Contents(1)

- **Goal and Scope**
- **Basic Concepts on OOT**
  - Basic Concepts to represent the world
  - Basic Concepts for Reuse
  - Information Hiding Principle and Java Program
  - Superiority of OOT
- **Modeling Techniques**
  - Static Model: Class and Association
  - Dynamic Model: State Machine
  - Dynamic Model: Interaction Diagram
  - Concurrency Description: Active Object and Multi-thread Programming
  - Outline of UML2.0

UML2（UML2.0）

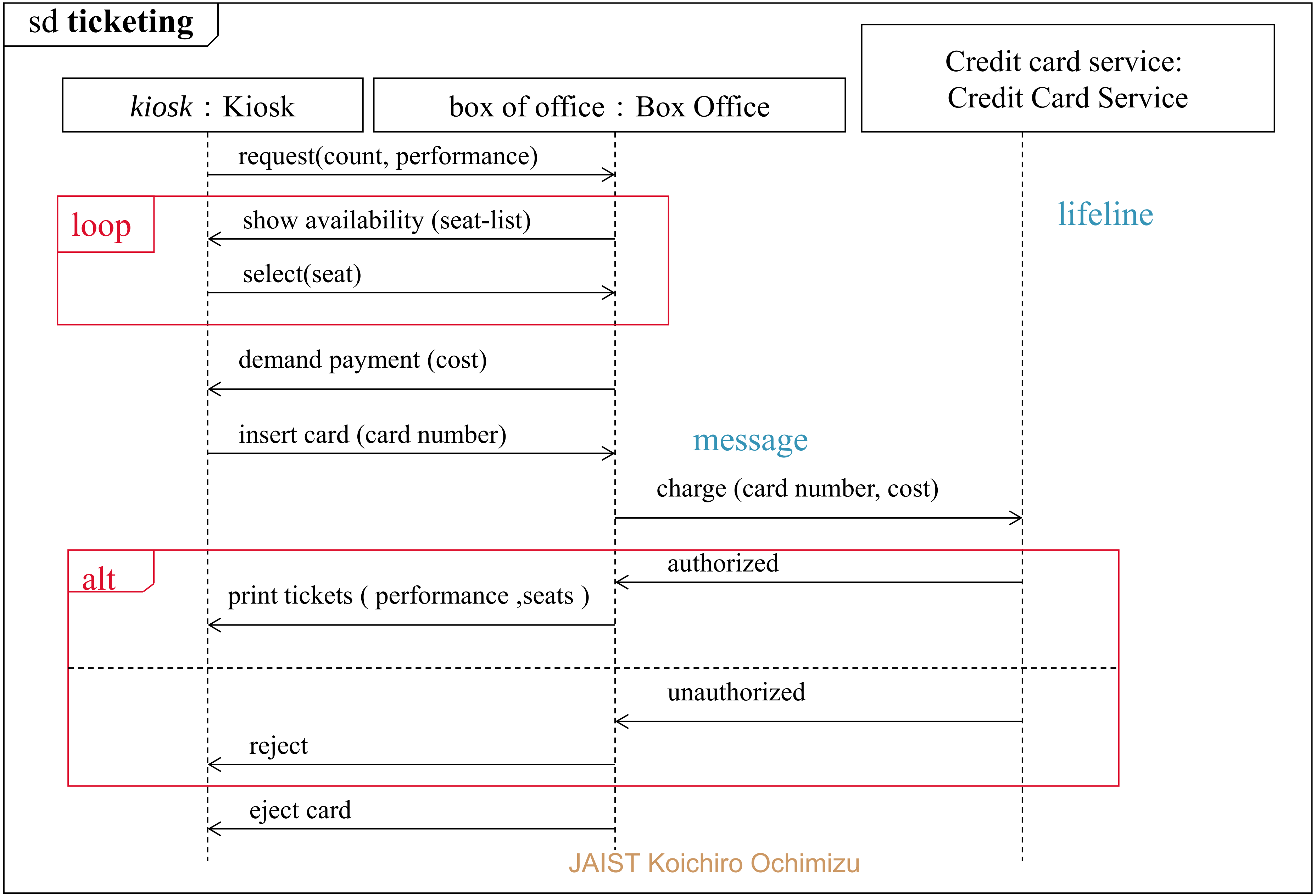
James Rumbaugh, Ivar Jacobson, Grady Booch,
“The Unified Modeling Language Reference Manual, Second Edition”,
Addison-Wesley, 2005.

Koichiro Ochimizu
Japan Advanced Institute of
Science and technologies
School of Information Science

New Features of UML2.0

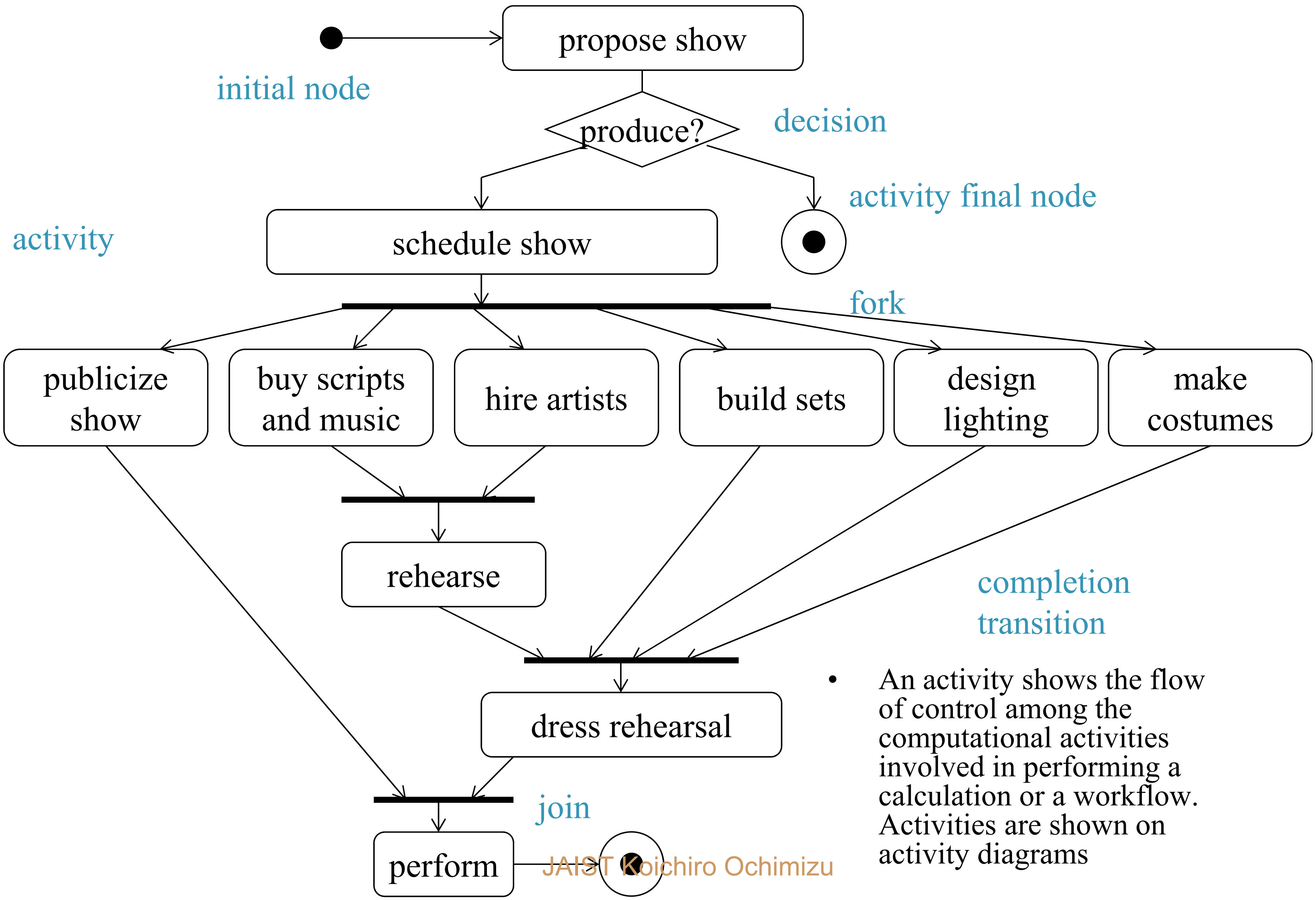
- Sequence Diagram constructs and notation based largely on the ITU（International Telecommunication Union）Message Sequence Chart standard, adapted to make it more object-oriented. MDA
- Decoupling of activity modeling concepts from state machines and use of notation popular in the business modeling community. Business Modeling
- Contextual modeling constructs for the internal composition of classes and collaborations. Theses constructs permit both loose and strict encapsulation and wiring of internal structures from smaller parts. Component Based Software Development
- Repositioning of components as design constructs and artifacts as physical entities that are deployed CBSD

Structured Control constructs in a Sequence Diagram


sd ticketing
kiosk : Kiosk
box of office : Box Office
Credit card service:
Credit Card Service
request(count, performance)
loop
show availability (seat-list)
select(seat)
lifeline
demand payment (cost)
insert card (card number)
message
charge (card number, cost)
alt
authorized
print tickets (performance ,seats)
unauthorized
reject
eject card

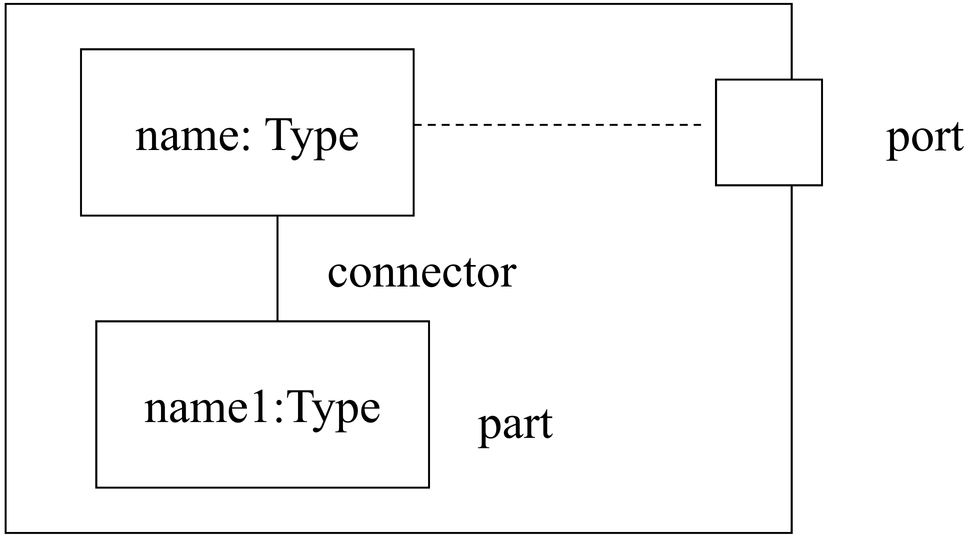
JAIST Koichiro Ochimizu

Activity View (Activity Diagram)



- An activity shows the flow of control among the computational activities involved in performing a calculation or a workflow. Activities are shown on activity diagrams

Structured Classifier

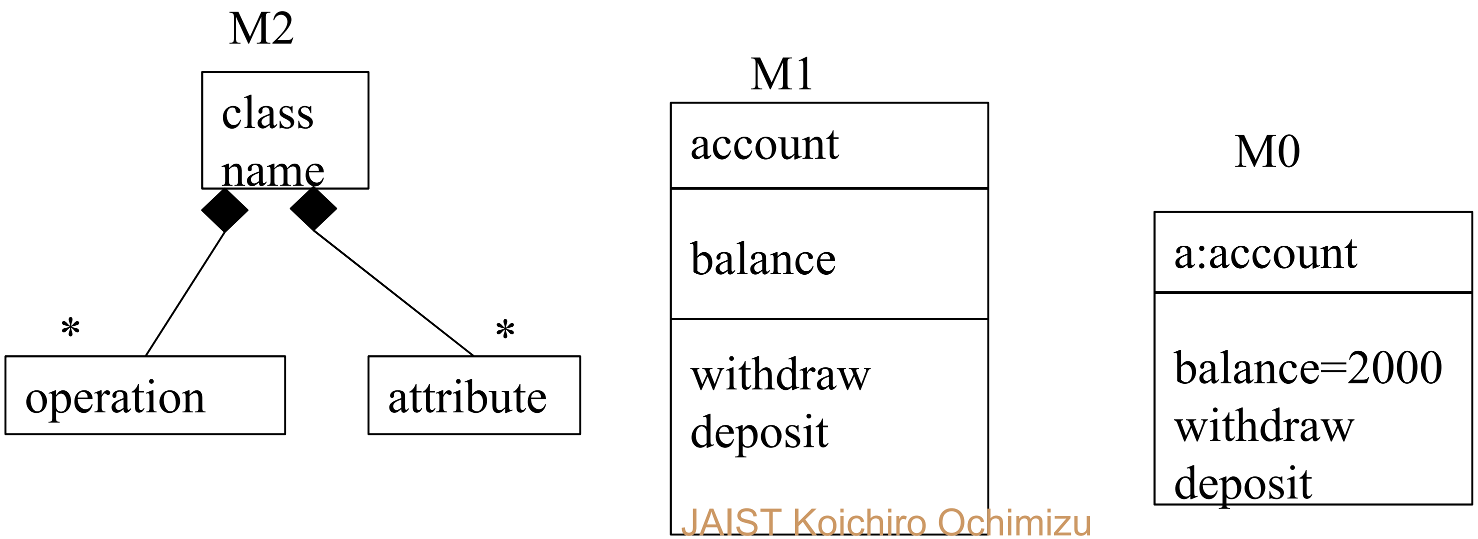


- A structured classifier is a classifier with internal structure.
- It contains a set of parts connected by connectors.
- An part has a type and a multiplicity within its container.
- An connector is a contextual relationship between two parts in a structured classifier.
- Structured classifiers may be tightly encapsulated by forcing all interactions between external environment and the internal parts to pass through ports.
- A port is an interaction point with well-defined interface.
- Messages received by a port are automatically forwarded to the part.
- Each port has a set of provides interfaces and required interfaces that define its external interactions.

J.Rumbaugh, I.Jacobson, G.Booch,"The Unified Modeling Language Reference Manual, Second Edition" Addison-Wesley 2005

Meta model

- A metamodel is the description of a model
- A UML metamodel defines the structure of UML models.
- A UML structured classifier is a type of *classifier* that is similar to a *class*. The difference being, it shows the internal wiring of classes through ports, connectors, and parts.
- A classifier is different from a *class*. In fact, classes are types of classifiers. Classifiers are the parent class of several elements in the UML, including *classes*, *use cases*, *artifacts*, and *components*.



M2

class name

*

operation

*

attribute

M1

account

balance

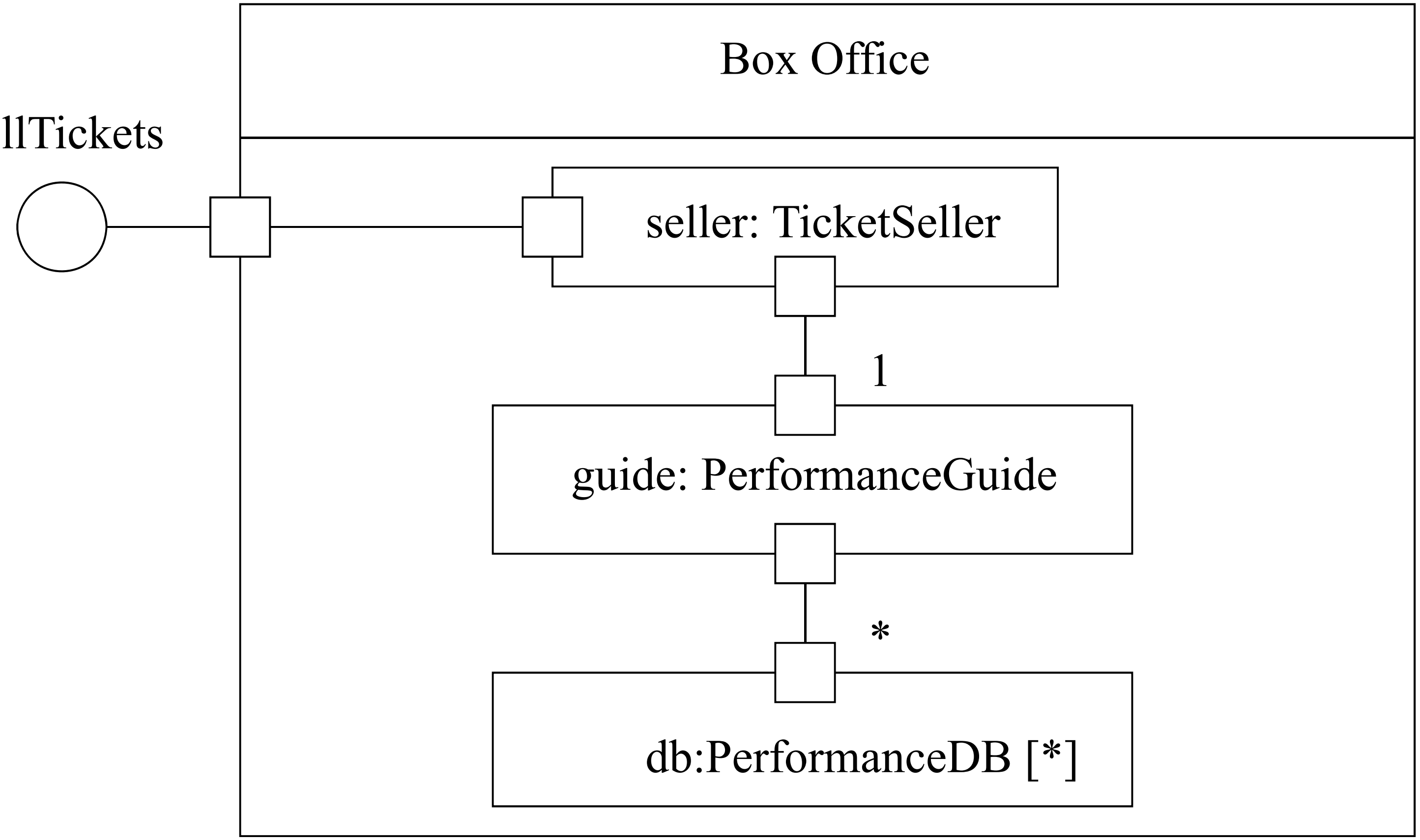
withdraw
deposit

M0

a:account

balance=2000
withdraw
deposit

Design View (Internal structure diagram)



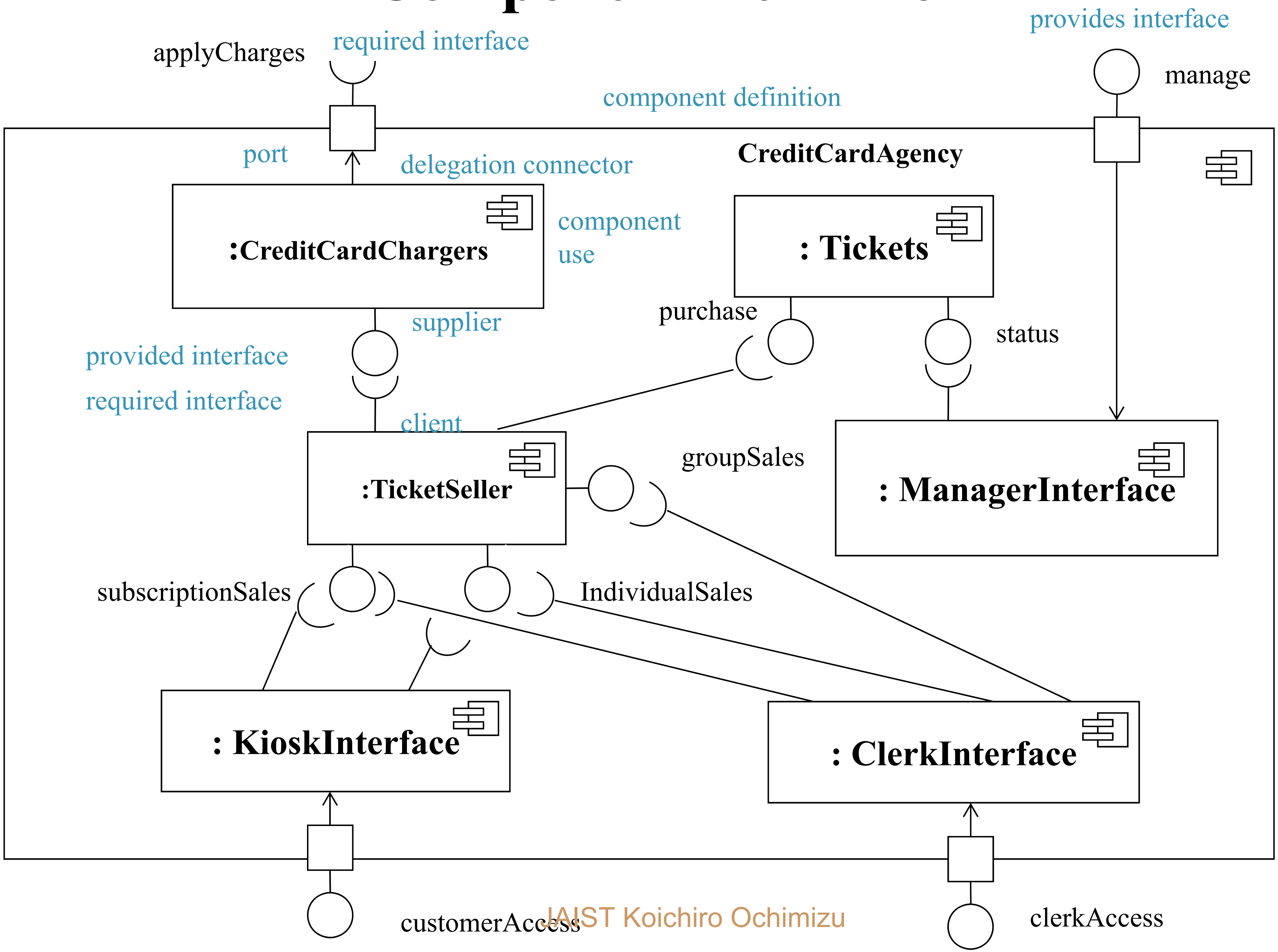
- Each port has a set of provides interfaces and required interfaces that define its external interactions. A provided interface specifies the services that a message to the port may request. A required interface specifies the services that a message from the port may require from the external environment.

J.Rumbaugh, I.Jacobson, G.Booch,"The Unified Modeling Language Reference Manual, Second Edition" Addison-Wesley 2005

Design View (component diagram)

- A component diagram is a kind of structured classifier, so its internal structure may be defined on an internal structure diagram.
- A component diagram shows the components in a system – that is, the software units from which the application is constructed. A small circle attached to a component or a class is a provided interface- a coherent set of services made available by a component or class.
- A small semicircle attached to a component or a class is a required interface – a statement that the component or class needs to obtain services from an element that provides them.

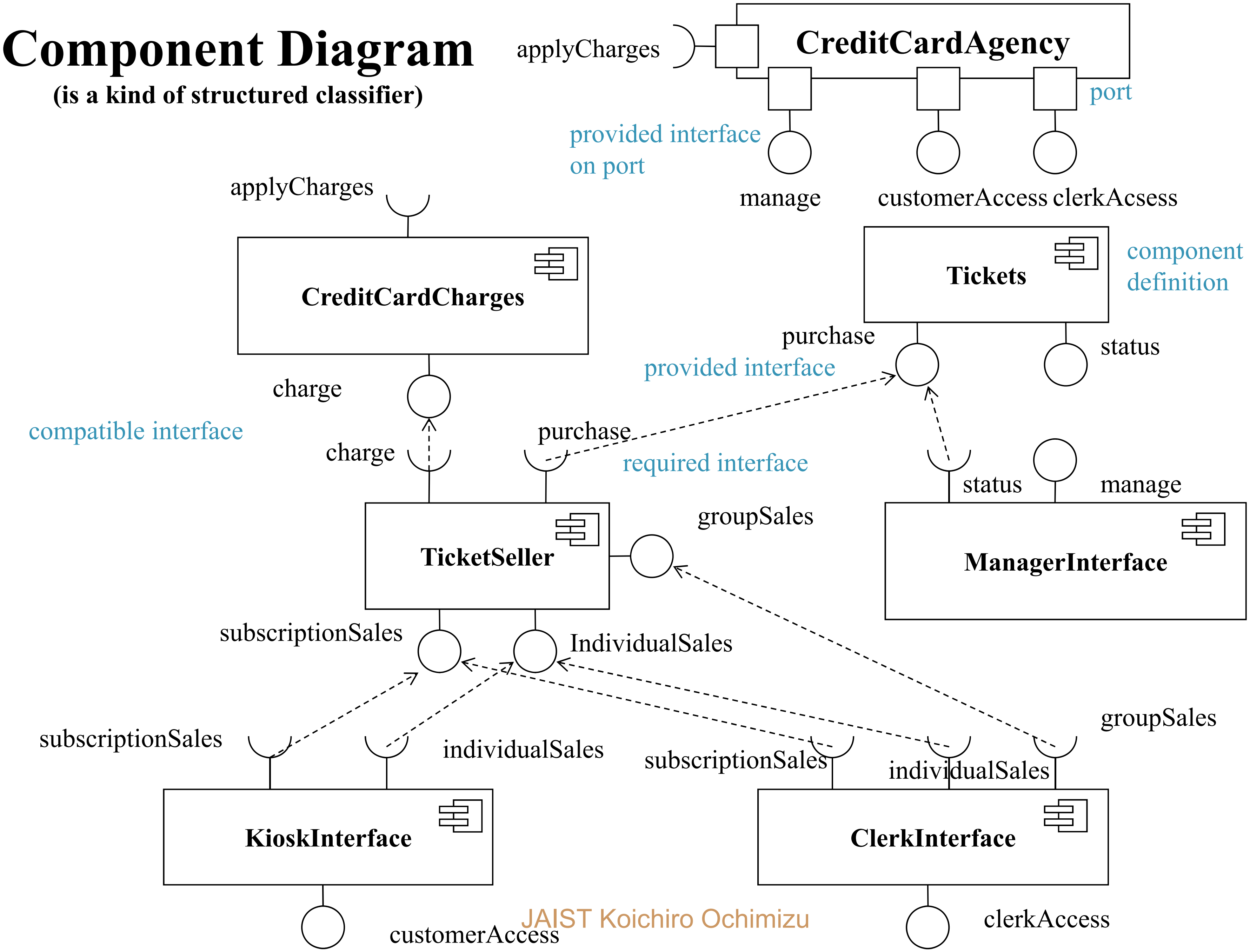
Component Definition



JAIST Koichiro Ochimizu

Component Diagram

(is a kind of structured classifier)

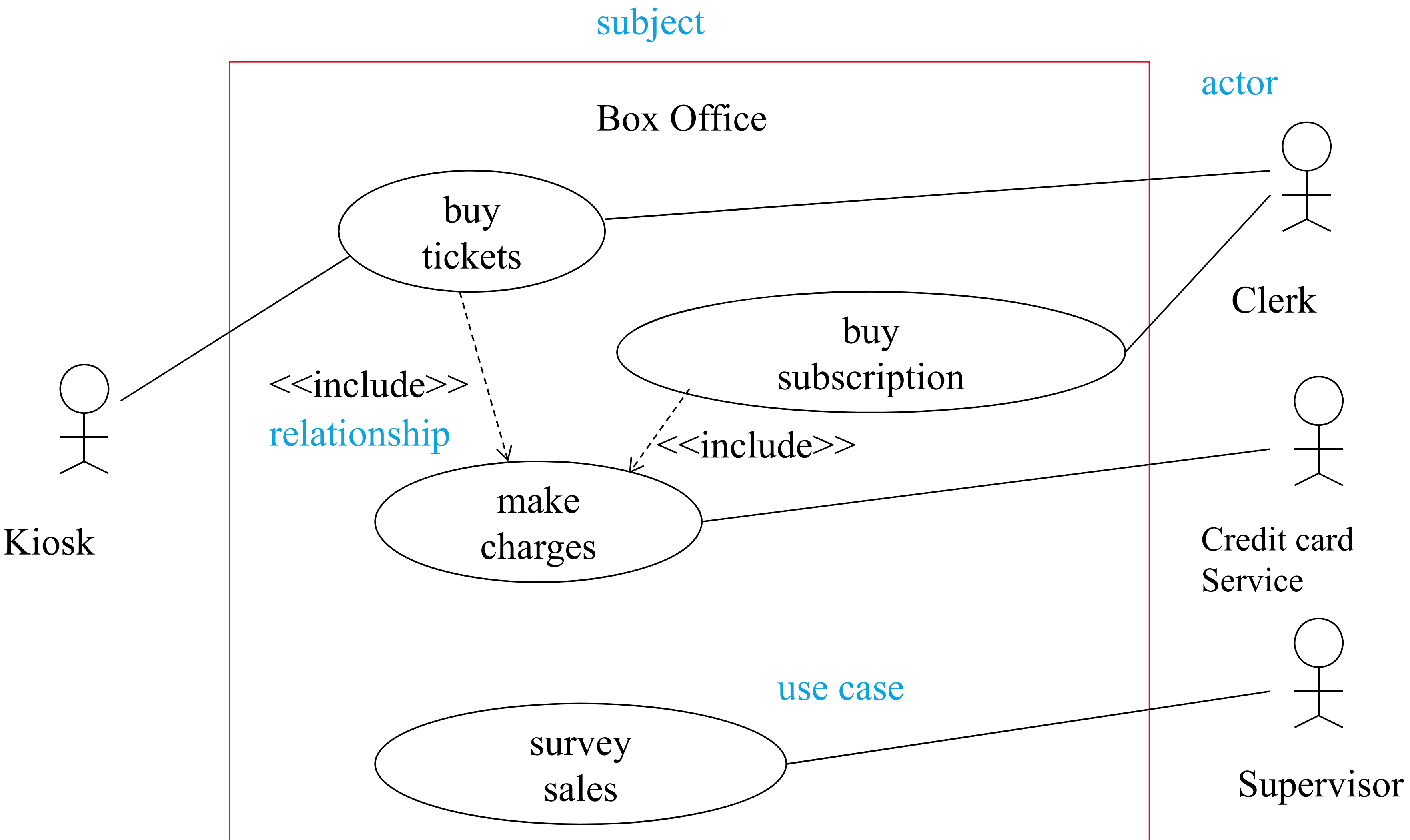


UML2.0 Views

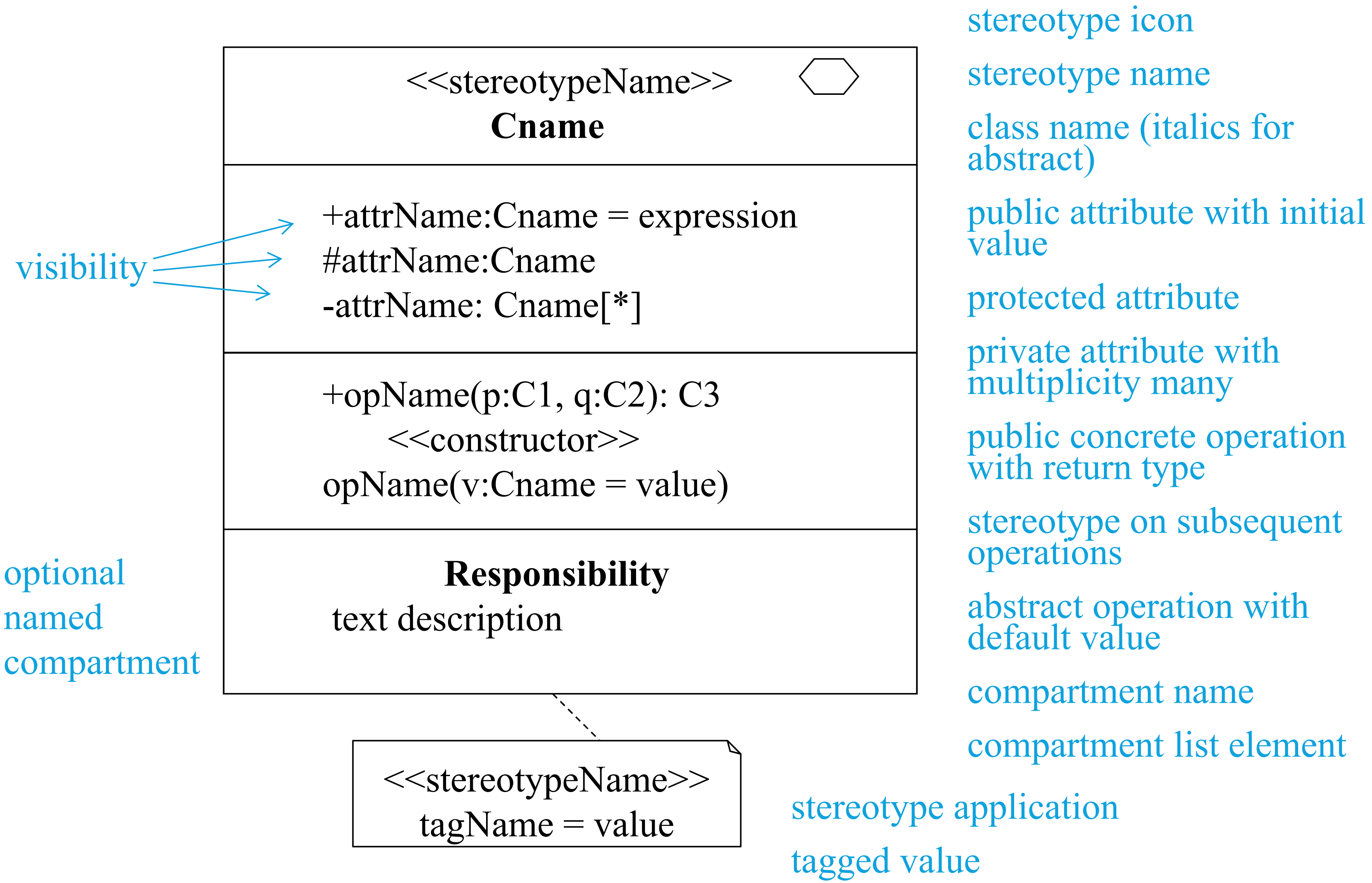
- Major Area,
  - View
    - Diagram
      - Main Concepts
- structural
  - static view ： class diagram
  - design view ： internal structure (connector, interface, part, port, provided interface, role, required interface), collaboration diagram (connector, collaboration use, role), component diagram (component, dependency, port, provided interface, realization, required interface, subsystem)
  - use case view ： usecase diagram
- dynamic
  - state machine view ： state machine diagram
  - activity view ： activity diagram
  - interaction view ： sequence diagram, communication diagram
- physical
  - deployment view ： deployment diagram
- model management
  - model management view ： package diagram
  - profile ： package diagram

Other Modifications

Use Case View (Use case diagram)



Class Content



<<stereotypeName>>
Cname

+attrName:Cname = expression
#attrName:Cname
-attrName: Cname[*]

+opName(p:C1, q:C2): C3
<<constructor>>
opName(v:Cname = value)

Responsibility
text description

<<stereotypeName>>
tagName = value

visibility

optional named compartment

stereotype icon

stereotype name

class name (italics for abstract)

public attribute with initial value

protected attribute

private attribute with multiplicity many

public concrete operation with return type

stereotype on subsequent operations

abstract operation with default value

compartment name

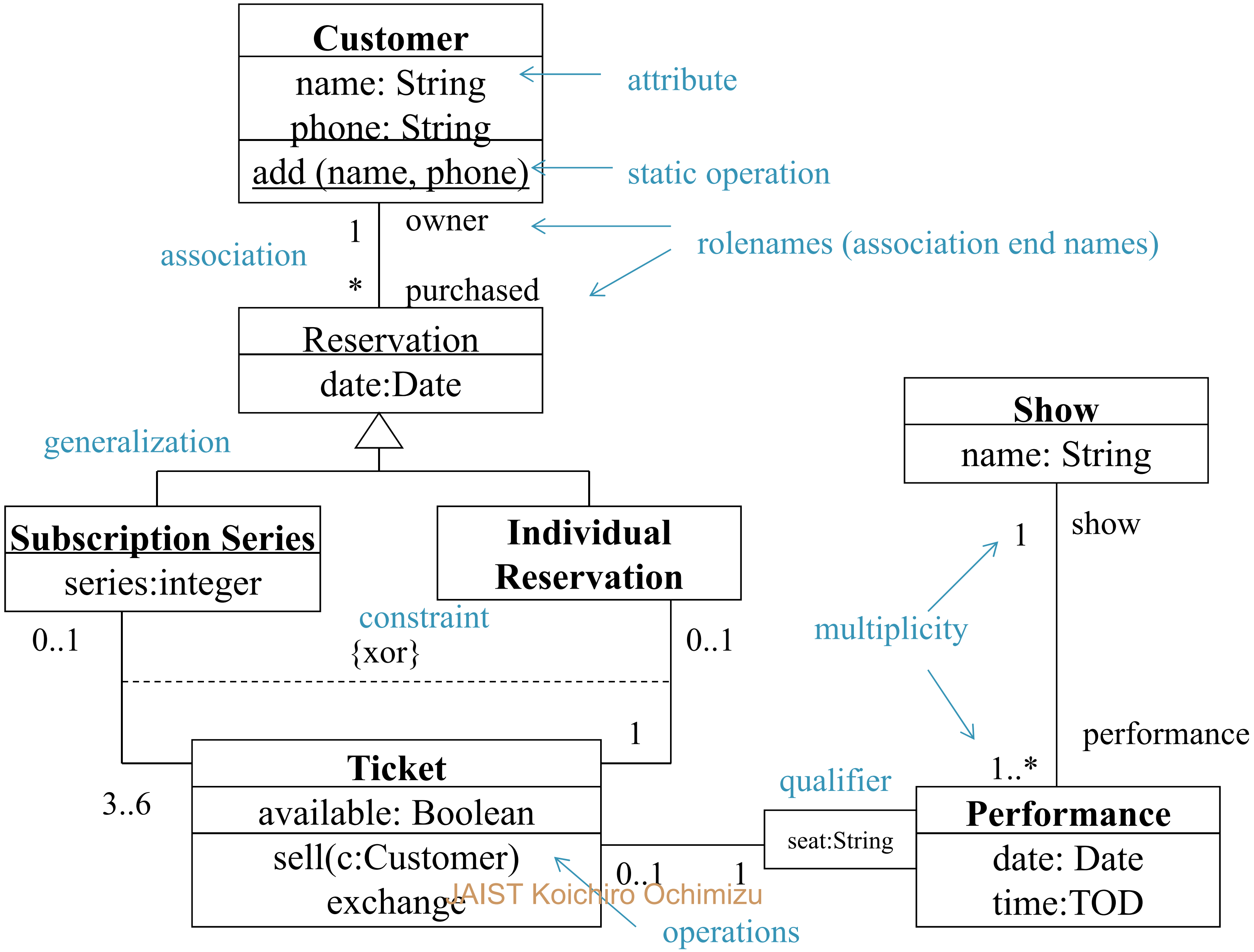
compartment list element

stereotype application

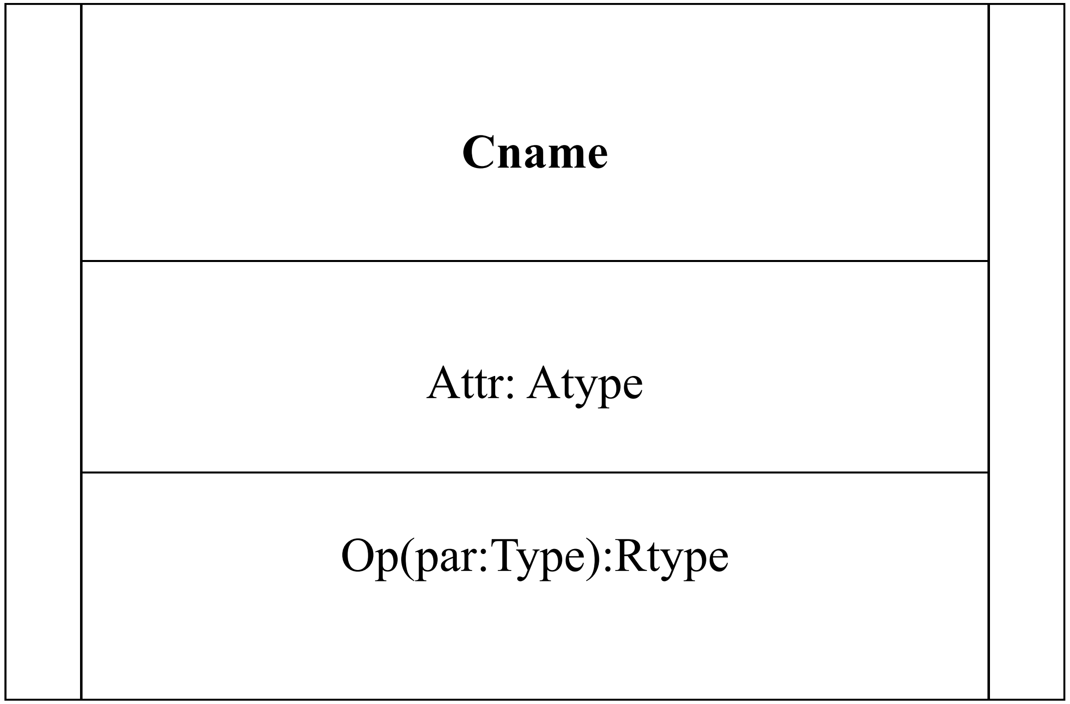
tagged value

J.Rumbaugh, I.Jacobson, G.Booch,"The Unified Modeling Language Reference Manual, Second Edition" Addison-Wesley 2005

Static View (class diagram)



Active Class



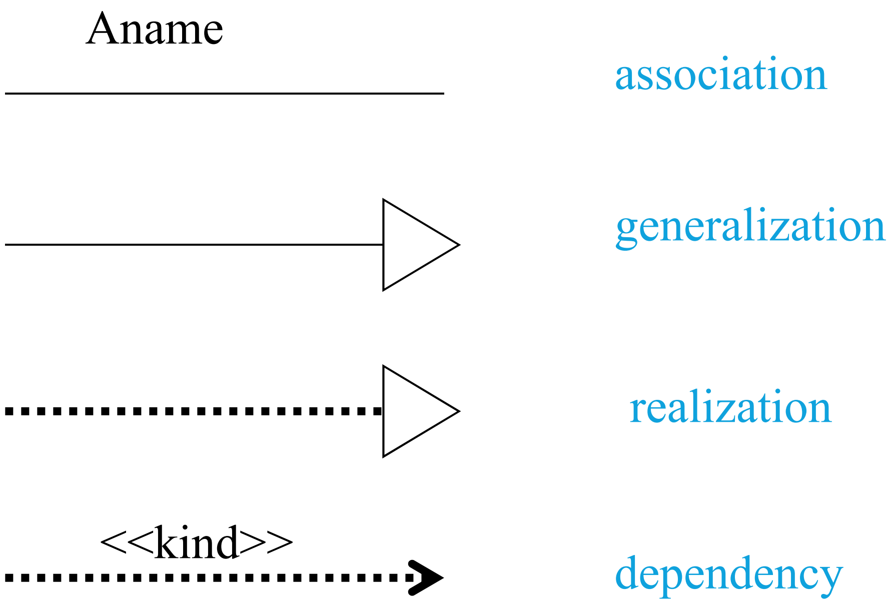
Cname

Attr: Atype

Op(par:Type):Rtype

J.Rumbaugh, I.Jacobson, G.Booch,”The Unified Modeling Language Reference Manual, Second Edition” Addison-Wesley 2005

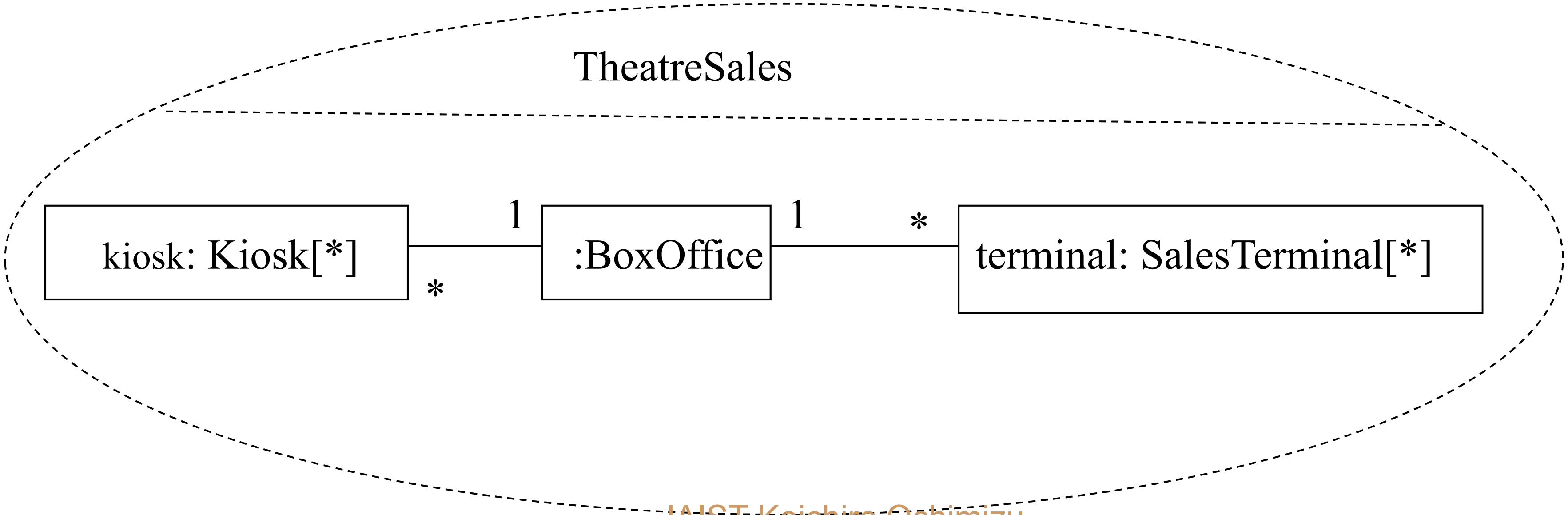
Relationship



J.Rumbaugh, I.Jacobson, G.Booch,"The Unified Modeling Language Reference Manual, Second Edition" Addison-Wesley 2005

Design View (collaboration diagram)

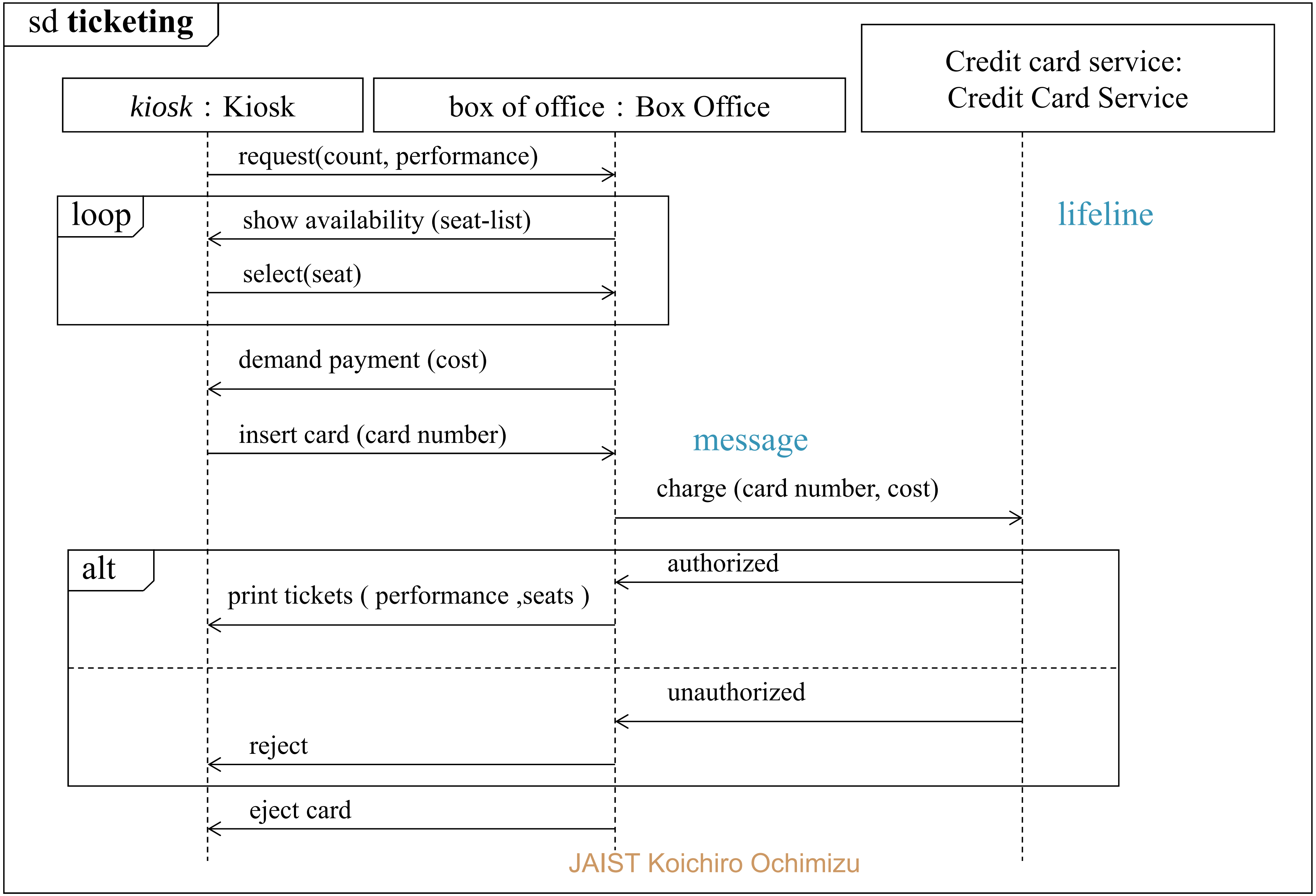
- A collaboration is a contextual relationship among a set of objects that work together to fullfill some purpose.
- It contains a collection of roles – contextual slots within a generic pattern that can be played by, or bound to, individual objects.



Interaction View

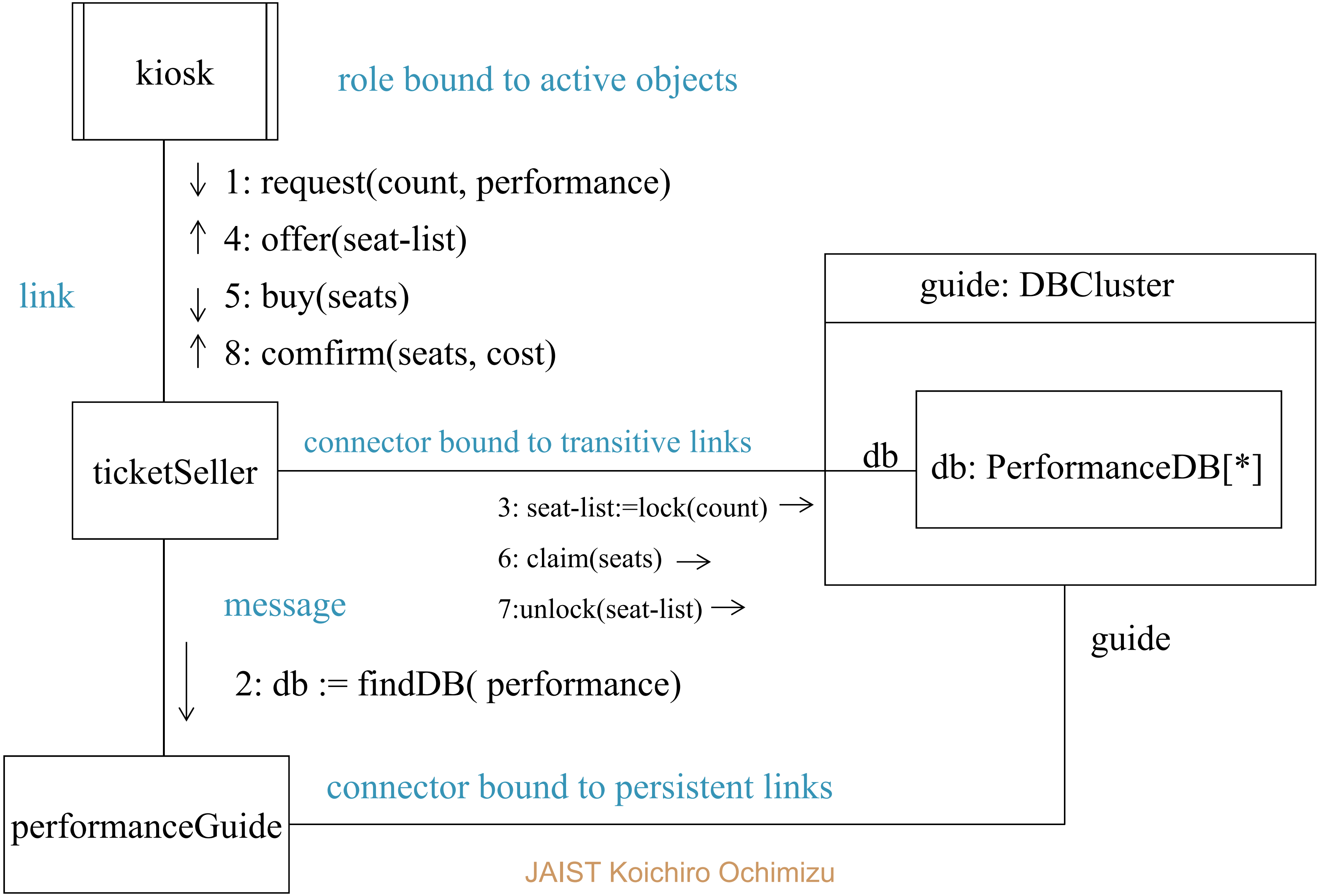
- The interaction view describes sequence of message exchanges among the parts of a system.
- An interaction is based on a structured classifier or a collaboration.
- A role is a slot that may be filled by objects in a particular us of an interaction.
- Interaction view shows the flow of control across many objects and is displayed in two diagrams focused on different aspects: sequence diagrams and communication diagrams. The communication diagram is called a collaboration diagram in UML1.5.

Interaction View (Sequence Diagram)


sd ticketing
kiosk : Kiosk
box of office : Box Office
Credit card service:
Credit Card Service
request(count, performance)
loop
show availability (seat-list)
select(seat)
lifeline
demand payment (cost)
insert card (card number)
message
charge (card number, cost)
alt
authorized
print tickets (performance ,seats)
unauthorized
reject
eject card

JAIST Koichiro Ochimizu

Interaction View (communication diagram)



role bound to active objects

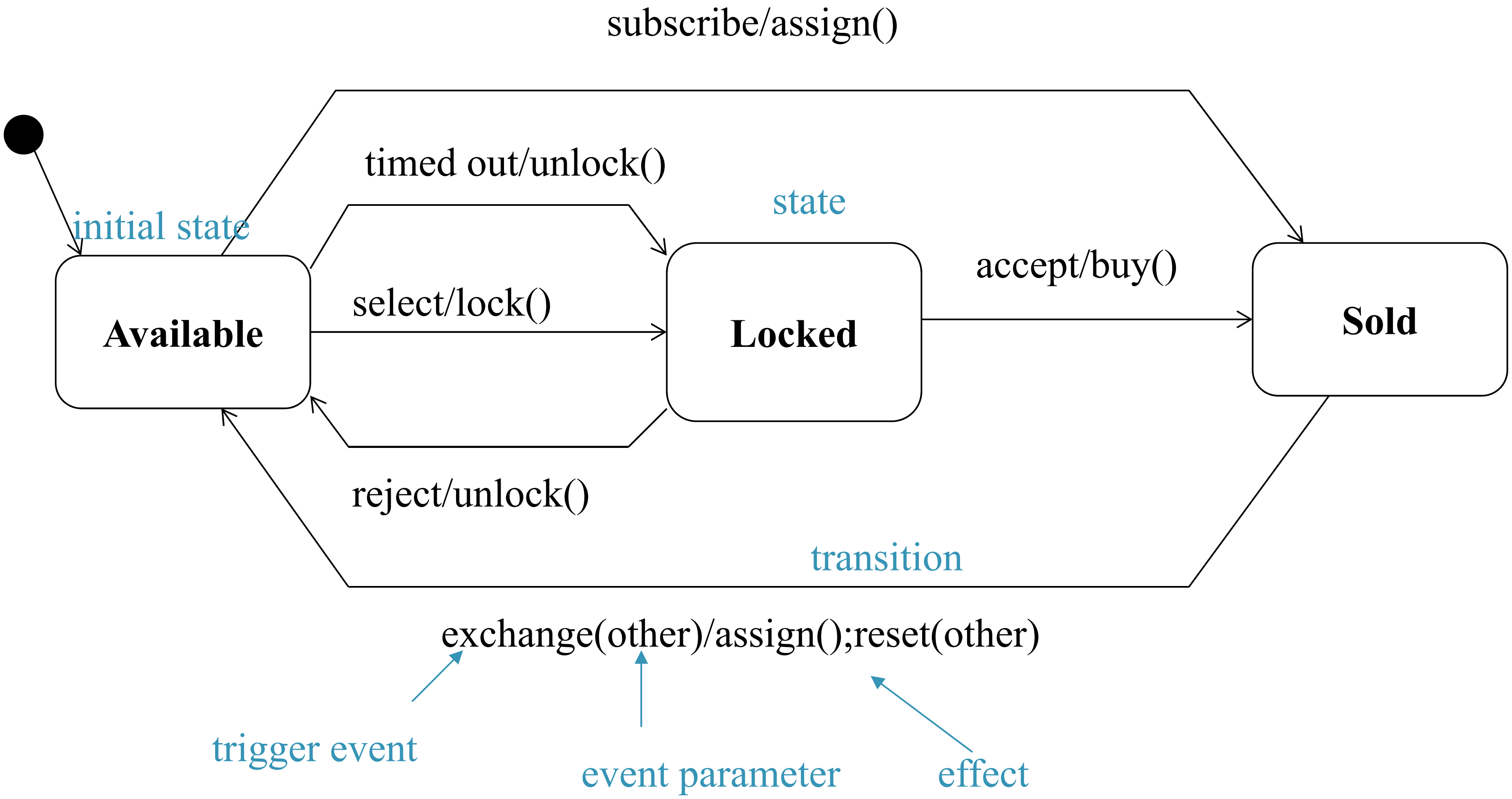
link

connector bound to transitive links

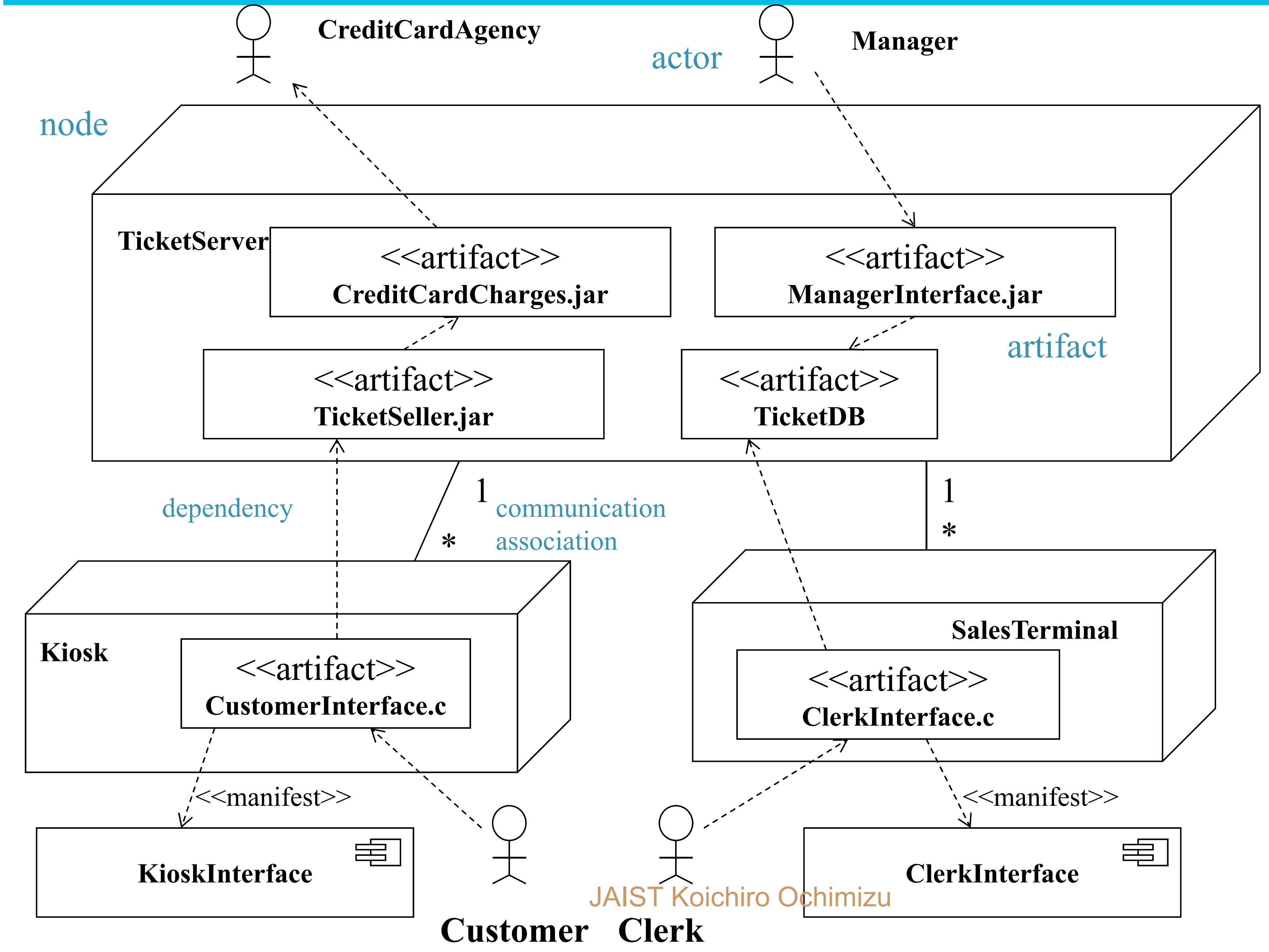
message

connector bound to persistent links

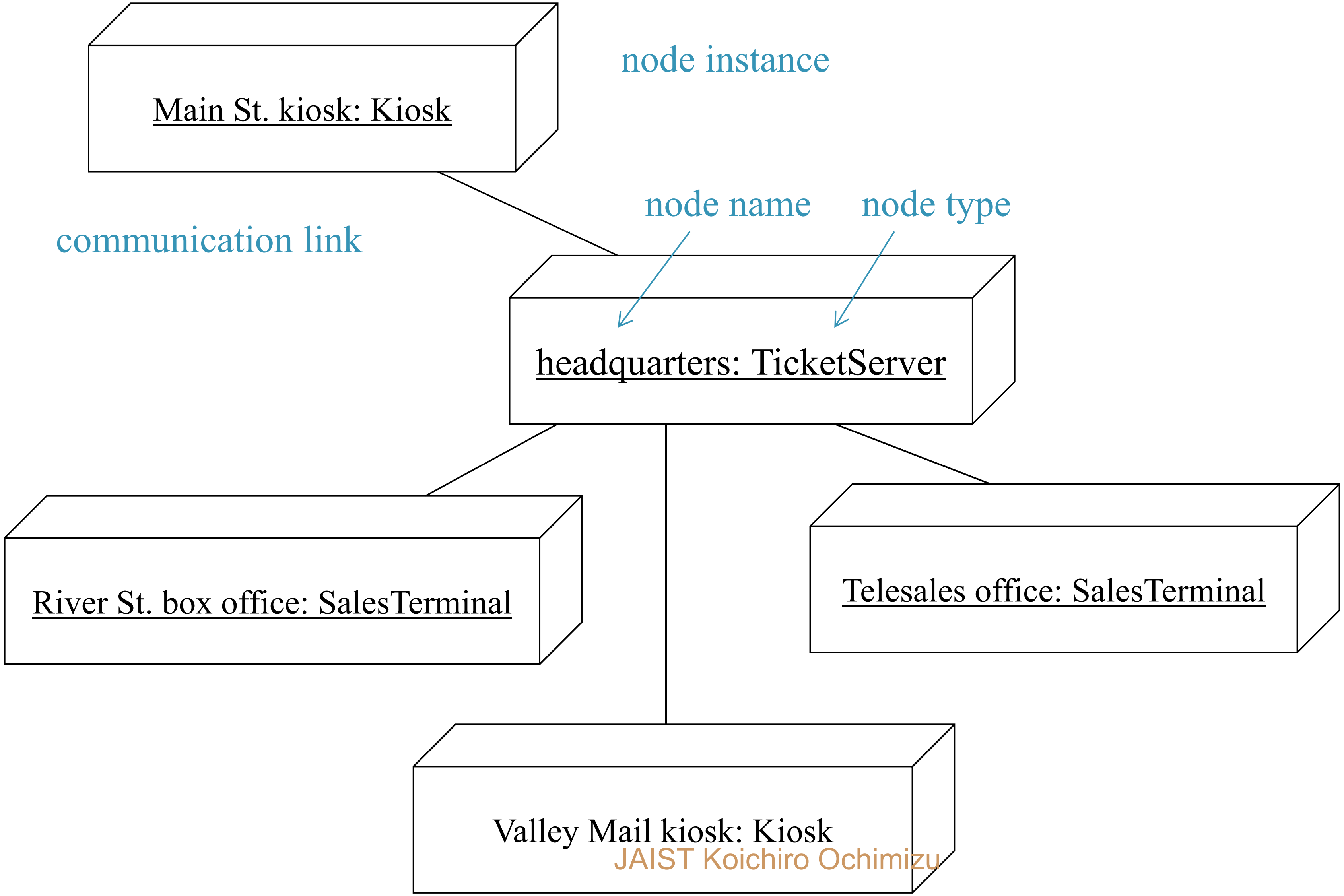
State Machine View (state machine diagram)



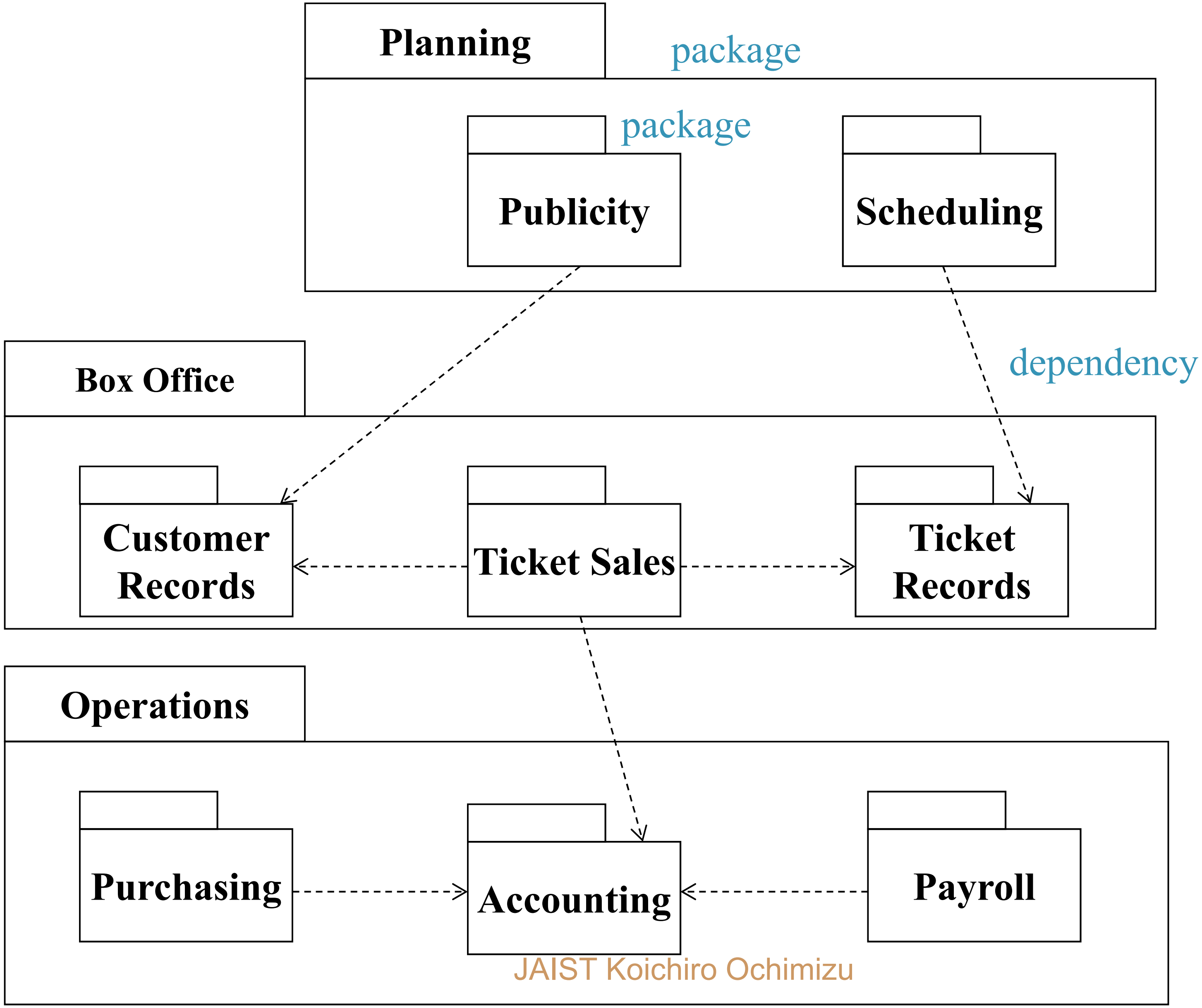
Deployment View (deployment diagram – descriptor level)



Deployment View (deployment diagram – instance level)



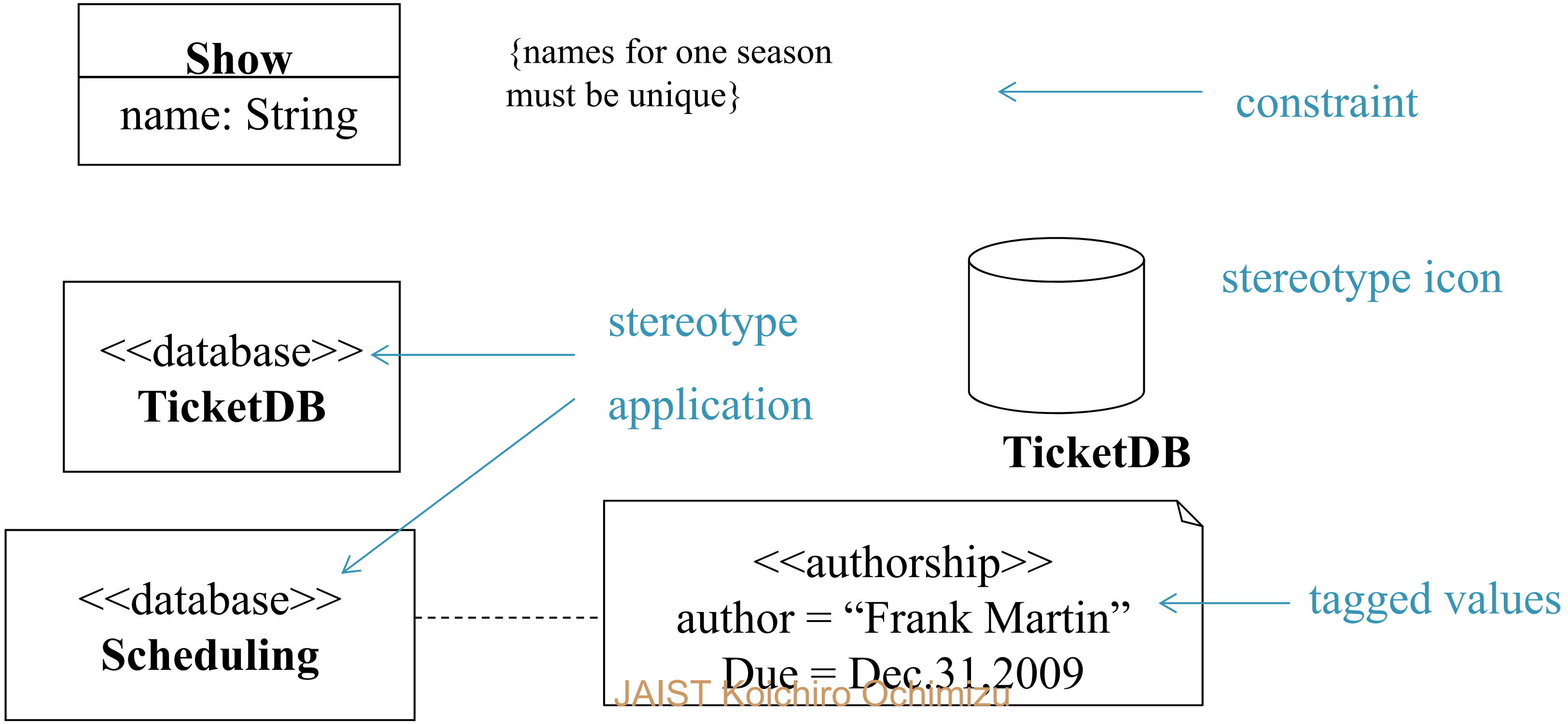
Model Management View (package diagram)



JAIST Koichiro Ochimizu

Model Management View (Profile)

- The profile mechanism permits limited changes to UML without modifying the underlying metamodel.
- UML includes three main extensibility constructs: constraints, stereotypes, and tagged


Show
name: String
{names for one season must be unique}
constraint
<<database>>
TicketDB
stereotype
application
stereotype icon
TicketDB
<<database>>
Scheduling
<<authorship>>
author = “Frank Martin”
Due = Dec.31.2009
tagged values