# Business Object Modeling with Collaboration Patterns

Jill Nicola

jill@streamlinedmodeling.com

## Introduction

Most people would protest if asked to give a presentation without knowing any details about the audience, setting, or available audio-visual equipment. Without a clear picture of the audience, setting, and equipment the presenter cannot logically decide the level, tone, length, and format of the presentation. He can only guess and hope that he guessed correctly.

Remarkably, we often put our software developers in the same position as that unlucky presenter by asking them to write software with only a fuzzy picture of the business processes and rules involved. Through a combination of guessing, trial and error, and prototypes for the customer, good developers eventually arrive at an understanding of the business domain, but rarely is it written down for new developers joining the team.

Business object modeling documents business rules and processes using object-oriented concepts and notations that most developers recognize and understand. The resulting business object model becomes the starting point for creating reusable and extensible frameworks, and helps transmit understanding of the business domain rapidly to new developers. This paper presents twelve (12) fundamental object patterns for building business object models. The patterns come from our book, *Streamlined Object Modeling,* © 2002, Prentice Hall. By themselves or in combinations with one another these twelve patterns can model all business domains.

## Patterns for Business Object Modeling

Patterns for business object modeling are not the same as design patterns which aim to increase reuse and framework pluggability. Business patterns, also known as analysis patterns, focus on creating an object model that clearly communicates the business requirements. Most well-built business object models can be read by a non-technical domain expert after a few minutes instruction in the notation. Using a business object model to get early customer verification instead of waiting until a working prototype is built saves significant time and rework. Once verified by the customer, the business object model may be refactored with design patterns to increase its reuse and pluggability; however the resulting model is more difficult for non-technical people to read.

The key to modeling business processes is not to focus on the steps of the process, which often are reordered and modified, but to instead focus on the people, places, things, and events involved in the process. At its core every business process involves some number of interactions of people with things at given places. Thus, business object modeling looks for four kinds of objects: people, places, things, and events, which describe details of interactions. By examining hundreds of past business object models, and giving some thought to how to best represent the real-world with object think, we discovered the best representations for modeling people, places, things, and events, and generalized these into twelve patterns.

## People

The important aspect of modeling people is to capture the different ways they can participate in the system. To do this first consider that every action by a person happens within a context. The context defines the information and rules necessary to ensure the only qualified people are participating with only permissible things them and at only available places. Since a person can participate in many different contexts, two kinds of objects are required to model a single person: (1) an **actor** object to model the permanent characteristics and behaviors of the person, and (2) a **role** object to model the person's participation in a context. The role object has the permissions, IDs, and passwords necessary to distinguish and track that person within a particular context, and the role has the history of interactions the person undertook within that context. The actor object has the characteristics and behaviors that are constant across all domains.

> **EXAMPLE—**An online auction site has sellers, who post things for sale in auctions, and bidders, who bid in auctions. A single person can be both a seller and a bidder, but slightly different information is required for each activity, and the history of interactions differs. Use separate role objects to model the seller and bidder , and an actor object to model the information about the person common to both roles. (See Figure 1.)

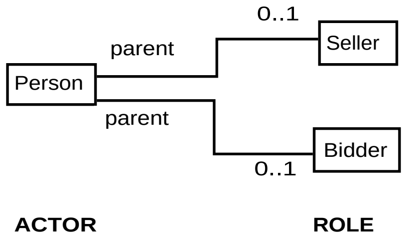

Figure 1. People can participate two ways in an online auction.

## Places

In business object modeling places are locations where events happen. With simple systems it is tempting to ignore places because all events may happen at the same location, but that is risky should the system may expand to multiple locations later. Often the position of a place is local to some larger region, and events are rolled up for statistical measuring. For these reasons, places are modeled hierarchical, with the lowest place being the location of some relevant event. We model places with two kinds of objects: (1) a **place** object that is the location of an event, and (2) an **outer place** that is the container of a place. In some cases where the level of nesting is many levels deep, an outer place may also act like a place and be the location of an event.

> **EXAMPLE—**A manufacturer has multiple warehouses that receive deliveries in designated loading areas. A delivery is made up of delivery loads, and each delivery load is deposited in a loading bin with its loading area. We model the manufacturing warehouse as an outer place object

> containing multiple load areas where deliveries occur. A loading area is both a place and outer place object because it contains loading bins where delivery loads are deposited. (See Figure 2.)

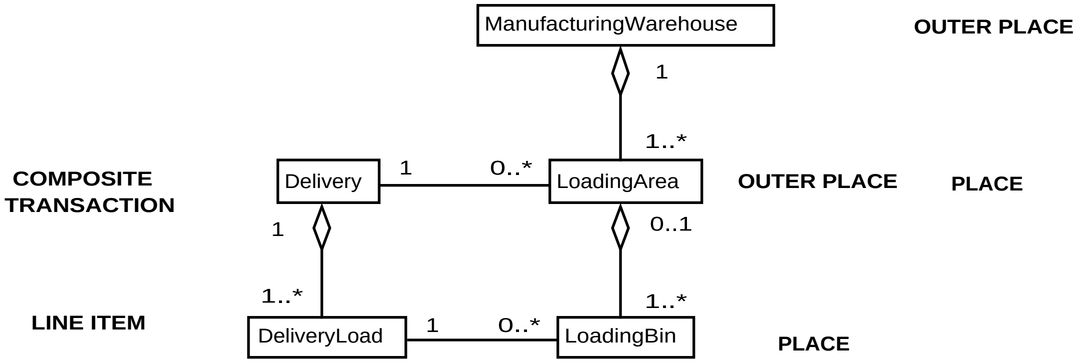

Figure 2. A warehouse contains several levels of places.

## Things

Things are the subjects of interactions; they are the entities being acted on by a person or other thing. Like people, a single thing usually requires two objects to describe it: (1) an **item** object to capture characteristics of it shared by other things, and (2) a **specific item** object to capture details of the thing particular only to itself. The item object defines a set of things, and the specific item is one element in that set.

> **EXAMPLE**—A video title describes a set of videotapes available for renting. Each video title object has information about the movie and the movie's classification within the video store catalog. Each video tape object has a unique identifier and remembers the history of its rental transactions. (See Figure 3.)

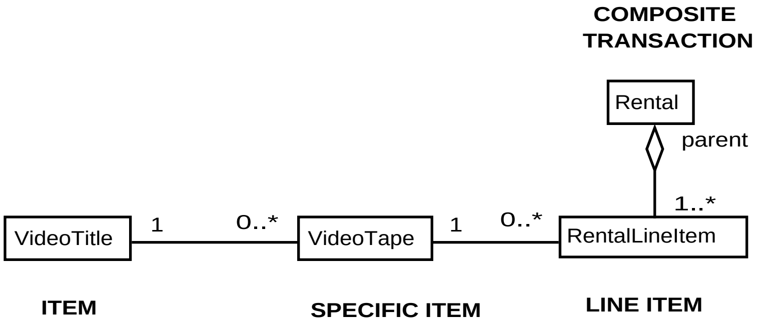

Figure 3. A video title is a generic description for a particular video tape.

The item and specific item objects model the generic and specific descriptions of a single thing. Things also have aggregate descriptions, of which three are relevant to business object modeling: (1) an ensemble of things, (2) a receptacle of things, and (3) a classification of things. Each aggregate description involves two objects, a whole and a part. An ensemble is modeled as an assembly object containing one or more parts. An assembly must have at least one part to exist and a part can only belong to one assembly at a time. A receptacle is modeled as a container object containing zero to many content objects. Unlike an assembly, a container can exist without anything inside it, but a content can only be in one container. A classification is modeled as a group object containing zero to many member objects. Unlike parts and contents, a member object can be in multiple groups.

> **EXAMPLE**—An e-commerce site has a catalog of products where each product can be classified in multiple categories, and each category can be classified within larger categories. A delivery truck containers pallets, and each pallet contains some number of cases. A workstation is assembled out of components, and some components are assembled out of smaller components. (See Figure 4.)

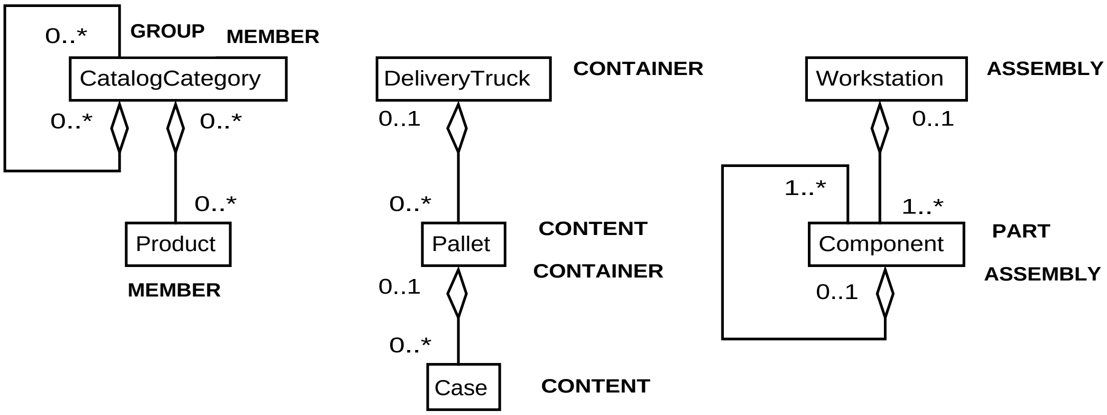

Figure 4. Different examples of aggregate things.

## Events

An event describes an interaction between people, places and things. An interaction of people with only one thing is modeled as a **transaction** object, and an interaction with multiple things is modeled with a **composite transaction** object. Both kinds of transactions record details about the interaction, such as date, time, and current status. Since transactions happen within contexts, a role object is used to record a person's involvement in a transaction. Because multiple things are involved in its interaction, a composite transaction contains **line items** to track individual interaction details for each thing involved. Often transactions are followed by later subsequent interactions involving some of the same people and things. Subsequent interactions are modeled with follow-up **transactions**.

> **EXAMPLE**—A commodity buyer submits a price inquiry for a particular quantity of commodity and gets back a price quote from a broker. (See Figure 5.)

> **EXAMPLE**—An online auction by an auction house includes multiple auction lots per auction. The auction lots include the number of the lot an information such as minimum bid and payment details demanded by the seller. (See Figure 6.)

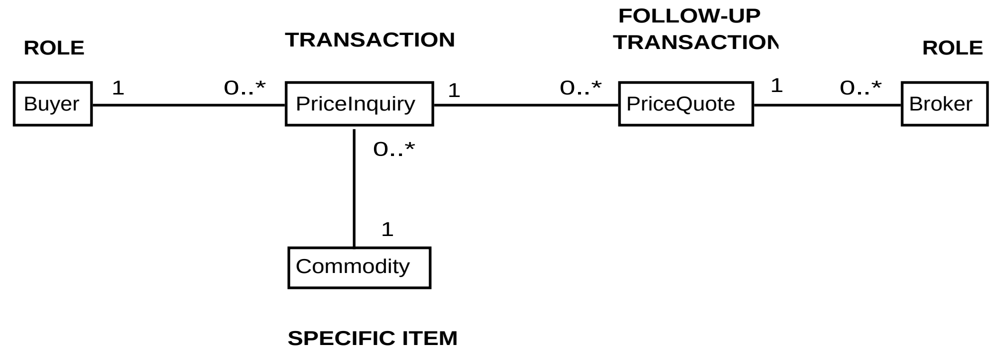

Figure 5. A broker returns a price quote for a buyer's price inquiry.

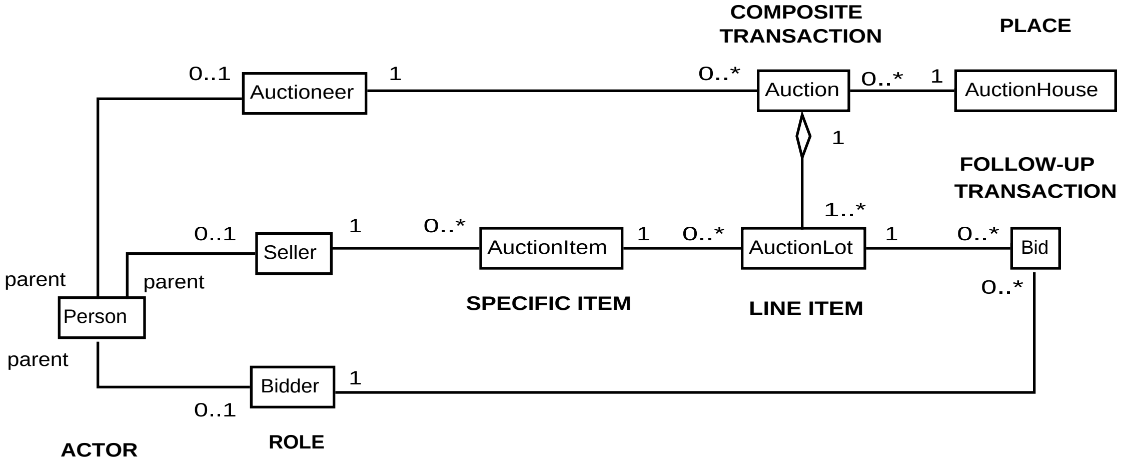

Figure 6. An auction house auctions items in lots.

## Collaboration Patterns

Summarizing the four categories of business objects into twelve object patterns of two objects each. (See Figure 7.) That two objects were required to model each people, place, things, and event object is not entirely coincidental. We have discovered that most business rules regulate whether a person can participate in a context, interact with other people and things at given places, and determine which things can belong in particular assemblies, groups, and containers. What this means for our patterns, is that most business rules can be expressed as pre-conditions for deciding whether two objects can form a collaboration or whether two objects can dissolve their collaboration.

Because these twelve patterns describe collaborations involved in typical business processes patterns they are called collaboration patterns. Collaboration patterns are ideal patterns for a business rules methodology which expresses business rules as constraints on whether two objects can collaborate or cease collaborating. The parallelism of the business rules and the patterns facilitates the development of strategies for mapping business rules to the patterns. These strategies then apply to all object models built with the collaboration patterns. Our book, *Streamlined Object Modeling*, develops such a rules methodology in rigorous detail, including a framework for checking rules and coding templates for implementing them.

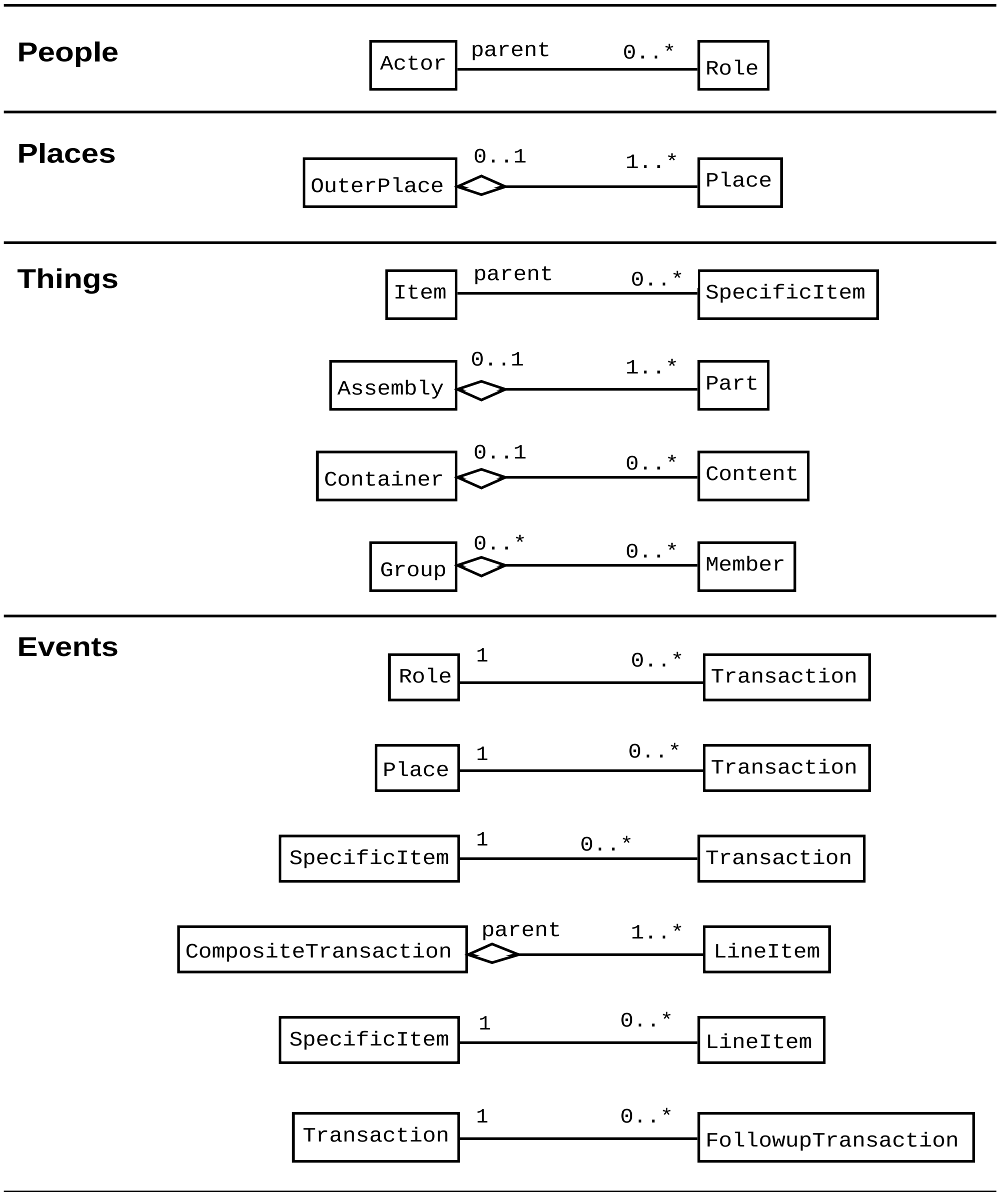

Figure 7. The twelve collaboration patterns.

## Conclusion

Business object modeling uses object concepts and notations to document the business processes and rules underlying an application. The key to modeling business processes is to focus on people, places, things, and events involved. We can model these business objects with twelve collaboration patterns. These patterns are ideal because they easily support business rules which regulate whether objects can collaborate we model people, places, things, and events.

## References

(1) Streamlined Object Modeling, Jill Nicola, Mark Mayfield, and Mike Abney, 2002, Prentice Hall.

(2) Object Models: Strategies, Patterns, & Applications (2nd ed.), Peter Coad, David North, and Mark Mayfield, 1997, Prentice Hall.