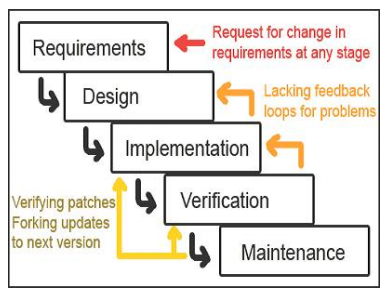
CS309 OBJECT-ORIENTED ANALYSIS AND DESIGN — OAD

Lecture2 UML（Unified Modeling Language）

Waterfall Model

Requirements Engineering

Design

Implementation

Testing（Verification）

Maintenance（维护优化）

既有先后顺序，又有相互重叠

Requirements is Important

The foundation must be laid well and planning should be adequate

The overall goal of requirements engineering is risk reduction

Not really an exact “science” though much formalism exists（只是工程实践要灵活运用）

Modeling（建模：表达需求）

Describing a system at a high level of abstraction

Used for requirements and specification

Many notations have existed over time

State machines（状态机）

Entity-relationship diagrams（实体关系图 — ER图）

Dataflow diagrams（数据流图）

UML

Unified Modeling Language（“Union of all Modeling Languages”）

Enormous language

Many loosely related styles under one roof

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

Nearly everything in UML is optional

Models are rarely complete

UML is “open to interpretation”

UML is designed to be extended

Static Modeling in UML

Static modeling captures the fixed, code-level relationships in the system

Class diagrams（widely used） 类图

Package diagrams 套件图

Component diagrams 组件图

Composite structure diagrams 组合结构图

Deployment diagrams 部署图

Behavioral Modeling with UML

Behavioral diagrams are used to capture the dynamic execution of a system

Use case diagrams（widely used） 用例图

Interaction diagrams 交互图

Sequence diagrams（widely used） 序列图

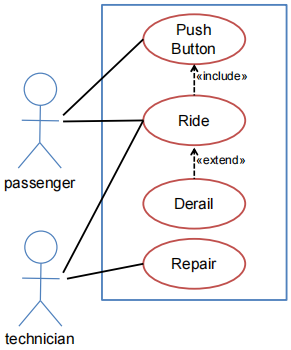
Collaboration diagrams 协作图

State diagrams（widely used） 状态图

Activity diagrams（widely used） 活动图

Use case diagrams

capture the requirements of a system from the user’s perspective

Actors

A role that a user takes when invoking a use case

Use cases

a particular piece of functionality that the system must provide

Edges

from actor to use case

showing that the actor is involved in that use case

Use cases have relationships to each other

Inclusion（包含）

Generalization（泛化）/ specialization（特化）

Extension（拓展）expresses an exceptional variation of a use case

Supplemental information

usually in a separate document

Entry/exit conditions（pre-conditions and post-conditions）

Nonfunctional requirements

Statechart Diagrams

Show the various stages of an entity during its lifetime

Can be used to show the state transitions of methods, objects, components, etc.

State

a condition of a modeled entity

The initial state is represented as a solid black circle

Action

an atomic execution

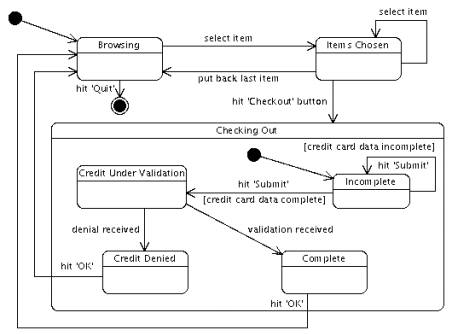
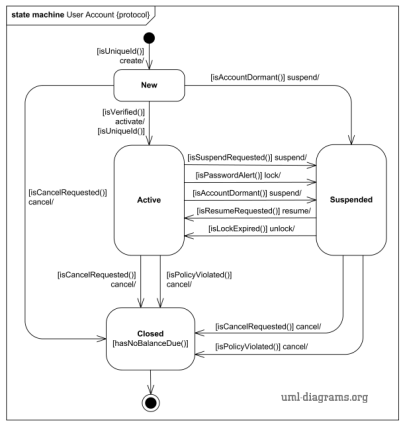
atomic means it completes without interruption

Activity

a more complex collection of behavior that may run for a long duration

Transition

between two states is represented as an arc from one state to another



UML Class Diagram

Lecture4 Information Hiding

Software Design Caveats

There is no definite formulation

There is no stopping rule

Solutions are not simply true or false

Abstraction

Modularity, coupling, and cohesion

Information hiding

Limited complexity

Hierarchical structure