The Unified Modeling Language Reference Manual

Second Edition

James Rumbaugh
Ivar Jacobson
Grady Booch

Addison-Wesley

Boston • San Francisco • New York • Toronto • Montreal
London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Contents

Preface . . . . . **xiii**

Part 1: Background

Chapter 1: UML Overview **3**

Brief Summary of UML *3*
UML History *4*
Goals of UML *10*
Complexity of UML *10*
UML Assessment *12*
UML Concept Areas *12*

Chapter 2: The Nature and Purpose of Models **15**

What Is a Model? *15*
What Are Models For? *15*
Levels of Models *17*
What Is in a Model? *19*
What Does a Model Mean? *21*

Part 2: UML Concepts

Chapter 3: UML Walkthrough **25**

UML Views *25*
Static View *28*
Design Views *29*
Use Case View *34*
State Machine View *35*
Activity View *37*
Interaction View *37*
Deployment View *40*
Model Management View *42*
Profiles *43*

Chapter 4: Static View **47**
Overview *47*
Classifier *48*
Relationships *52*
Association *53*
Generalization *57*
Realization *61*
Dependency *62*
Constraint *65*
Instance *66*

Chapter 5: Design View **69**
Overview *69*
Structured Classifier *70*
Collaboration *71*
Patterns *73*
Component *73*

Chapter 6: Use Case View **77**
Overview *77*
Actor *77*
Use Case *78*

Chapter 7: State Machine View **81**
Overview *81*
State Machine *81*
Event *82*
State *84*
Transition *85*
Composite State *89*

Chapter 8: Activity View **95**
Overview *95*
Activity *96*
Activities and Other Views *98*
Action *98*

Chapter 9: Interaction View **101**
Overview *101*
Interaction *101*
Sequence Diagram *102*
Communication Diagram *106*

Chapter 10: Deployment View **109**
Overview *109*
Node *109*
Artifact *109*

Chapter 11: Model Management View **111**
Overview *111*
Package *111*
Dependencies on Packages *112*
Visibility *113*
Import *113*
Model *114*

Chapter 12: Profiles **115**
Overview *115*
Stereotype *116*
Tagged Value *117*
Profile *118*

Chapter 13: UML Environment **121**
Overview *121*
Semantics Responsibilities *121*
Notation Responsibilities *122*
Programming Language Responsibilities *123*
Modeling with Tools *124*

Part 3: Reference

Chapter 14: Dictionary of Terms **129**

Part 4: Appendices

Appendix A: UML Metamodel **685**

Appendix B: Notation Summary **689**

Bibliography **703**

Index **707**

Chapter 3: UML Walkthrough 25

UML Views 25
Static View 28
Design Views 29
Use Case View 34
State Machine View 35
Activity View 37
Interaction View 37
Deployment View 40
Model Management View 42
Profiles 43

Table 3-1: *UML Views and Diagrams*

Major Area	*View*	*Diagram*	*Main Concepts*
structural	static view	class diagram	association, class, dependency, generalization, interface, realization
	design view	internal structure	connector, interface, part, port, provided interface, role, required interface
		collaboration diagram	connector, collaboration, collaboration use, role
		component diagram	component, dependency, port, provided interface, realization, required interface, subsystem
	use case view	use case diagram	actor, association, extend, include, use case, use case generalization

Table 3-1: *UML Views and Diagrams (continued)*

Major Area	*View*	*Diagram*	*Main Concepts*
dynamic	state machine view	state machine diagram	completion transition, do activity, effect, event, region, state, transition, trigger
	activity view	activity diagram	action, activity, control flow, control node, data flow, exception, expansion region, fork, join, object node, pin
	interaction view	sequence diagram	occurrence specification, execution specification, interaction, interaction fragment, interaction operand, lifeline, message, signal
		communication diagram	collaboration, guard condition, message, role, sequence number

physical	deployment view	deployment diagram	artifact, dependency, manifestation, node
model manage-ment	model management view	package diagram	import, model, package
	profile	package diagram	constraint, profile, stereotype, tagged value

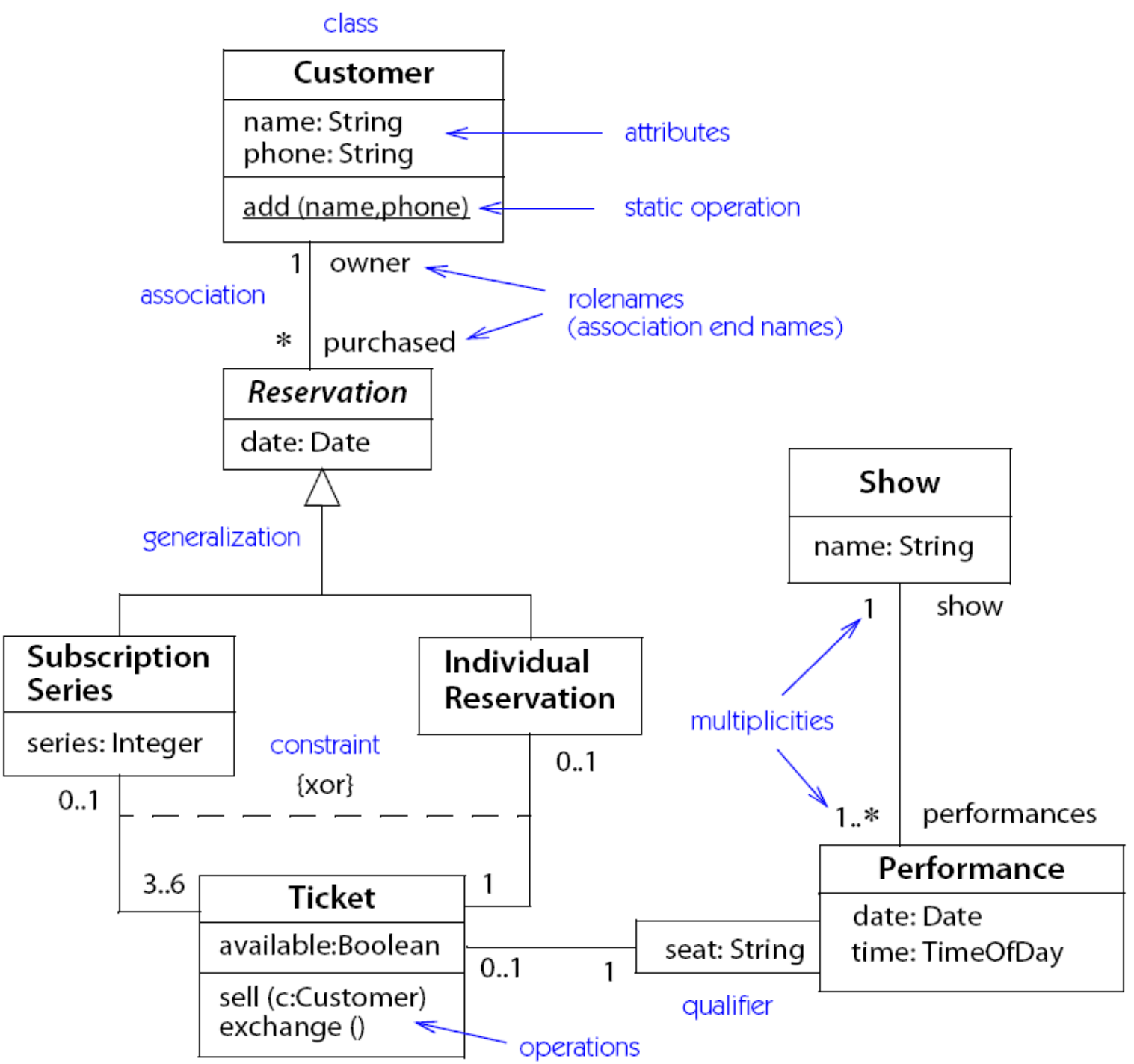


Figure 3-1. *Class diagram*

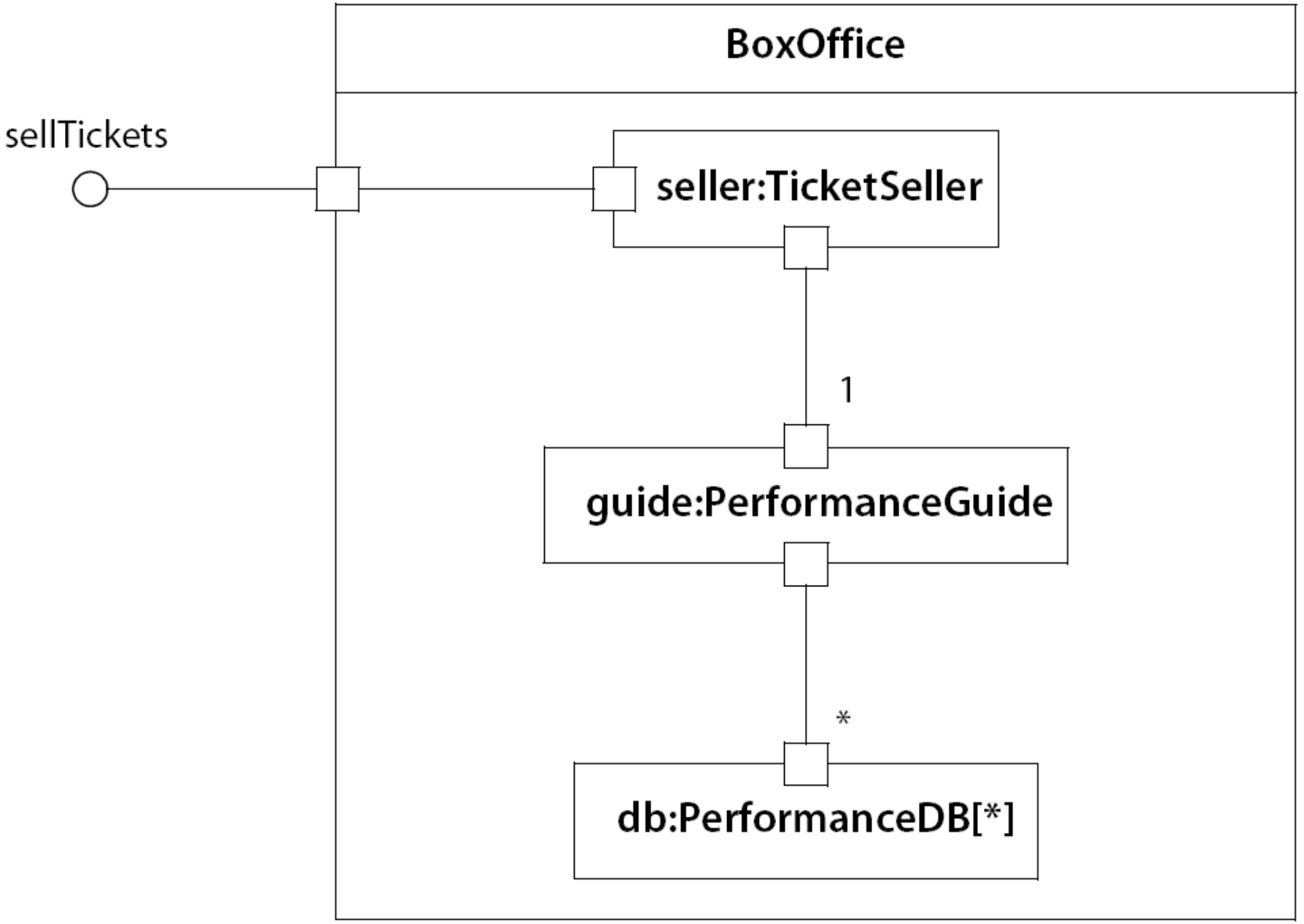


Figure 3-2. *Internal structure diagram*

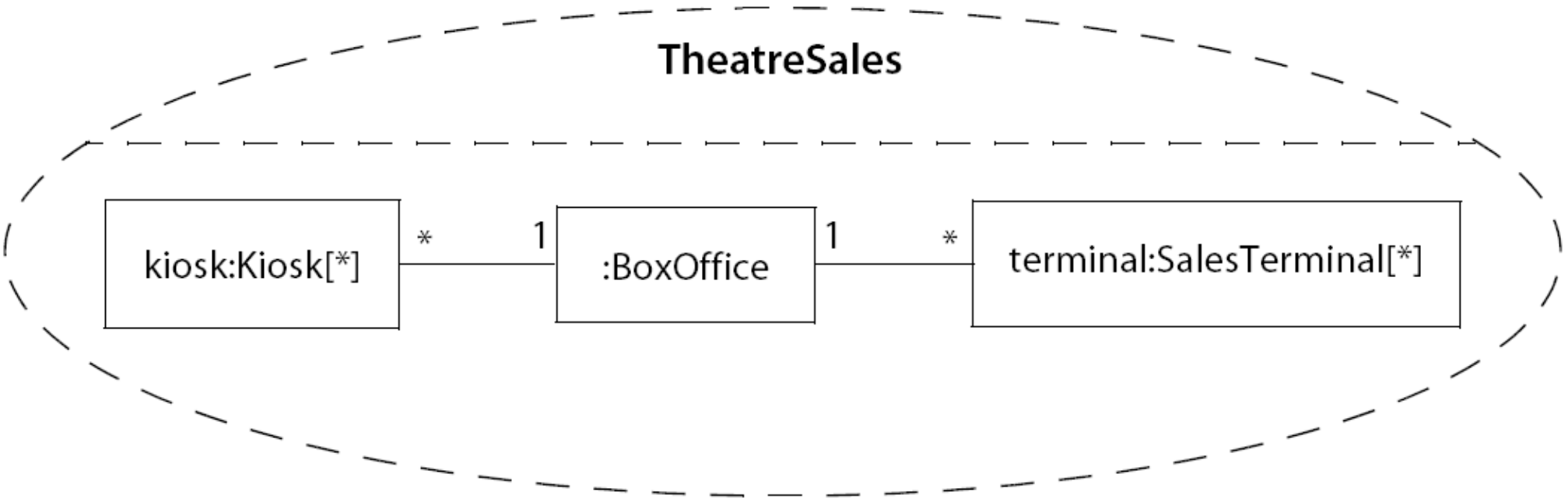


Figure 3-3. *Collaboration diagram*

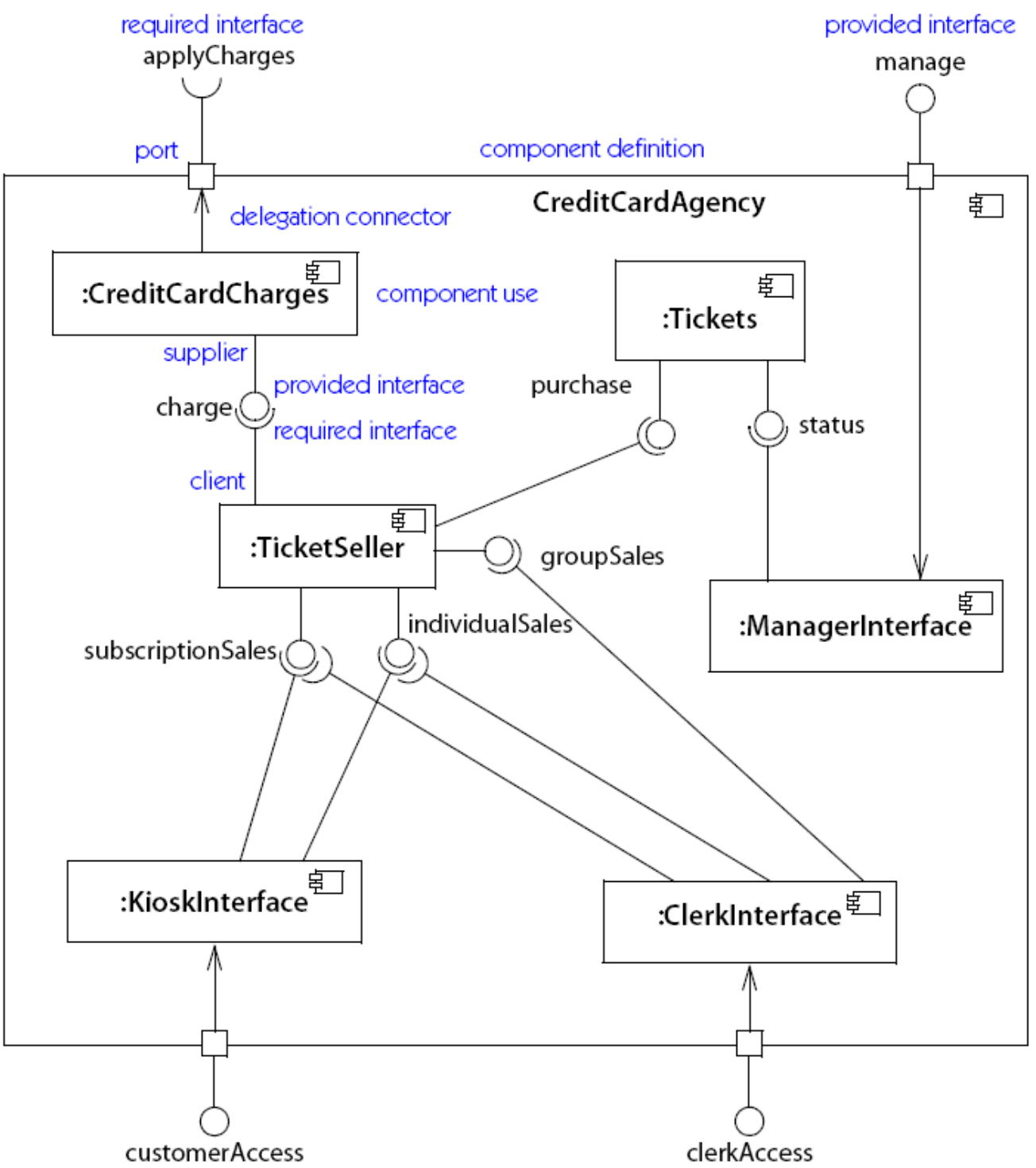


Figure 3-4. *Component definition*

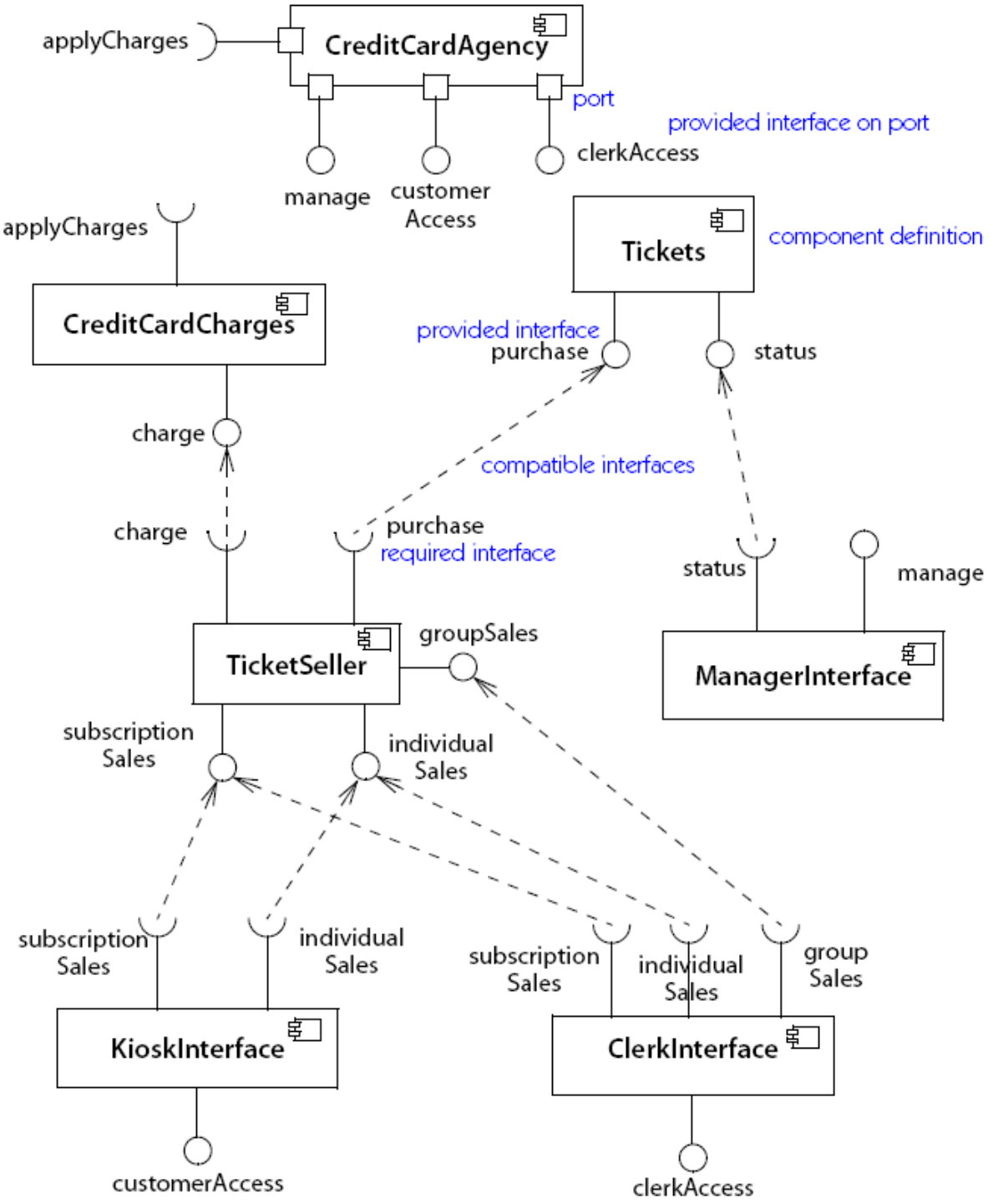


Figure 3-5. *Component diagram*

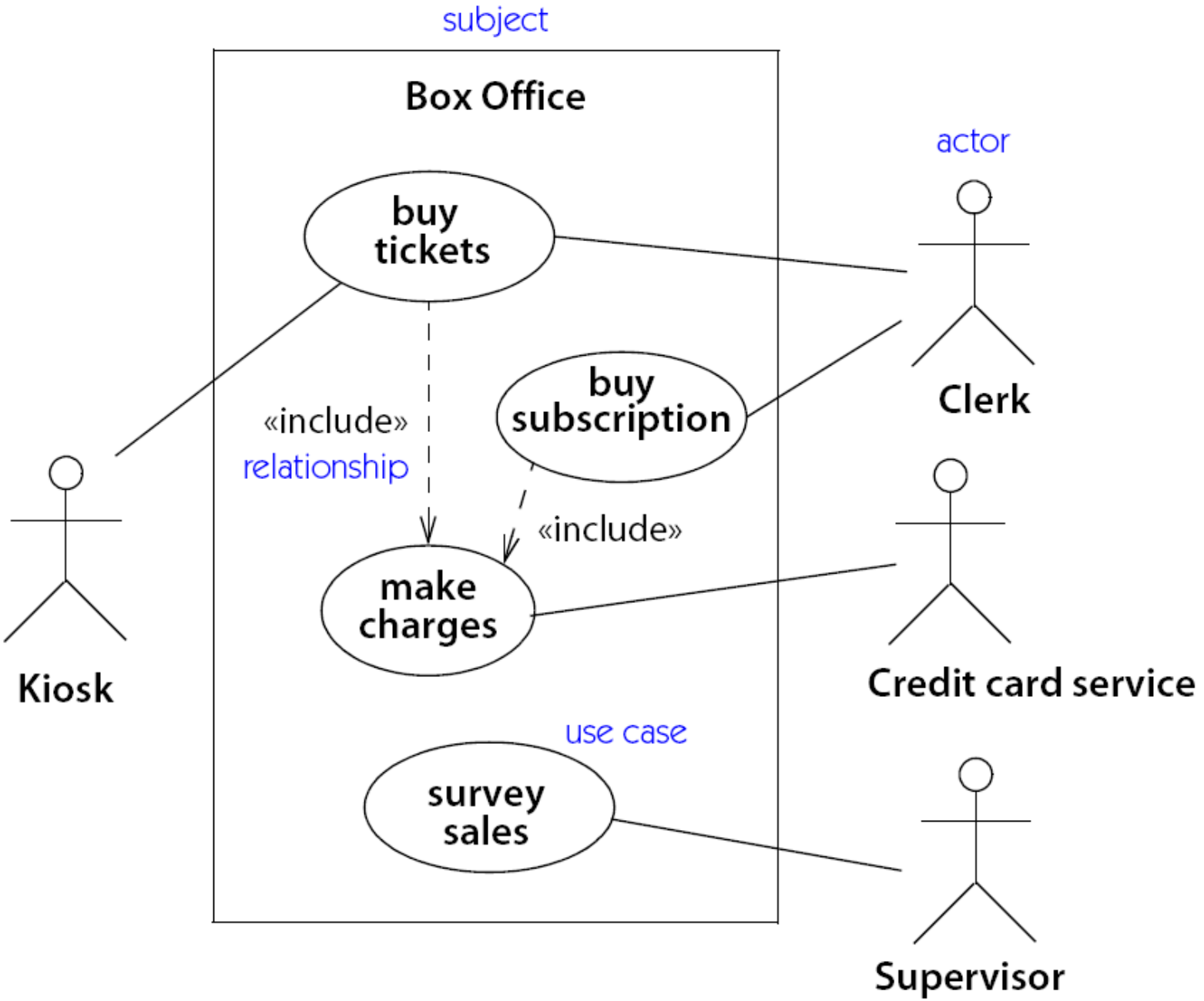


Figure 3-6. *Use case diagram*

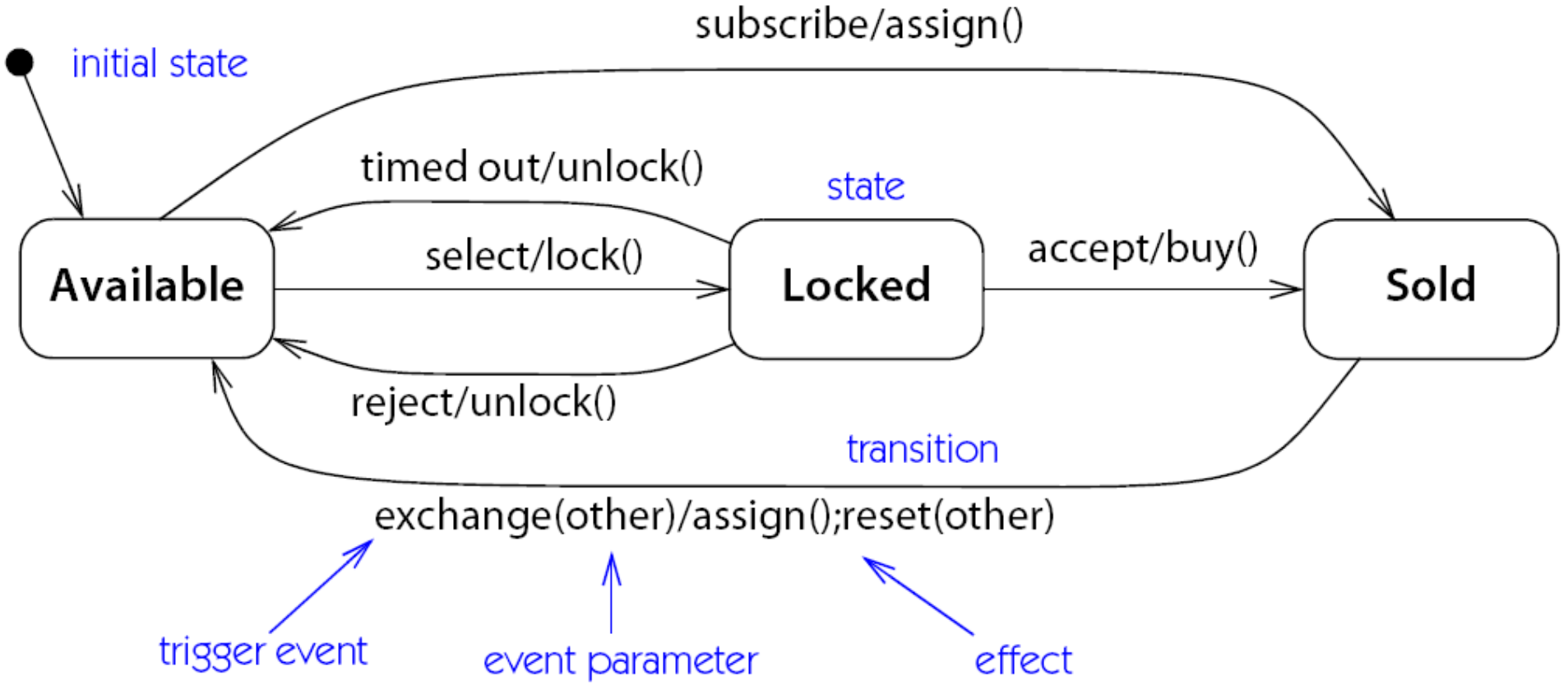


Figure 3-7. *State machine diagram*

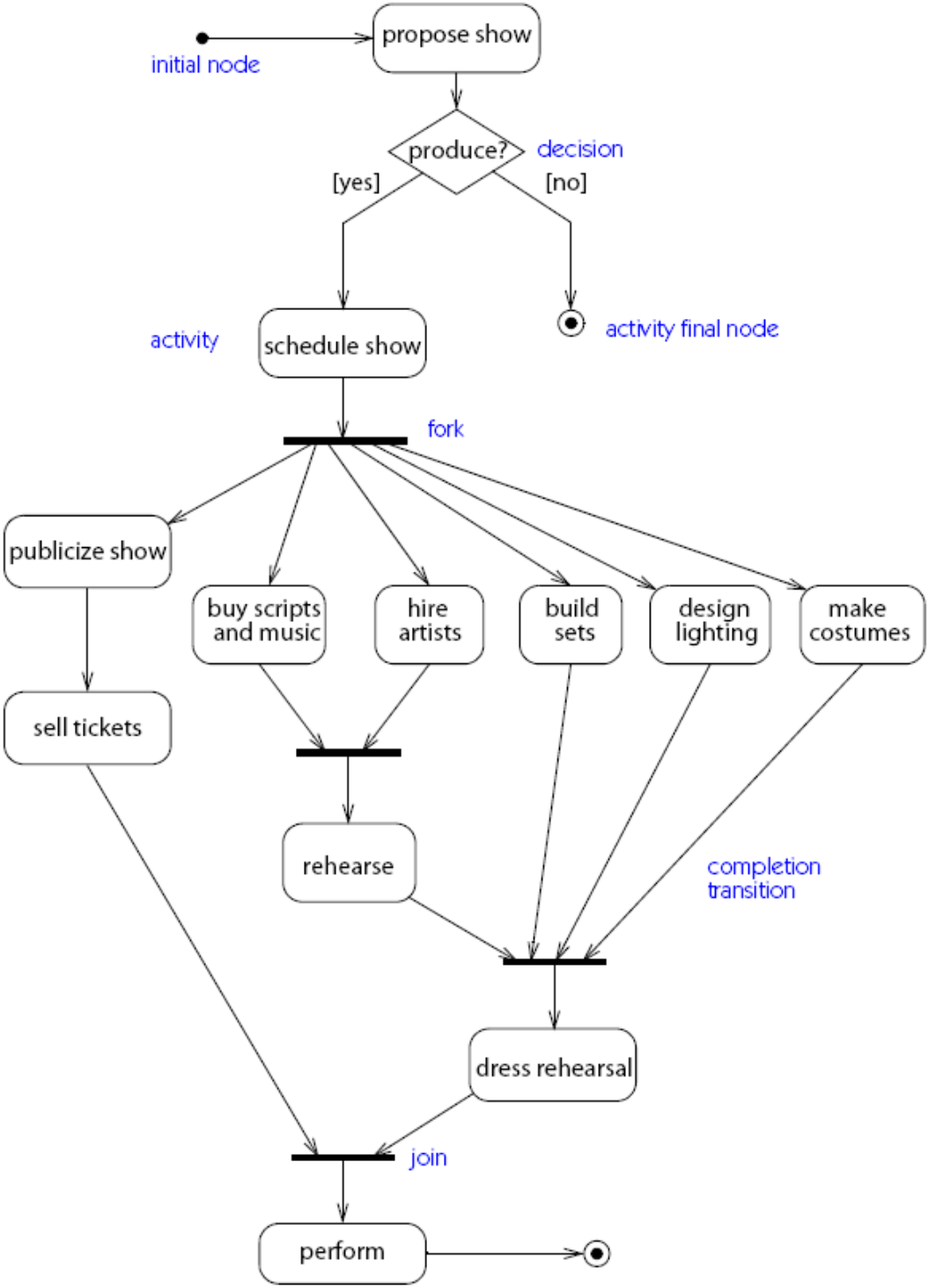


Figure 3-8. *Activity diagram*

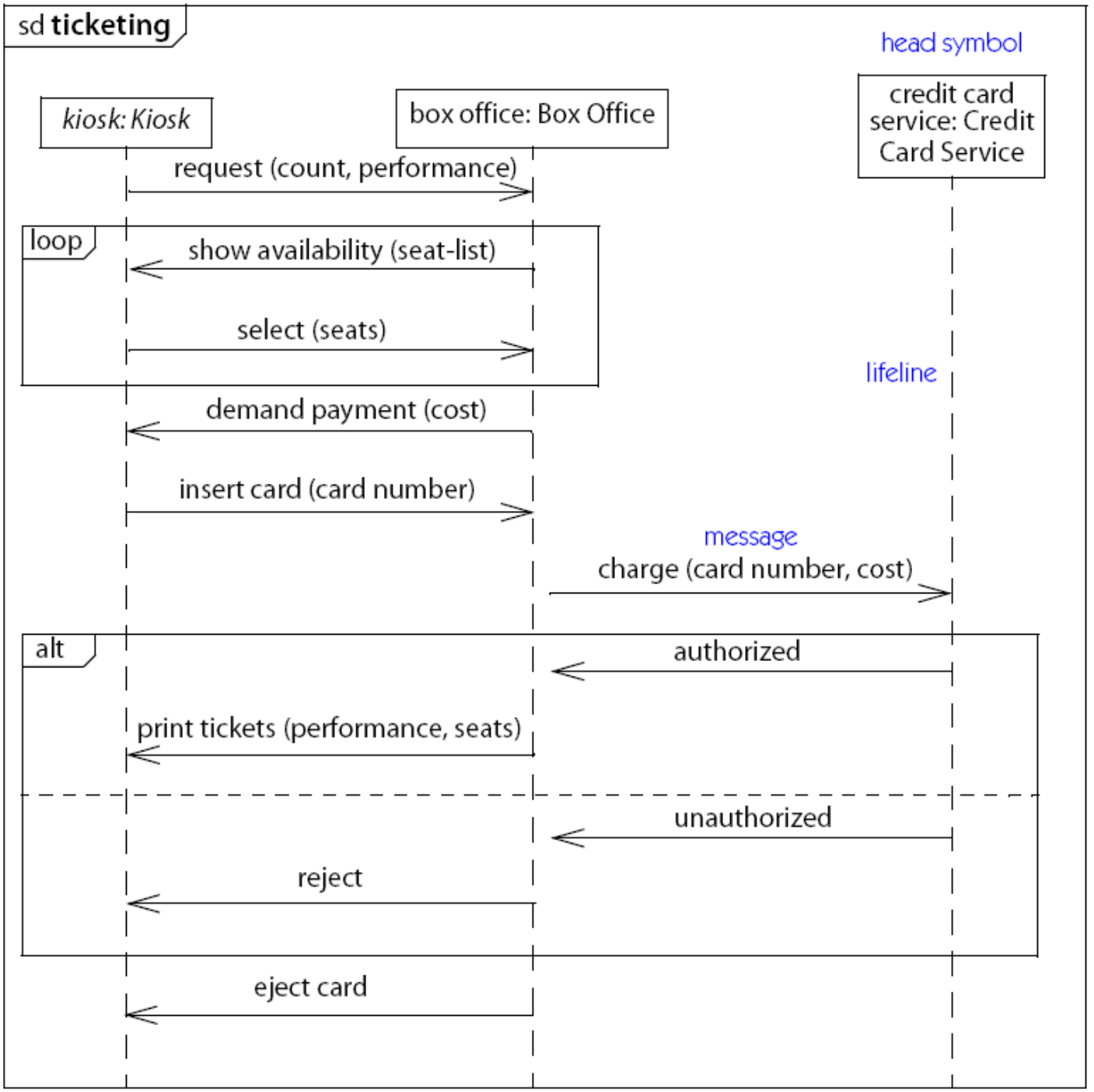


Figure 3-9. *Sequence diagram*

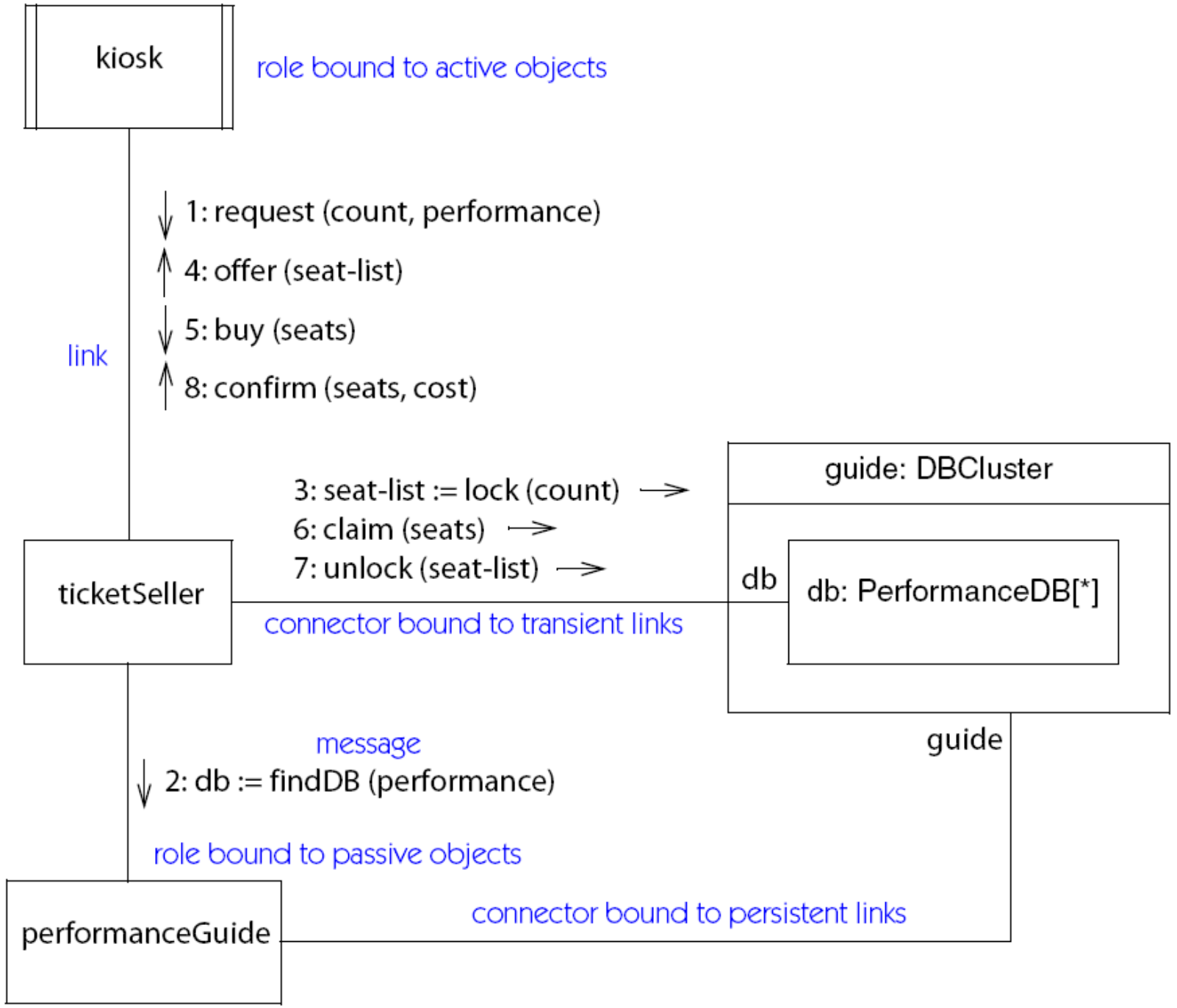


Figure 3-10. *Communication diagram*

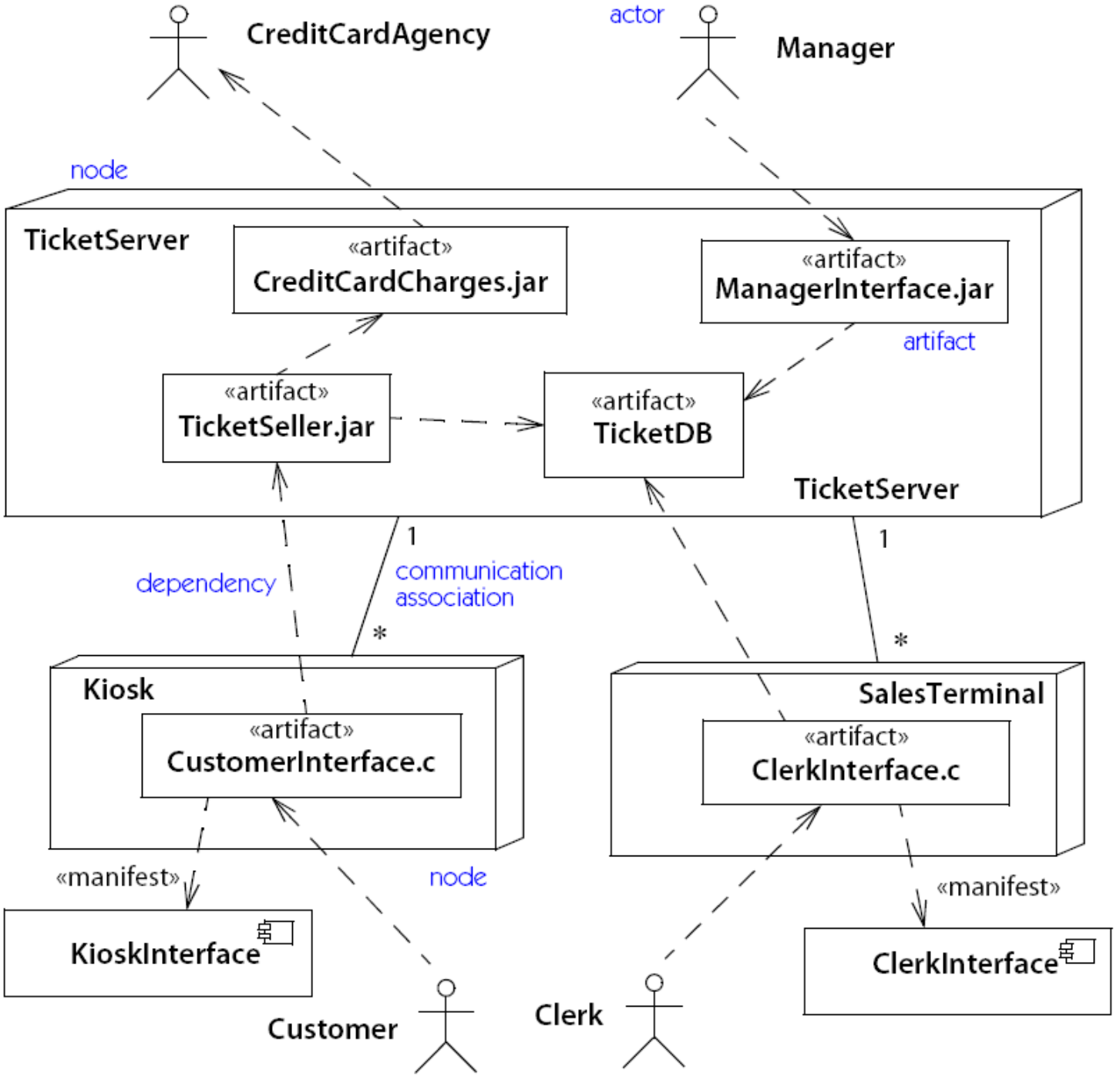


Figure 3-11. *Deployment diagram (descriptor level)*

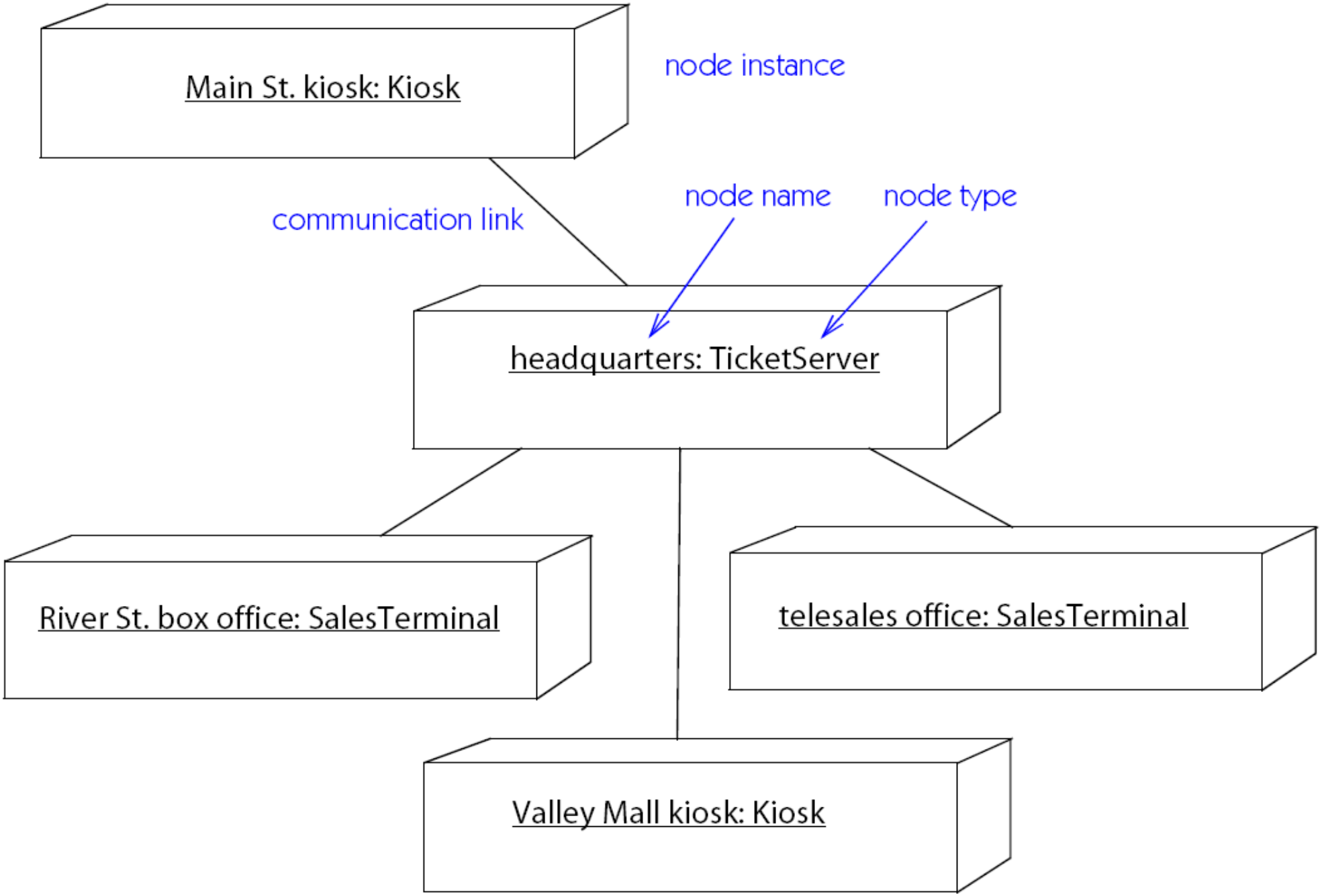


Figure 3-12. *Deployment diagram (instance level)*

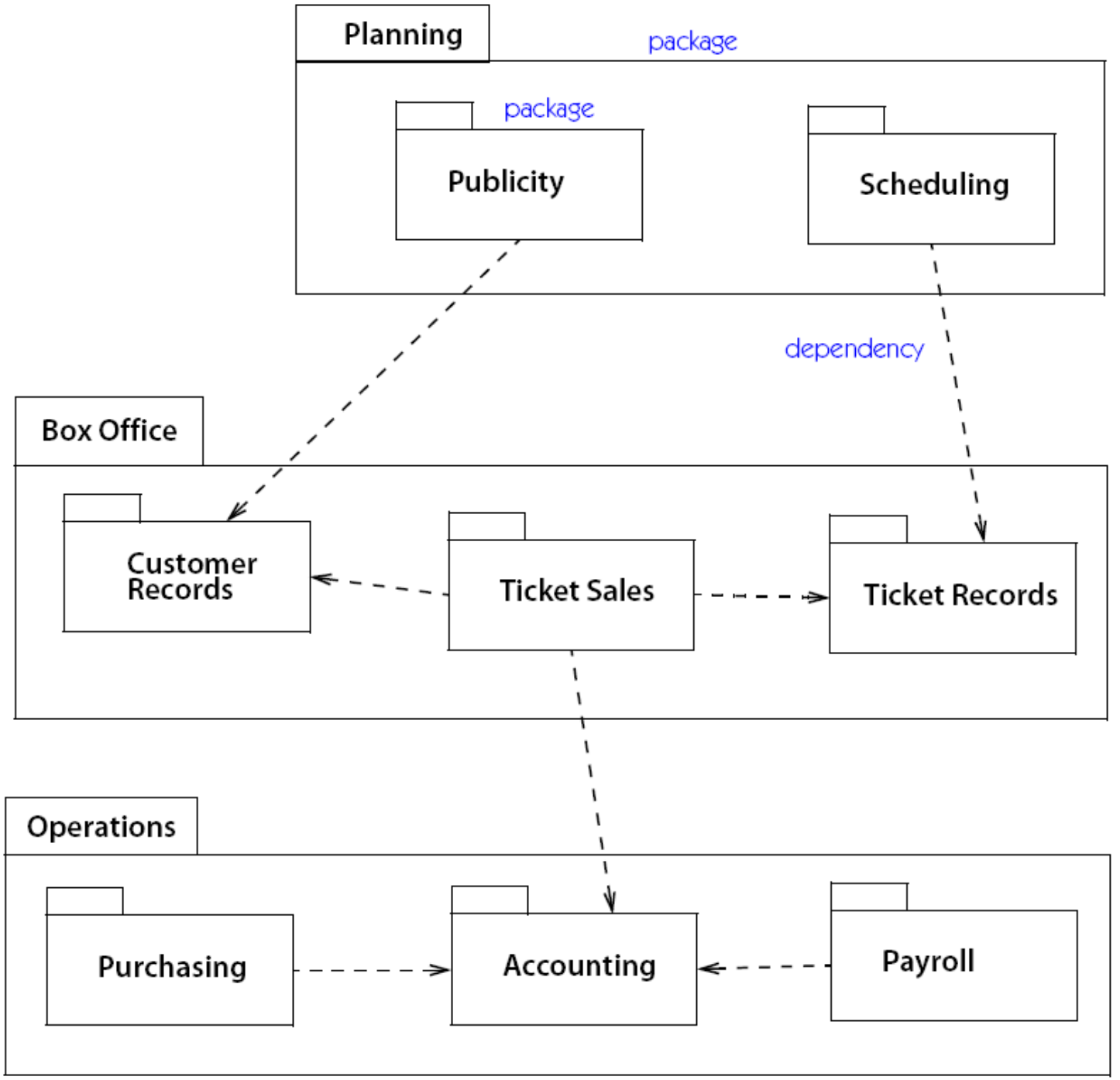


Figure 3-13. *Package diagram*

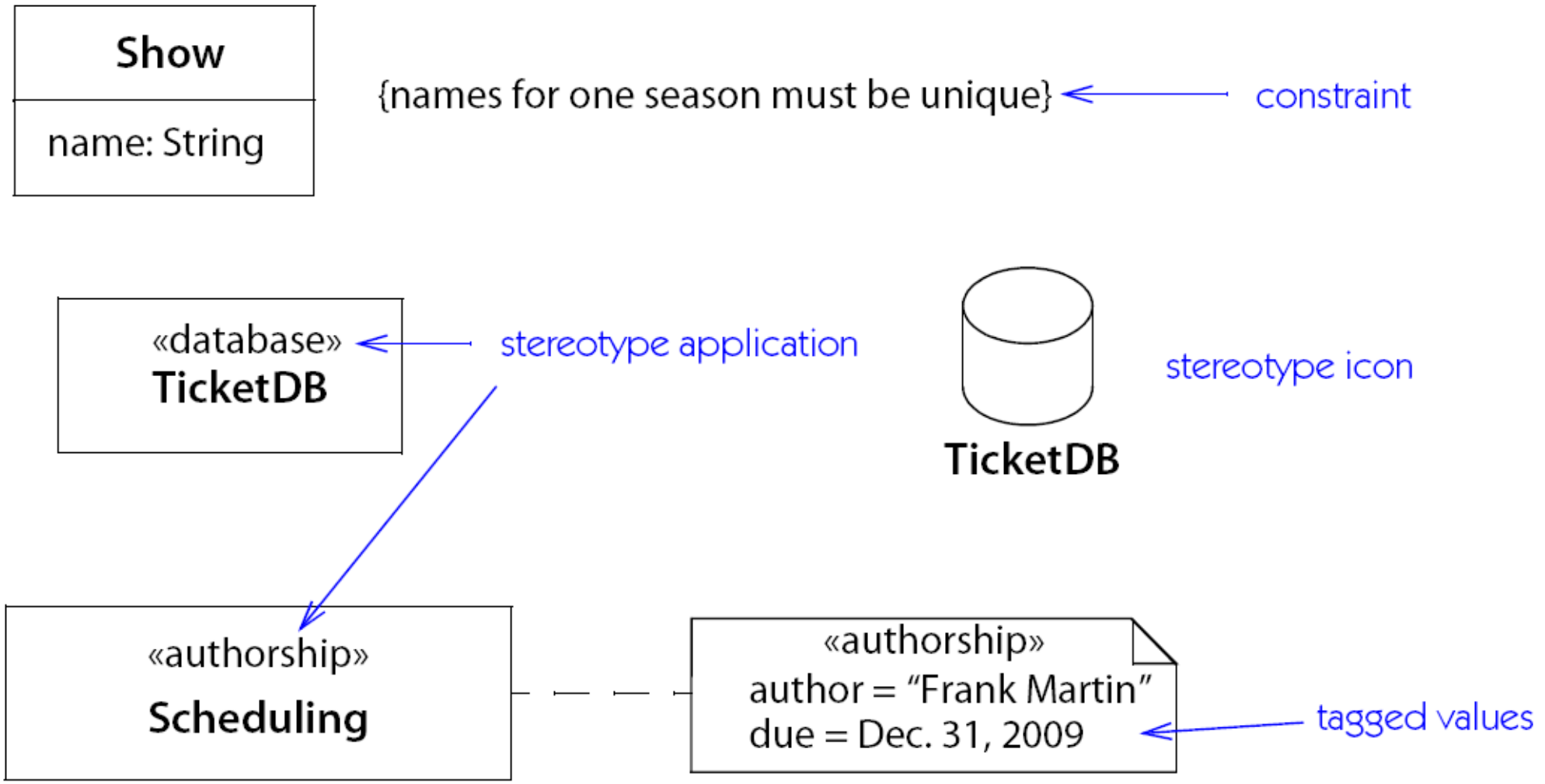


Figure 3-14. *Extensibility constructs*

Chapter 4: Static View 47

Overview 47
Classifier 48
Relationships 52
Association 53
Generalization 57
Realization 61
Dependency 62
Constraint 65
Instance 66

Table 4-1: *Kinds of Classifiers*

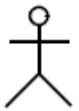
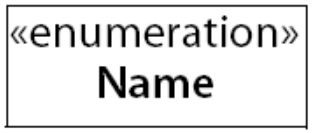
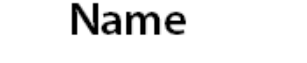
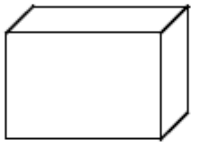
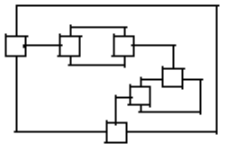
Classifier	*Function*	*Notation*
actor	An outside user of a system	
artifact	A physical piece of system information	«artifact» **Name**
class	A concept from the modeled system	**Name**

Table 4-1: *Kinds of Classifiers (continued)*

Classifier	*Function*	*Notation*
collaboration	A contextual relationship among objects playing roles	
component	A modular part of a system with well-defined interfaces	
enumeration	A data type with predefined literal values	
primitive type	A descriptor of a set of primitive values that lack identity	
interface	A named set of operations that characterize behavior	

node	A computational resource	
role	An internal part in the context of a collaboration or structured classifier	
signal	An asynchronous communication among objects	
structured classifier	A classifier with internal structure	
use case	A specification of the behavior of an entity in its interaction with outside agents	

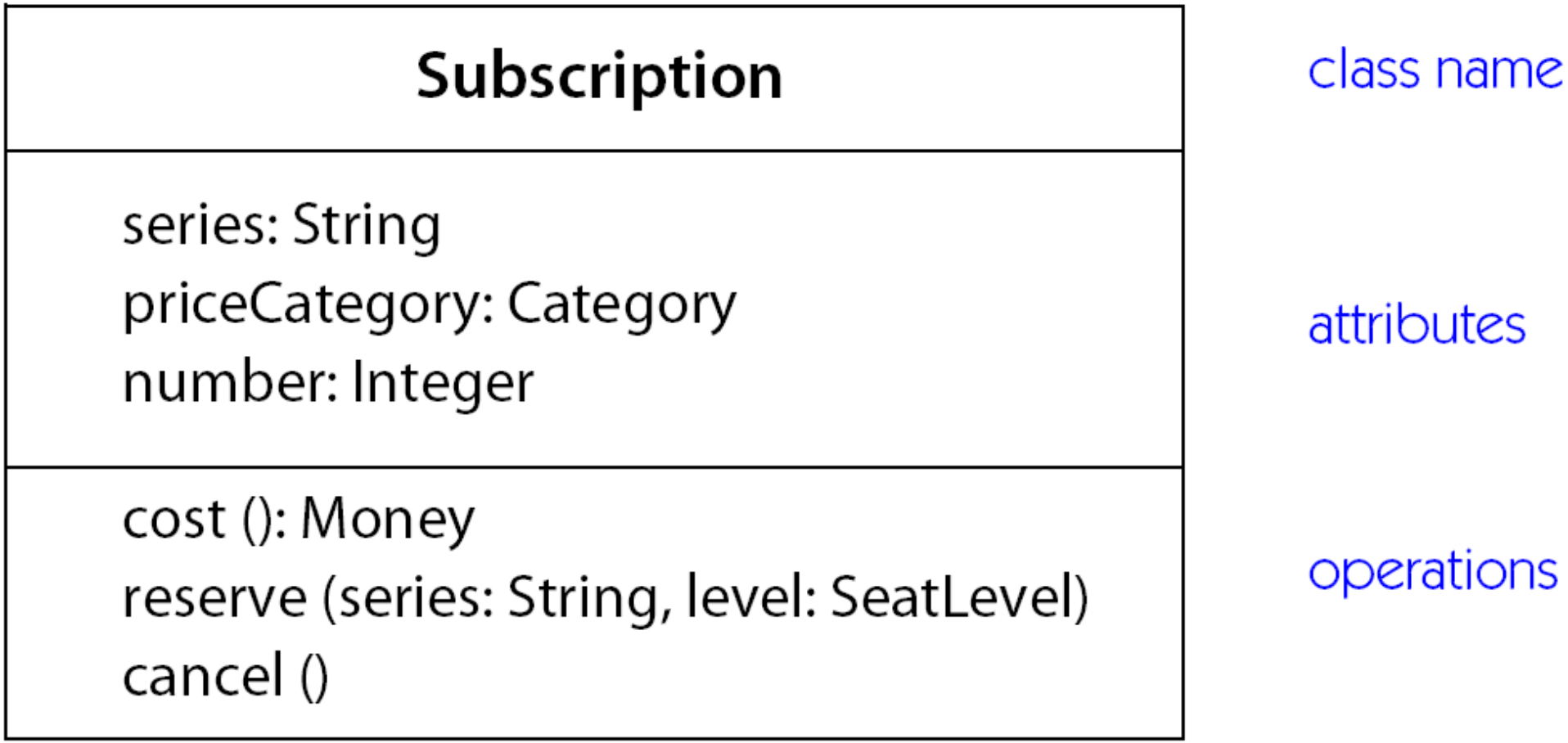
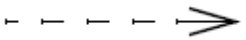
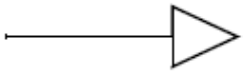
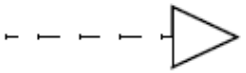
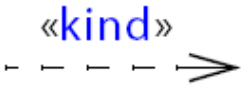


Figure 4-1. *Class notation*

Table 4-2: *Kinds of Relationships*

Relationship	*Function*	*Notation*
association	A description of a connection among instances of classes	
dependency	A relationship between two model elements	
generalization	A relationship between a more specific and a more general description, used for inheritance and polymorphic type declarations	
realization	Relationship between a specification and its implementation	
usage	A situation in which one element requires another for its correct functioning	

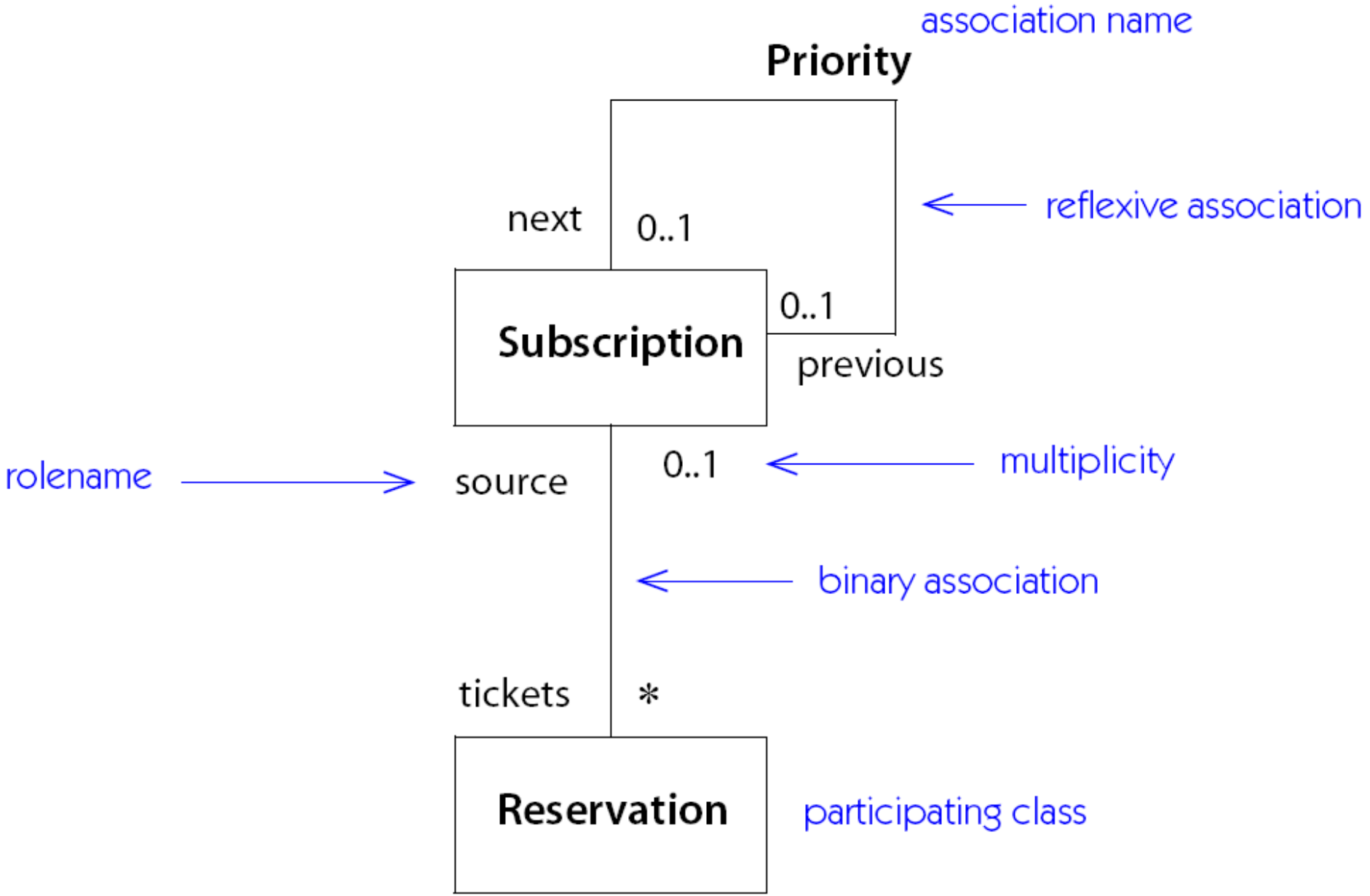


Figure 4-2. *Association notation*

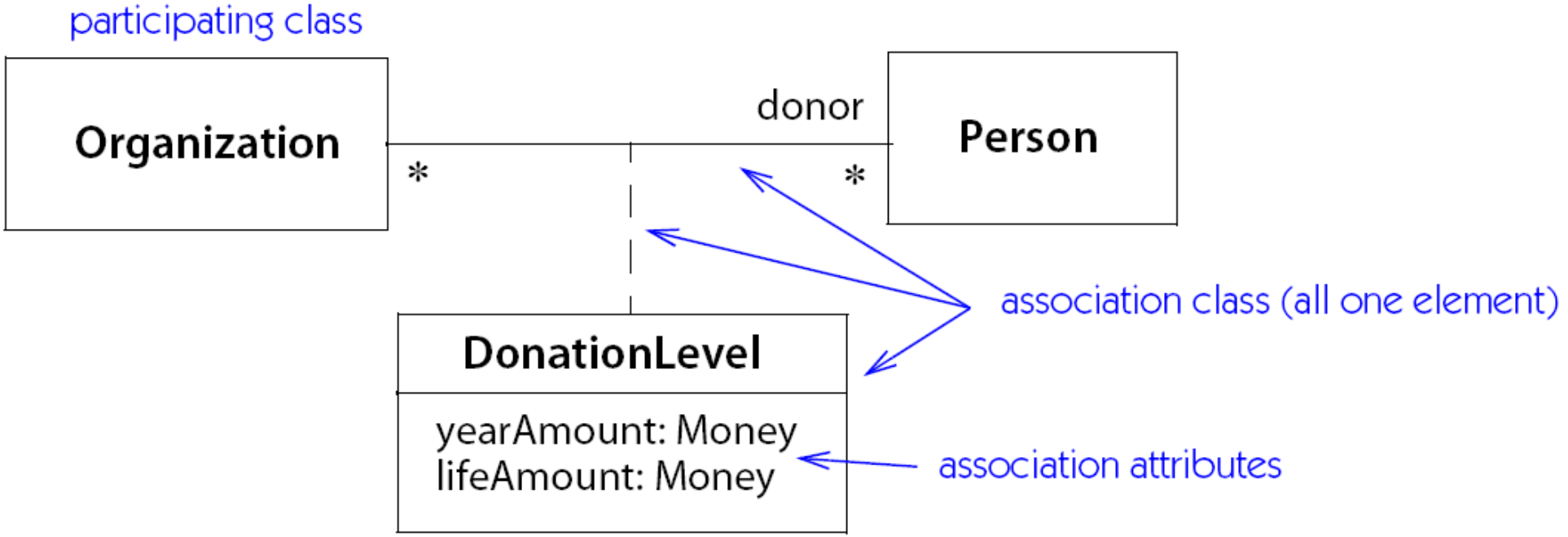


Figure 4-3. *Association class*

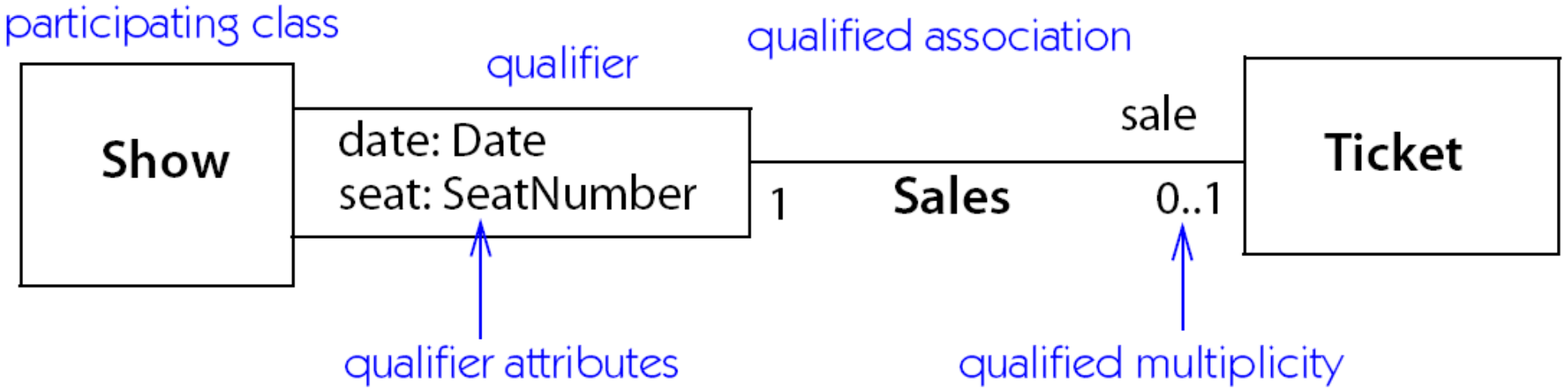


Figure 4-4. *Qualified association*

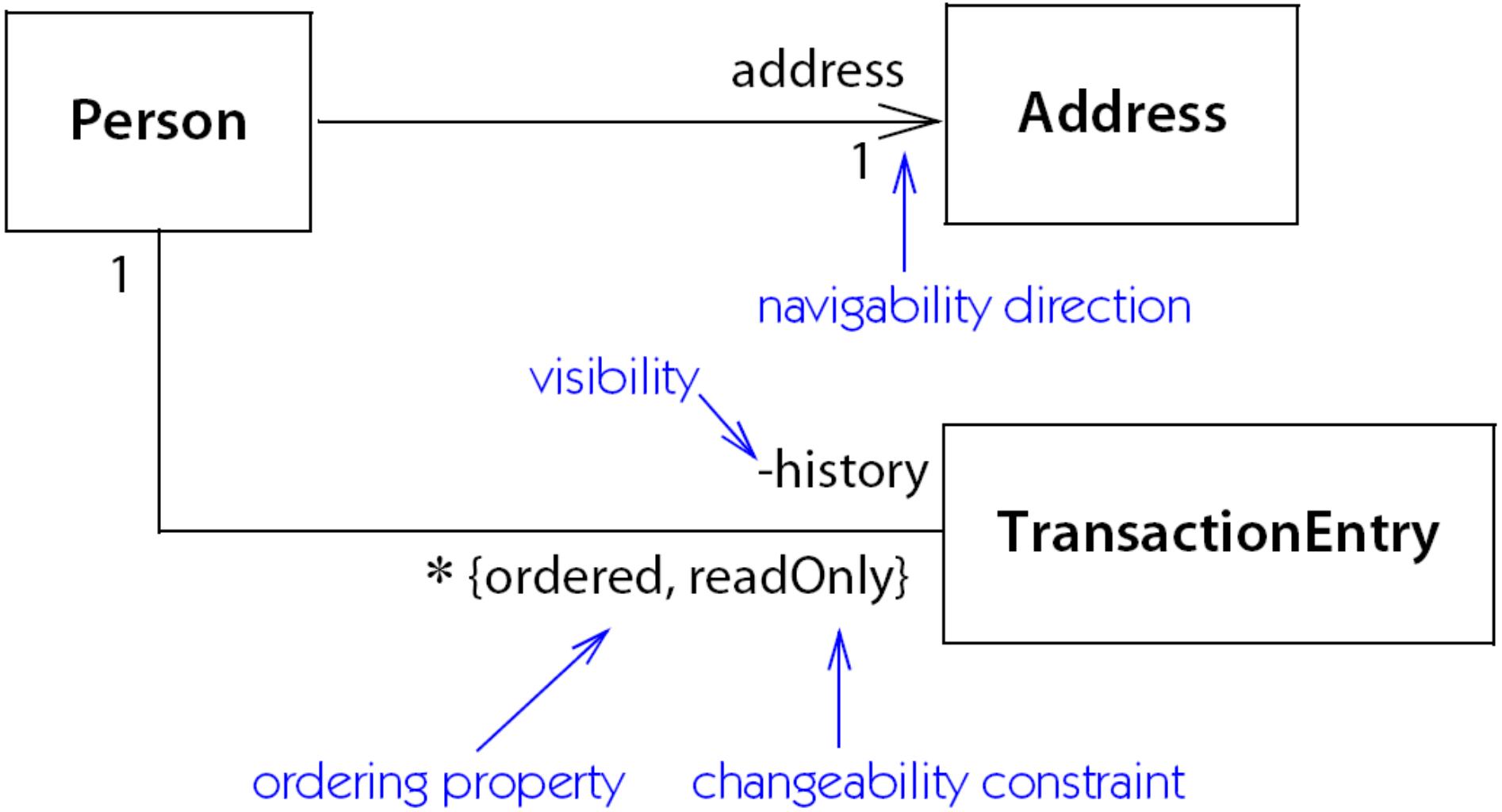


Figure 4-5. *Design properties of association*

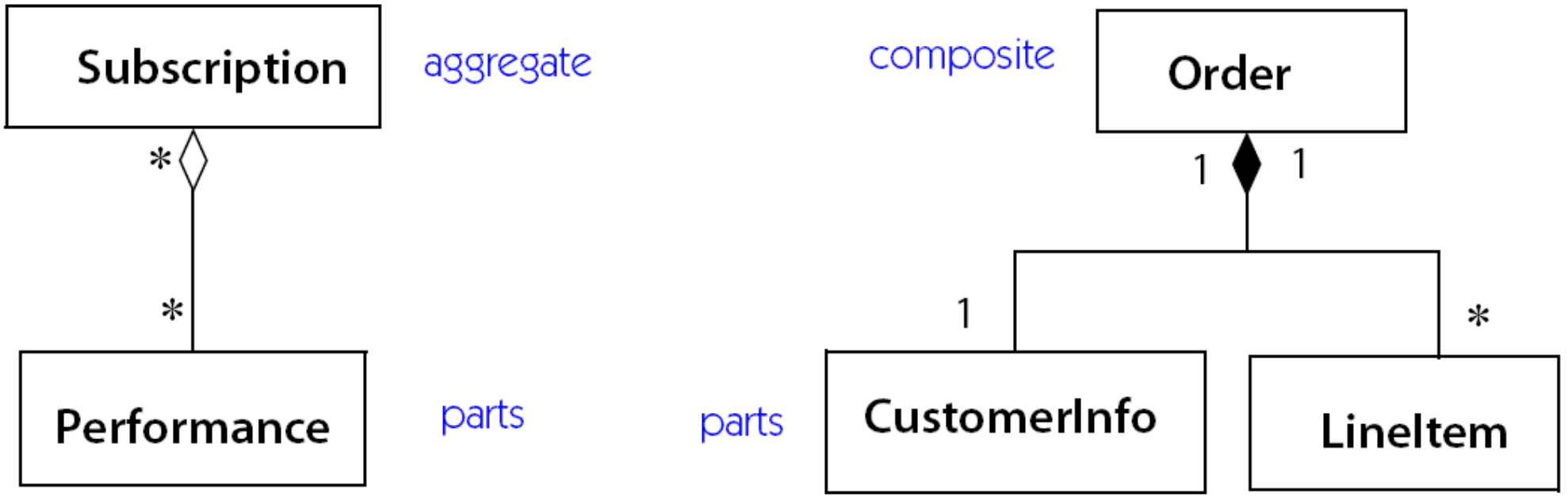


Figure 4-6. *Aggregation and composition*

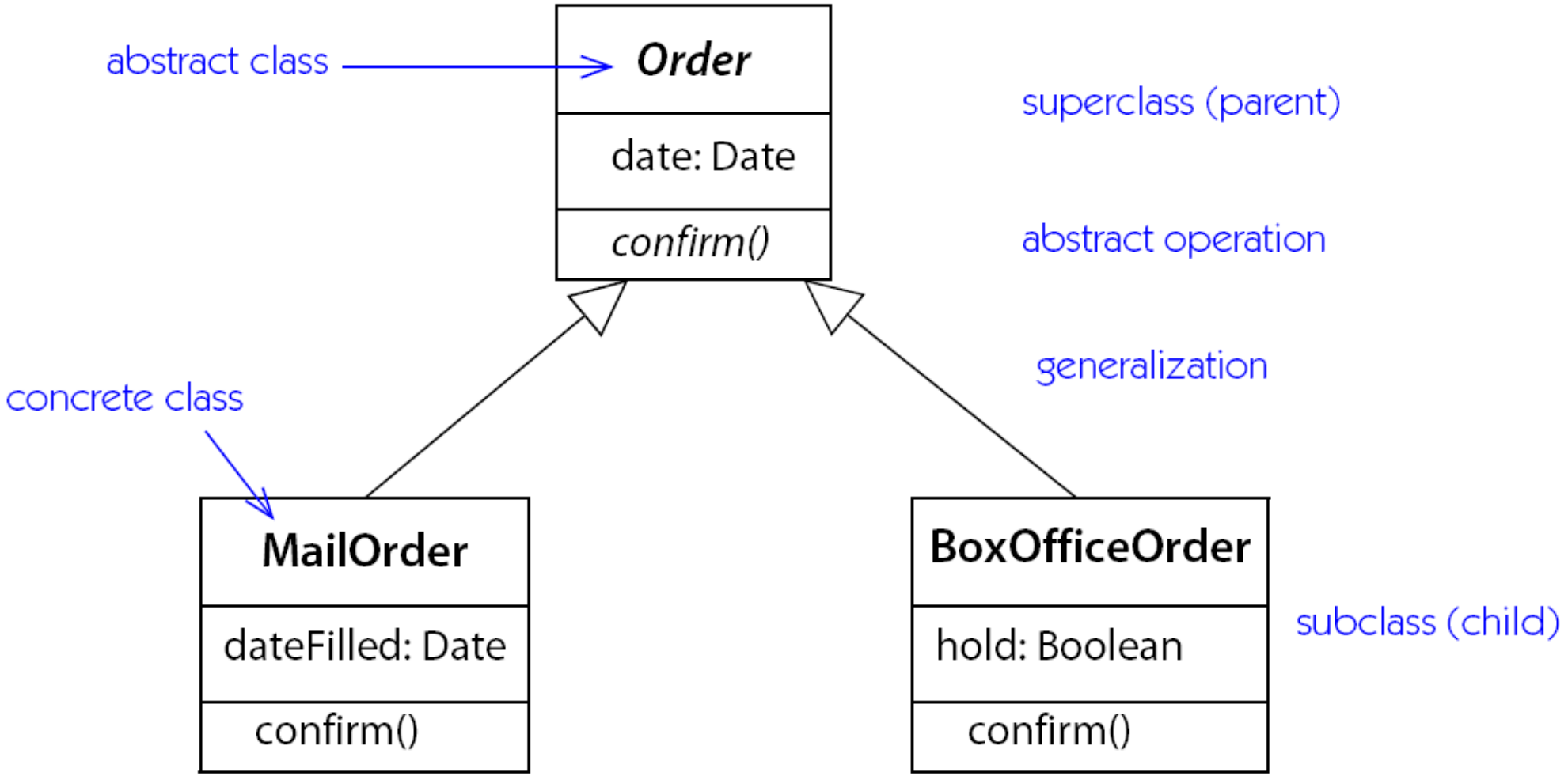


Figure 4-7. *Generalization notation*

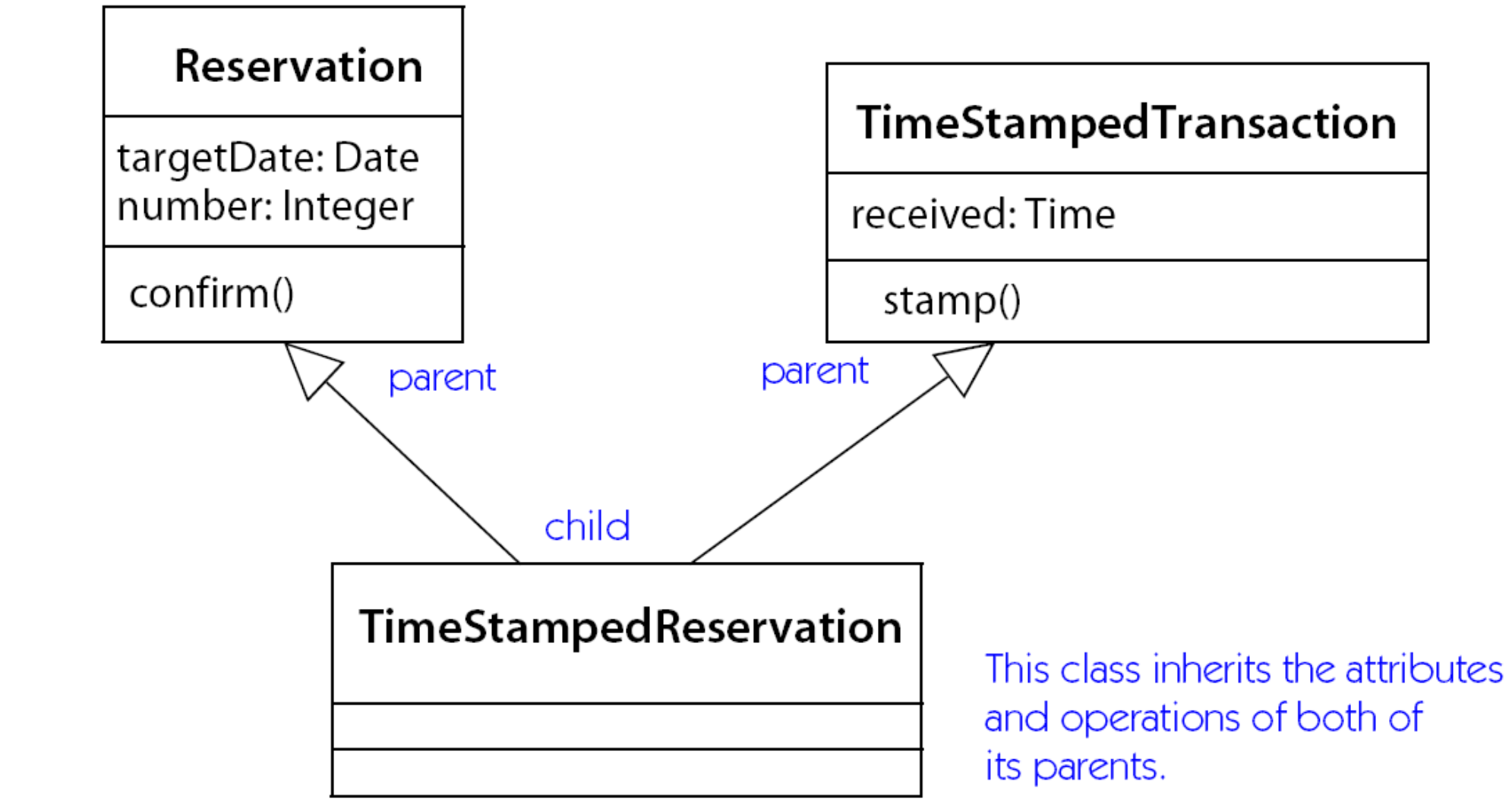


Figure 4-8. *Multiple inheritance*

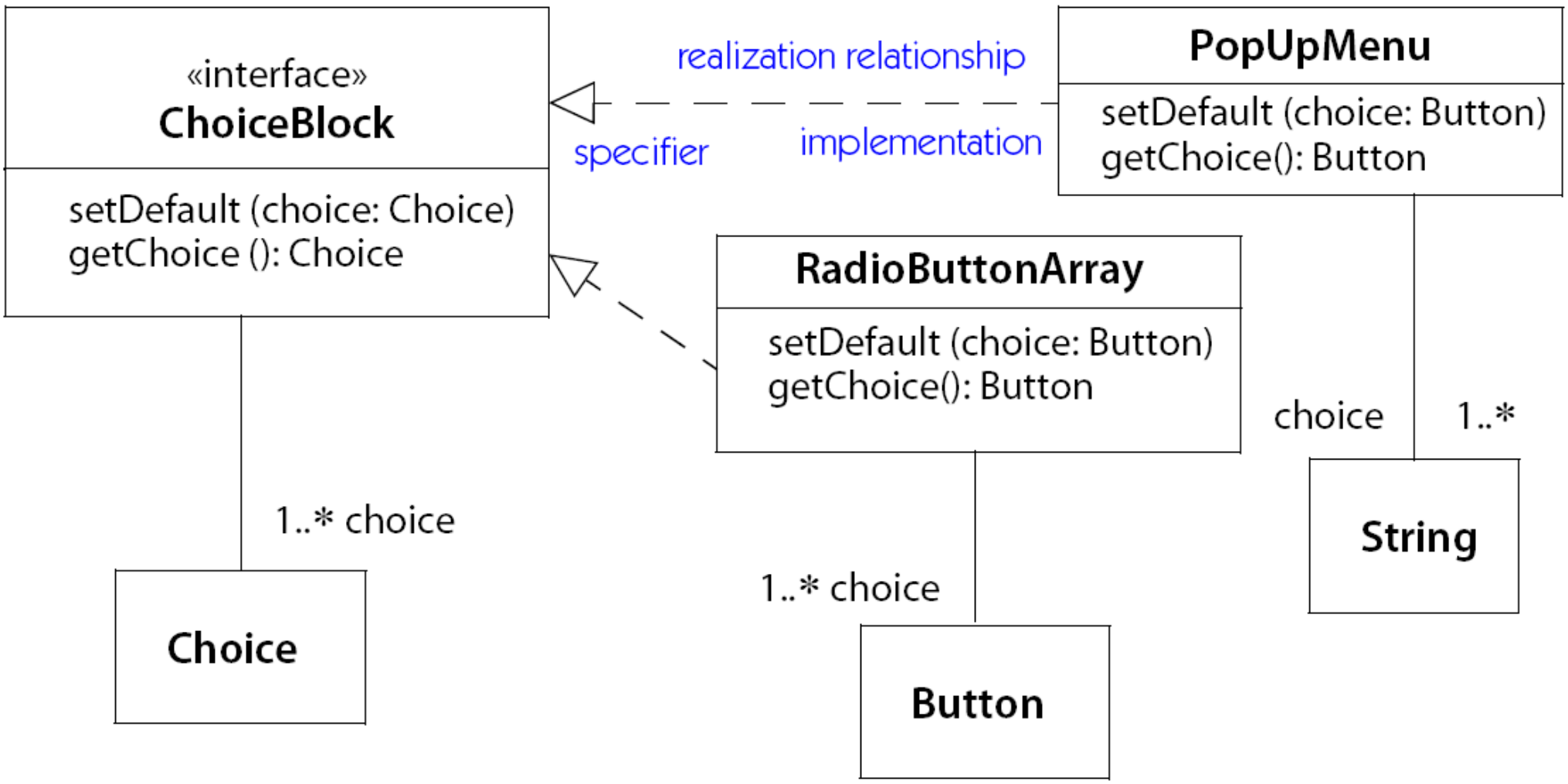


Figure 4-9. *Realization relationship*

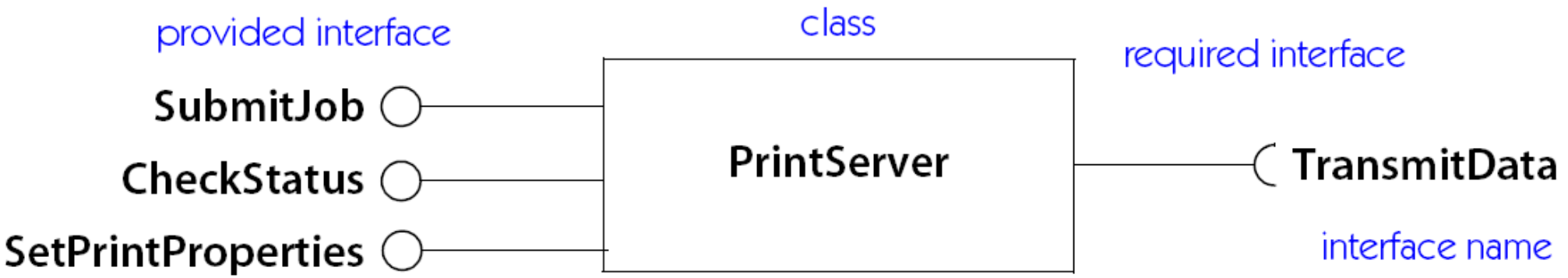


Figure 4-10. *Provided and required interfaces*

Table 4-3: *Kinds of Dependencies*

Dependency	*Function*	*Keyword*
access	A private import of the contents of another package	access
binding	Assignment of values to the parameters of a template to generate a new model element	bind
call	Statement that a method of one class calls an operation of another class	call
creation	Statement that one class creates instances of another class	create
derivation	Statement that one instance can be computed from another instance	derive
instantiation	Statement that a method of one class creates instances of another class	instantiate
permission	Permission for an element to use the contents of another element	permit

realization	Mapping between a specification and an implementation of it	realize
refinement	Statement that a mapping exists between elements at two different semantic levels	refine
send	Relationship between the sender of a signal and the receiver of the signal	send
substitution	Statement that the source class supports the interfaces and contracts of the target class and may be substituted for it	substitute
trace dependency	Statement that some connection exists between elements in different models, but less precise than a mapping	trace
usage	Statement that one element requires the presence of another element for its correct functioning (includes call, creation, instantiation, send, and potentially others)	use

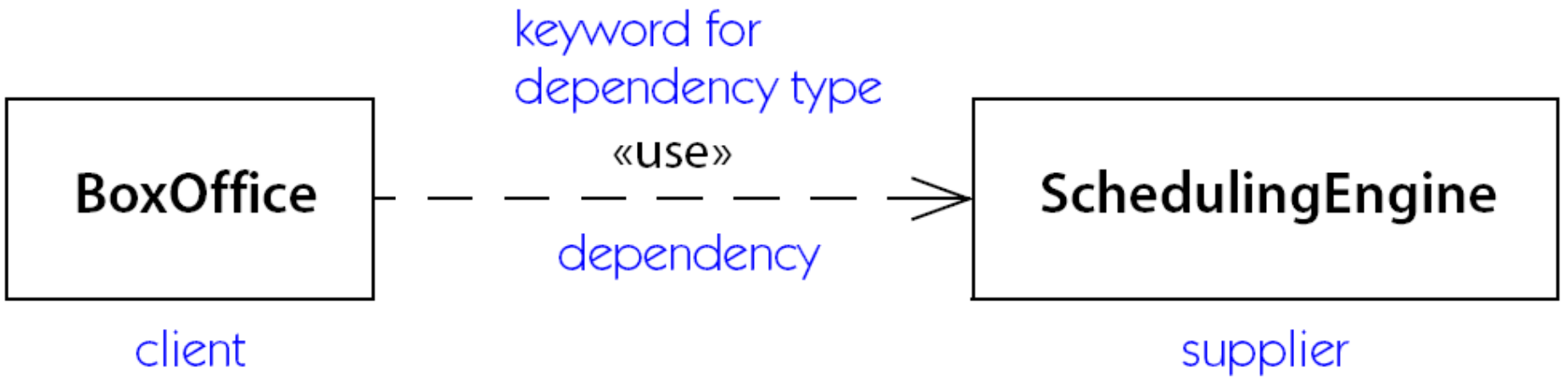


Figure 4-11. *Dependencies*

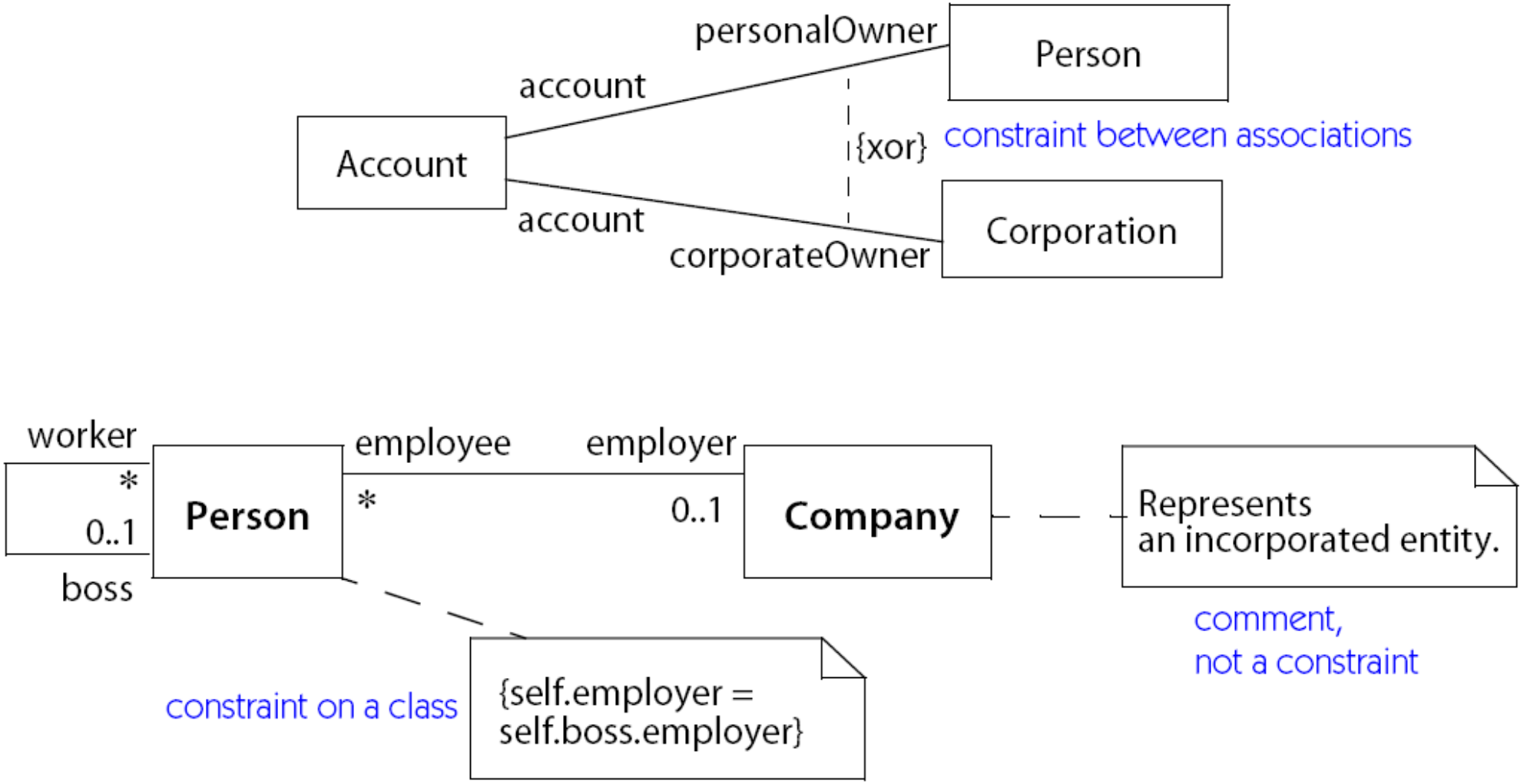


Figure 4-12. *Constraints*

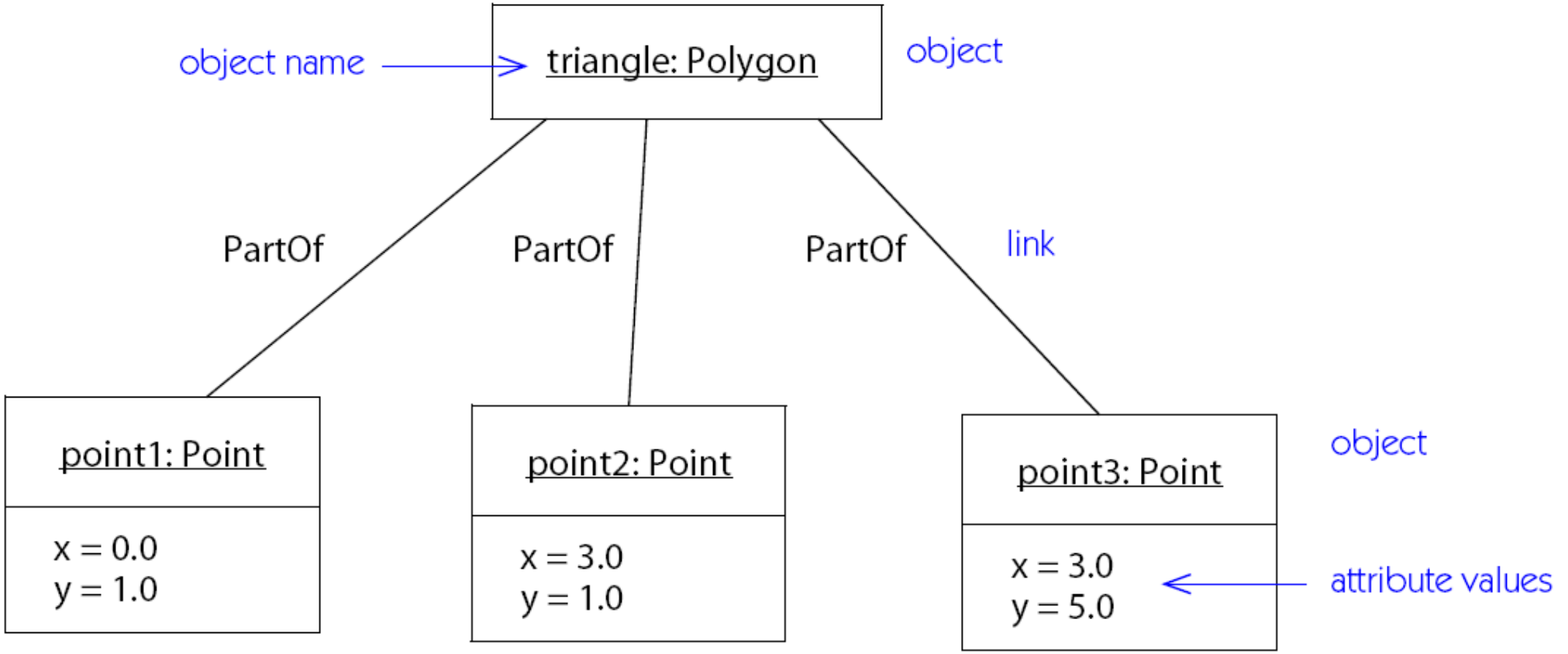


Figure 4-13. *Object diagram*

Chapter 5: Design View 69

Overview 69
Structured Classifier 70
Collaboration 71
Patterns 73
Component 73

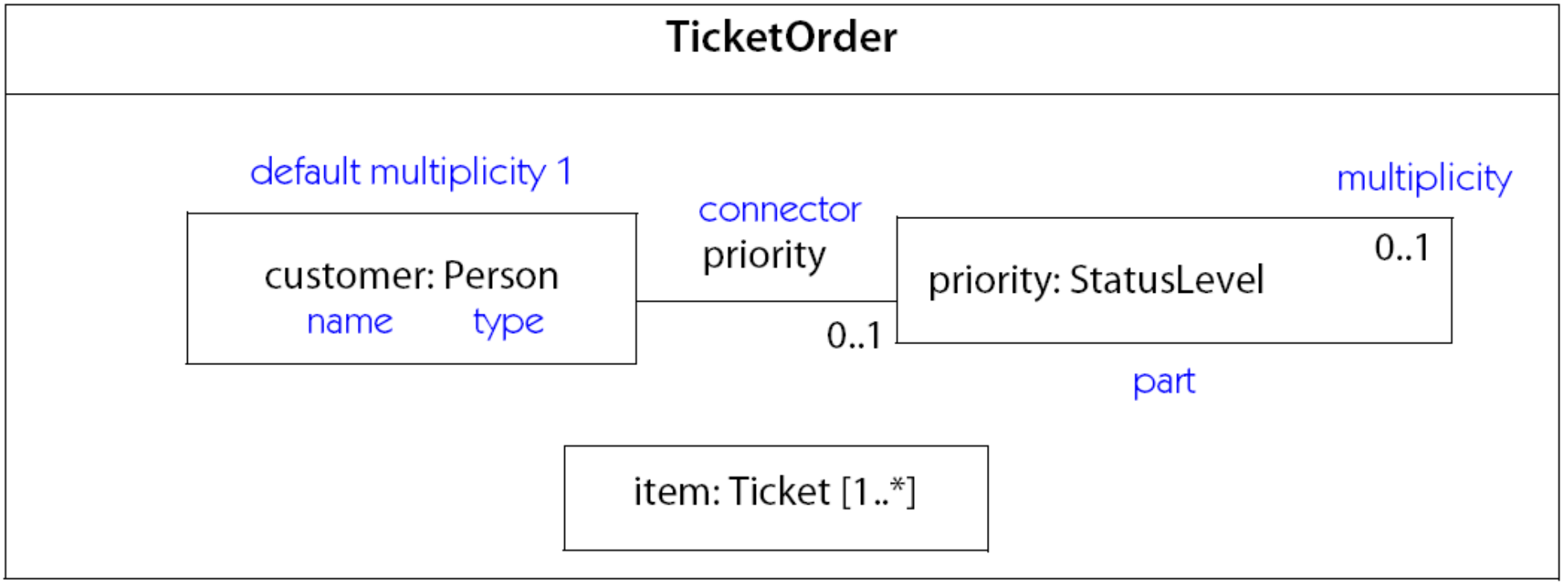


Figure 5-1. *Structured class*

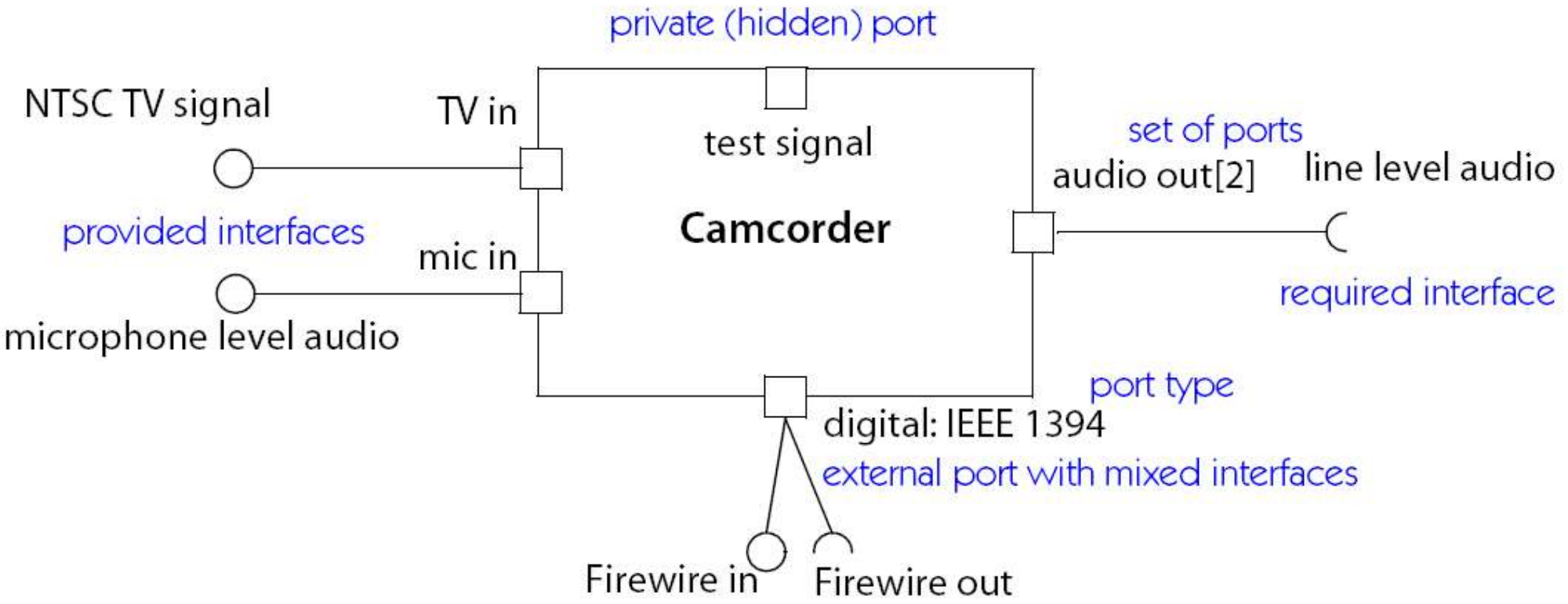


Figure 5-2. *Structured class with ports*

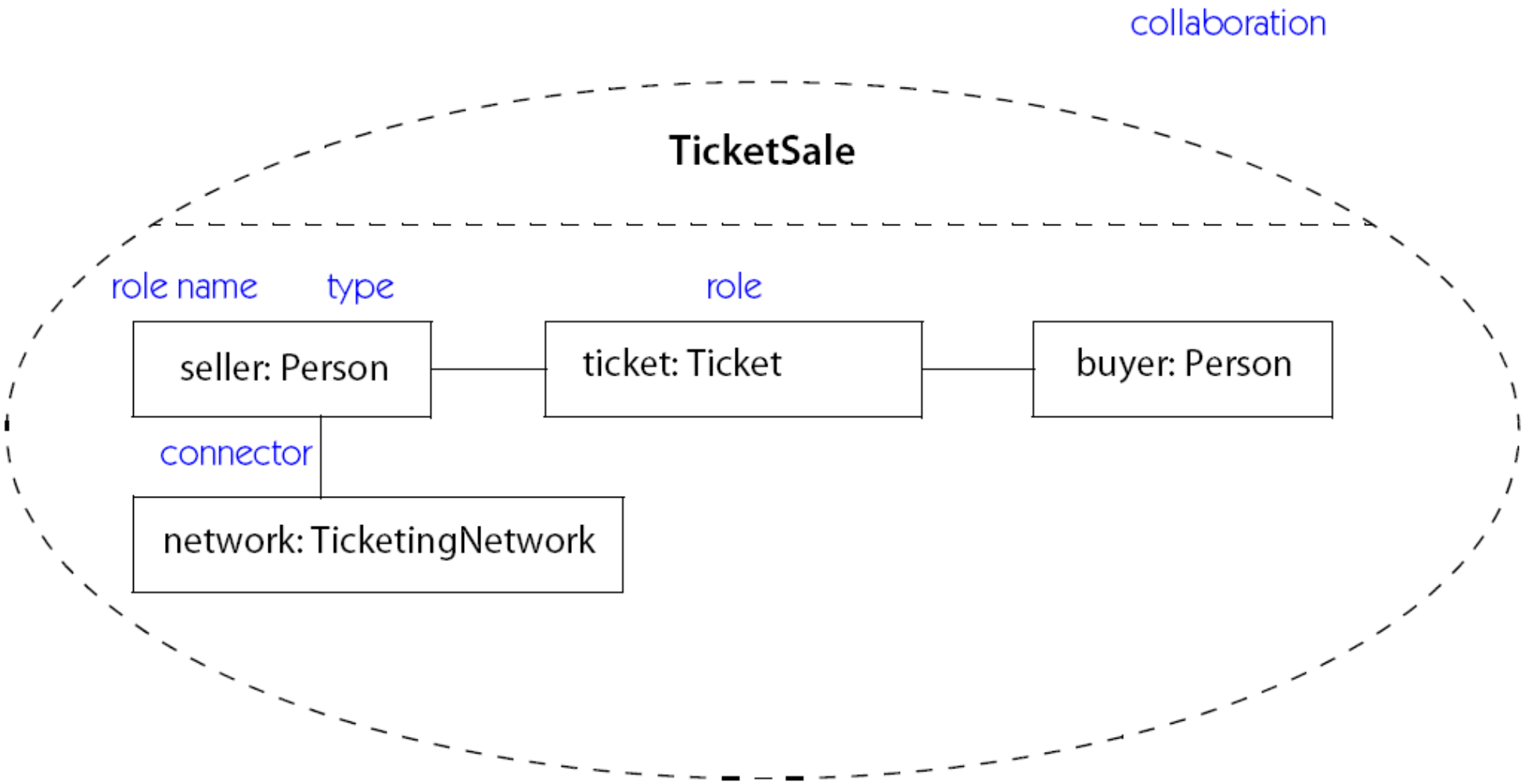


Figure 5-3. *Collaboration definition*

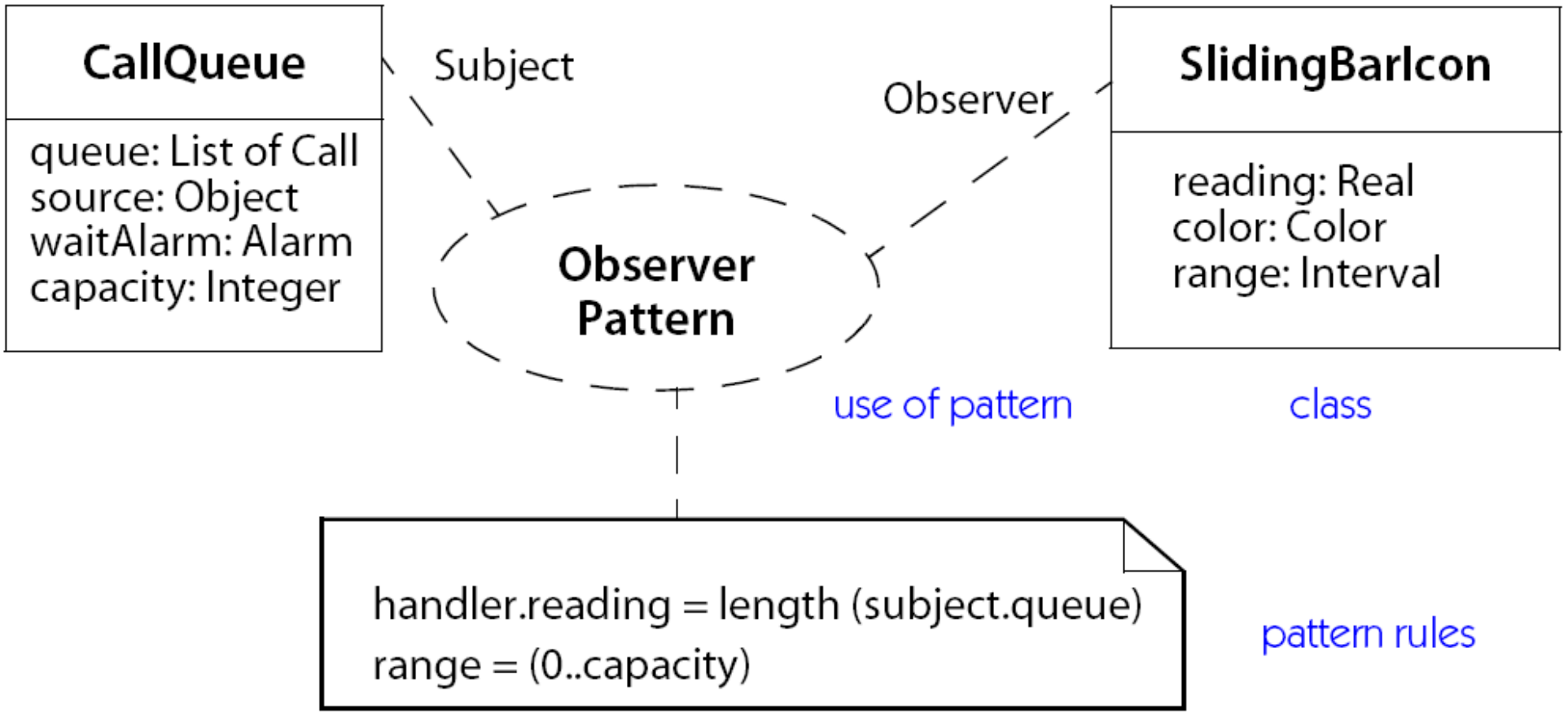


Figure 5-4. *Pattern usage*

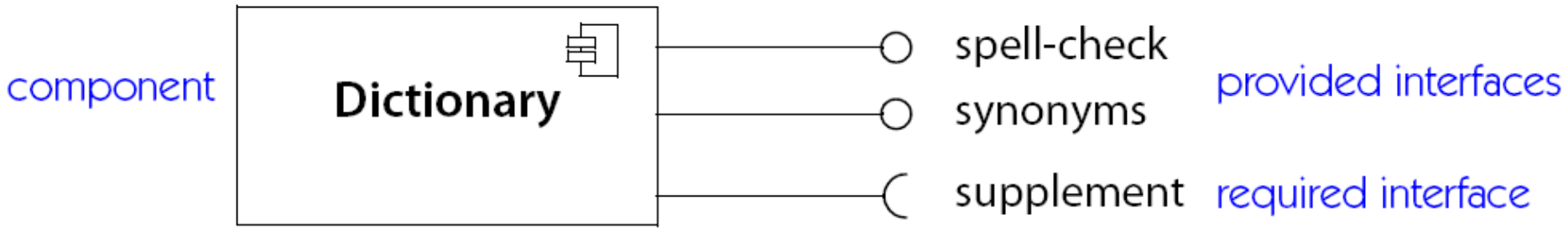


Figure 5-5. *Component with interfaces*

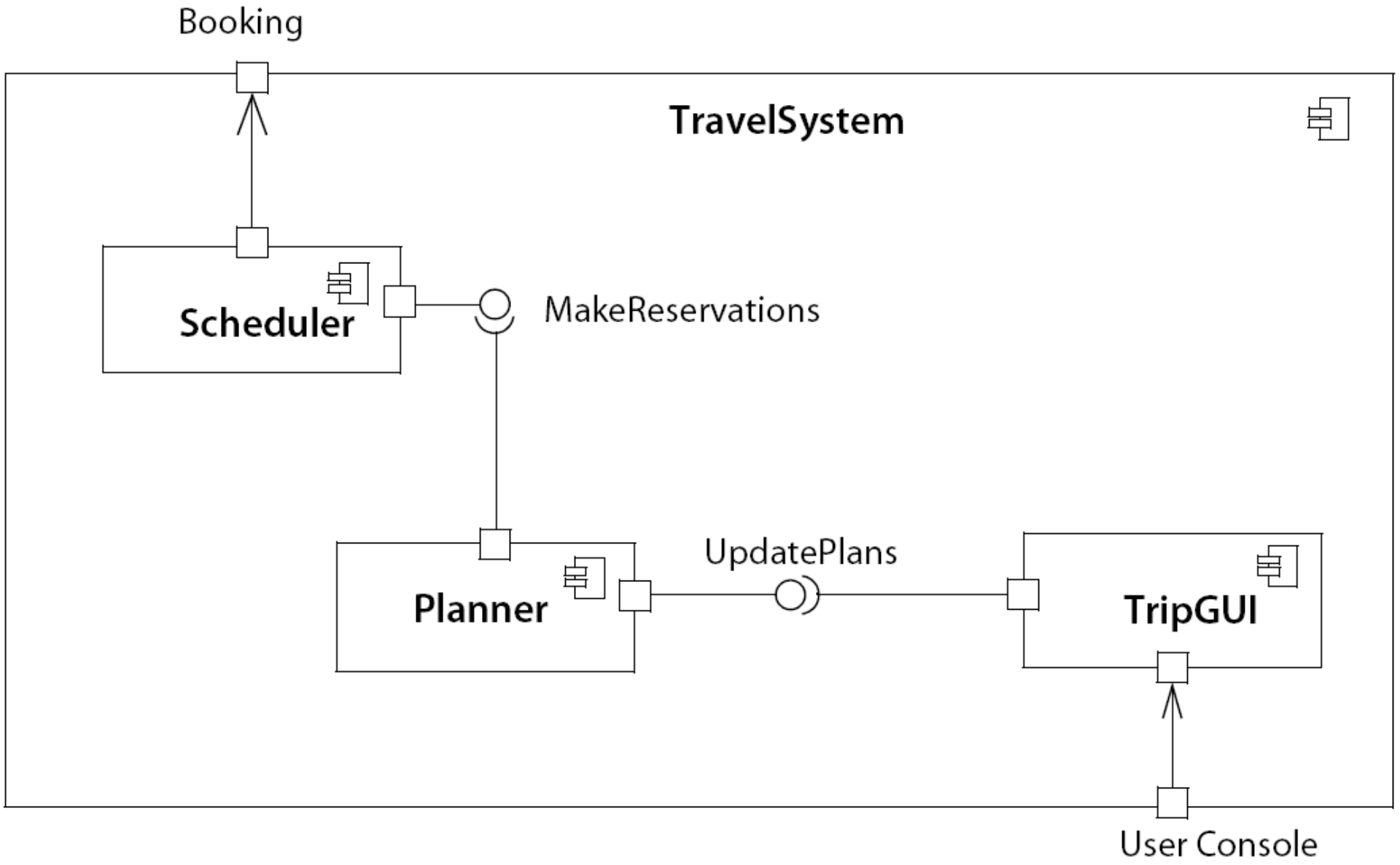


Figure 5-6. *Component internal structure*

Chapter 6: Use Case View 77
Overview 77
Actor 77
Use Case 78

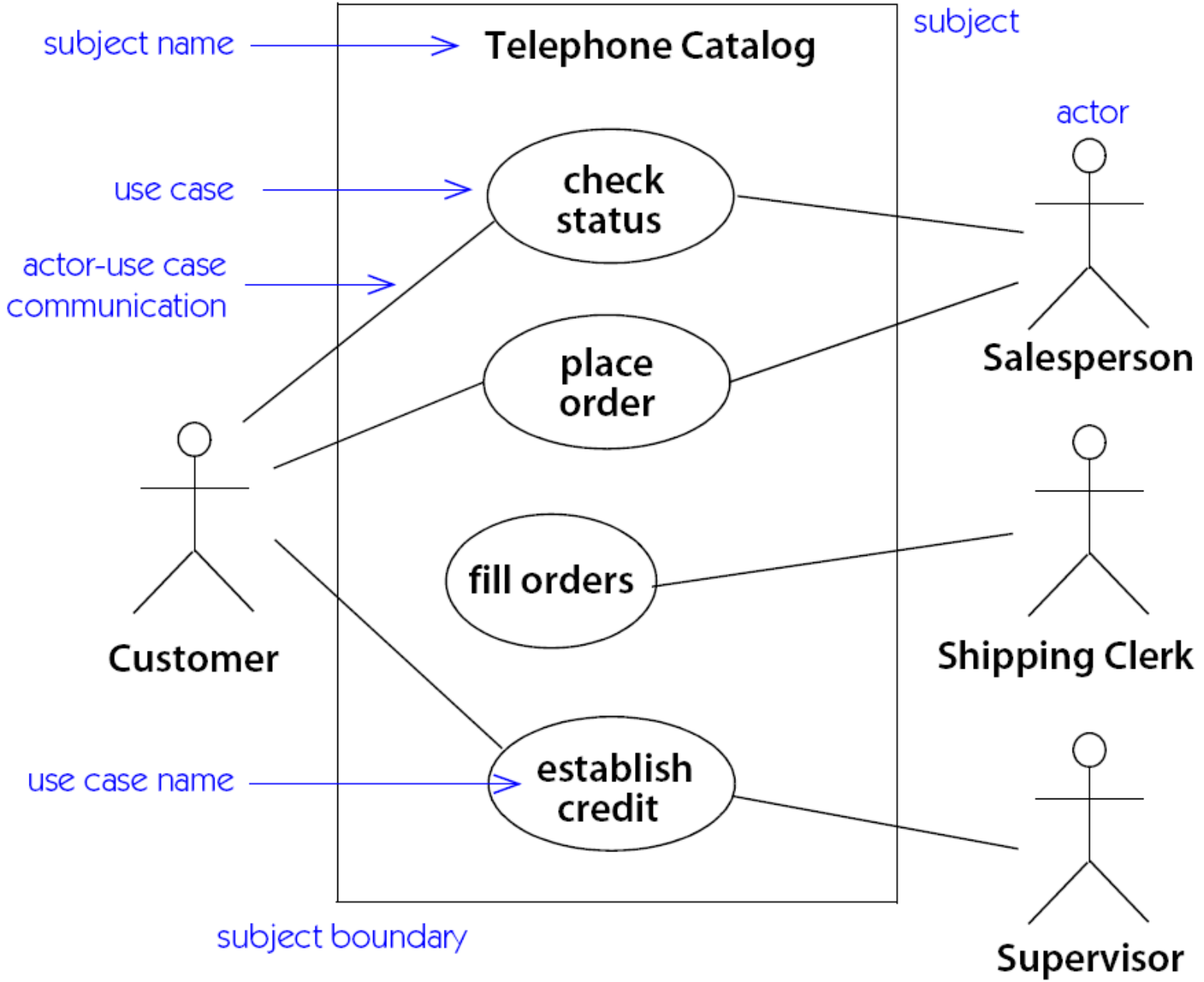
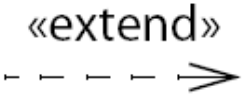
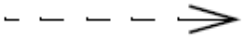
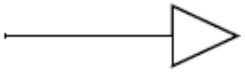


Figure 6-1. *Use case diagram*

Table 6-1: *Kinds of Use Case Relationships*

Relationship	*Function*	*Notation*
association	The communication path between an actor and a use case that it participates in	
extend	The insertion of additional behavior into a base use case that does not know about it	
include	The insertion of additional behavior into a base use case that explicitly describes the insertion	
use case generalization	A relationship between a general use case and a more specific use case that inherits and adds features to it	

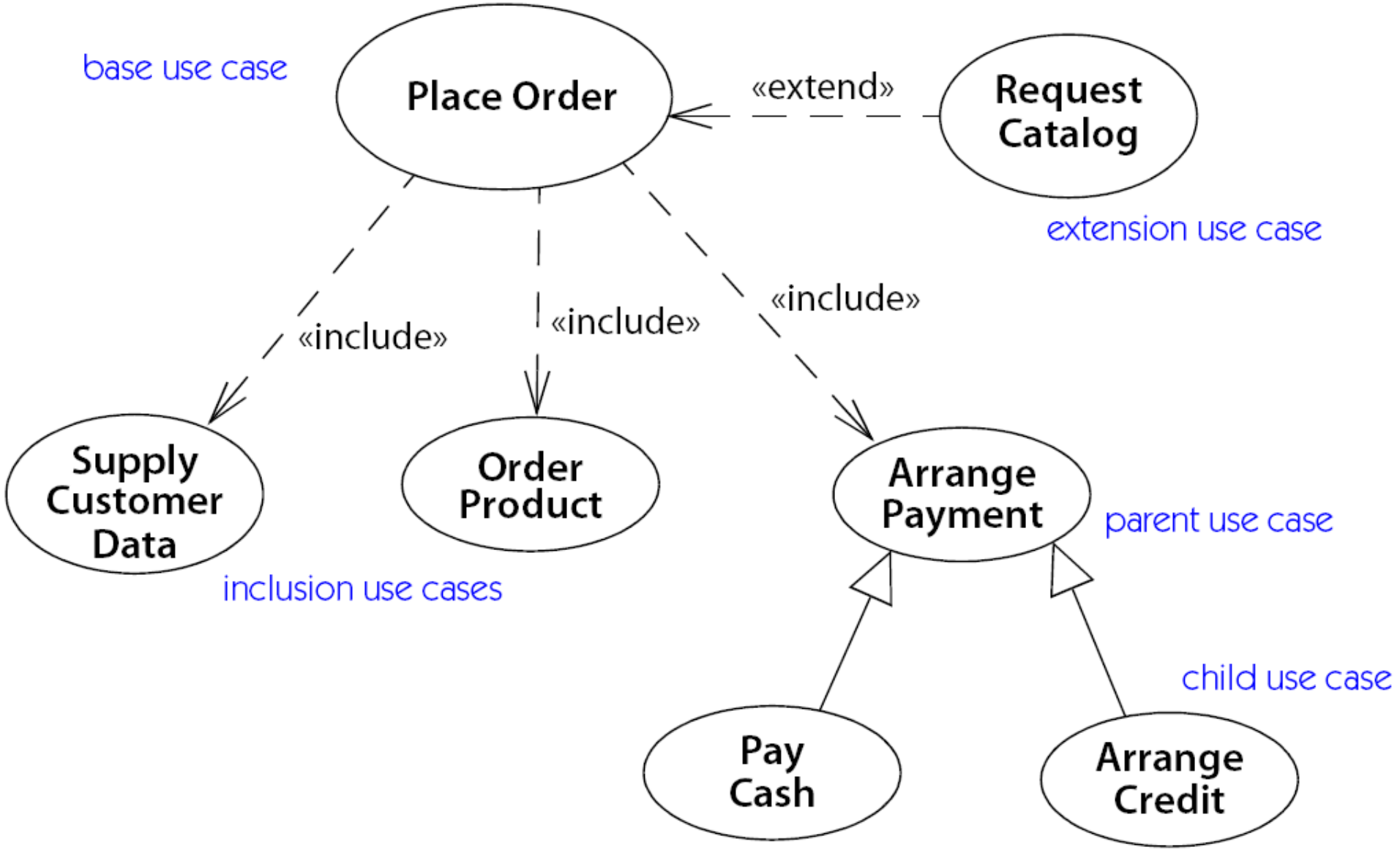


Figure 6-2. *Use case relationships*

Chapter 7: State Machine View 81

Overview 81
State Machine 81
Event 82
State 84
Transition 85
Composite State 89

Table 7-1: *Kinds of Events*

Event Type	*Description*	*Syntax*
call event	Receipt of an explicit synchronous call request by an object	op (a:T)
change event	A change in value of a Boolean expression	**when** (exp)
signal event	Receipt of an explicit, named, asynchronous communication among objects	sname (a:T)
time event	The arrival of an absolute time or the passage of a relative amount of time	**after** (time)

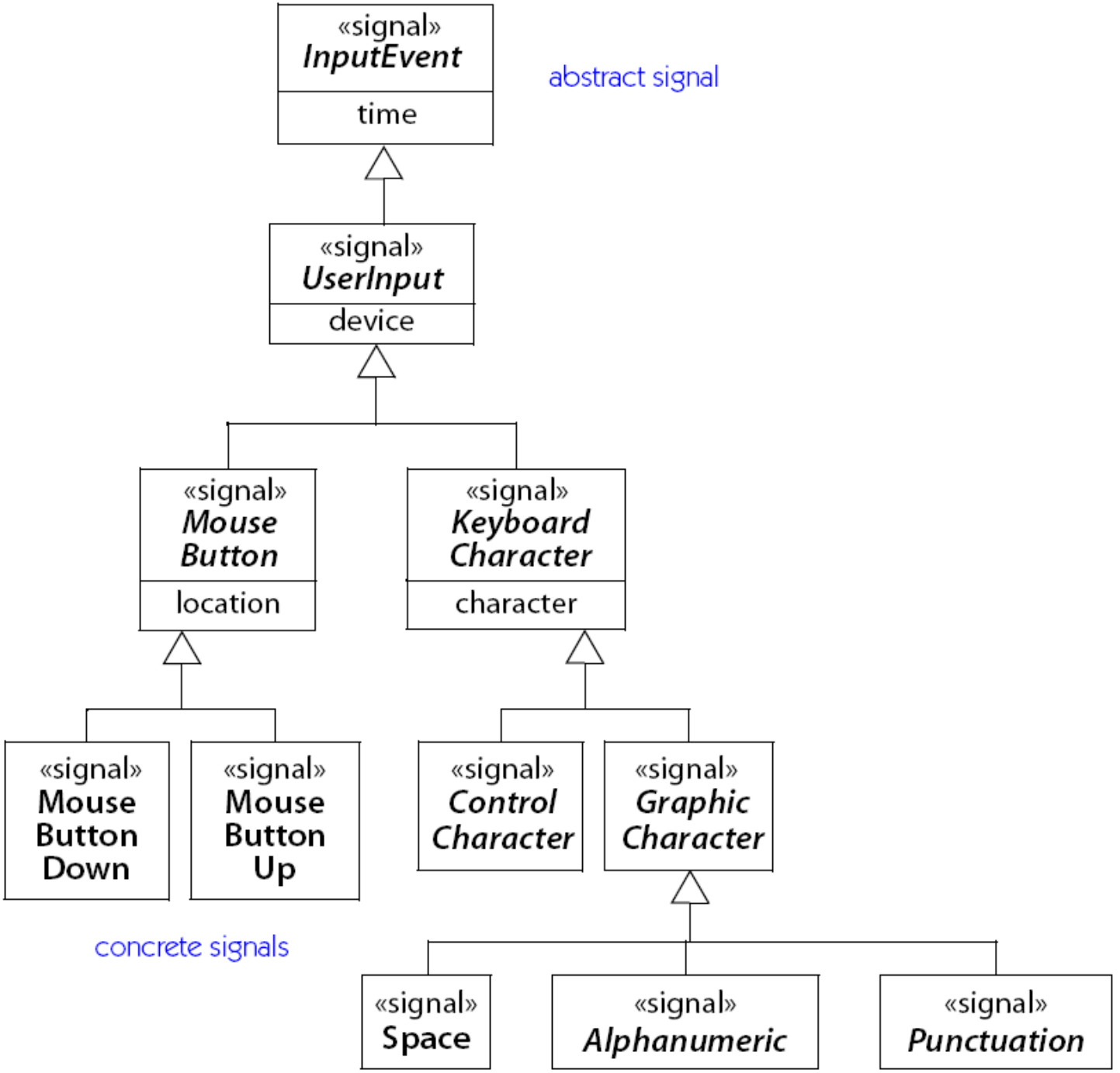


Figure 7-1. *Signal hierarchy*

Figure 7-2. *State*

Table 7-2: *Kinds of Transitions and Implicit Effects*

Transition Kind	*Description*	*Syntax*
entry transition	The specification of an entry activity that is executed when a state is entered	entry/ activity
exit transition	The specification of an exit activity that is executed when a state is exited	exit/ activity
external transition	A response to an event that causes a change of state or a self-transition, together with a specified effect. It may also cause the execution of exit and/or entry activities for states that are exited or entered.	e(a:T)[guard]/activity
internal transition	A response to an event that causes the execution of an effect but does not cause a change of state or execution of exit or entry activities	e(a:T)[guard]/activity

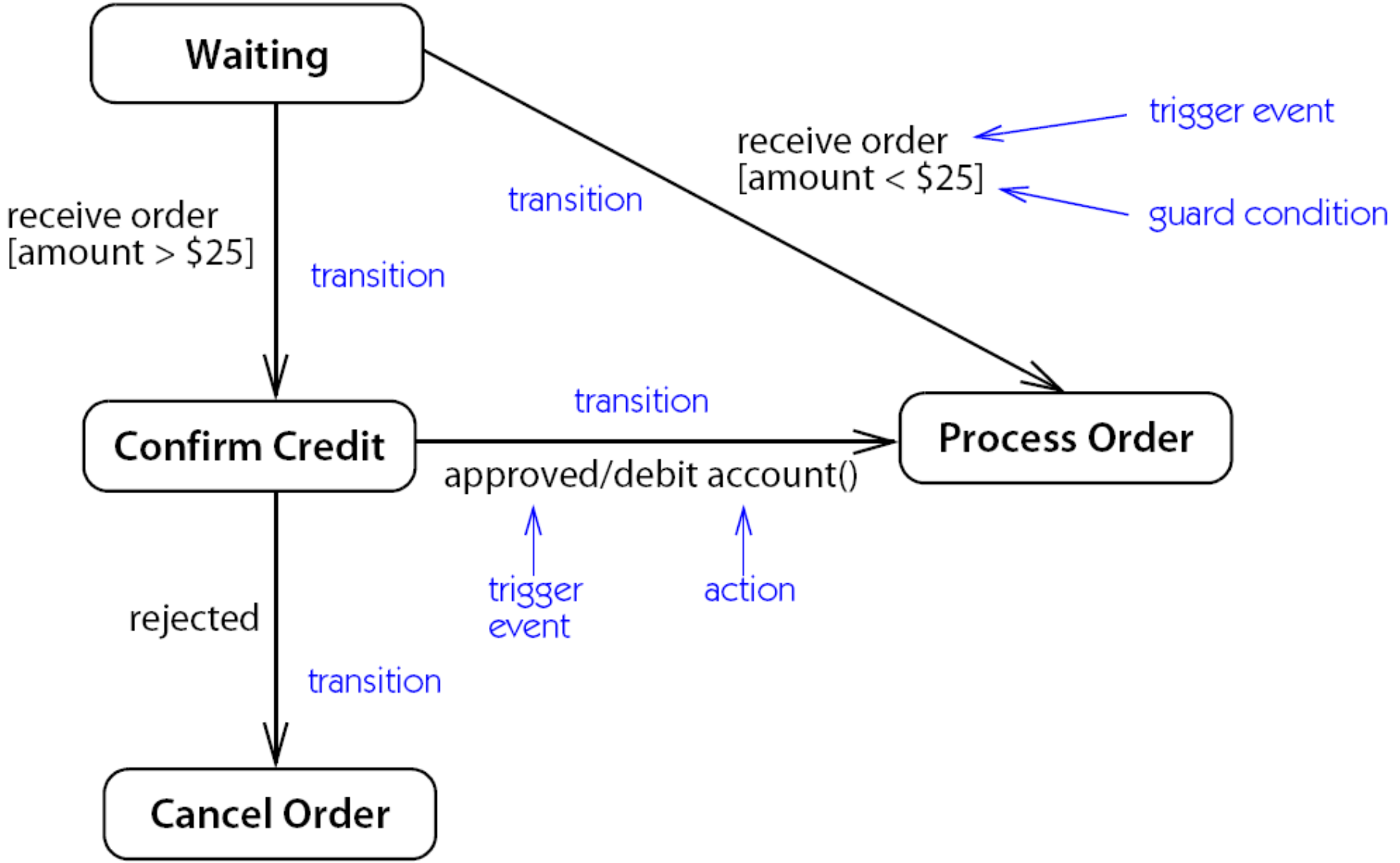


Figure 7-3. *External transitions*

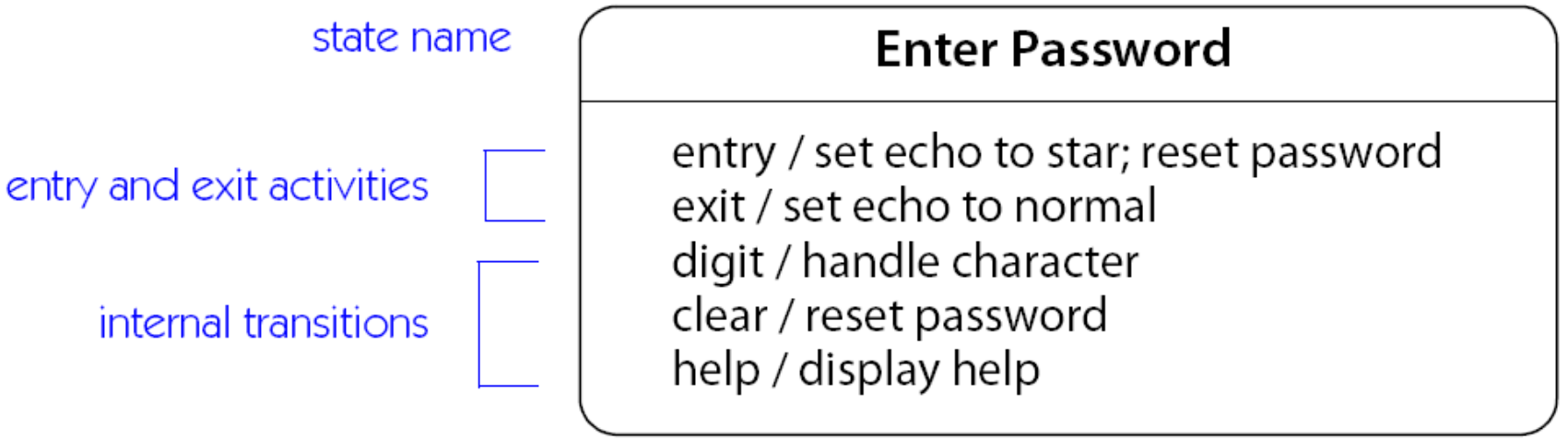
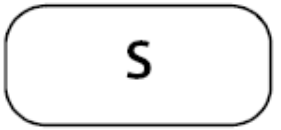
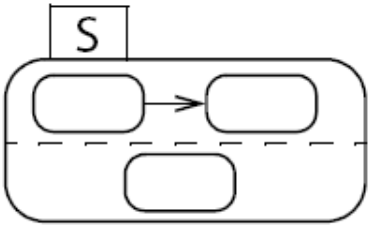
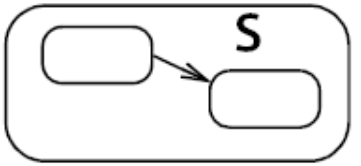


Figure 7-4. *Internal transitions, and entry and exit actions*

Table 7-3: *Kinds of States*

State Kind	*Description*	*Notation*
simple state	A state with no substructure	
orthogonal state	A state that is divided into two or more regions. One direct substate from each region is concurrently active when the composite state is active.	
nonorthogonal state	A composite state that contains one or more direct substates, exactly one of which is active at one time when the composite state is active	
initial state	A pseudostate that indicates the starting state when the enclosing state is invoked	

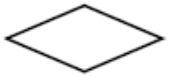
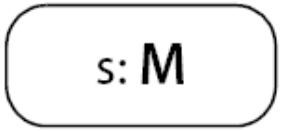
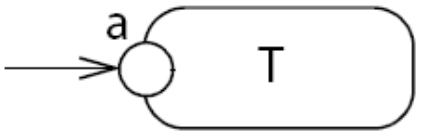
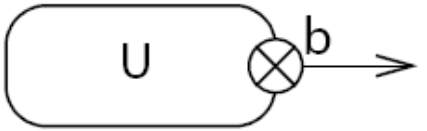
final state	A special state whose activation indicates the enclosing state has completed activity	
terminate	A special state whose activation terminates execution of the object owning the state machine	
junction	A pseudostate that chains transition segments into a single run-to-completion transition	
choice	A pseudostate that performs a dynamic branch within a single run-to-completion transition	
history state	A pseudostate whose activation restores the previously active state within a composite state	

Table 7-3: *Kinds of States*

State Kind	*Description*	*Notation*
submachine state	A state that references a state machine definition, which conceptually replaces the submachine state	
entry point	A externally visible pseudostate within a state machine that identifies an internal state as a target	
exit point	A externally visible pseudostate within a state machine that identifies an internal state as a source	

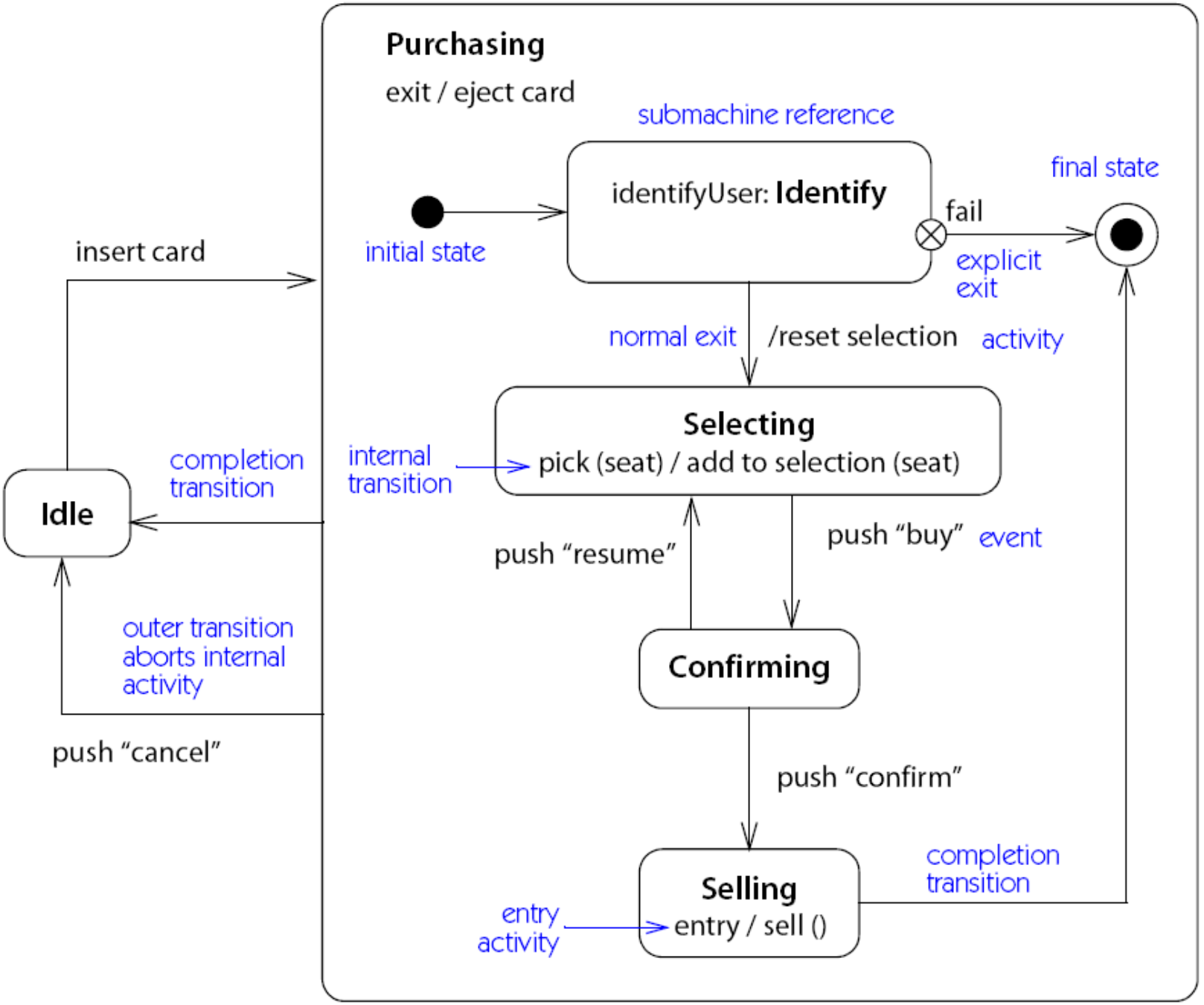


Figure 7-5. *State machine*

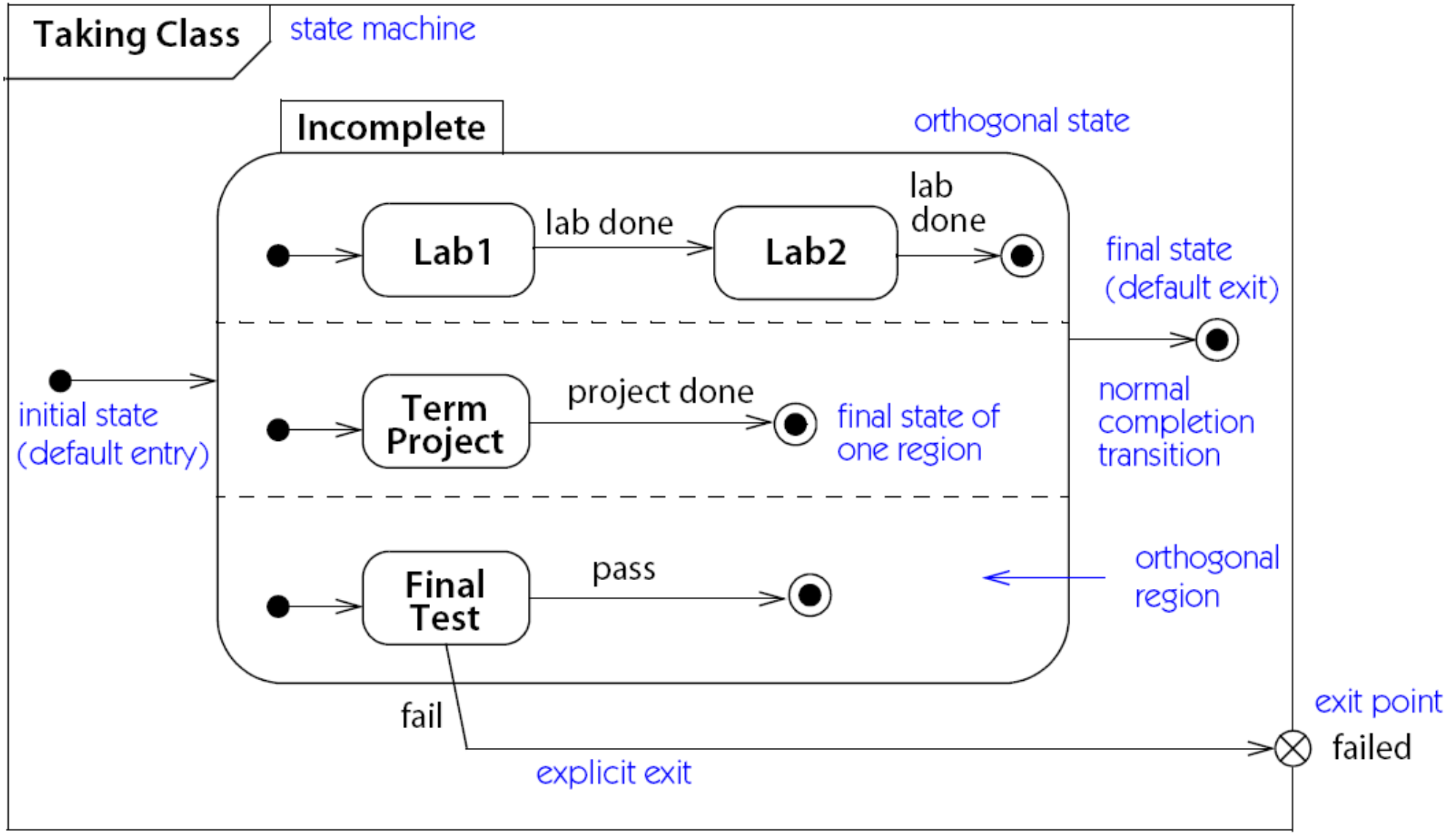


Figure 7-6. *State machine with orthogonal composite state*

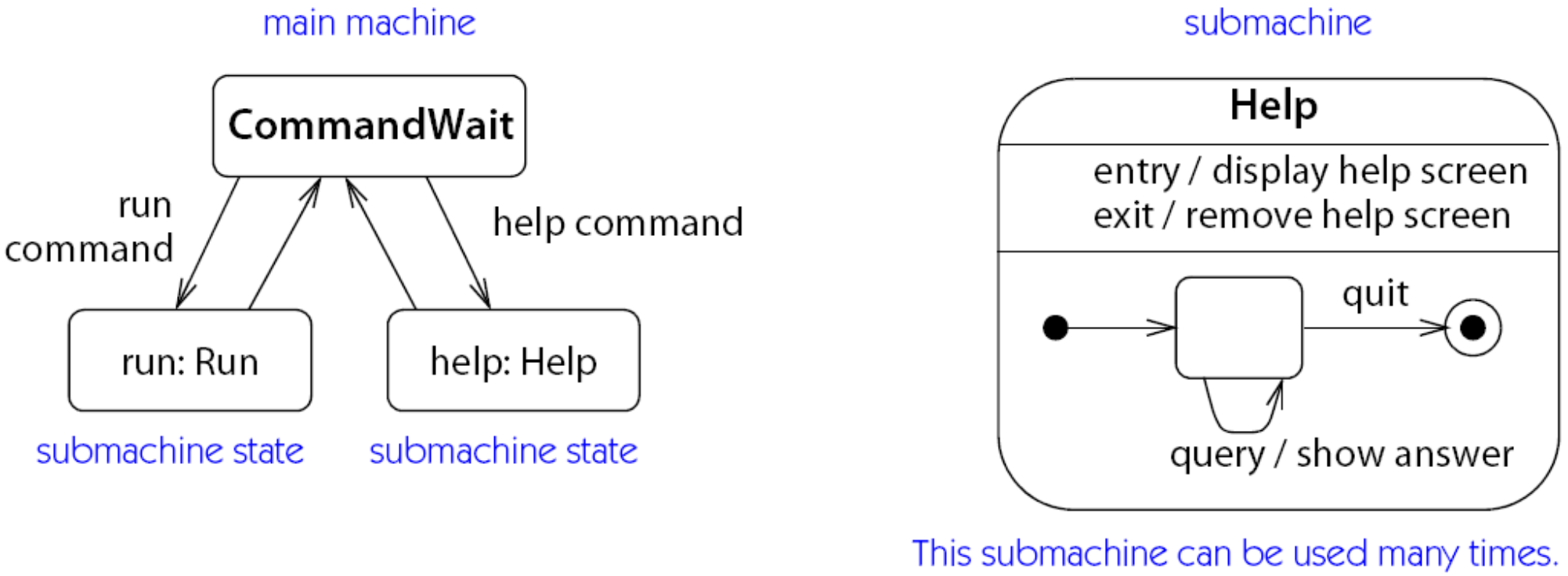


Figure 7-7. *Submachine state*

Chapter 8: Activity View 95
Overview 95
Activity 96
Activities and Other Views 98
Action 98

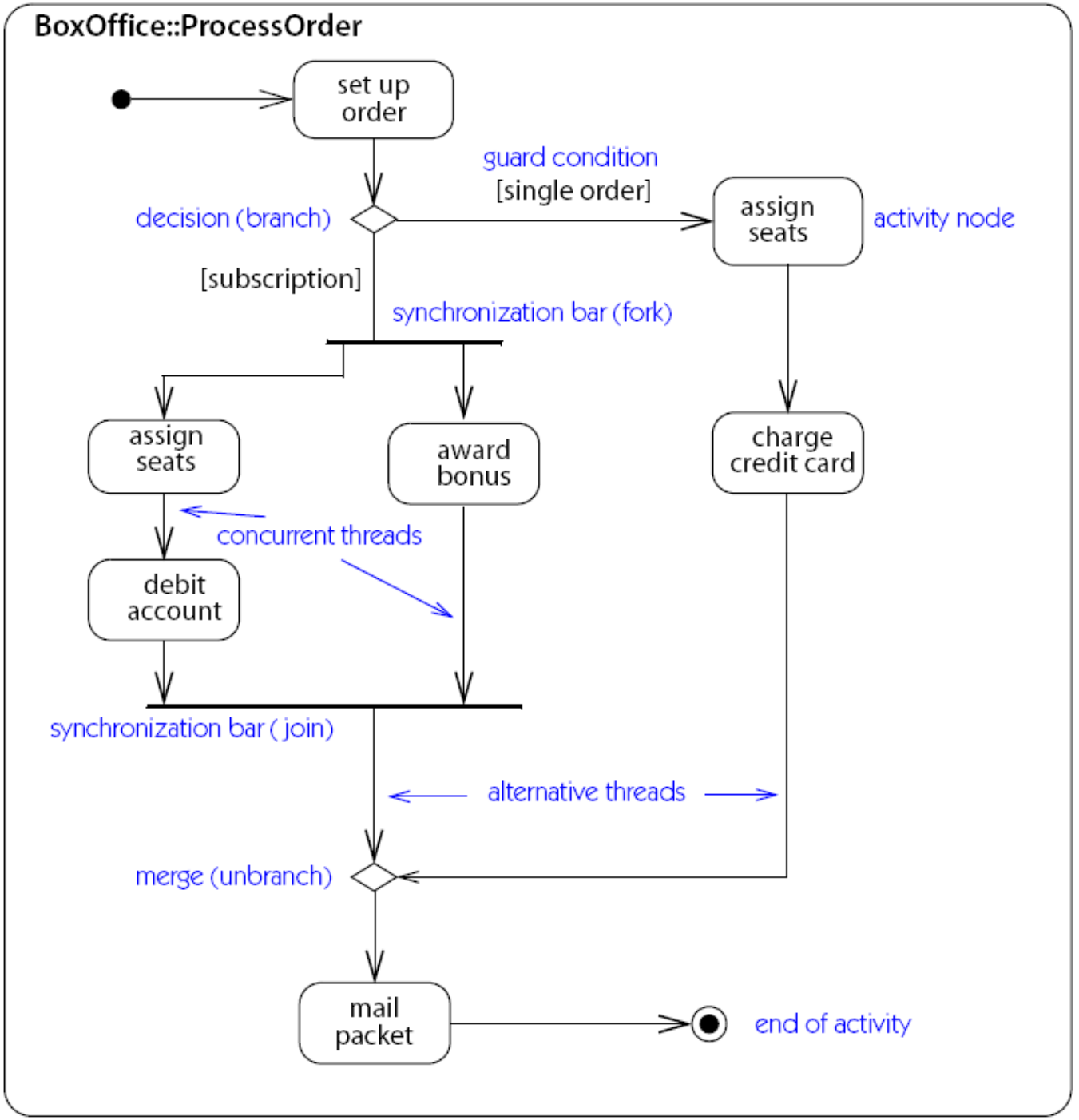


Figure 8-1. *Activity diagram*

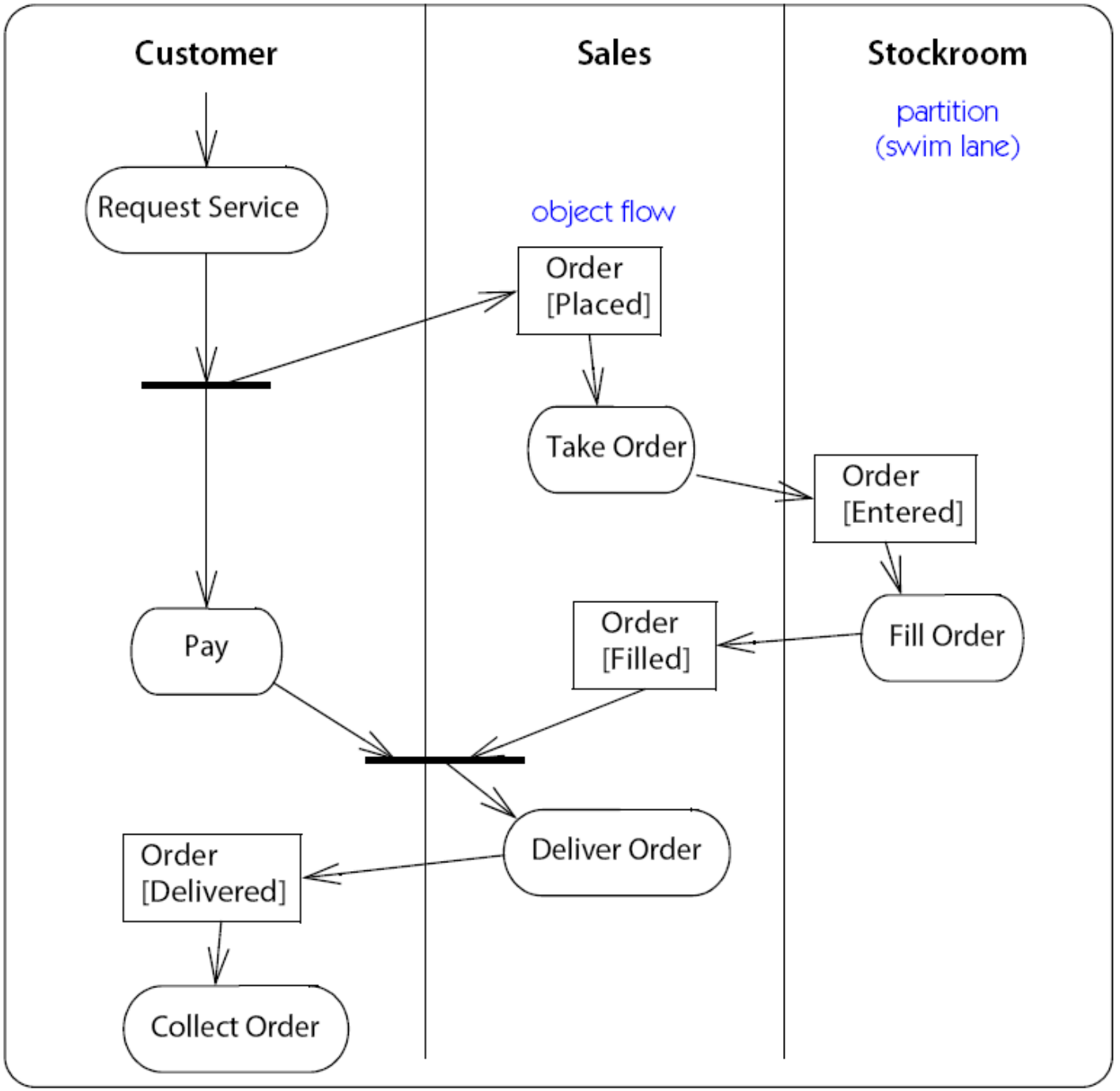


Figure 8-2. *Partitions and object flows*

Table 8-1: *Kinds of actions*

Category	*Actions*	*Purpose*
classification	readIsClassifiedObject reclassifyObject testIdentity	test classification change classification test object identity
communication	broadcastSignal callOperation reply (implicit) return sendObject sendSignal	broadcast normal call reply after explicit accept implicit action on activity end send signal as object send signal as argument list

Table 8-1: *Kinds of actions (continued)*

Category	*Actions*	*Purpose*
computation	accept call accept event addVariableValue applyFunction callBehavior clearVariable readSelf readVariable removeVariableValue writeVariable	inline wait for call inline wait for event add additional value to set mathematical computation nested behavior reset value in procedure obtain owning object identity obtain value in procedure remove value from set set value in procedure
control	startOwnedBehavior	explicit control
creation	createLinkObject createObject	create object from association create normal object
destruction	destroyObject	destroy object
exception	raiseException	raise exception in procedure

read	readExtent	get all objects
	readLink	get link value
	readLinkObjectEnd	get value from association class
	readLinkObject-EndQualifier	get qualifier value
	readStructuralFeature	get attribute value
time	durationObservation	measure time interval
	timeObservation	get current time
write	addStructuralFeature-Value	set attribute value
	clearAssociation	clear links
	clearStructuralFeature	clear attribute value
	createLink	add a link
	destroyLink	remove a link
	removeStructural-FeatureValue	remove value from set

Chapter 9: Interaction View 101

Overview 101
Interaction 101
Sequence Diagram 102
Communication Diagram 106

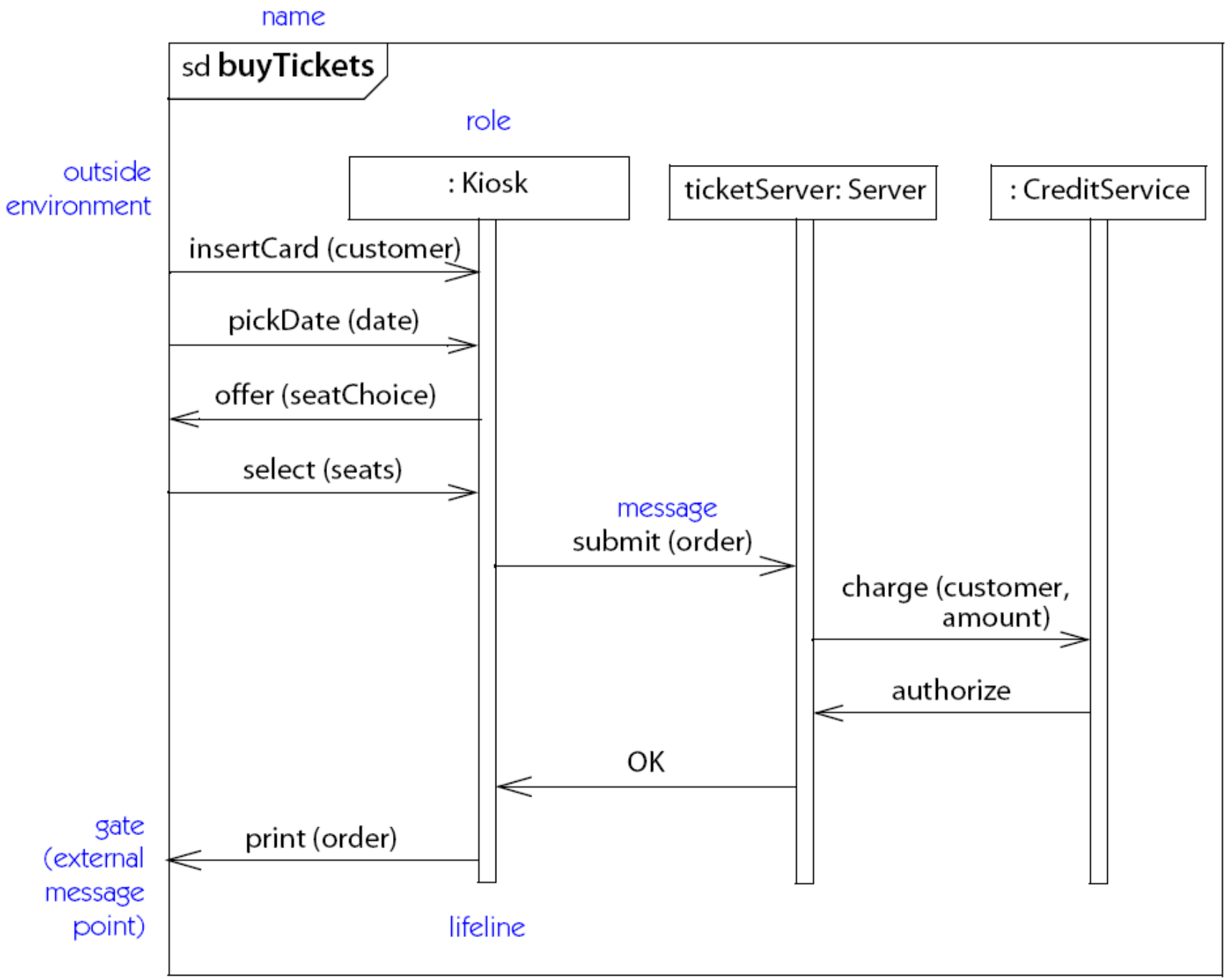


Figure 9-1. *Sequence diagram*

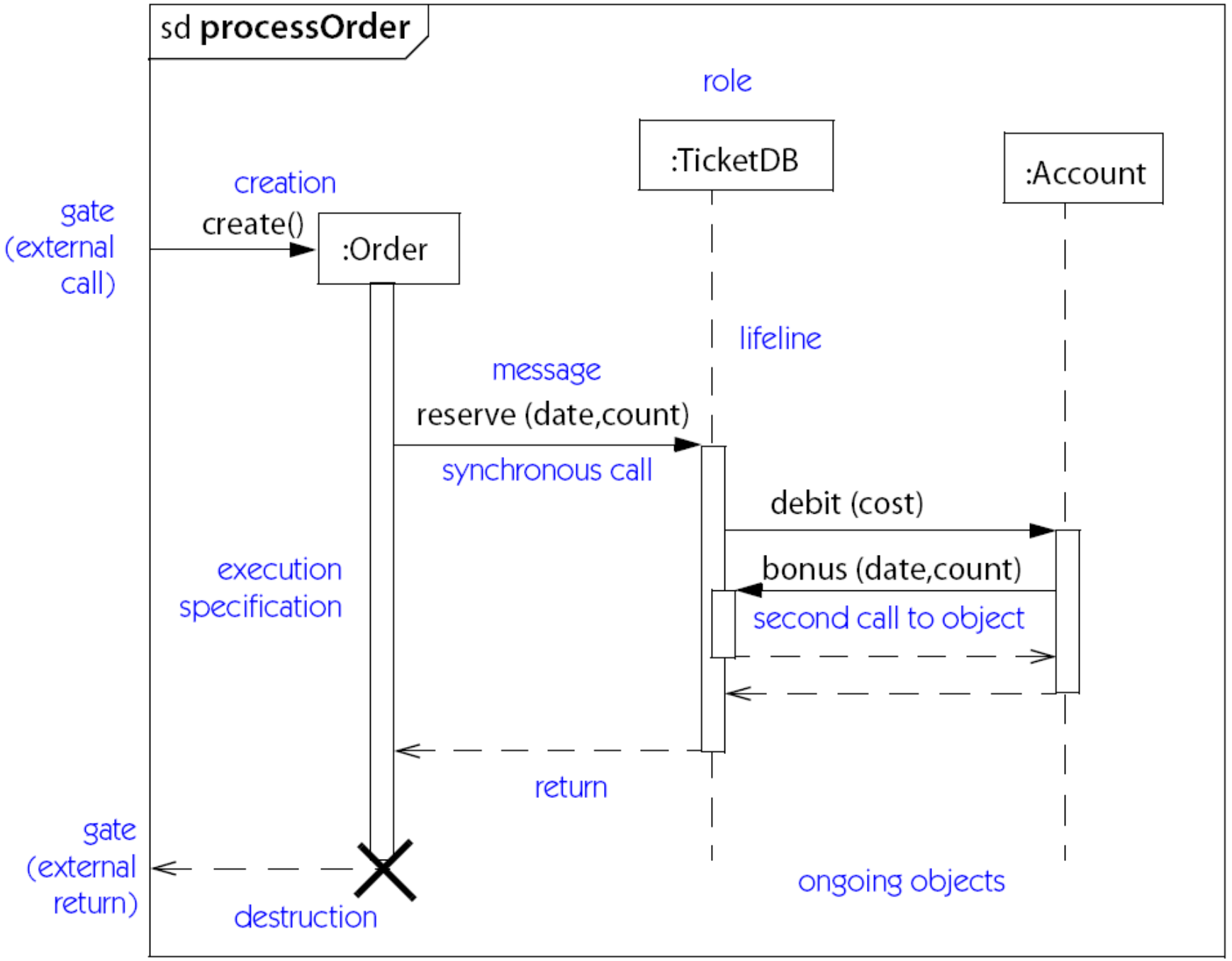


Figure 9-2. *Sequence diagram with execution specifications*

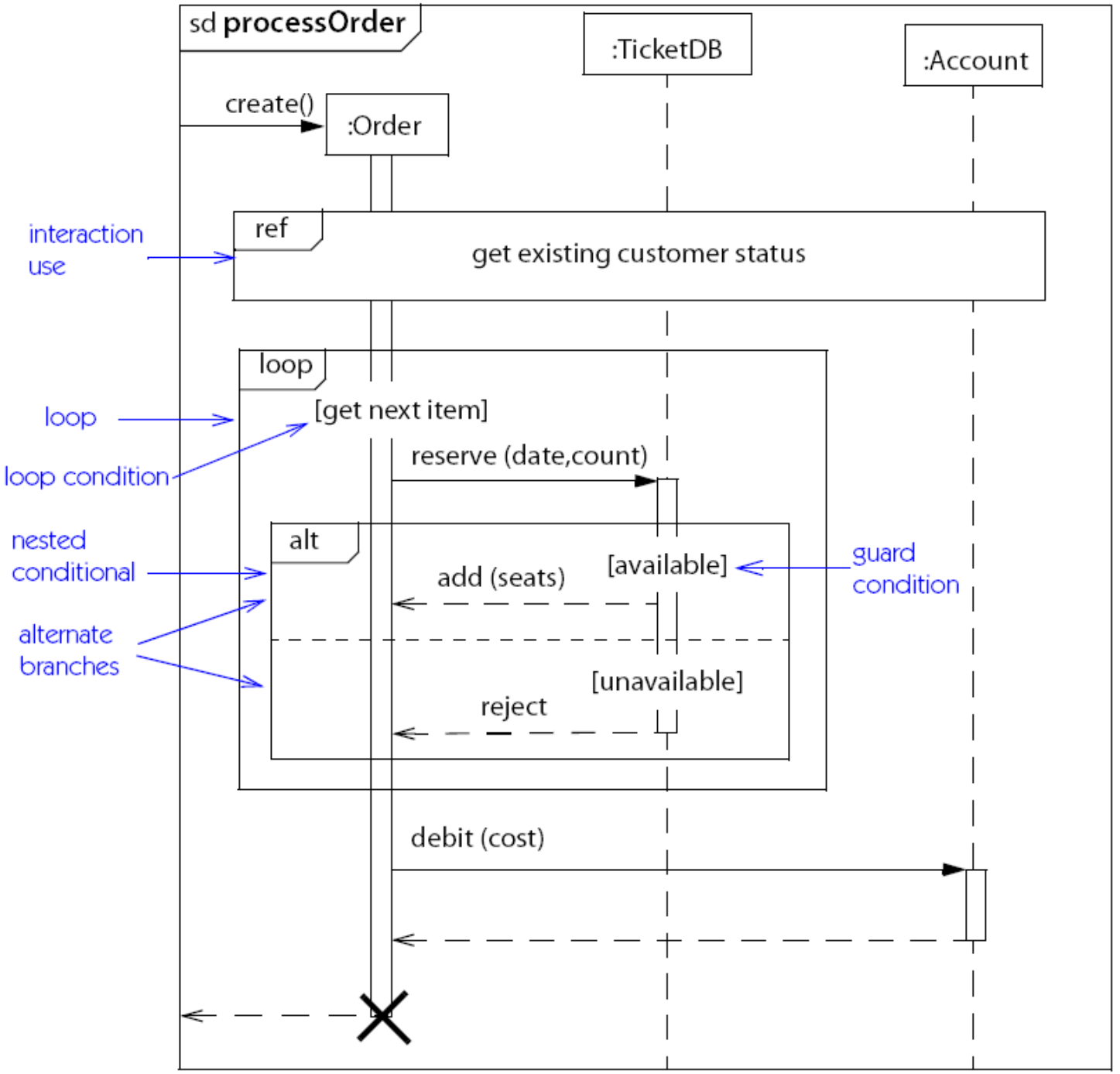


Figure 9-3. *Structured control constructs*

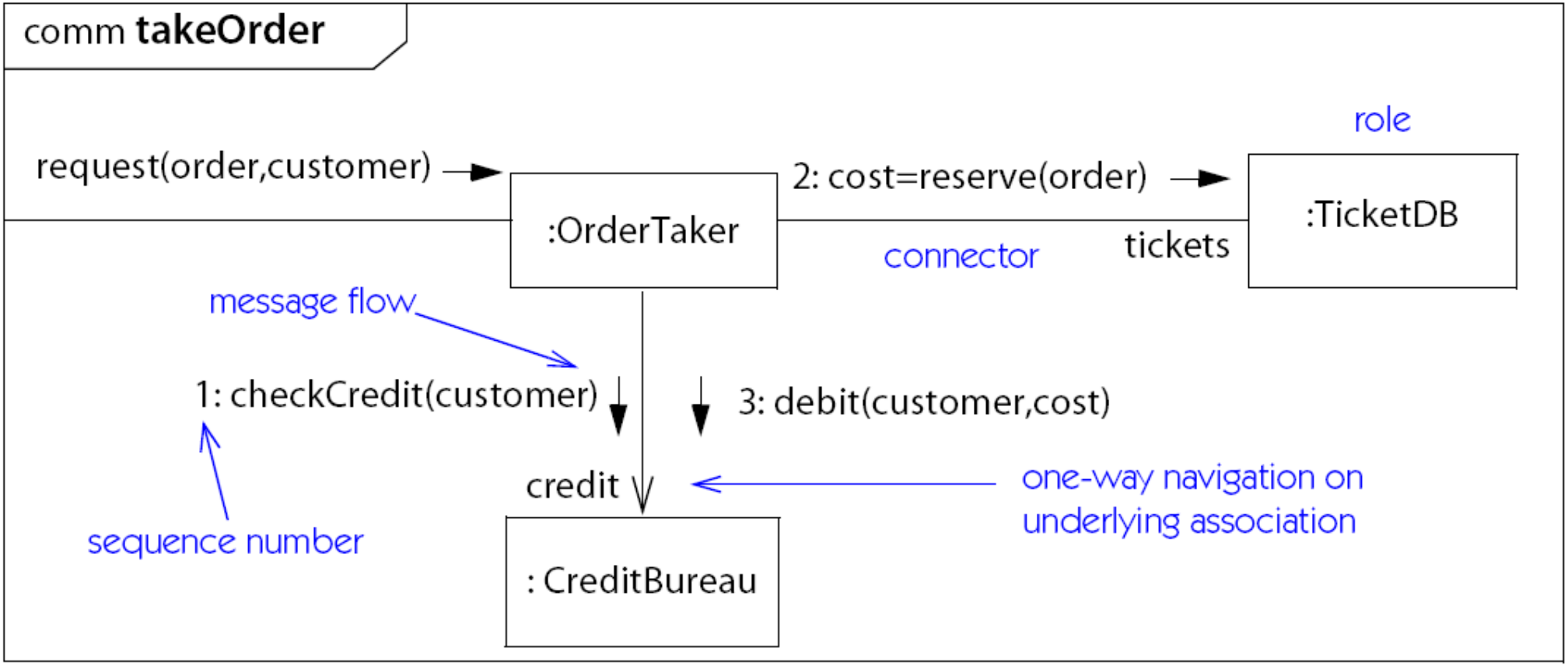


Figure 9-4. *Communication diagram*

Chapter 10: Deployment View 109

Overview 109
Node 109
Artifact 109

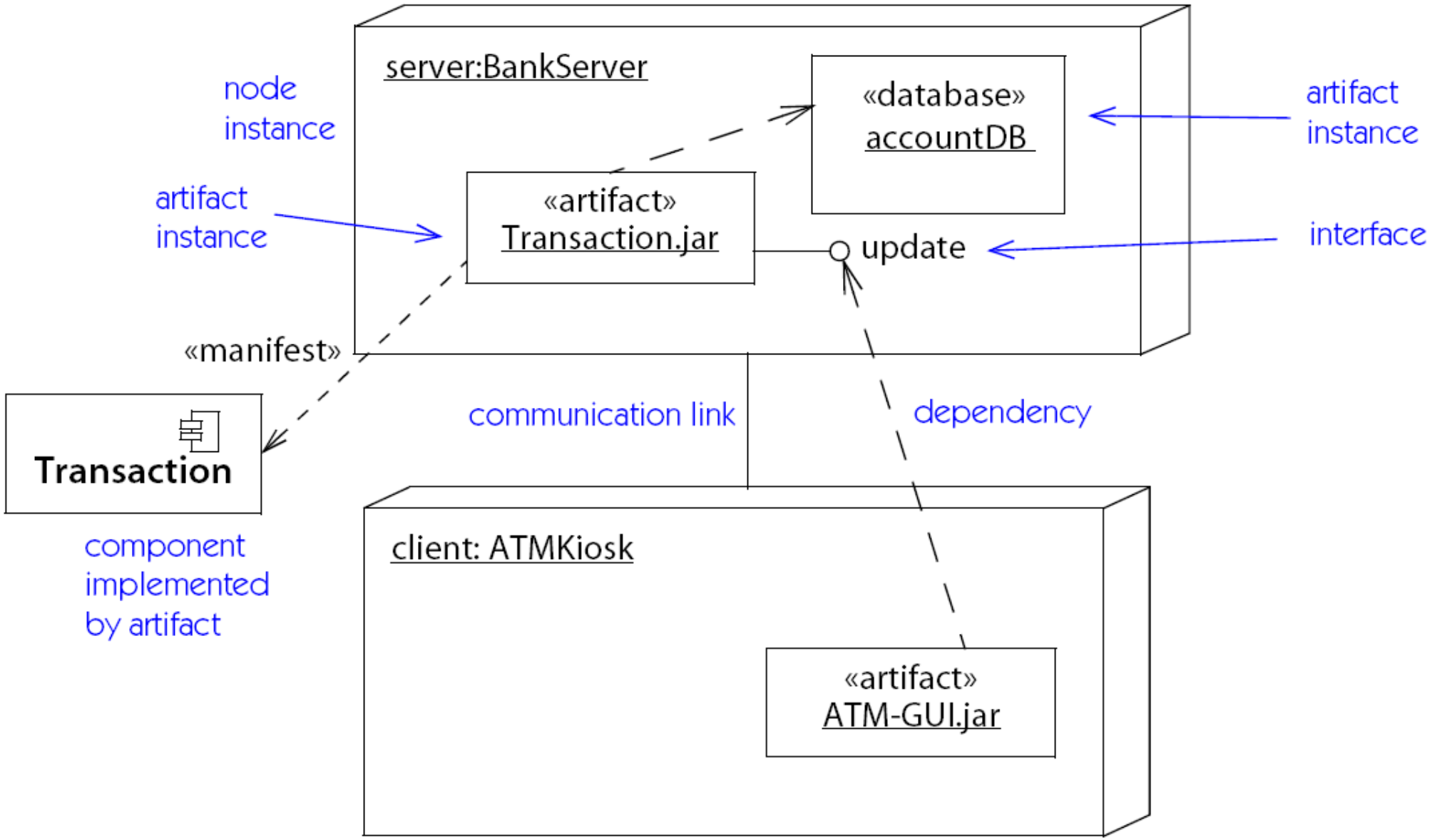


Figure 10-1. *Deployment diagram*

Chapter 11: Model Management View 111

Overview 111
Package 111
Dependencies on Packages. 112
Visibility. 113
Import 113
Model 114

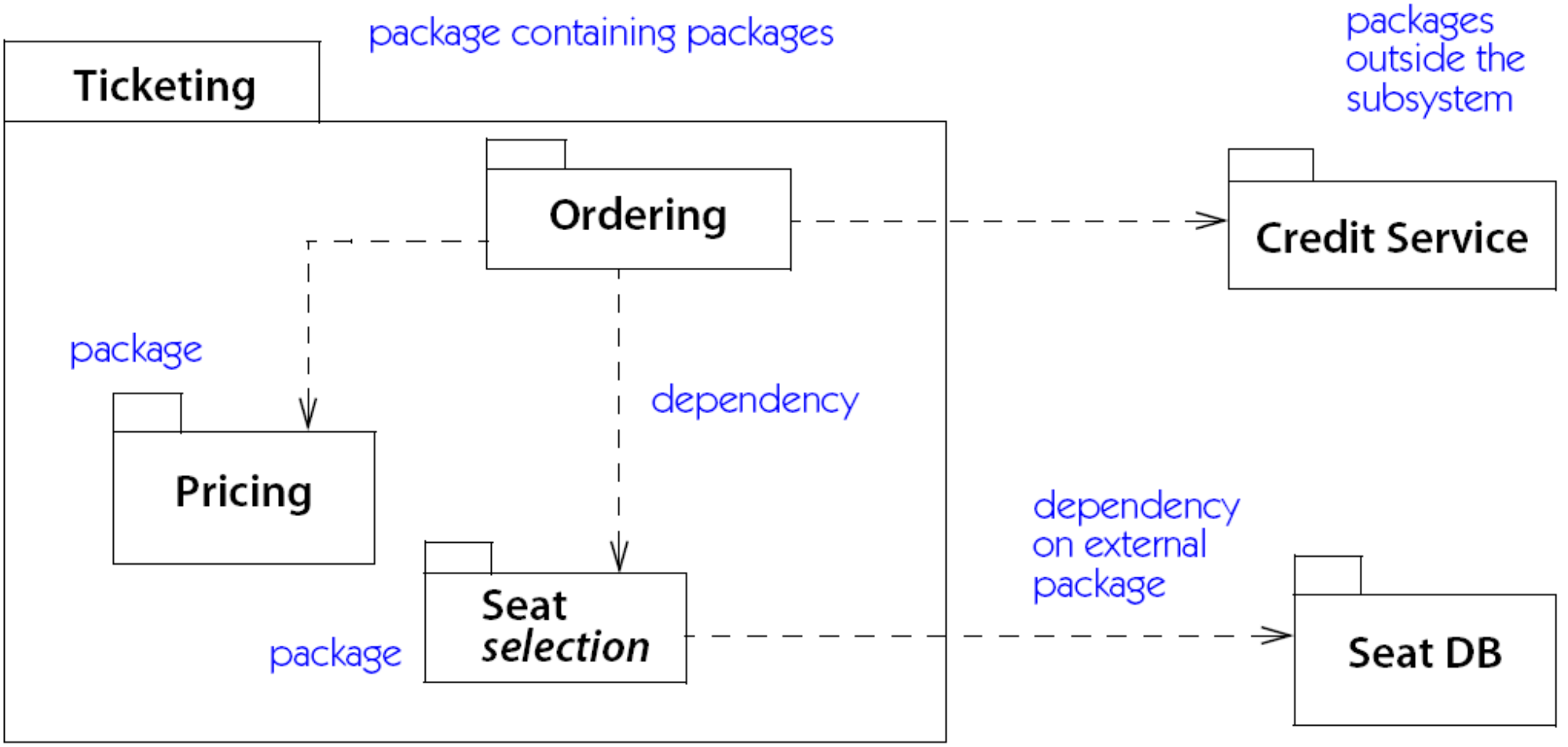


Figure 11-1. *Packages and their relationships*

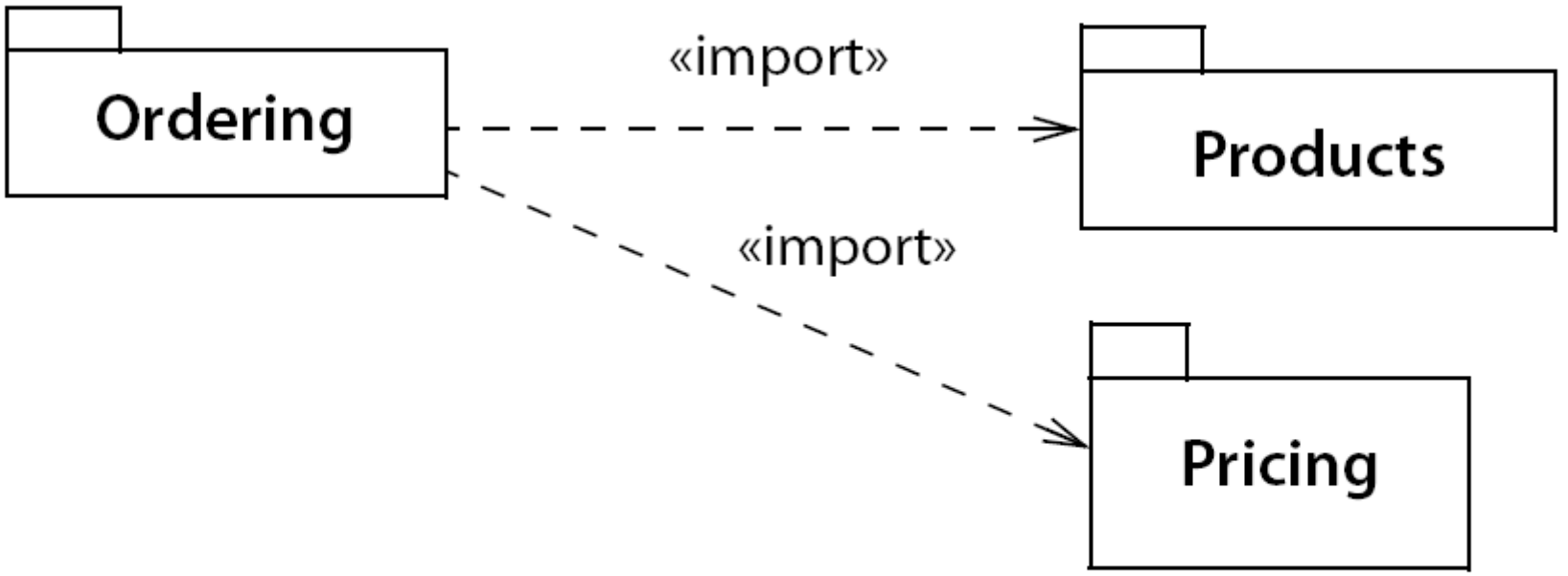


Figure 11-2. *Package import*

Chapter 12: Profiles 115
Overview 115
Stereotype 116
Tagged Value 117
Profile 118

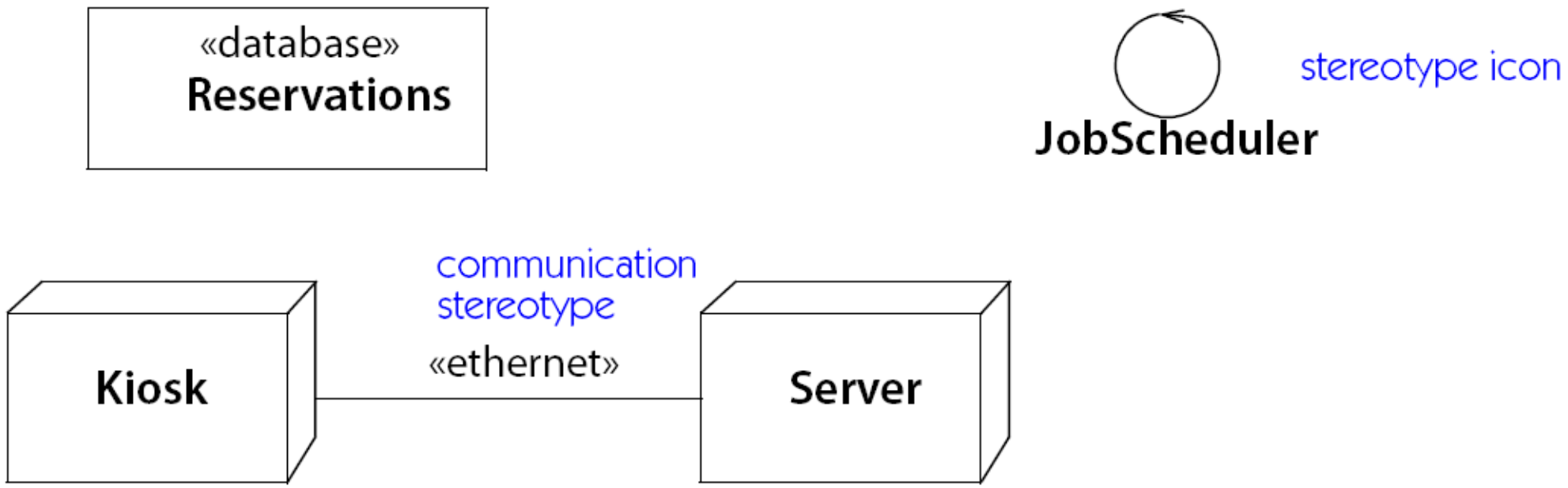


Figure 12-1. *Stereotypes*

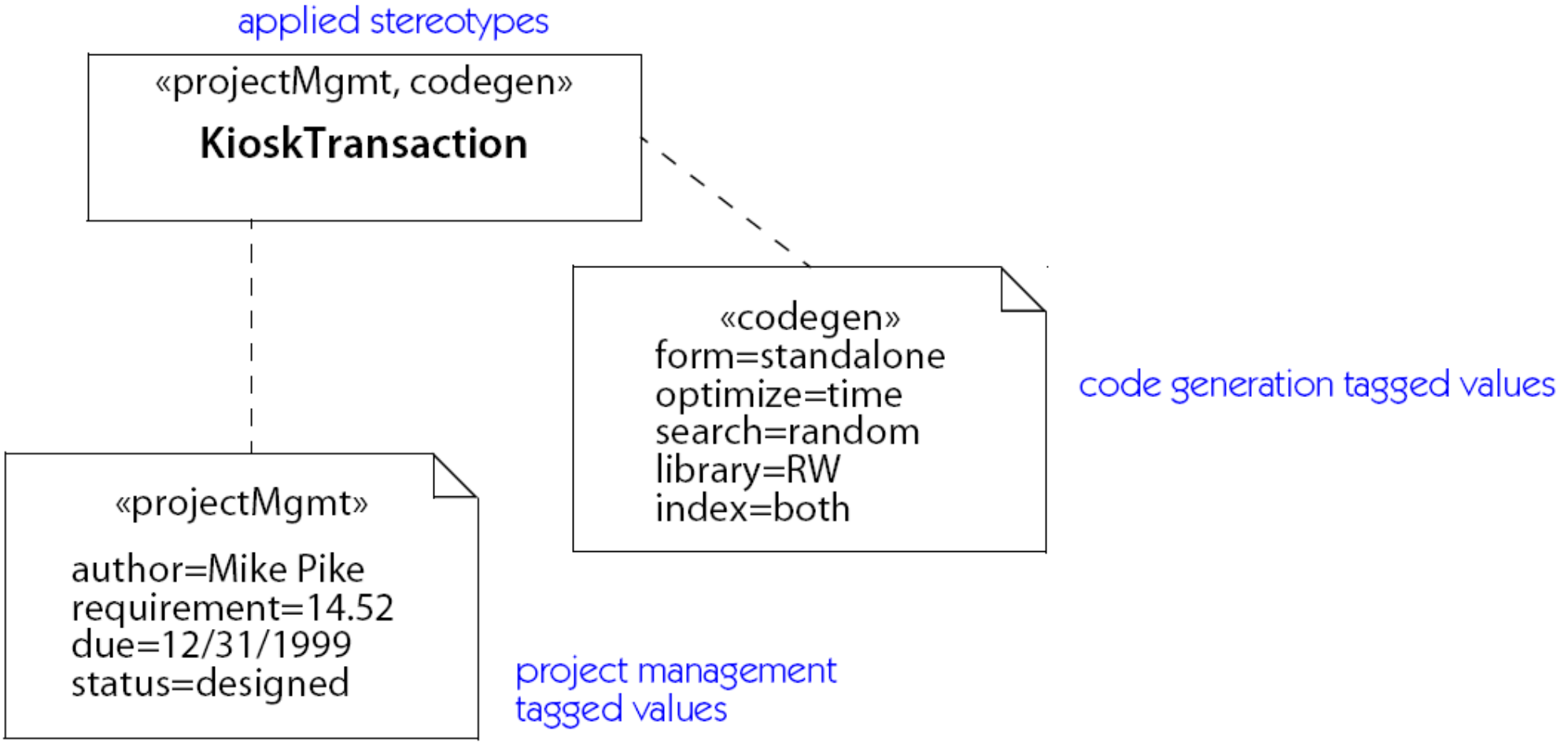


Figure 12-2. *Stereotypes and tagged values*

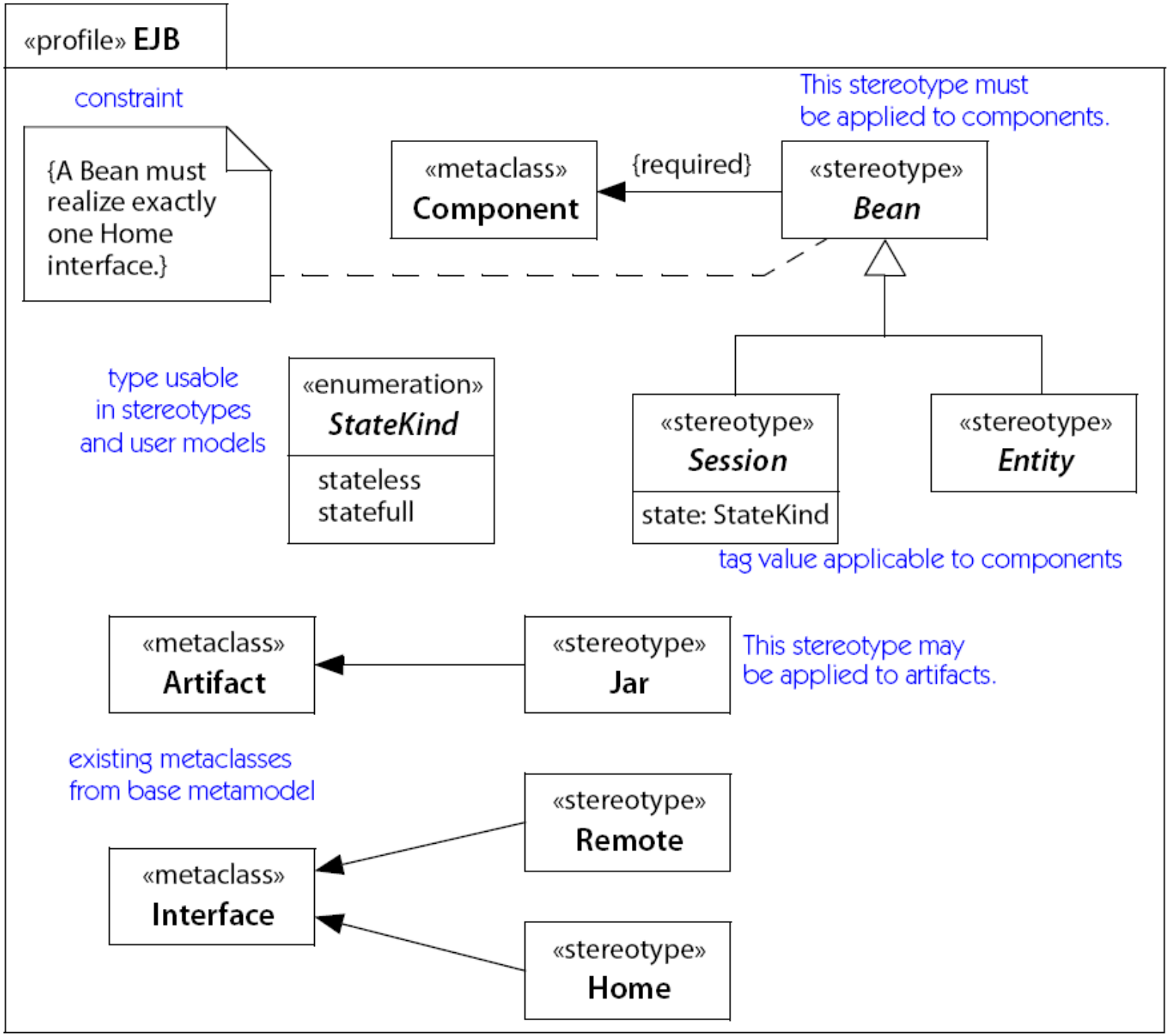


Figure 12-3. *Profile definition*

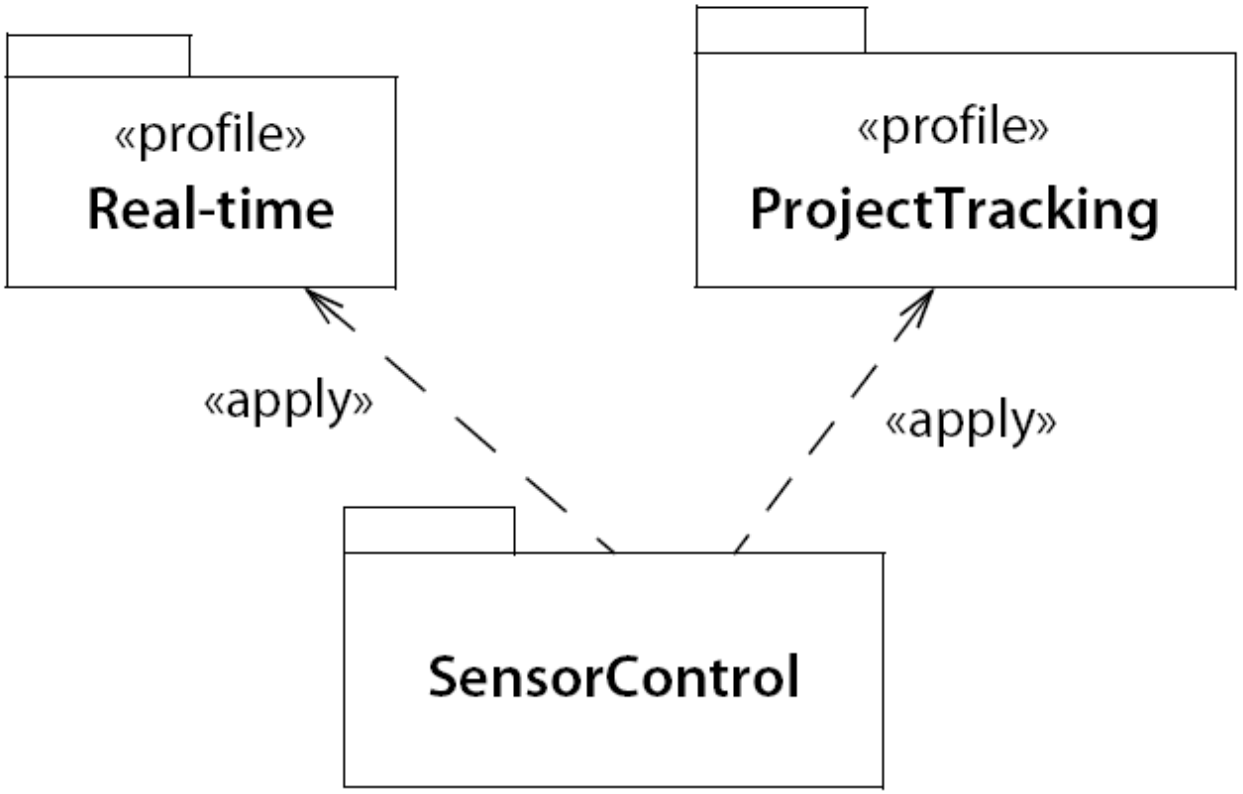


Figure 12-4. *Profile application*