UML Extensions 4

This chapter includes the *UML Extension for Software Development Processes* and the *UML Extension for Business Modeling*.

Contents

This chapter contains the following topics.

Part 1 - UML Extension for Software Development Processes

4.1 Overview

User-defined extensions of the UML are enabled through the use of stereotypes, tagged values, and constraints. Two extensions are defined currently: 1) Software Development Processes and 2) Business Modeling.

The UML is broadly applicable without extension, so companies and projects should define extensions only when they find it necessary to introduce new notation and terminology. Extensions will not be as universally understood, supported, and agreed upon as the UML itself. In order to reduce potential confusion around vendor implementation, the following terms are defined:

- UML Variant - a language with well-defined semantics that is built on top of the UML metamodel, as a metamodel. It specializes the UML metamodel, without changing any of the UML semantics or redefining any of its terms. For example, it could reintroduce a class called State.
- UML Extension - a predefined set of Stereotypes, TaggedValues, Constraints, and notation icons that collectively extend and tailor the UML for a specific domain or process.

4.2 Introduction

This section defines the UML Extension for Software Development Processes, defined in terms of the UML's extension mechanisms, namely Stereotypes, TaggedValues, and Constraints.

See the UML Semantics chapter for a full description of the UML extension mechanisms.

This chapter describes a UML extension that has been found useful in software development processes. Although this extension was based on the Objectory process, it is general-purpose and may also be applied to other software development processes. It is not meant to be a comprehensive description of all software development processes, but is an example of one such process.

4.3 Summary of Extension

Table 4-1 Stereotypes

Metamodel Class	Stereotype Name
Model	use case model
Model	analysis model
Model	design model
Model	implementation model

Table 4-1 Stereotypes

Package	use case system
Subsystem	analysis system
Subsystem	design system
Package	implementation system
Subsystem	analysis subsystem
Subsystem	design system
Package	implementation system
Package	use case package
Subsystem	analysis service package
Subsystem	design service package
Class	boundary
Class	entity
Class	control
Association	communicates
Association	subscribes
Collaboration	use case realization

4.3.1 TaggedValues

Currently, this extension does not introduce any new TaggedValues.

4.3.2 Constraints

Currently, this extension does not introduce any new Constraints, other than those associated with the well-formedness semantics of the stereotypes introduced.

4.3.3 Prerequisite Extensions

This extension requires no other extensions to the UML for its definition.

4.4 Stereotypes and Notation

4.4.1 Model, Package, and Subsystem Stereotypes

A software development process comprises several different, but related models. Software engineering process models are characterized by the lifecycle stage that they represent. The different models are stereotypes of model:

- Use Case

- Analysis
- Design
- Implementation

Use Case

A Use Case Model is a model in which the top-level package is a use case system.

A Use Case System is a top-level package. A use case system contains use case packages and/or use cases and/or actors and relationships.

A Use Case Package is a package containing use cases and actors with relationships. A use case is not partitioned over several use case packages.

Analysis

An Analysis Model is a model whose top-level package is an analysis system.

An Analysis System is a top-level subsystem. An analysis system contains analysis subsystems, and/or analysis service packages, and/or analysis classes (i.e., entity, boundary, and control), and relationships.

An Analysis Subsystem is a subsystem containing other analysis subsystems, analysis service packages, analysis classes (i.e., entity, boundary, and control), and relationships.

An Analysis Service Package is a subsystem containing analysis classes (i.e., entity, boundary, and control) and relationships.

Design

A Design Model is a model whose top-level package is a design system.

A Design System is a top-level subsystem. A design system contains design subsystems, and/or design service packages, and/or design classes, and relationships.

A Design Subsystem is a subsystem containing other design subsystems, design service packages, design classes, and relationships.

A Design Service Package is a subsystem containing design classes and relationships.

Implementation

An Implementation Model is a model whose top-level package is an implementation system.

An Implementation System is a top-level package. An implementation system contains implementation subsystems, and/or components, and relationships.

An Implementation Subsystem is a package containing implementation subsystems, and/or components, and relationships.

Notation

Package stereotypes are indicated with stereotype keywords in guillemets («stereotype name»). There are no stereotyped icons for packages.

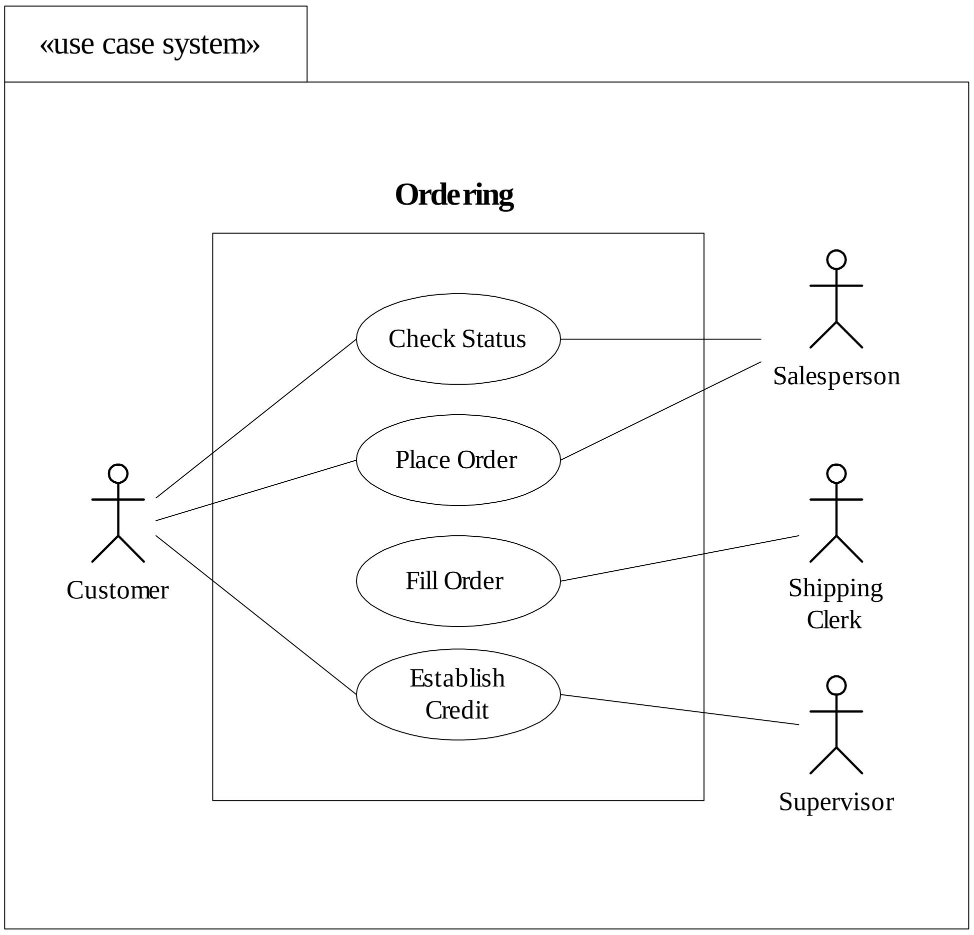

Figure 4-1 Objectory Packages

4.4.2 Class Stereotypes

Analysis classes come in the following three kinds: 1) entity, 2) control, and 3) boundary. Design classes are not stereotyped in the process.

Entity

Entity is a class that is passive; that is, it does not initiate interactions on its own. An entity object may participate in many different use case realizations and usually outlives any single interaction.

Control

Control is a class, an object of which denotes an entity that controls interactions between a collection of objects. A control class usually has behavior specific for one use case and a control object usually does not outlive the use case realizations in which it participates.

Boundary

A Boundary is a class that lies on the periphery of a system, but within it. It interacts with actors outside the system as well as objects of all three kinds of analysis classes within the system.

Notation

Class stereotypes can be shown with keywords in guillemets. They can also be shown with the following special icons.

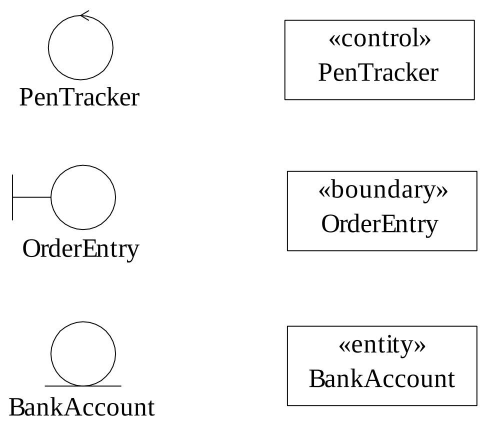

Figure 4-2 Class Stereotypes

4.4.3 Association Stereotypes

The following are special associations between classes.

Communicates

Communicates is an association between actors and use cases denoting that the actor sends messages to the use case and/or the use case sends messages to the actor. This may be one-way or two-way navigation. The direction of communication is indicated by the navigability of the association.

Subscribes

Subscribes is an association whose source is a class (called the subscriber) and whose target is a class (called the publisher). The subscriber specifies a set of events. The subscriber is notified when one of those events occurs in the target.

Notation

Association stereotypes are indicated by keywords in guillemets. There are no special stereotype icons. The stereotype «communicates» on Actor-Use Case associations may be omitted, since it is the only kind of relationships between actors and use cases.

4.5 Well-Formedness Rules

Stereotyped model elements are subject to certain constraints, in addition to the constraints imposed on all elements of their kind.

4.5.1 Generalization

All the modeling elements in a generalization must be of the same stereotype.

4.5.2 Association

Apart from standard UML combinations, the following combinations are allowed for each stereotype.

Table 4-2 Valid Association Stereotype Combinations

To: **From::**	**actor**	**boundary**	**entity**	**control**
actor		communicates		
boundary	communicates	communicates	communicates subscribes	communicates
entity			communicates subscribes	
control		communicates	communicates subscribes	communicates

Part 2 - UML Extension for Business Modeling

4.6 Introduction

The UML Extension for Business Modeling is defined in terms of the UML's extension mechanisms, namely Stereotypes, TaggedValues, and Constraints. See the UML Semantics chapter for a full description of the UML extension mechanisms.

This section describes stereotypes that can be used to tailor the use of UML for business modeling. All of the UML concepts can be used for business modeling, but providing business stereotypes for some common situations provides a common terminology for this domain. Note that UML can be used to model different kinds of systems (software systems, hardware systems, and real-world organizations). Business modeling models real-world organizations.

This section is not meant to be a complete definition of business modeling concepts and how to apply them, but it serves the purpose of registering this extension, including its icons.

4.7 Summary of Extension

4.7.1 Stereotypes

Table 4-3 Metamodel Class Stereotypes

Metamodel Class	Stereotype Name
Model	use case model
Package	use case system
Package	use case package
Model	object model
Subsystem	object system
Subsystem	organization unit
Subsystem	work unit
Class	worker
Class	case worker
Class	internal worker
Class	entity
Collaboration	use case realization
Association	subscribes

4.7.2 Tagged Values

This extension does not currently introduce any new TaggedValues.

4.7.3 Constraints

This extension does not currently introduce any new Constraints, other than those associated with the well-formedness semantics of the stereotypes introduced.

4.7.4 Prerequisite Extensions

This extension requires no other extensions to the UML for its definition.

4.8 Stereotypes and Notation

4.8.1 Model, Package, and Subsystem Stereotypes

A business system comprises several different, but related models. The models are characterized by being exterior or interior to the business system they represent. Exterior models are use case models and interior models are object models. A large business system may be partitioned into subordinate business systems. The following are the model stereotypes.

Use Case

A Use Case Model is a model that describes the business processes of a business and their interactions with external parties such as customers and partners.

A use case model describes:

- the businesses modeled as use cases.
- parties exterior to the business (e.g., customers and other businesses) modeled as actors.
- the relationships between the external parties and the business processes.

A Use Case System is the top-level package in a use case model. A use case system contains use case packages, use cases, actors, and relationships.

A Use Case Package is a package containing use cases and actors with relationships. A use case is not partitioned over several use case packages.

Object

An Object Model is a model in which the top-level package is an object system. These models describe the things interior to the business system itself.

An Object System is the top-level subsystem in an object model. An object system contains organization units, classes (workers, work units, and entities), and relationships.

Organization Unit

Organization Unit is a subsystem corresponding to an organization unit of the actual business. An organization unit subsystem contains organization units, work units, classes (workers and entities), and relationships.

Work Unit

A Work Unit is a subsystem that contains one or more entities.

A work unit is a task-oriented set of objects that form a recognizable whole to the end user. It may have a facade defining the view of the work unit's entities relevant to the task.

Notation

Package stereotypes are indicated with stereotype keywords in guillemets («stereotype name»). There are no special stereotyped icons for packages.

4.8.2 Class Stereotypes

Business objects come in the following kinds:

- actor (defined in the UML)
- worker
- case worker
- internal worker
- entity

Worker

A Worker is a class that represents an abstraction of a human that acts within the system. A worker interacts with other workers and manipulates entities while participating in use case realizations.

Case Worker

A Case Worker is a worker who interacts directly with actors outside the system.

Internal Worker

An Internal Worker is a worker that interacts with other workers and entities inside the system.

Entity

An Entity is a class that is passive; that is, it does not initiate interactions on its own. An entity object may participate in many different use case realizations and usually outlives any single interaction. In business modeling, entities represent objects that workers access, inspect, manipulate, produce, and so on. Entity objects provide the basis for sharing among workers participating in different use case realizations.

Notation

Class stereotypes can be shown with keywords in guillemets within the normal class symbol. They can also be shown with the following special icons.

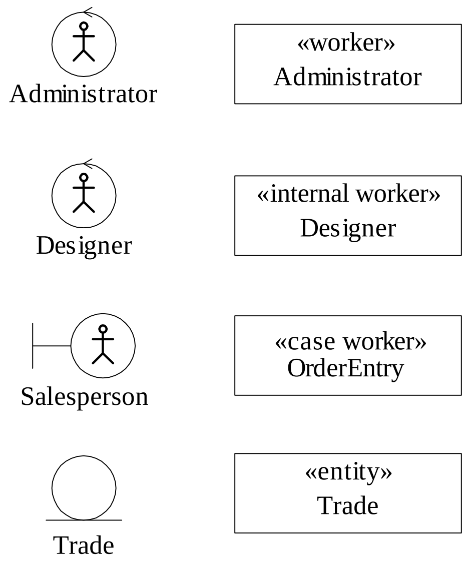

Figure 4-3 Class Stereotypes

The preceding icons represent common concepts useful in most business models.

Example of Alternate Notations

Tools and users are free to add additional icons to represent more specific concepts. Examples of such icons include icons for documents and actions, as shown in Figure 4-4.

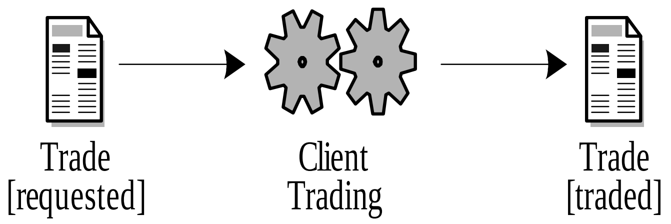

Figure 4-4 Example of Special Icons for Entities and Actions

In this example, "Trade [requested]" and "Trade [traded]" represent an entity in two states, where the Trade is the dominant entity of a Trade Document work unit. Client Trading is an action. The icons are designed to be meaningful in the particular problem domain.

4.8.3 Association Stereotypes

The following are special business modeling associations between classes:

Communicates

Communicates is an association used by two instances to interact. This may be one-way or two-way navigation. The direction of communication is the same as the navigability of the association.

Subscribes

Subscribes is an association whose source is a class (called the subscriber) and whose target is a class (called the publisher). The subscriber specifies a set of events. The subscriber is notified when one of those events occurs in the target.

Notation

Association stereotypes are indicated by keywords in guillemets. There are no special stereotype icons.

4.9 Well-Formedness Rules

Stereotyped model elements are subject to certain constraints in addition to the constraints imposed on all elements of their kind.

4.9.1 Generalization

All the modeling elements in a generalization must be of the same stereotype.

4.9.2 Association

Apart from standard UML combinations, the following combinations are allowed for each stereotype.

Table 4-4 Valid Association Stereotype Combinations

To: **From:**	**actor**	**case worker**	**entity**	**work unit**	**internal worker**
actor		communicates		communicatessubscribes	
case worker	communicates	communicates	communicatessubscribes	communicatessubscribes	communicates
entity			communicatessubscribes	communicates	
work unit	communicates	communicates	communicatessubscribes	communicatessubscribes	communicates
internal worker		communicates	communicatessubscribes	communicatessubscribes	communicates