Business Object Type Library

Draft Technical Specification

Version Information

Version 0.01, November 26, 2001

Executive Summary

The eBTWG Business Object Type Library (BOTL) provides a framework of standardized Business Object Type (BOT) packages. These packages describe behavior, attributes and purpose of persistent business objects that are the subject of business collaborations. For businesses to optimize their execution of supply chain relationships they must align their definitions of the subjects of their collaborations. Often these business objects are the subject of multiple collaborations which occur over time. These business objects usually have persistent existence in the databases of both trading partners, and alignment of these instantiations to a shared definition is critical to achieving alignment between the partners' instances during process execution. The BOT specification provides a Business Operations Map (BOM) and Business Requirements View (BRV) mechanism for identifying and elaborating these objects.

eBTWG and UMM context

The Business Object Type Library is intended to provide a resource for: 1) reusable definition of conditions (business object state) used for business collaboration preconditions, postconditions, and transitions; and 2) identification of business semantic for the referenced business object. In the context of the UMM this allows a more formal declaration and elaboration of the guards on start and end states, and the guards on transitions between business transactions in the definition of a complex collaboration. In the context of eBTWG this establishes a mechanism for mapping the collaborations onto an integrated and reusable set of domain definitions instead of the current complex conditionals that reference elements of transitory business messages. The BOT are essential to meeting the objectives of eBTWG:

- Partner Information Alignment: *"You know what I think you know"*
- Partner Process Alignment: *"You are doing what I think you are doing"*
- Partner Expectations Alignment: *"We will accomplish what we expect to accomplish"*

Essence of BOTL

- Understanding the behavior and lifecycle of business objects is critical to the objectives of the eBTWG
- Business objects have identity and persist over time in the public collaboration domain

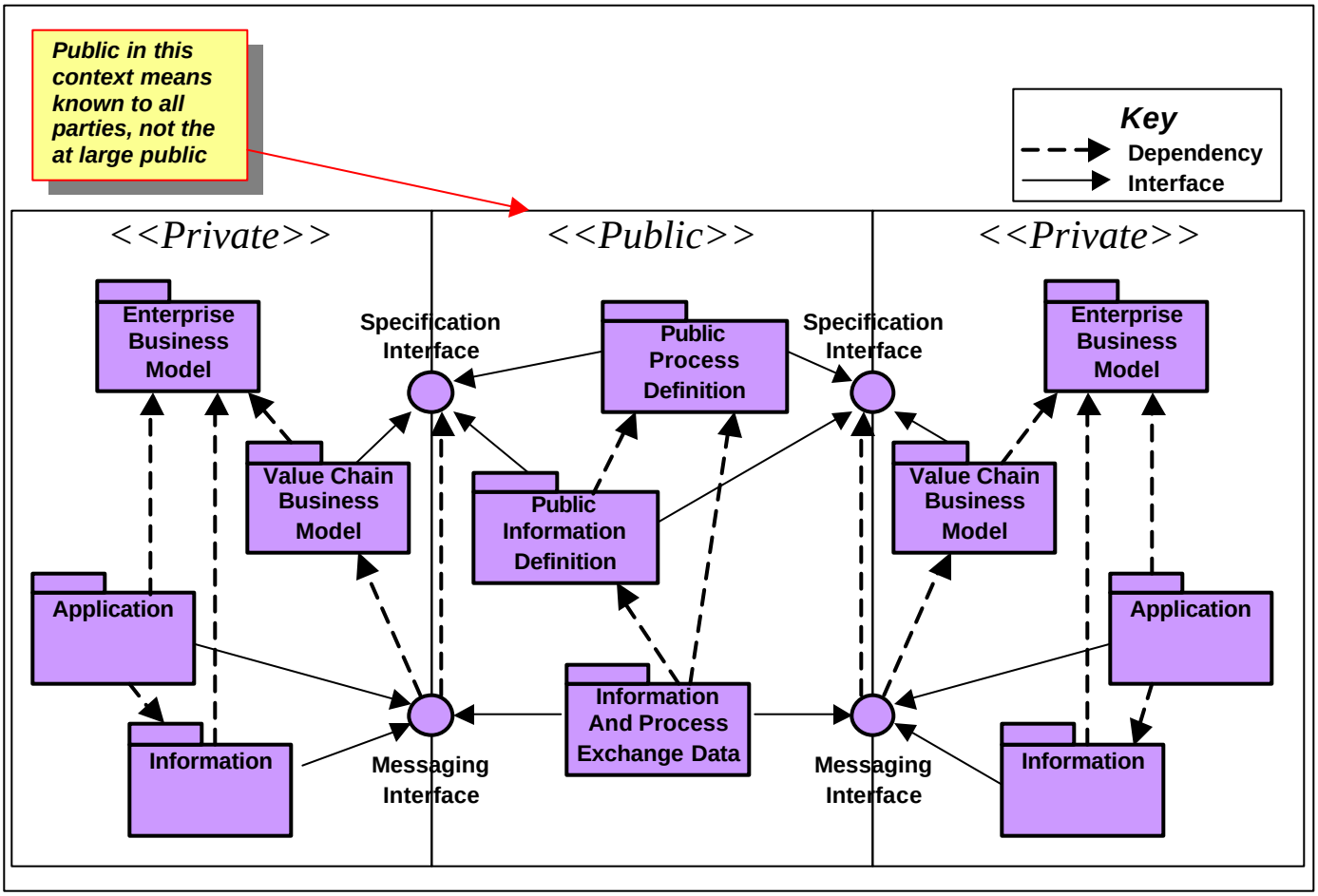
- BOTL provides a model exposing business object state space in the public collaboration domain
- Business collaboration use cases take business objects from one reference state to another reference state
- There is business semantic (business objective) attached to business objects. Classifying business objects according to REA provides significant value to the construction and use of entity representations of these business objects

BOTL working hypothesis

- The objective of a business collaboration is best described by a set of business objects reaching specific states. *(For example, the objective of a procurement collaboration is a delivered product and a paid invoice)*
- These states are directly related to the business semantic (business objective) of that object in the collaboration
- The collaboration specification must identify reference business objects and express their reference states
- Normalization of these reference states and the relationship of the state to the business semantic adds value to understanding and acting upon collaboration dependencies
- Business objects will go through a lifecycle of states, changing state through exposed behavior
- The goal of a business collaboration protocol is to achieve alignment between partners on these states and business semantics

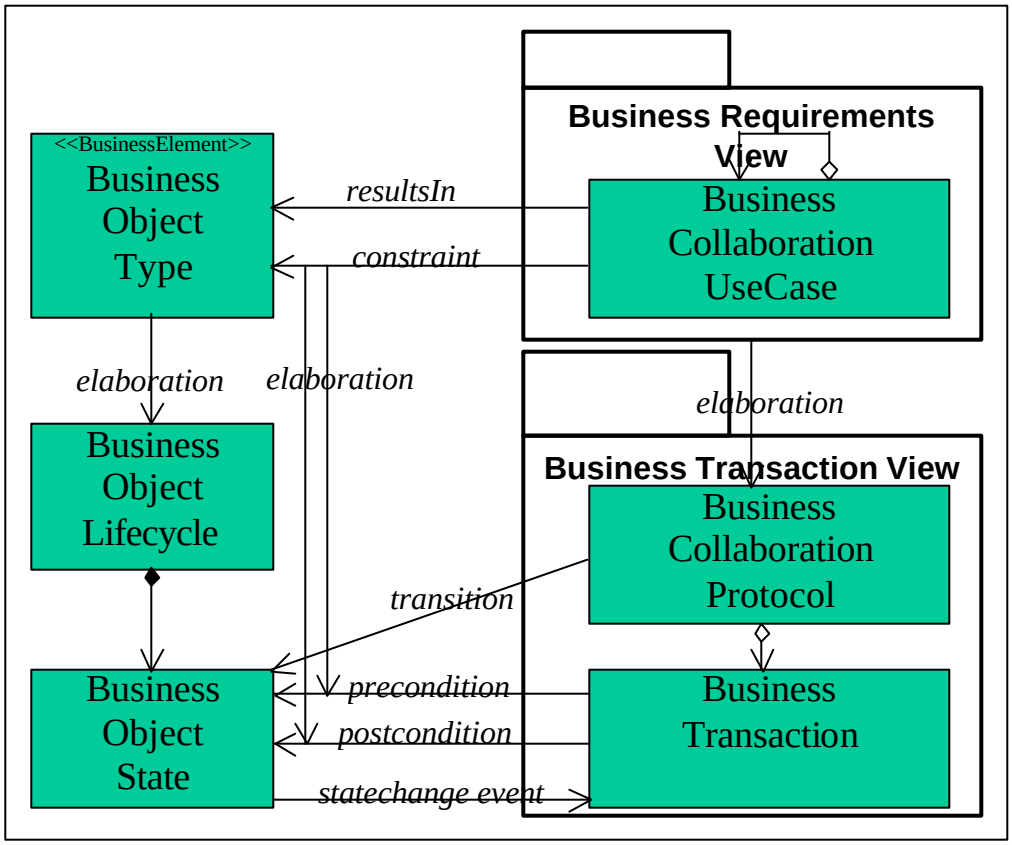
Domain of alignment

eBTWG is providing two primary interfaces for partner interaction: 1) Definition interface for a language for alignment (models, specifications, ontologies); and 2) Execution interface for artifacts of alignment (messages, services). These interfaces facilitate communication between partners. The packages in the "collaboration domain" provide a language for this communication. The purpose of this diagram is to place the next two slides in context. Slide 9 elaborates on the public process definition and slide 10 elaborates on the public information definition. Remember that "public" in this case means visible to involved parties, not visible to at large public.



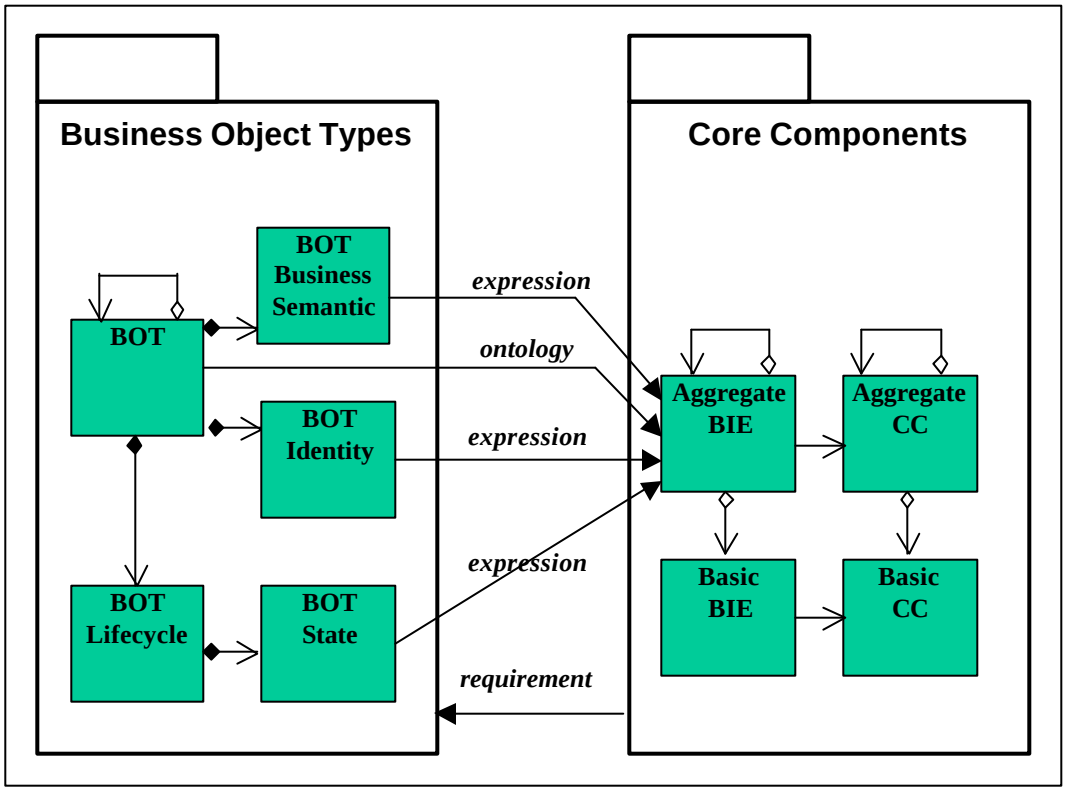
Use of BOT in process construction

The business object states identified and formalized in the business object lifecycle provide the domain for arguments in the condition statements used to guard transitions and define start and end states. The events of state change can be used as the domain for expressing the triggers for business transaction execution. The existence of an ontology for these states allows the BRV model to be completed without referencing specific BusinessEntity properties which should instead be elaborated in the analysis (BTV) phase.



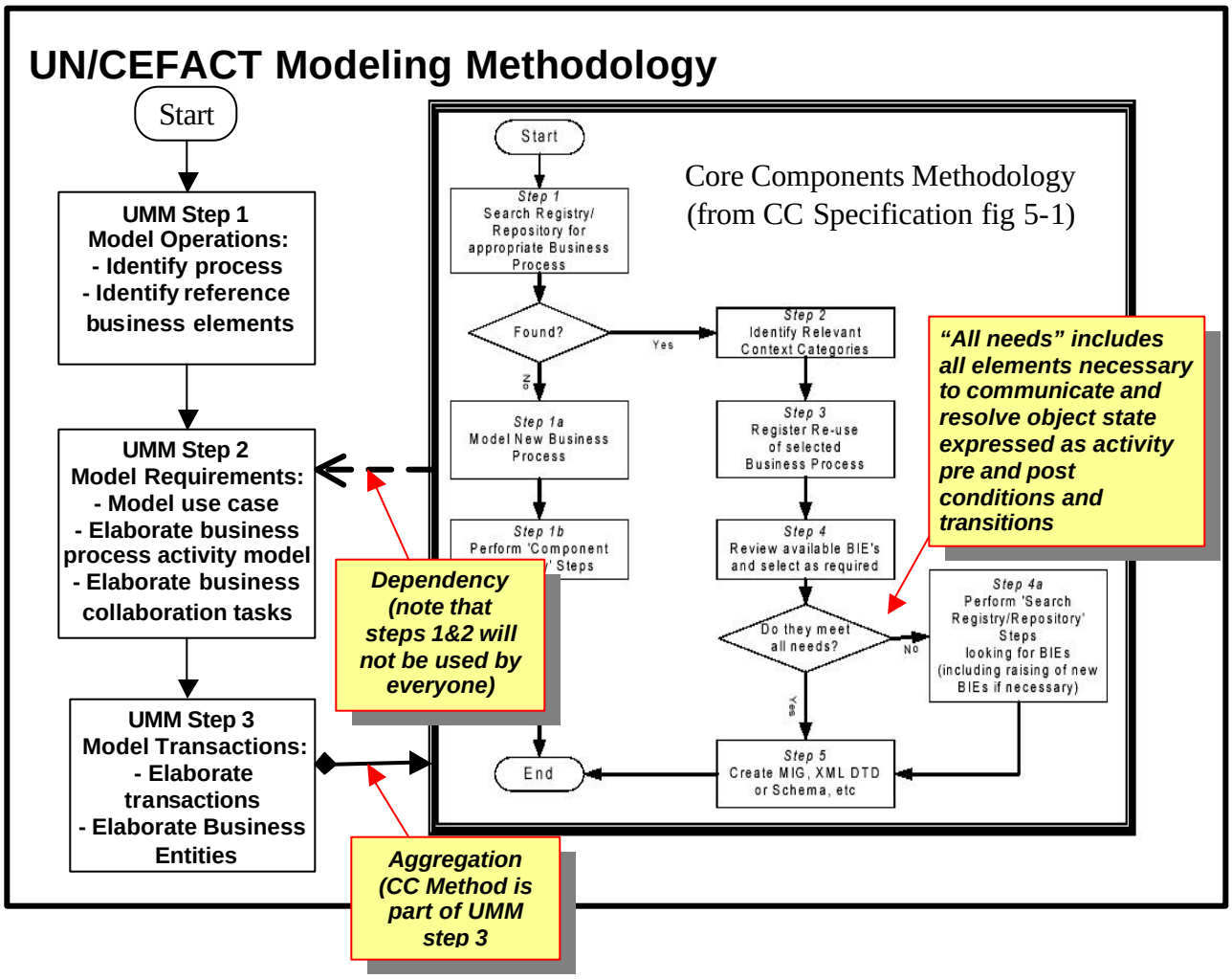
Use of BOT in information construction

Business Object Type identity, semantic and state definition are requirements which are elaborated into specific business entity attributes. These entity attributes carry business information realizations of the identity, semantic, and state. There are specific relationships between the elements of the BOTL and the elements of Core Componenets: 1) The BusinessInformationEntity namespace should provide the ontological basis for naming the subset of Entities that are Objects (have instance identity). 2) Identification of instance of a BOT is expressed (elaborated) using BIE. 3) Identification of state of a BOT is expressed (elaborated) using BIE. 4) BOT will provide requirements for additional BIE and additional CC content to particular BIE (to render them capable of expressing the required states).



BOT relationship to Core Components in UMM

This diagram places Business Object Types and the Core Components process in the context of an simplified and abbreviated UMM process. Note that Step 2 of the UMM includes identification of business semantic and business process transitions. Note that Step 3 of the UMM process contains the elaboration of the BusinessEntities required by the BusinessTransactions, and that the CoreComponents process provides a detailed method for this elaboration (CC process is part of UMM step 3). If the entire UMM is being followed, then there is a dependency between the CC process and BOT definition in UMM step 2, since step 2 would identify all of the processes that are elaborated and the states that are expressed (preconditions, postconditions, and transition conditions) by instances of BIE.

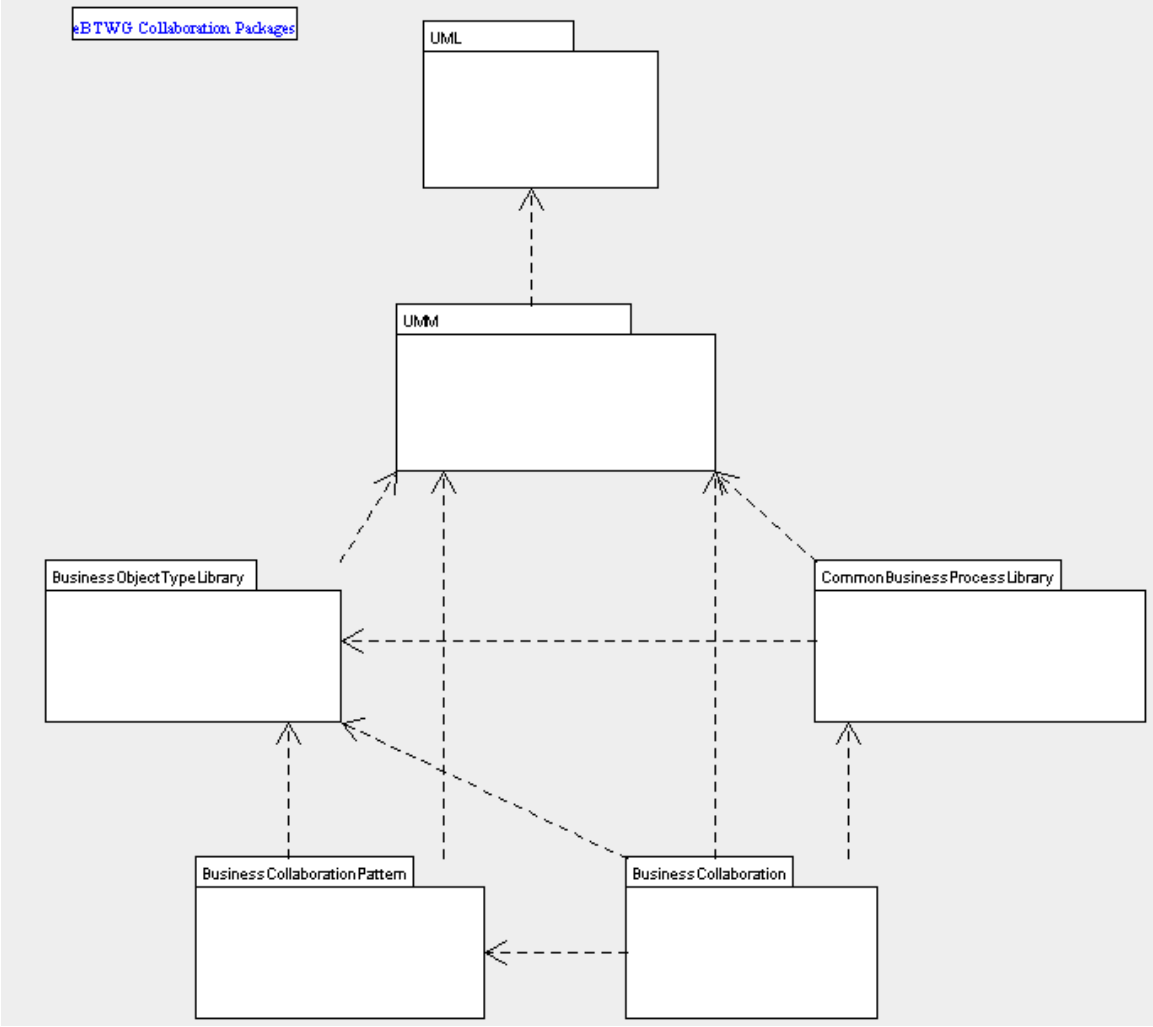


BOTL Packaging

As a library, packaging is a critical part of the usability of the library for constructing business collaboration specifications.

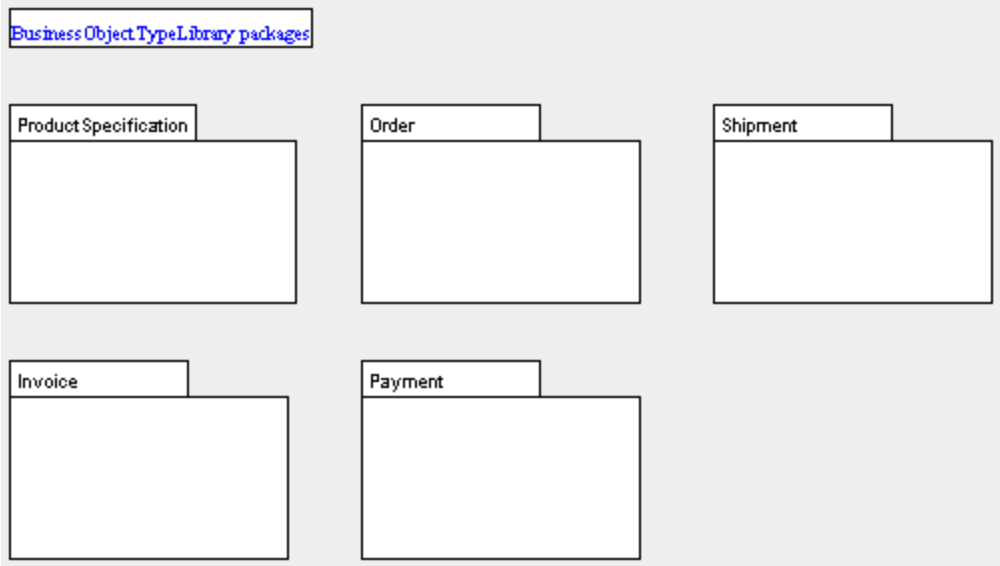
BOTL - eBTWG dependencies

The Business Object Type Library will be referenced by Common Business Process Library packages (conditions), by Business Collaboration Pattern packages (business semantics), and by Business Collaboration model packages (conditions, business semantics, and identification)



BOT packages

Business Object Type packages represent common business objects that have persistence, identity, state, and business semantic.

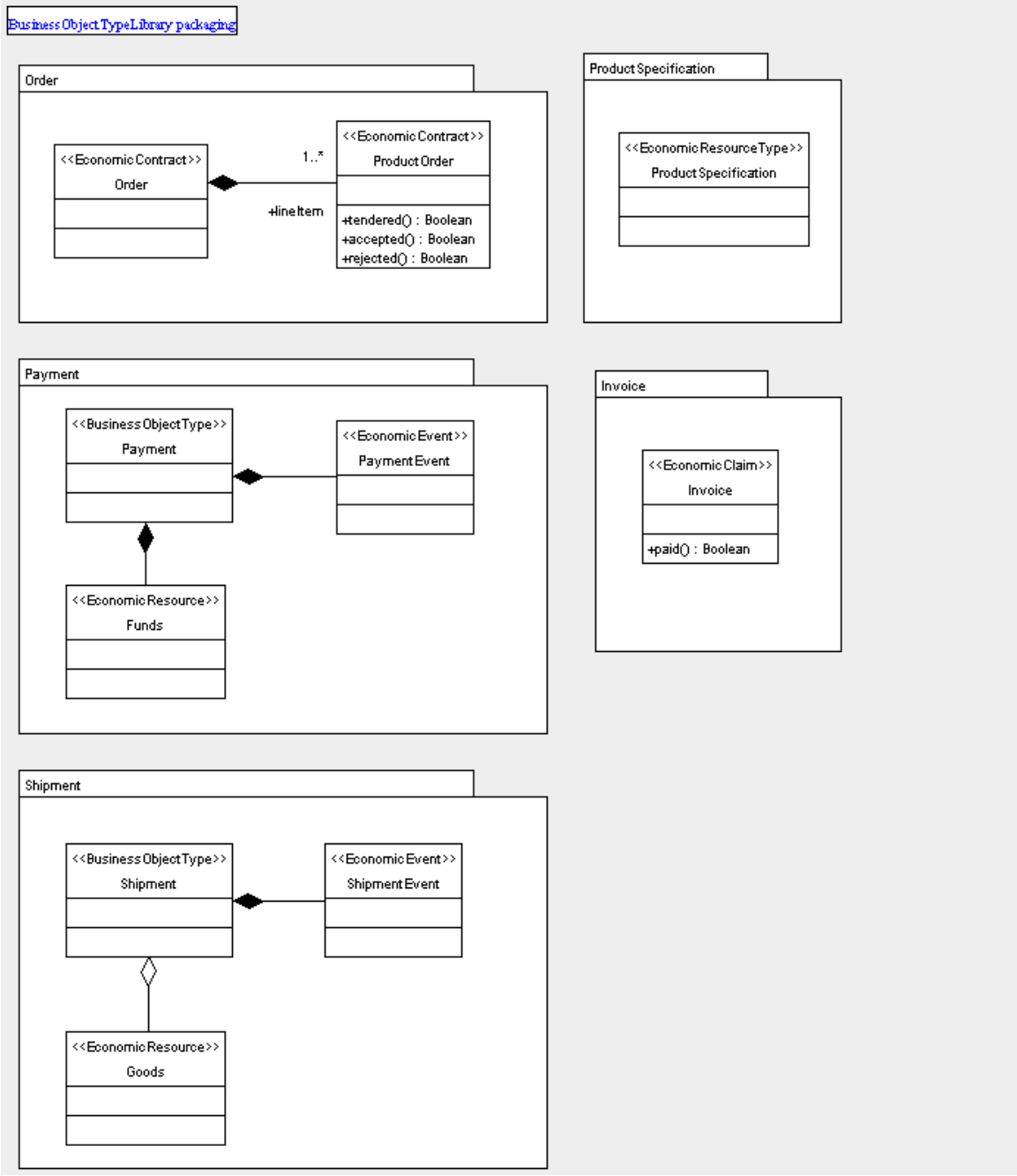


Business Object Types

Business Object Types are normalized common business objects. Often there will be packages of BOT where there are composite BOT within the BOT package. These composite BOT will often have their own identity, lifecycle, and business semantic despite the fact they are a compositional element of another BOT.

BOT business semantic

BOT will have business semantic. This business semantic will relate to the context of the business commitment within which the BOT provides that semantic. That semantic describes the role of the BOT in fulfilling the business commitment. The business semantics are expressed using REA economic element stereotypes.



BOT states accessible through Boolean methods on BOT class

Note the delivered() and paid() methods on the ProductShipment and ProductPayment classes. These methods return Boolean values with respect to the respective BOT state.

There are currently threads discussing how events and transitions express conditions related to state. These BOT states are made available using methods which return a Boolean...

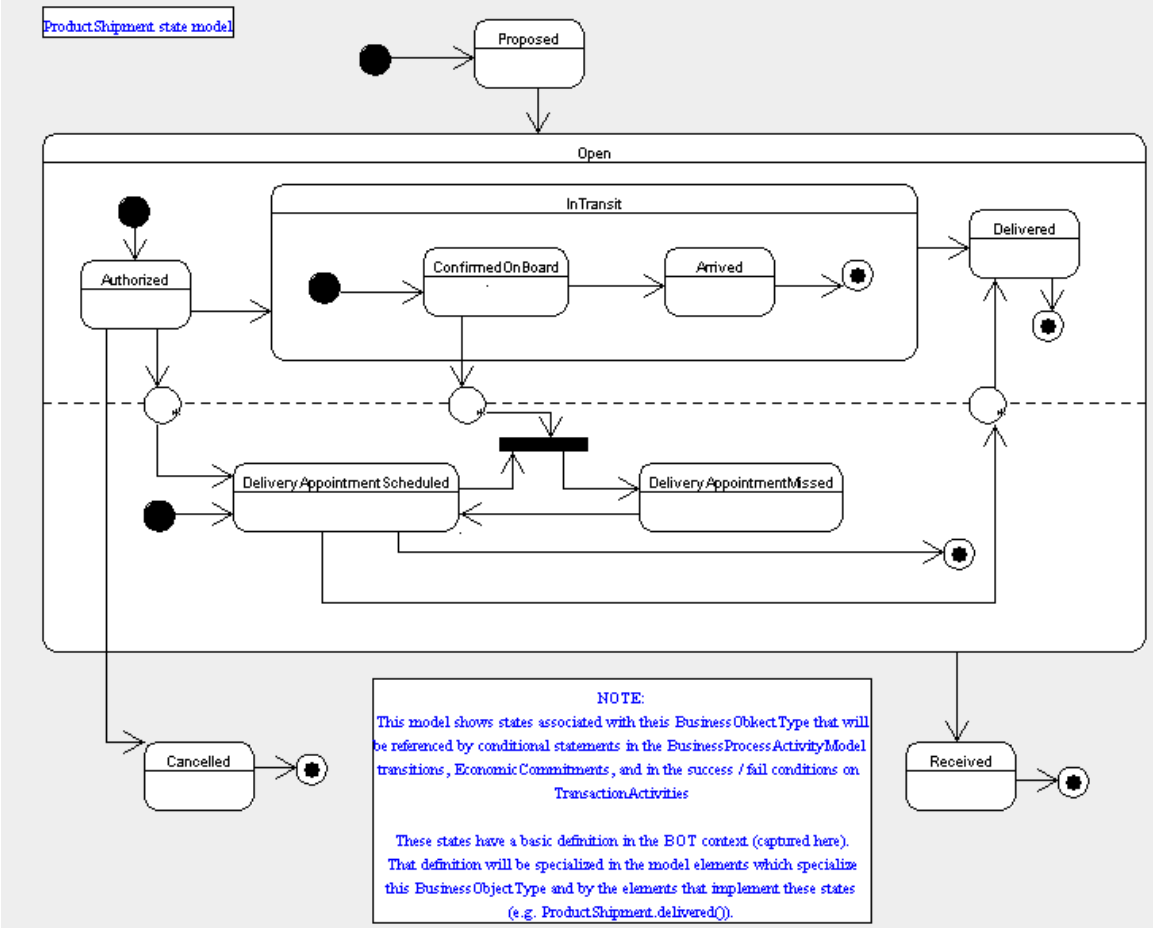
e.g. ProductOrder.ShortShip() : Boolean

Then in the models, the method is defined by an OCL statement:

Context ProductOrder
post:
ProductShipment.ConfirmedOnBoard() and
Sum(ProductShipment.ShipedQuantity) < ProductOrder.OrderQuantity

BOT lifecycle – state model

A Business Object Type will have a state model expressing the lifecycle of that BOT. Only states that are referenced in the collaborative domain are expressed in these models.



BOT state used as transition conditions

The defined state on a BOT are expressed as Boolean methods named for the state. The method returns true when the state is active. The following diagram uses object state methods as an expression of transition condition.