CS 451
Software Engineering

Yuanfang Cai

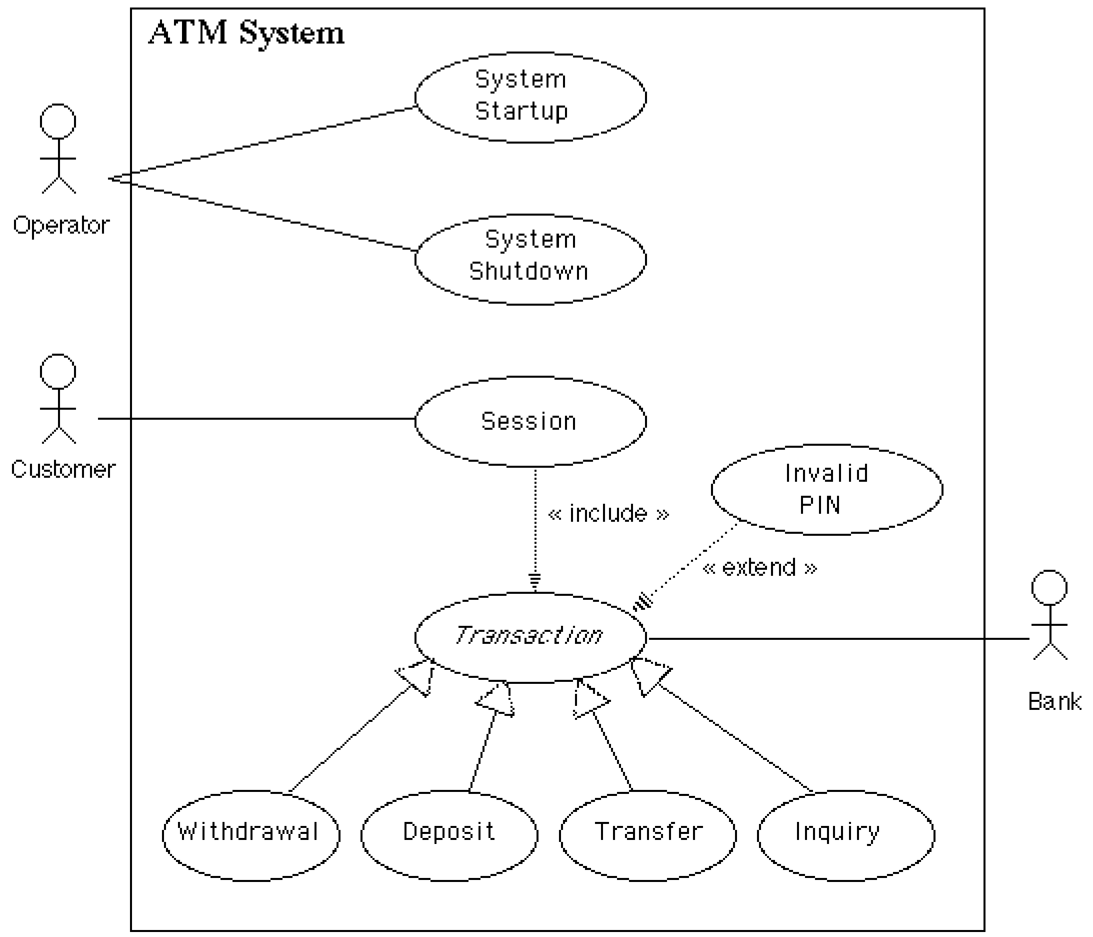
Room 104, University Crossings
215.895.0298

yfcai@cs.drexel.edu

Use Case

- A sequence of transactions performed by a system that yields a measurable result of values for a particular actor
  - What are the tasks of each actor?
  - Will any actor create, store, change, remove or read information in the system?
  - What use cases will create, store, change, remove or read this information?
  - Will any actor need to inform the system about sudden, external changes?
  - Does any actor need to be informed about certain occurrences in the system?
  - What use cases will support and maintain the system?
  - Can all functional requirements be preformed by the use cases?
- A use case typically represents a major piece of functionality that is complete from beginning to end. A use case must deliver something of value to an actor.

Another Example



Observed problems

- Mixing up a use case with its detailed subflows
- Trying to show sequential relations among use cases
- Trying to show conditions in use case diagrams
- Not using <<include>> or <<extend>> stereotypes correctly.
- Not using generalization relation correctly.

- The implicit actor, registrar, is missing
- Not all requirements are covered
- Relations among use cases are not labeled.
- Too much details

Template for Flow of Events

X Flow of Events for the <name> Use Case

X.1 Preconditions

What needs to happen (in another use case before this use case can start?

X.2 Main Flow

X.3 Subflows

Break “normal” flow into pieces

X.4 Alternative Flows

Things that happen outside of the “normal” flow

Flow of Events vs Scenario

- Flow of events enumerates all subflows and exception flows.
- Scenario is one path through your flow of events – choose one set of subflows and maybe an exception.
- When you’re testing, make sure you cover a reasonable (80%??) set of your scenarios.

Intersection Problem Statement

You will be simulating automobile traffic flow/traffic signals at a typical intersection. Traffic flows in both directions on each of the cross streets. Cars form into eight different queues at the intersection: N: from the north, headed straight south; NL: from the north, headed east (left at intersection); E: from the east, headed straight west; EL: from the east, headed south (left at intersection); S: from the south, headed straight north; SL: from the south, headed west (left at intersection); W: from the west, headed straight east; WL: from the west, headednorth (left at intersection).

Intersection Problem Statement

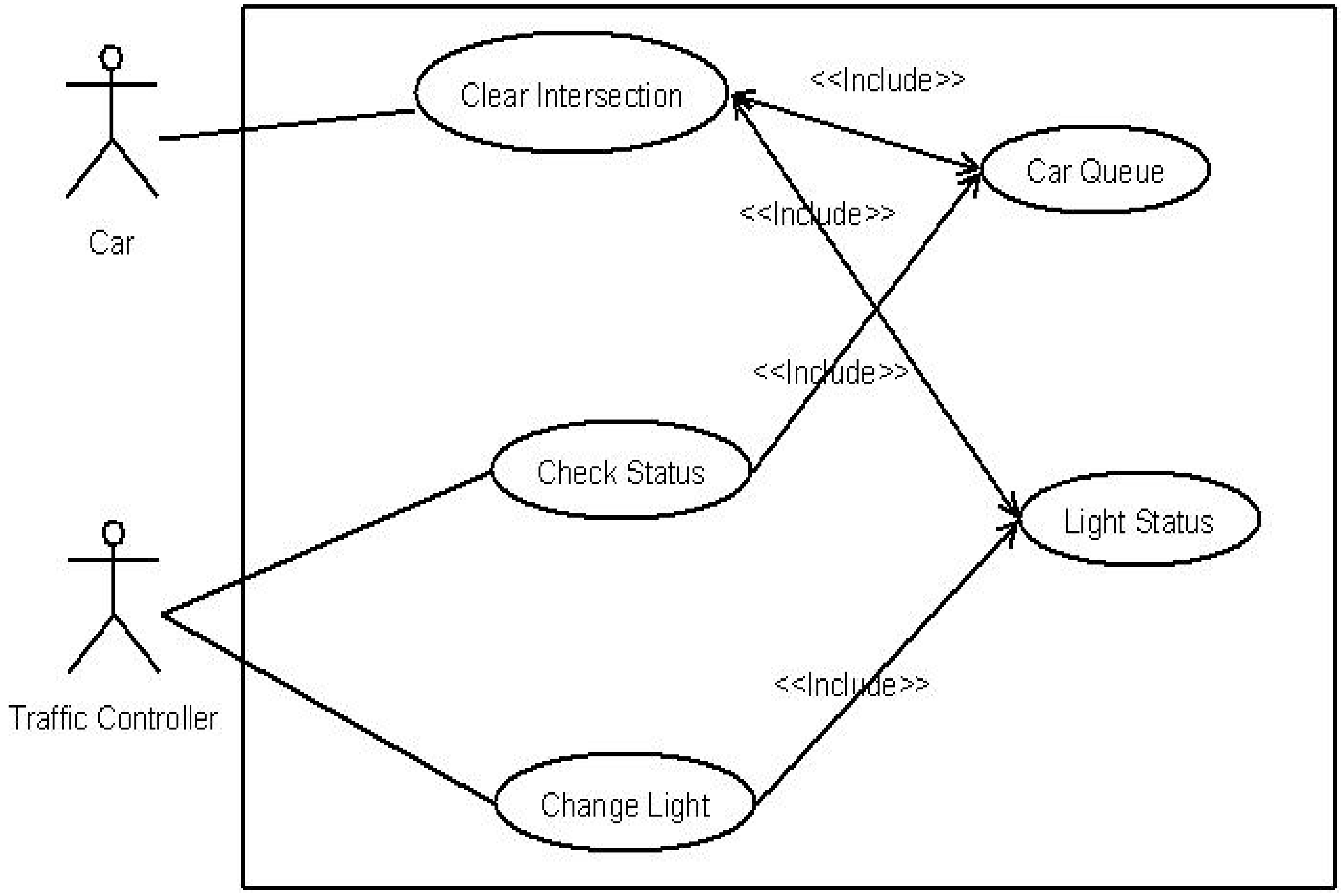
- When a car enters the intersection, if the queue there is empty and the light is green, they can clear the intersection. Else, they join the appropriate queue.
- When the system is initiated, the traffic signal allows traffic to flow from NL and SL. Next it allows traffic to flow from N and S. Then, it allows traffic flow from EL and WL. Lastly, it allows traffic to flow from E and W – then starts again with NL and SL and so forth.
- When a signal light changes it can allow up to five cars to pass through the intersection (get out of the queue).

Clear Intersection Example

- User wants to drive through an intersection.

- The user can only clear through the intersection if the traffic light is green and there are no cars in the intersection. Otherwise, the car needs to join a queue.

Clear Intersection Use Case Diagram



1. Flow of Events for the *Clear Intersection* Use Case

1.1 Preconditions

Traffic light has been initialized.

1.2 Main Flow

This use case begins when a car enters the intersection (1). The car checks status (2, 3). If the light is green, and the queue is empty, the car clears the intersection (4). Otherwise, it joins a queue (5).

1.3 Subflows

1 Enter Intersection

The car enters the intersection.

2 Check Light

The car checks whether the light is green via the *Report Status* subflow of the Light Status use case.

3 Check Queue

The car checks whether it is at the front of the queue via the *Report Status* subflow of the Car Queue use case.

4 Go

The car clears the intersection.

5 Join a Queue

The car joins the queue via the *Add to Queue* subflow of the Car Queue use case.

1.4 Alternative Flows

Scenario:
Car approaches intersection with green light and no queue

1.1 Preconditions

Traffic light has been initialized.

1.2 Main Flow

This use case begins when a car enters the intersection (1). The car checks status (2, 3). If the light is green, and the queue is empty, the car clears the intersection (4). Otherwise, it joins a queue (5).

1.3 Subflows

1 Enter Intersection

The car enters the intersection.

2 Check Light

The car checks whether the light is green via the *Report Status* subflow of the Light Status use case.

3 Check Queue

The car checks whether it is at the front of the queue via the *Report Status* subflow of the Car Queue use case.

4 Go

The car clears the intersection.

5 Join a Queue

The car joins the queue via the *Add to Queue* subflow of the Car Queue use case.

1.4 Alternative Flows

Scenario:
Car approaches intersection with red light or the queue is not empty

1.1 Preconditions

Traffic light has been initialized.

1.2 Main Flow

This use case begins when a car enters the intersection (1). The car checks status (2, 3). If the light is green, and the queue is empty, the car clears the intersection (4). Otherwise, it joins a queue (5).

1.3 Subflows

1 Enter Intersection

The car enters the intersection.

2 Check Light

The car checks whether the light is green via the *Report Status* subflow of the Light Status use case.

3 Check Queue

The car checks whether it is at the front of the queue via the *Report Status* subflow of the Car Queue use case.

4 Go

The car clears the intersection.

5 Join a Queue

The car joins the queue via the *Add to Queue* subflow of the Car Queue use case.

1.4 Alternative Flows

1. Flow of Events for the *Clear Intersection* Use Case

1.1 Preconditions

1.2 Main Flow

This use case begins when a car enters the intersection (1). The car checks status (2, 3). If the light is green, and the queue is empty, the car clears the intersection (4). Otherwise, it joins a queue (5).

1.3 Subflows

1 Enter Intersection

The car enters the intersection.

2 Check Light

The car checks whether the light is green via the *Report Status* subflow of the Light Status use case.

3 Check Queue

The car checks whether it is at the front of the queue via the *Report Status* subflow of the Car Queue use case.

4 Go

The car clears the intersection.

5 Join a Queue

The car joins the queue via the *Add to Queue* subflow of the Car Queue use case.

1.4 Alternative Flows

E-1 Light Out: If S-2 determines that the light is not red, yellow, or green, the car must wait for a clear intersection and proceed with caution.

E-2 Accident: If an accident is blocking the intersection, the car will cause a traffic problem by rubbernecking as it slowly drives around it.

2 Flow of Events for the ***Check Status*** Use Case

2.1 Preconditions

None.

2.2 Main Flow

Traffic controller gets the status of the queues (S-1) and reports the status (S-2).

2.3 Subflows

1 Obtain Status: Traffic controller sends a message asking for the how many cars are in each queue via the *Report Status* subflow of the Car Queue use case.

2 Report Status: Traffic controller reports the number of cars in each queue.

2.4 Alternative Flows

3 Flow of Events for the *Change Light* Use Case

3.1 Preconditions
None

3.2 Main Flow
Traffic controller initializes the traffic lights (1). Traffic controller changes the lights change (2)

3.3 Subflows
1 Initiate Lights: Traffic controller initializes all lights to red except NL and SL. NL and SL are green.
2 Change Light: Traffic controller changes the lights in the following order and send a message to the *Update Status* subflow of the Light Status use case**.**
When the system is initiated, the traffic signal allows traffic to flow from NL and SL. Next it allows traffic to flow from N and S. Then, it allows traffic flow from EL and WL. Lastly, it allows traffic to flow from E and W -- then starts again with NL and SL and so forth.

3.4 Alternative Flows

4 Flow of Events for the *Light Status* Use Case

4.1 Preconditions

None.

4.2 Main Flow

The system updates (S-1) and reports (S-2) the status of a traffic light color.

4.3 Subflows

1 Update Status: The system changes the color of the lights and sends a message to the *Check Status* subflow of the Clear Intersection use case.

2 Report Status: The system returns a report indicating the status of the light for each queue.

4.4 Alternative Flows

Problems observed

- Not understand the requirements.
- Incorrect initialization
- Repeat the content of other use cases
- Not showing the relation to other use cases
- Show program constructs.

Why Scenarios and Use Cases?

- Utterly comprehensible by the user
  - Use cases model a system from the users’ point of view (functional requirements)
    - Define every possible event flow through the system
    - Description of interaction between objects
- Great tools to manage a project. Use cases can form basis for whole development process
  - User manual
  - System design and object design
  - Implementation
  - Test specification
  - Client acceptance test
- An excellent basis for incremental & iterative development