Chapter1. Unified Modeling Language
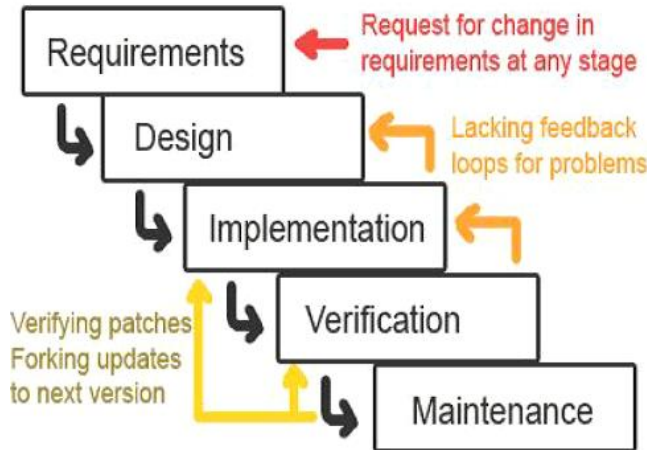
2019年12月5日　　10:11

NOTE TAKING AREA

==Waterfall Model==

Components: requirements, design, implementation, verification, maintenance.

- Requirements: are important: clearly a loaded question, better stated, overall goal of requirements engineering is **risk reduction**.
- Modeling: describing a system at a high level of abstraction, many notations have existed over time.

==UML==

Provides a **common, simple, graphical** representation of software design and implementation.

Allows developers, architects, and users to discuss the workings of the software. Modeling guidelines: everything in UML is optional, models are rarely complete, open to interpretation, design to be extended.

Static modeling (**fixed, code-level** relationships): class diagram, package diagram, component diagram, composite structure diagram, deployment diagram.

- UML class diagram: classes, multiplicity / singleton.
    - Class relationships: dependency, association, aggregation, composition, generalization, realization.
    - Operation descriptions: public +, private -, protected #, package ~.
    - Parameter list: $name: type$.

Behavioral modeling (**dynamic execution**): use case diagram, interaction diagram (sequence diagram), collaboration diagram, state diagram, activity diagram.

- Use case diagram: name, actor, entry condition (pre-condition), exit condition (post-condition), event flow, graph showing.
    Inclusion (include), generalization / specialization, extension (extend).
- State-chart diagram: state, action (atomic execution), activity, transition, initial state.
- Interaction diagram: communication between elements, sequence diagram, communication diagram, interaction overview diagram, timing diagram.
- UML sequence diagram: classes & actors, time steps, control / data flow, lifeline.
    Loops and alternatives: alt, loop, opt, par, region.

Good part of UML: a common language, visual syntax is useful, forces clarity to extend UML is precise, commercial tool support.
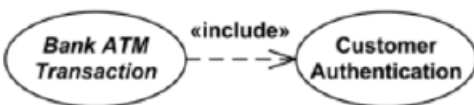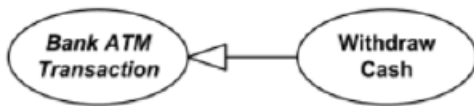
Bad part of UML: hodge podge of ideas, visual syntax does not scale well, semantics is not completely clear.

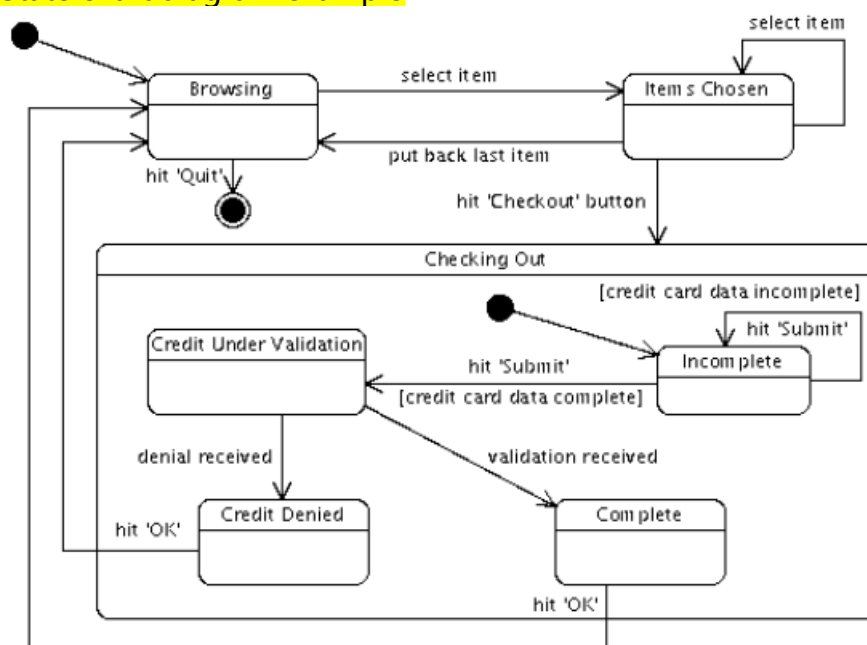**Use case diagram example**
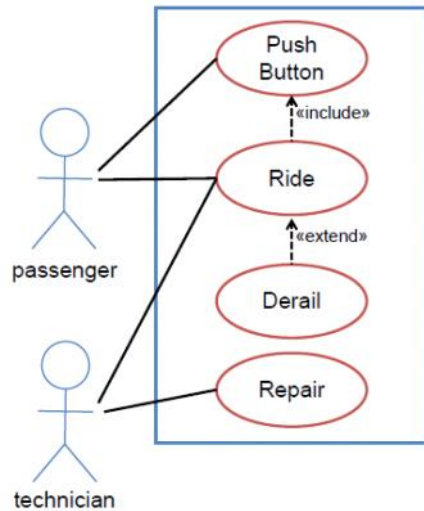
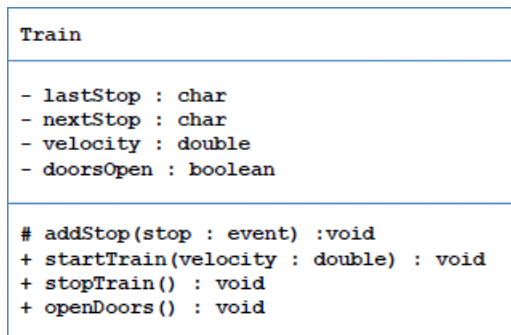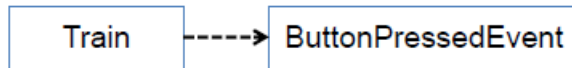**In an auxiliary example, relations are:**

State chart diagram example

UML class diagram example

```
Train

- lastStop : char
- nextStop : char
- velocity : double
- doorsOpen : boolean

# addStop(stop : event) :void
+ startTrain(velocity : double) : void
+ stopTrain() : void
+ openDoors() : void
```
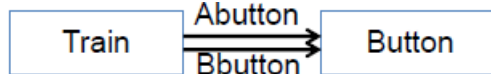
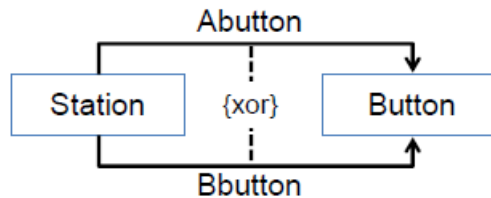- Dependency: uses.

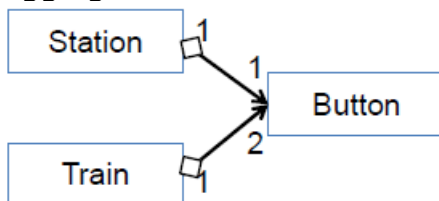- Association: has a, number means multiplicity.
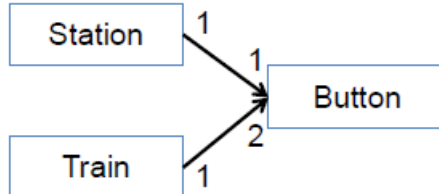
  ○ Associations with names:

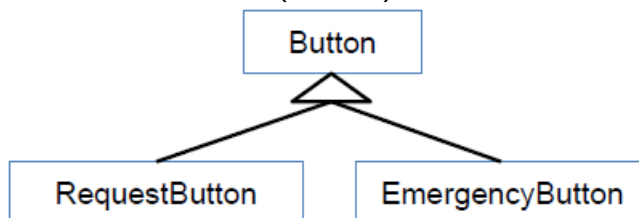 ○ Associations with condition:

- Aggregation: owns a.

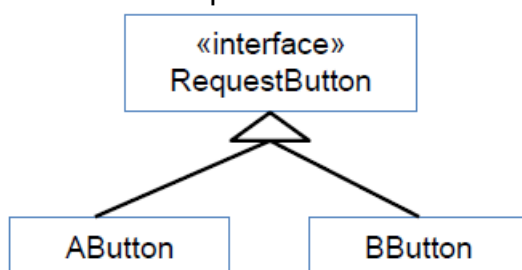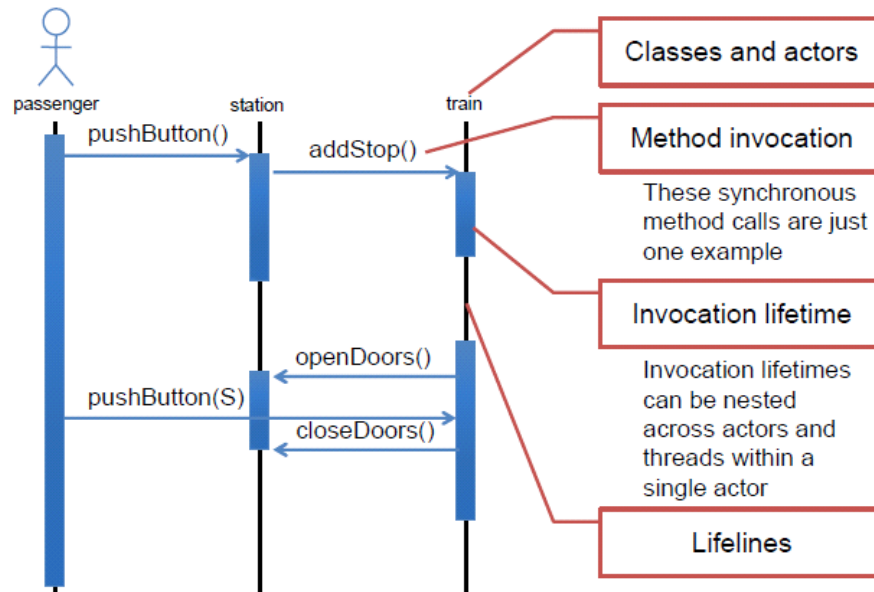

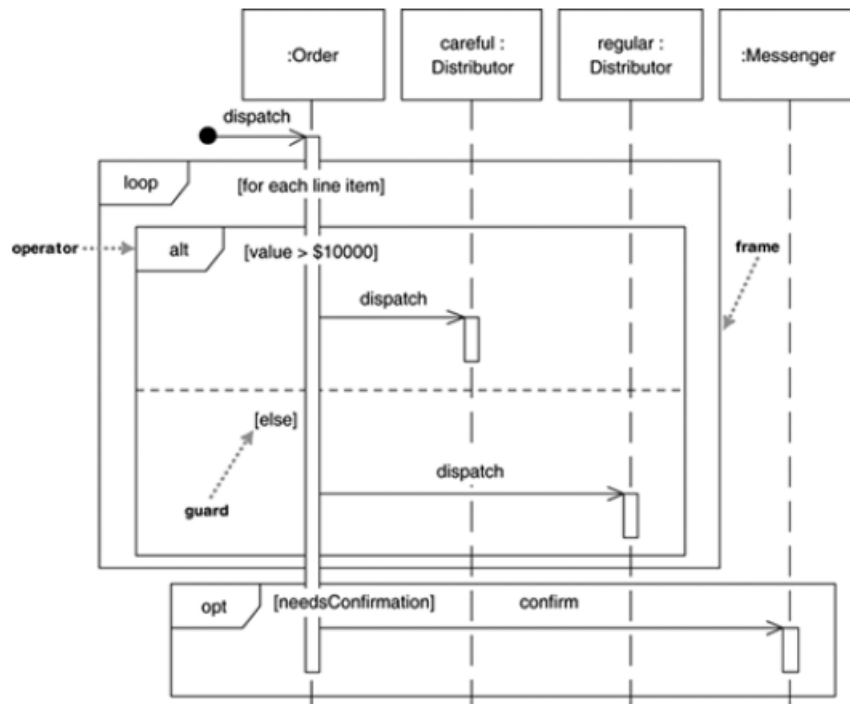- Composition: is made up of (inner class).

- Generalization: is a (extend).

- Realization: implements.

Loops and alternatives example:

- Description: par (parallel), region (single thread).

SUMMARIES

1. Waterfall model and its components.
2. 2 UML modeling (static, behavior) and some UML diagrams.
   a. UML class diagram;
   b. Use case diagram;
   c. State-chart diagram;
   d. Interaction diagram;
   e. UML sequential diagram.