Unified Modeling Language

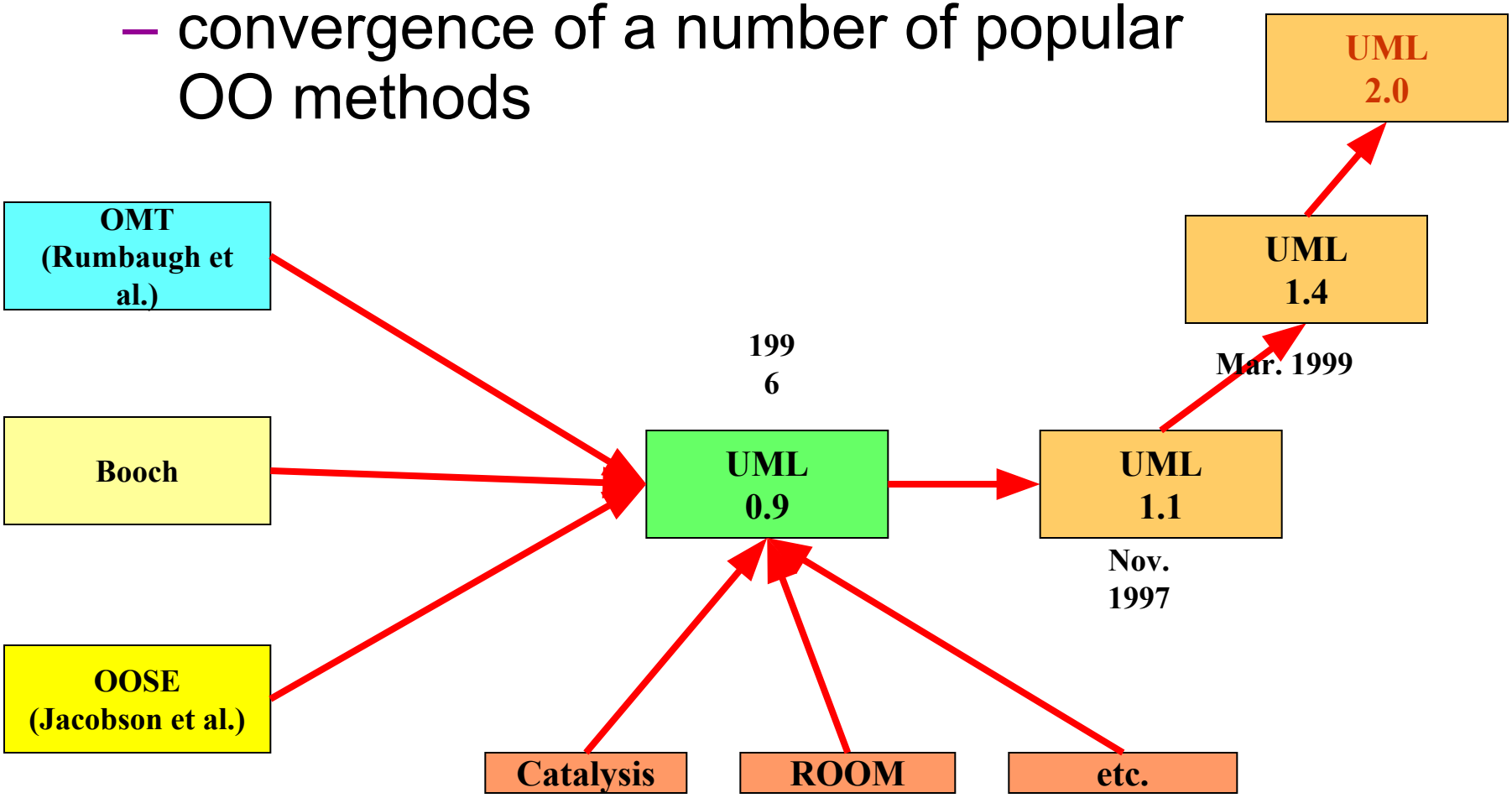
UML - a glance

UML

- UML stands for Unified Modeling Language
- The UML combines the best of the best from:
  - Data Modeling concepts (Entity Relationship Diagrams)
  - Business Modeling (Work Flow)
  - Object Modeling
  - Component Modeling
- The UML is a standard language for **specifying, visualizing, documenting and constructing the artifacts of a software**

UML Heritage

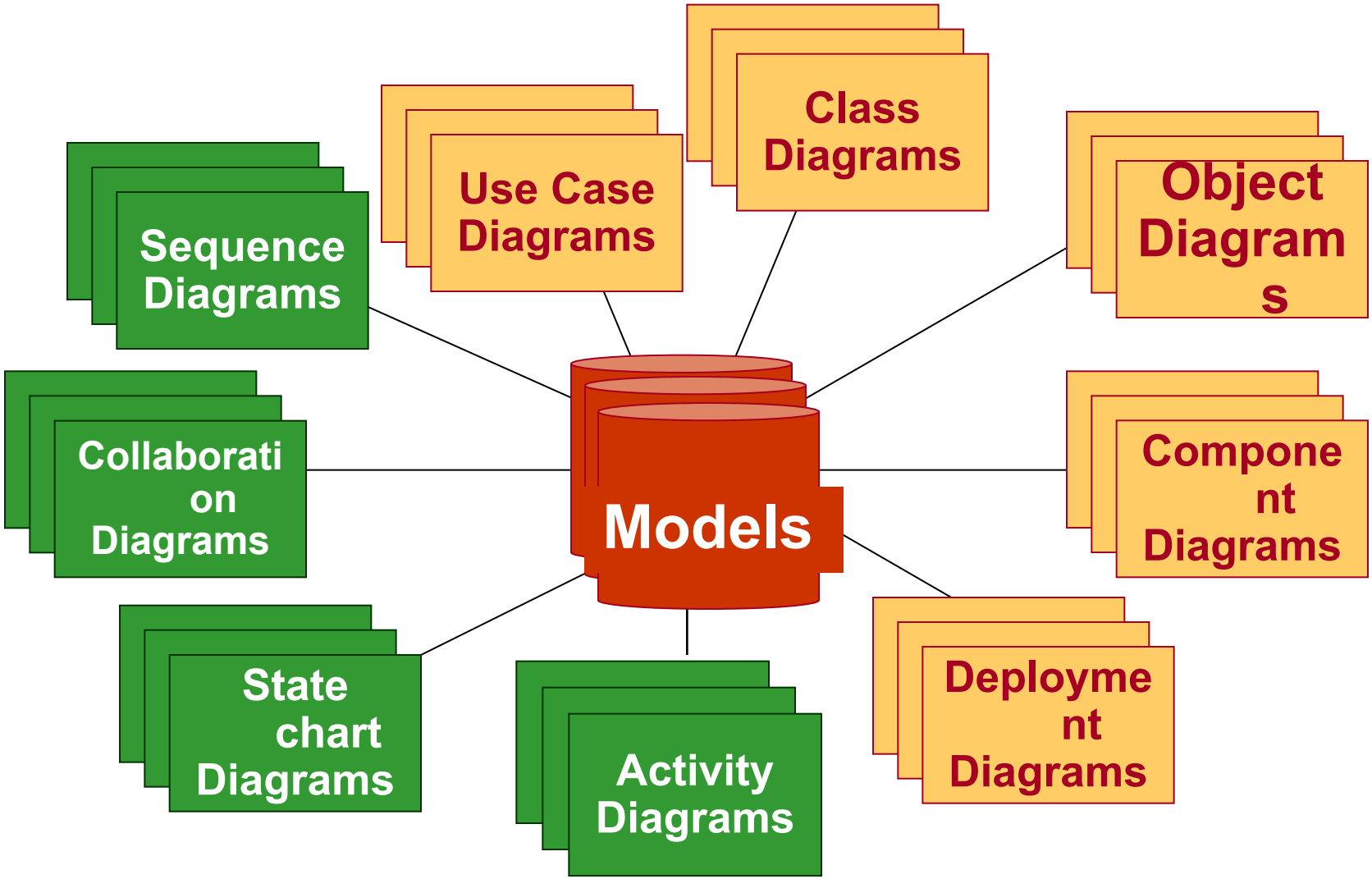
- General-purpose OO *modeling* language
  - convergence of a number of popular OO methods



UML Usage: Overview

- The UML may be used to:
  - Represent the Elements of a system or a domain and their Relationships in a Static Structure using class and object diagrams
  - Model the Behavior of objects with state transition diagrams
  - Reveal the Physical Implementation Architecture with component & deployment diagrams
  - Display the Boundary of a System & its major Functions using use cases and actors
  - Illustrate Use Case Realizations with interaction diagrams (Sequence, Activity Diagram & Collaboration)

UML Diagrams



UML Model Views

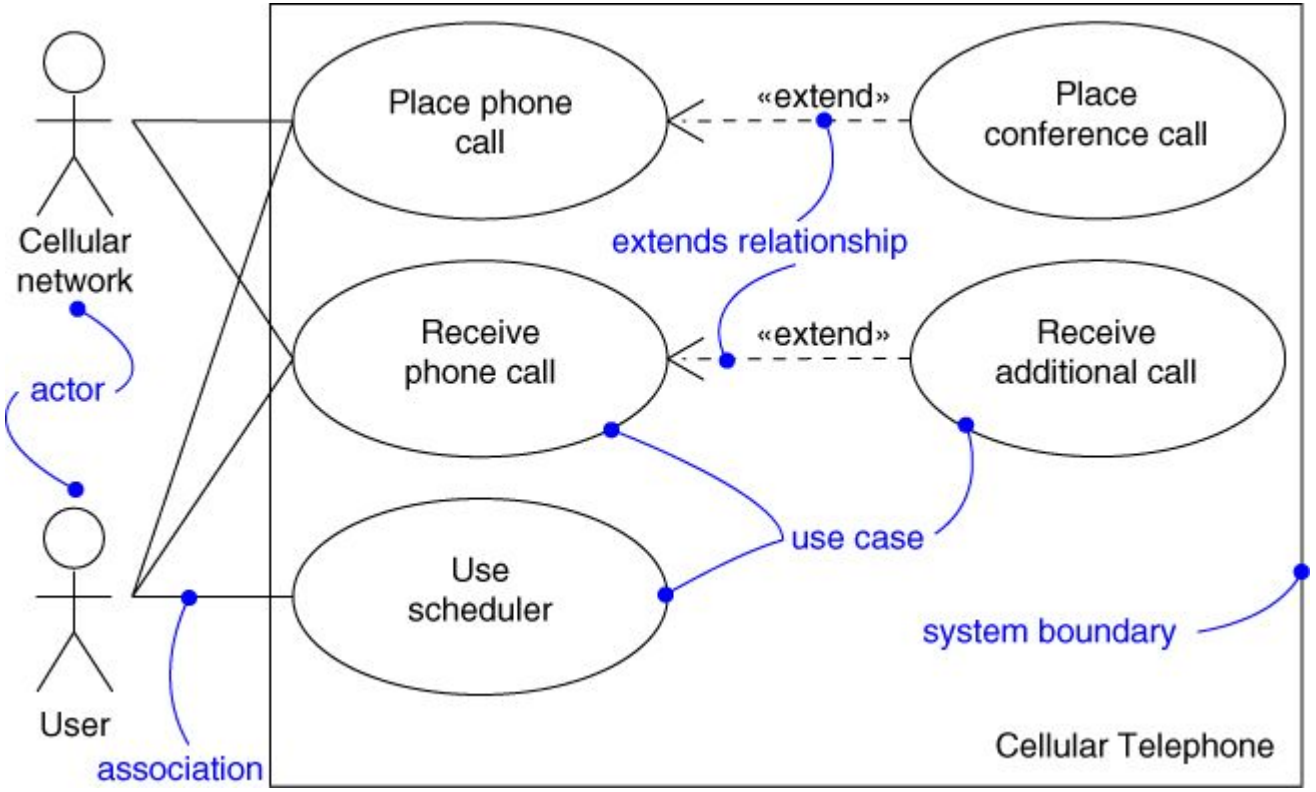
- **Requirements** (use case diagrams)
- **Static structure** (class diagrams)
  - kinds of objects and their relationships
- **Dynamic behavior** (state machines)
  - possible life histories of an object
- **Interactive behavior** (activity, sequence, and collaboration diagrams)
  - flow of control among objects to achieve system-level behavior
- **Physical implementation structures** (component and deployment diagrams)
  - software modules and deployment on physical nodes

Diagrams

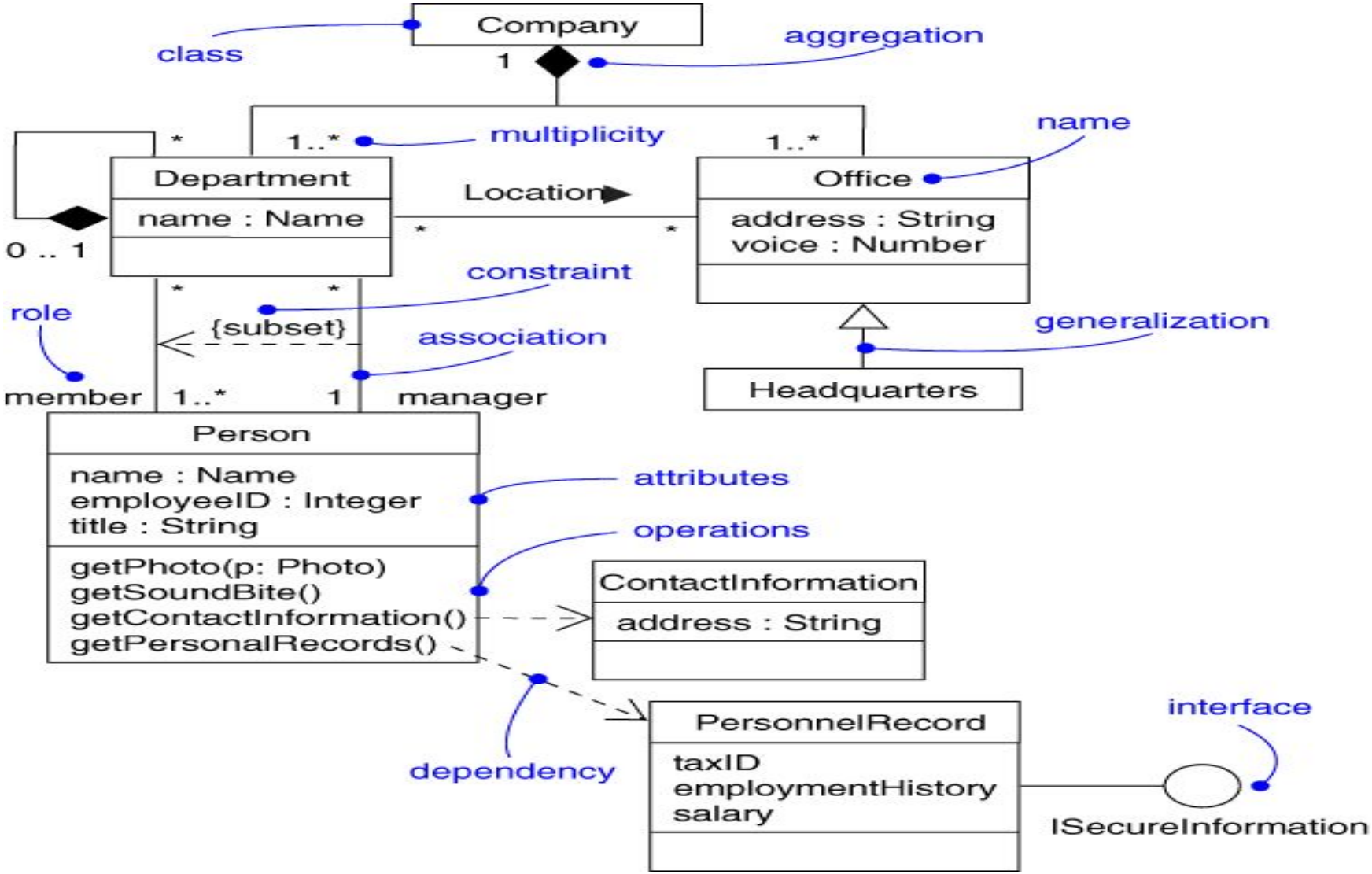
- In the UML, there are **9 standard diagrams**
  - **Static views**: **use case, class, object, component, deployment**
  - **Dynamic views**: **sequence, collaboration, state chart, activity**

Use Case Diagram

- **Captures system functionality as seen by users**

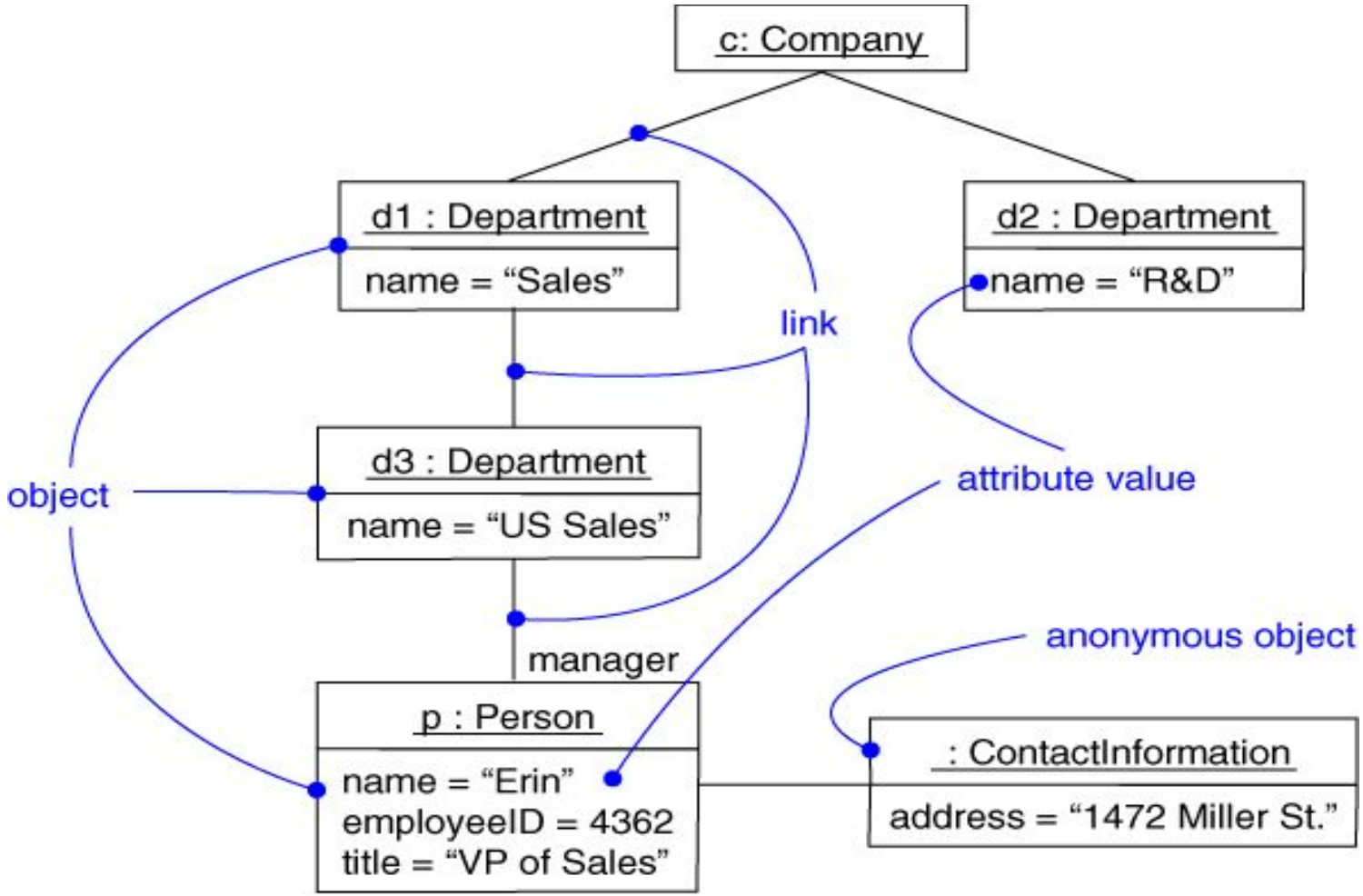


Class Diagram



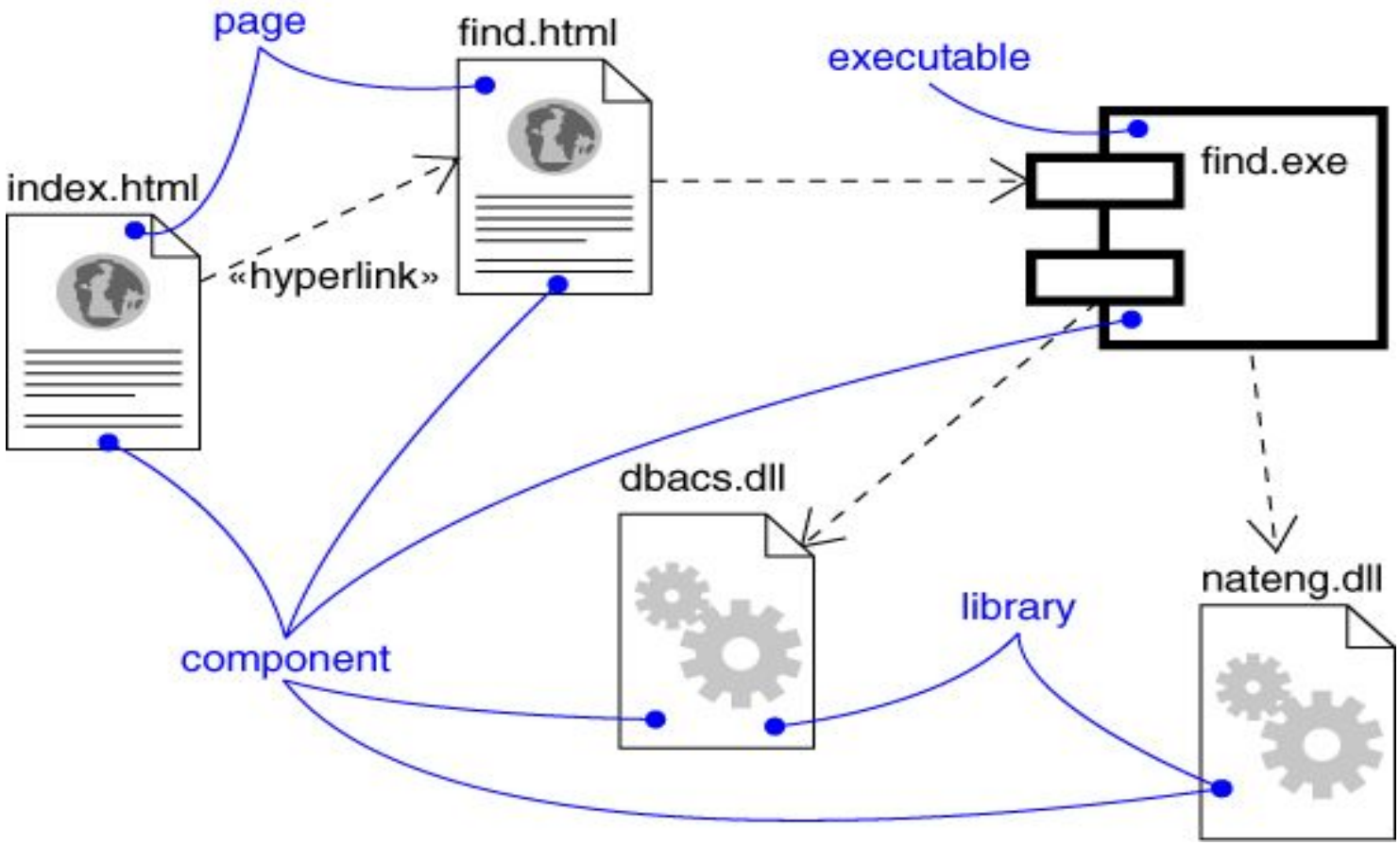
Object Diagram

- **Captures instances and links**



Component Diagram

- **Captures the physical structure of the implementation**

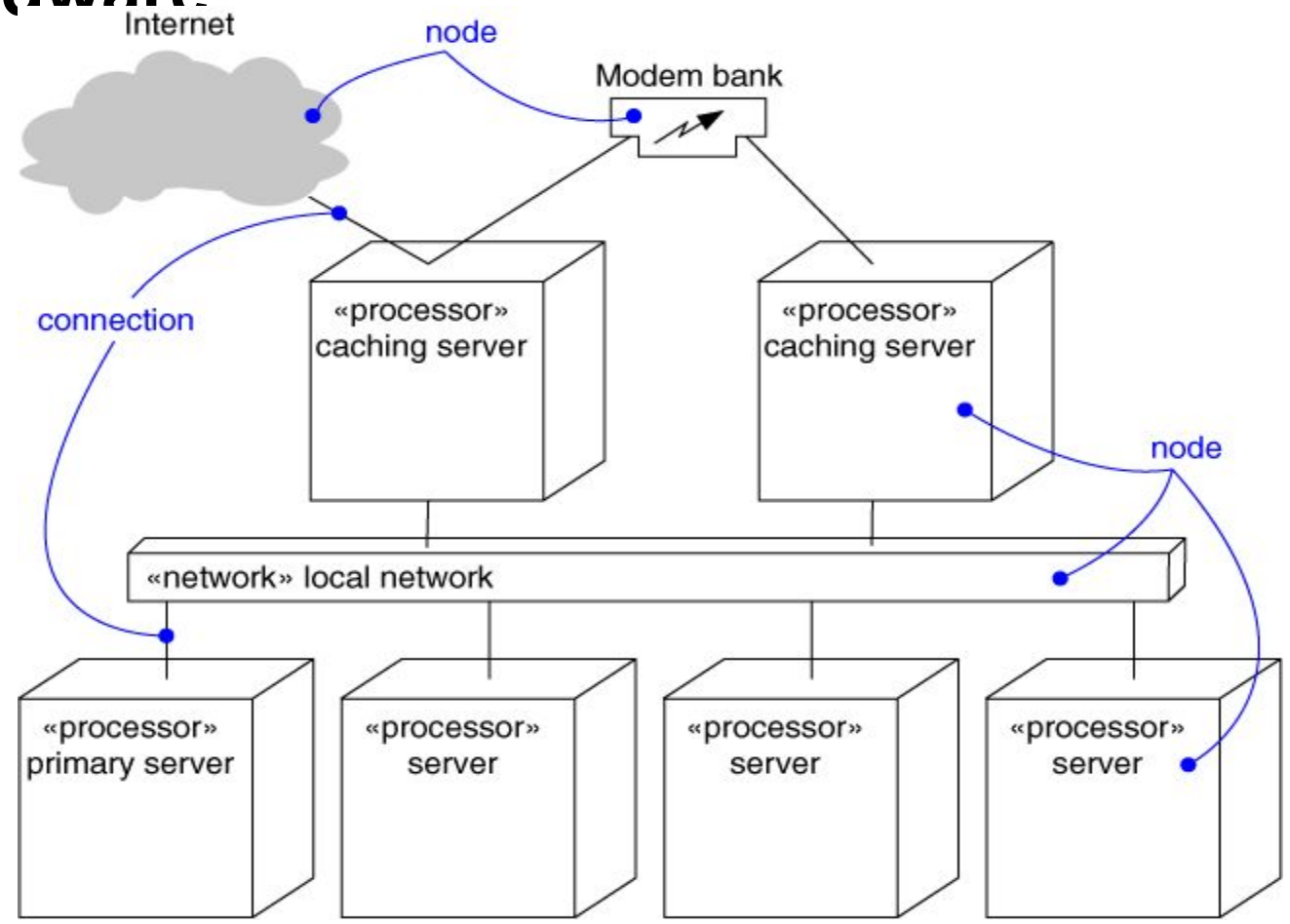


Component Diagram

- Captures the physical structure of the implementation
- Built as part of architectural specification
- Purpose
  - Organize source code
  - Construct an executable release
  - Specify a physical database
- Developed by architects and programmers

Deployment Diagram

- **Captures the topology of a system's hardware**

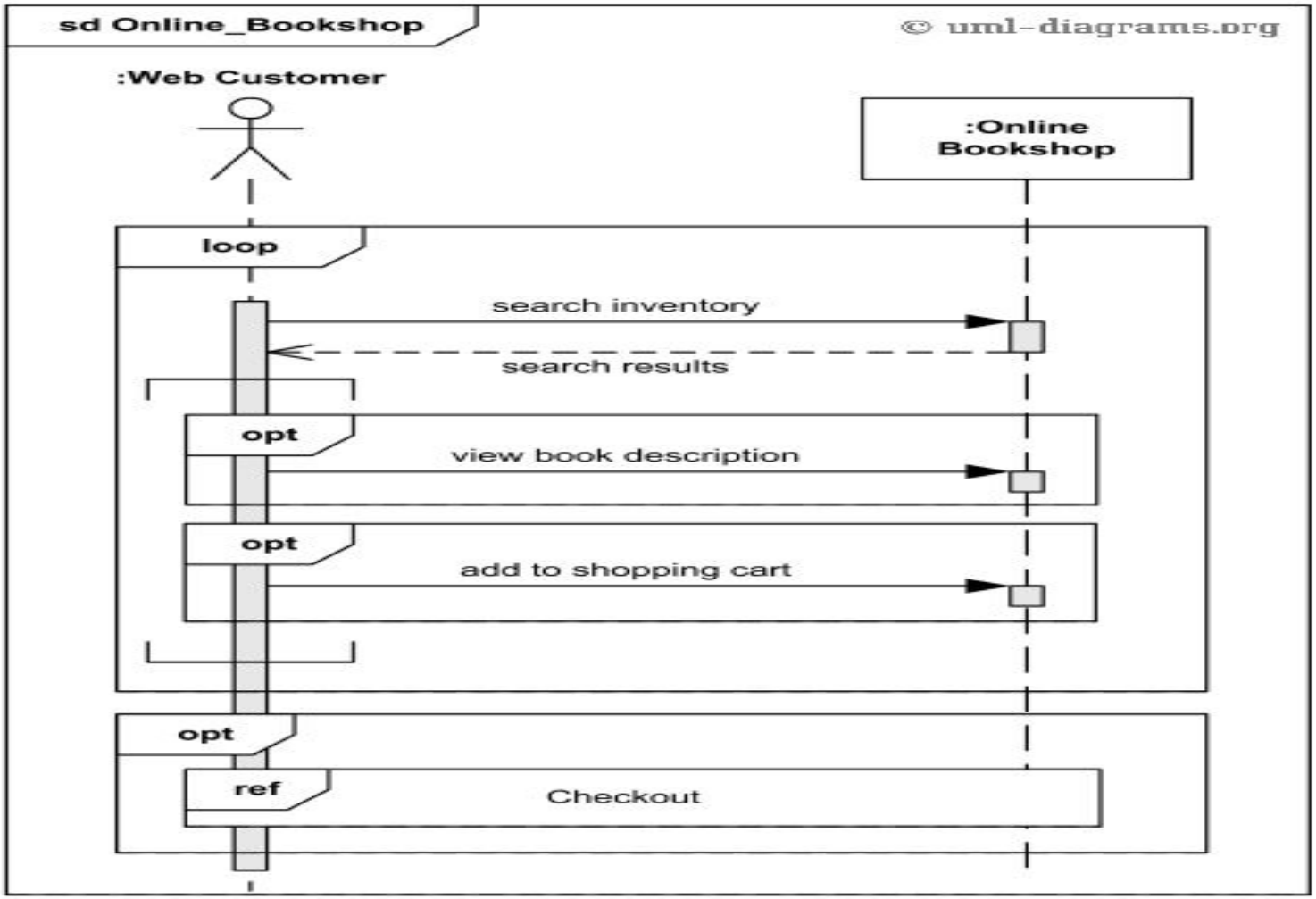


Deployment Diagram

- Captures the topology of a system's hardware
- Built as part of architectural specification
- Purpose
  - Specify the distribution of components
  - Identify performance bottlenecks
- Developed by architects, networking engineers, and system engineers

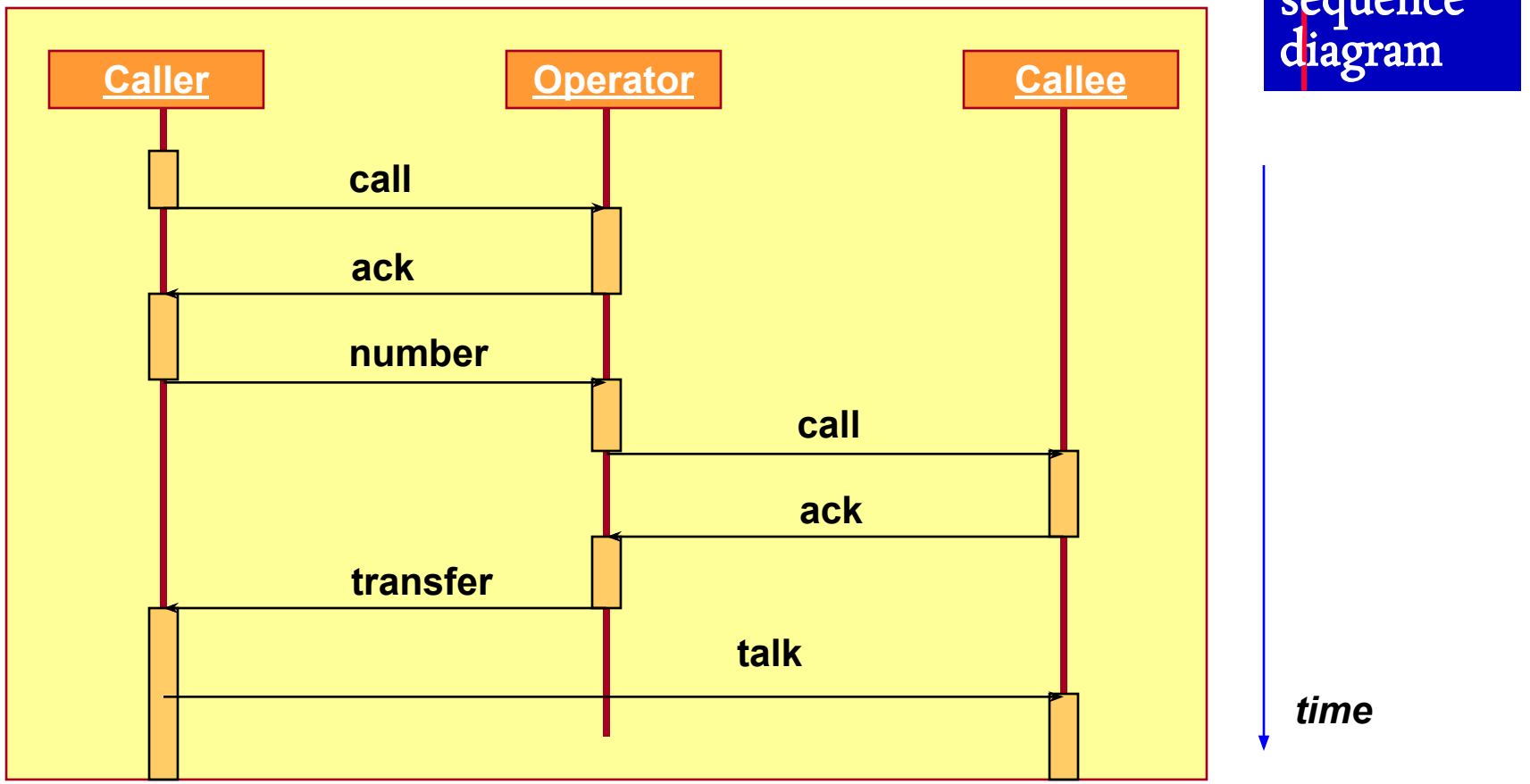
Sequence Diagram

- **Captures dynamic behavior (time-oriented)**

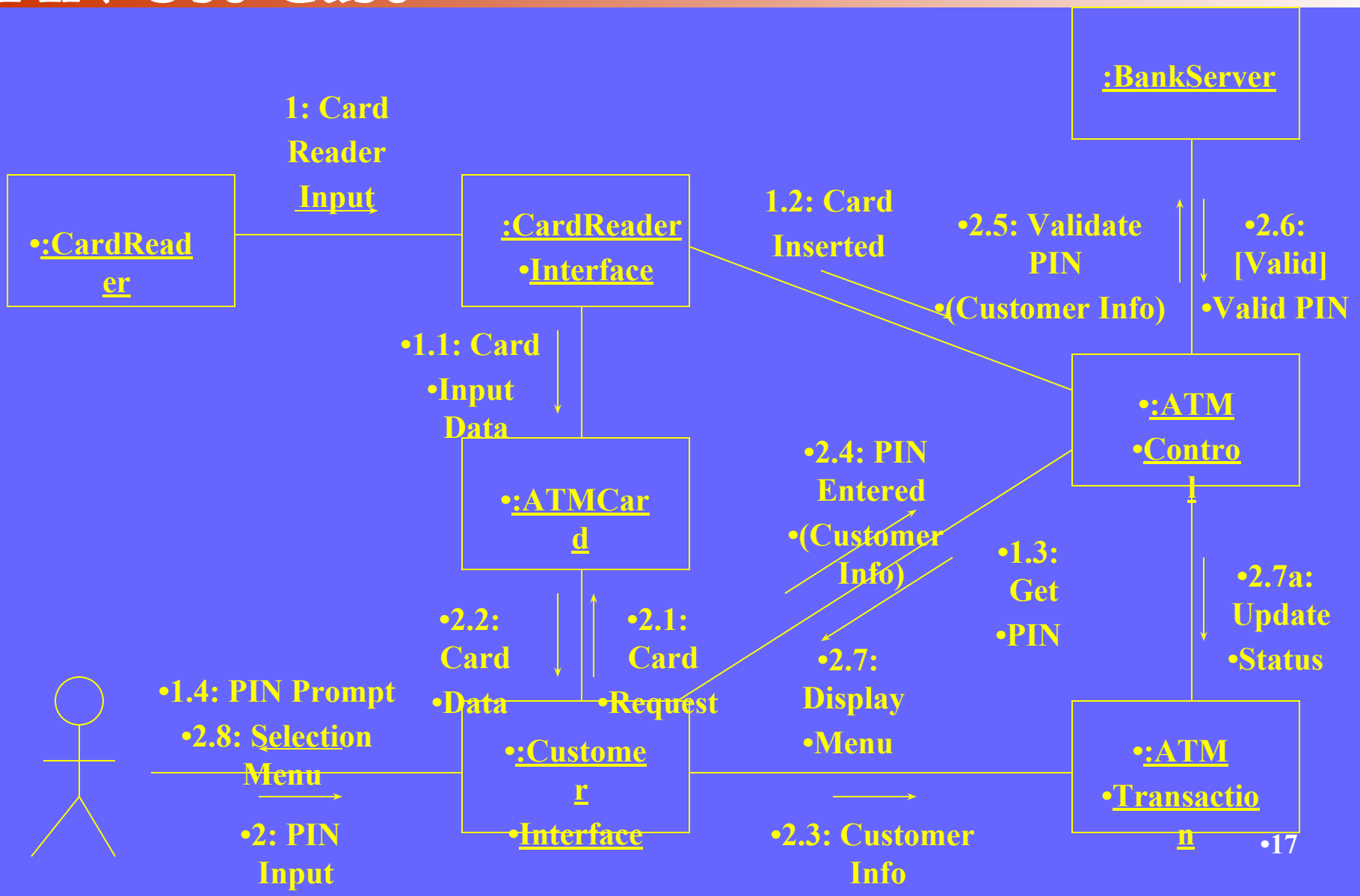


Example

- Assertions of legal interactions between objects (e.g., operator-assisted call)

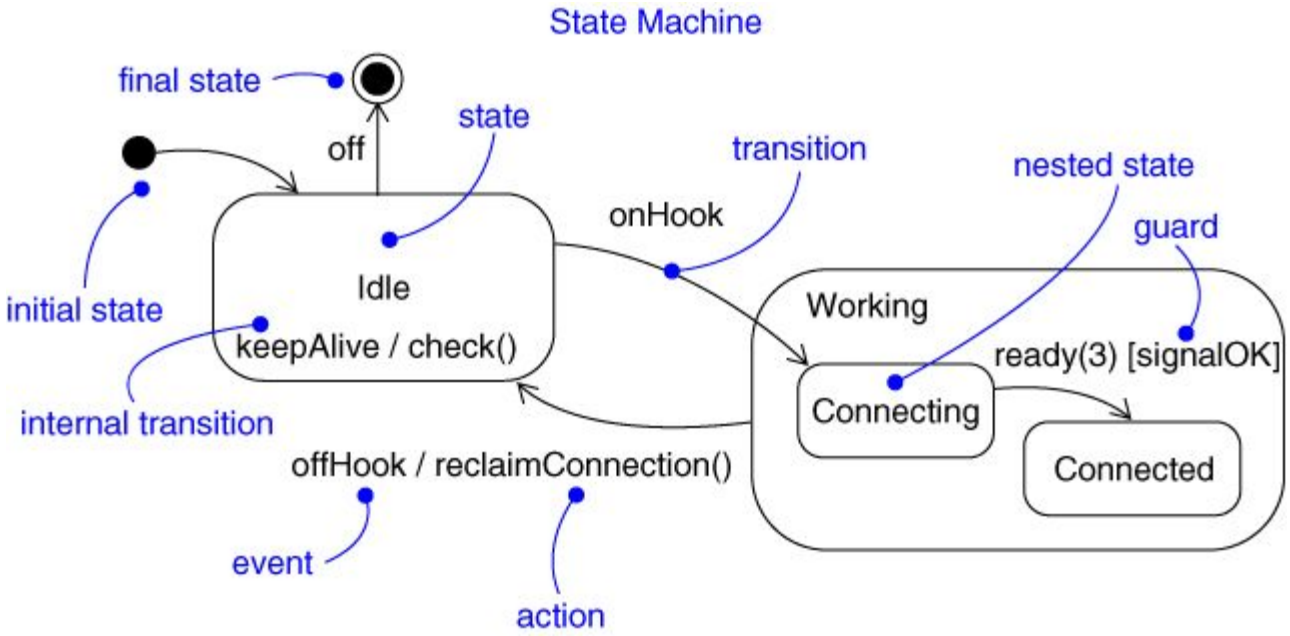


Collaboration Diagram: ATM Client Validate PIN Use Case

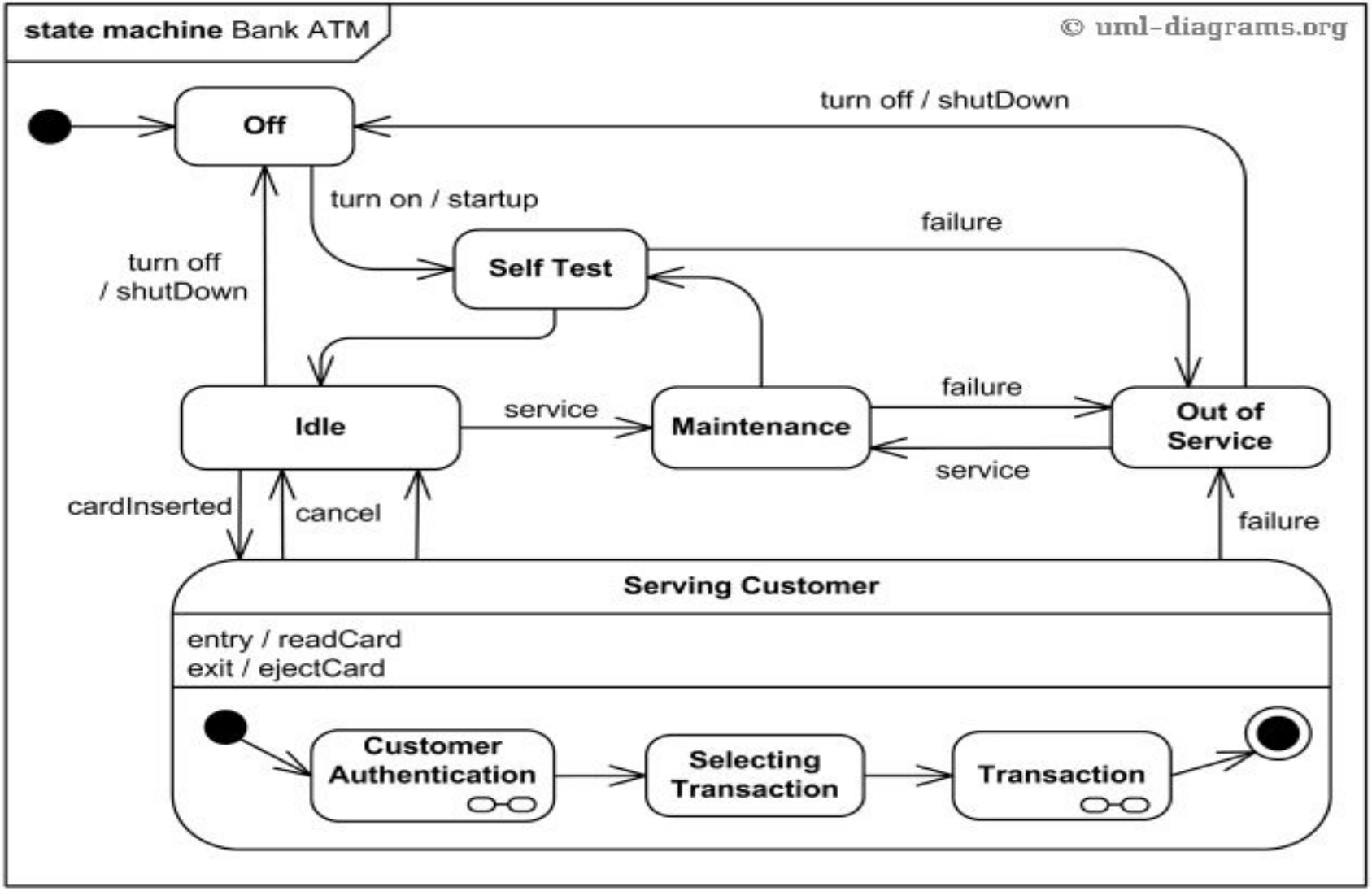


Diagram

- Captures dynamic behavior (event-oriented)

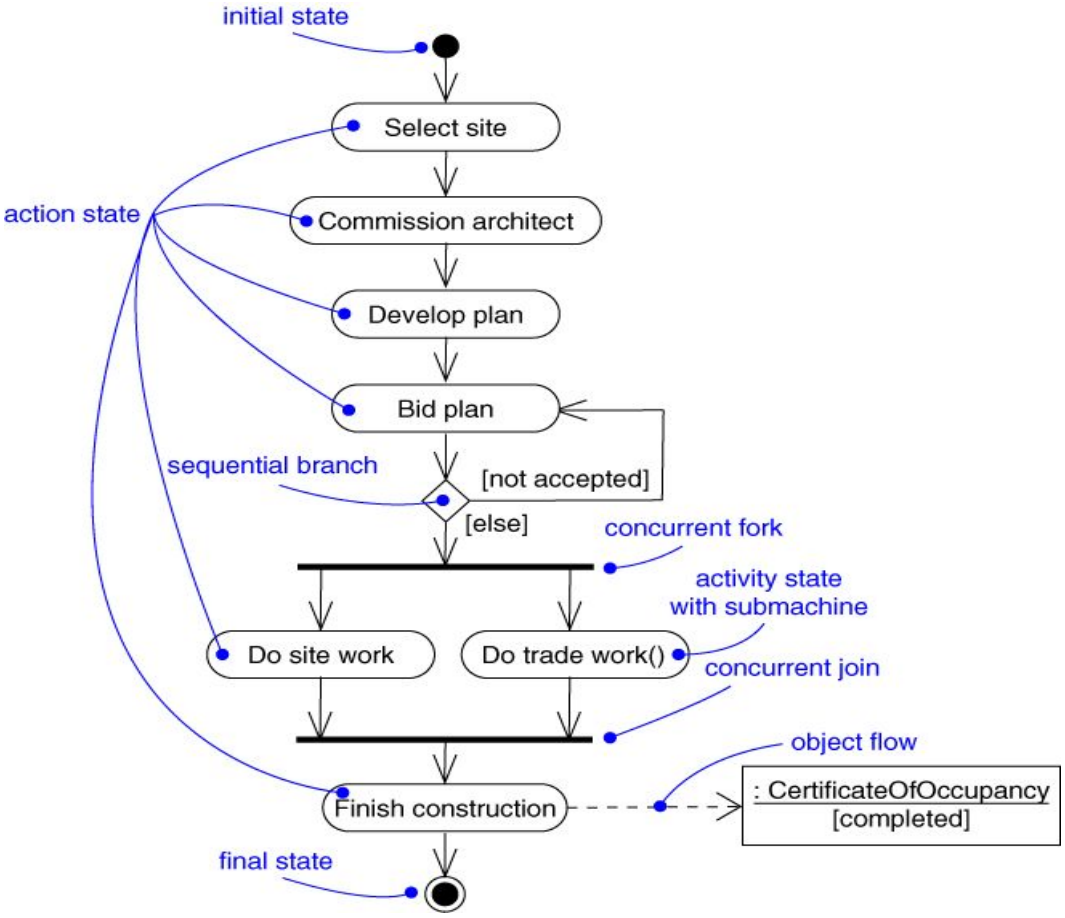


State Chart

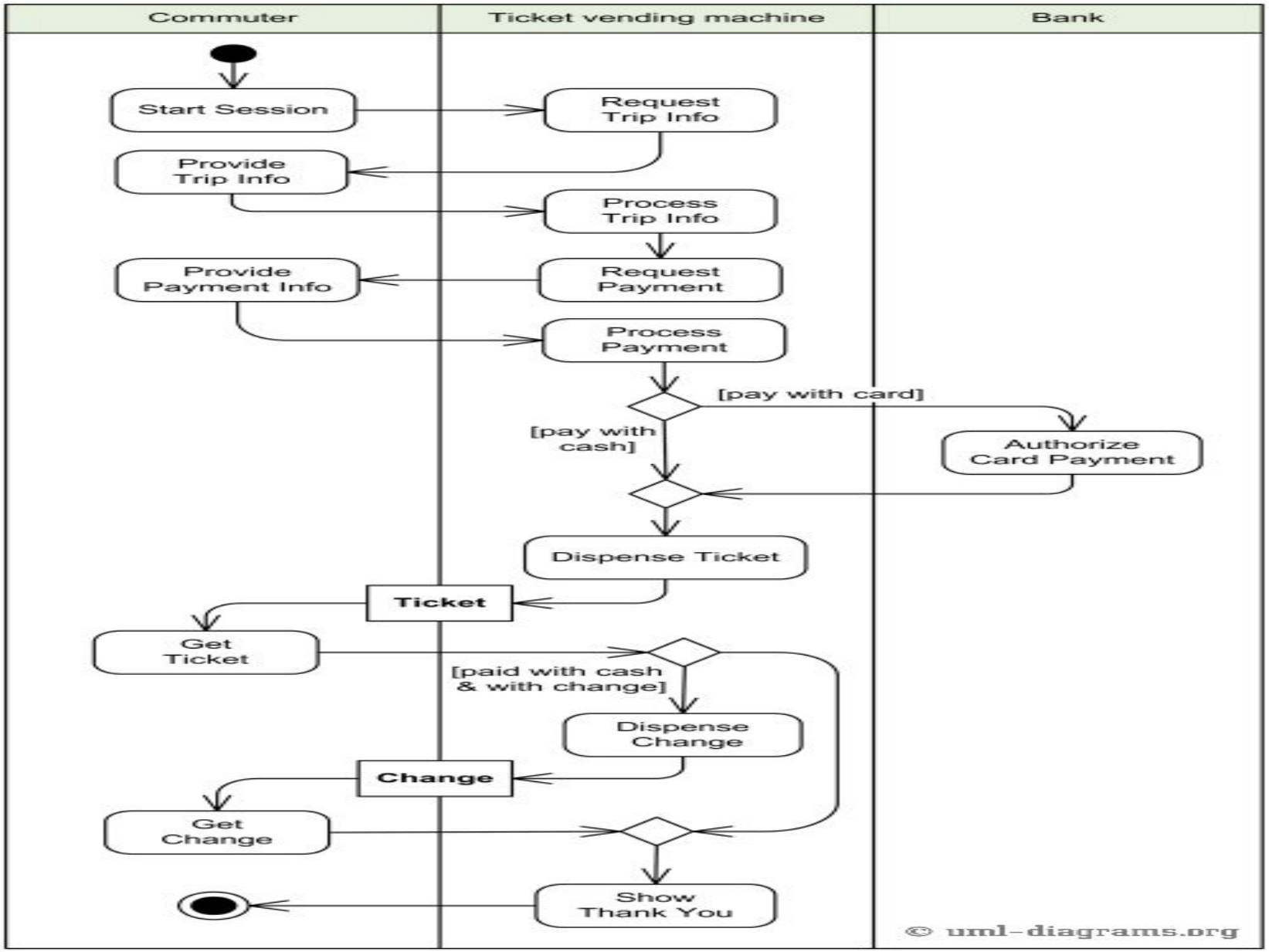

state machine Bank ATM
© uml-diagrams.org
Off
turn off / shutDown
turn on / startup
failure
Self Test
turn off
/ shutDown
Idle
service
Maintenance
failure
Out of
Service
service
cardInserted
cancel
failure
Serving Customer
entry / readCard
exit / ejectCard
Customer
Authentication
Selecting
Transaction
Transaction

Activity Diagram

- Captures dynamic behavior (activity-oriented)



Activity Diagram



Summary: UML

- An industry standard for analysis and design of object-oriented systems
  - based on extensive experience and best practices
  - gaining rapid acceptance (training, tools, books)
- Comprises:
  - set of modeling concepts
  - graphical notation
- Concepts are organized into diagram types
  - class, state machine, collaboration, use case, sequence, activity, component, deployment
- The UML can be used in many different domains to capture domain-specific concepts and ideas