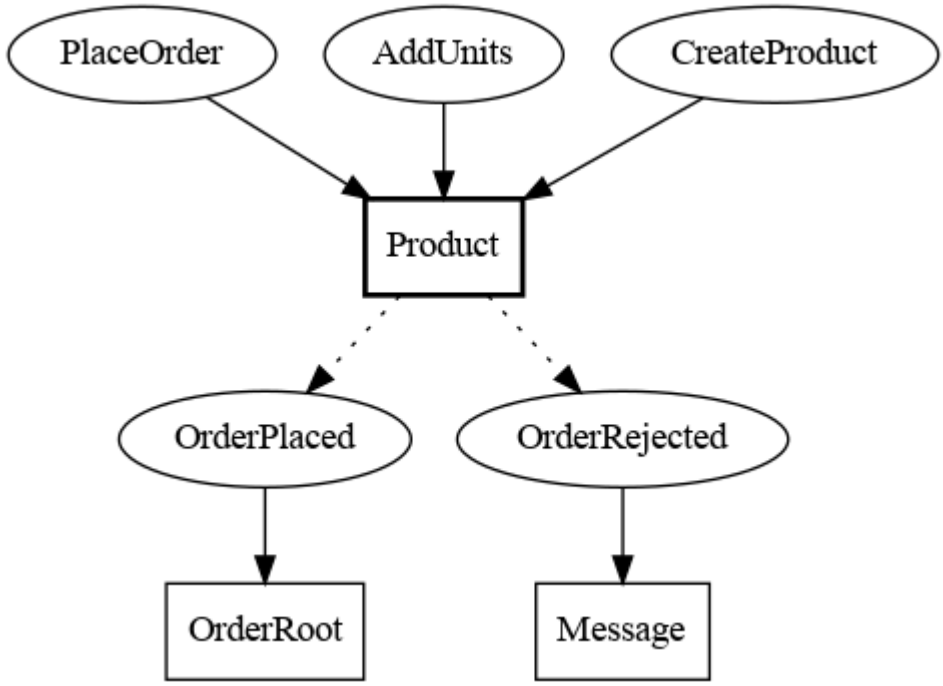
1.1.3.1. Events



1.1.4. Product

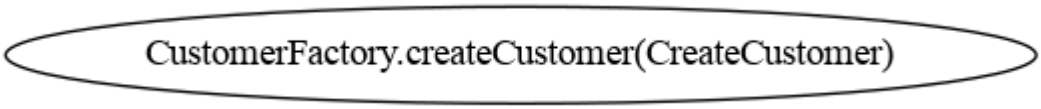


1.1.4.1. Events



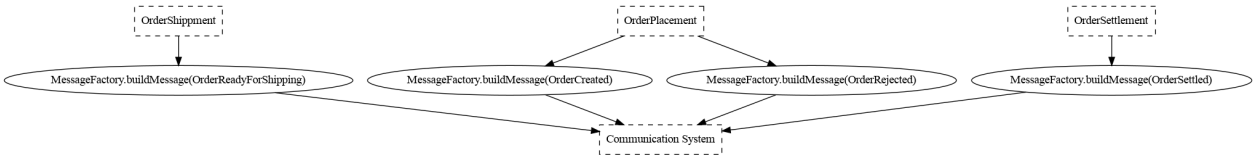
1.2. Domain Processes

1.2.1. CustomerCreation



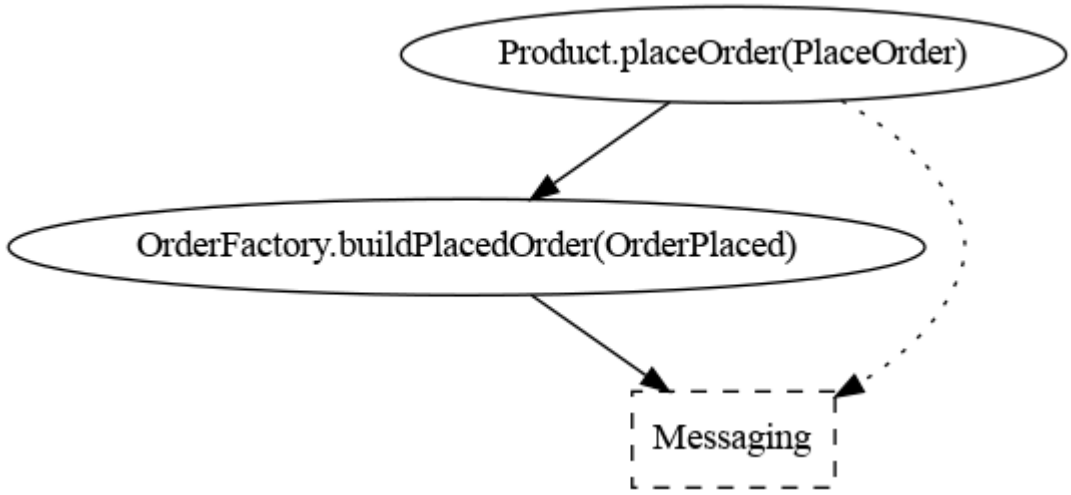
- **CustomerFactory.createCustomer(CreateCustomer):**

1.2.2. Messaging



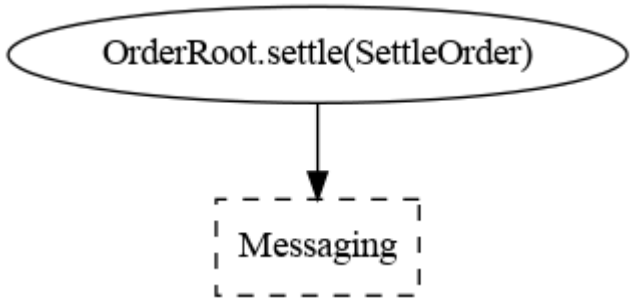
- **MessageFactory.buildMessage(OrderReadyForShipping):**
- **MessageFactory.buildMessage(OrderSettled):**
- **MessageFactory.buildMessage(OrderCreated):**
- **MessageFactory.buildMessage(OrderRejected):**

1.2.3. OrderPlacement



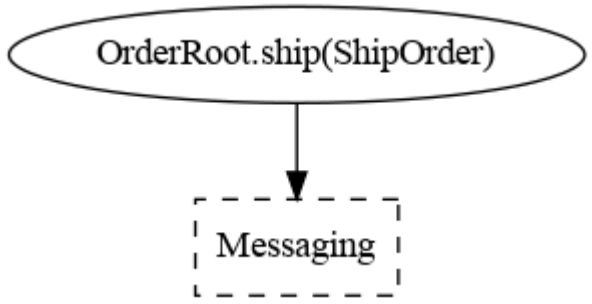
- **Product.placeOrder(PlaceOrder):**
- **OrderFactory.buildPlacedOrder(OrderPlaced):**

1.2.4. OrderSettlement



- **OrderRoot.settle(SettleOrder):**

1.2.5. OrderShippment



- **OrderRoot.ship(ShipOrder):**

1.2.6. ProductManagement

- **ProductFactory.buildProductWithNoStock(CreateProduct):**
- **Product.addUnits(AddUnits):**

Ubiquitous Language

ContentType (Shop), Value Object,
CustomerCreation (Shop), Domain Process,
CustomerRoot (Shop), Aggregate,
Message (Shop), Aggregate,
Messaging (Shop), Domain Process,
OrderPlacement (Shop), Domain Process,
OrderRoot (Shop), Aggregate,
OrderSettlement (Shop), Domain Process,
OrderShippment (Shop), Domain Process,
Product (Shop), Aggregate,
ProductManagement (Shop), Domain Process,
Shop, Module, Models an online shop where Customers may buy Products by placing Orders. Customers receive Message giving them an update about the handling of their Orders.