Use Case Modeling

ESOF 16/17

Agenda

- What is a Use Case?
- Benefits of the Use Cases
- Developing the Use Case model
  - System
  - Actor
  - Use Case
  - Use Case Relationships
- Example: TVRS Use Cases

What is a Use Case?

- Created by Ivar Jacobson (1994)
- "A use case is a sequence of transactions in a system whose task is to yield a measurable value to an individual actor of the system"
- Describes WHAT the system (as a "Black Box") does from a user's (actor) perspective
- The Use Case Model is NOT an inherently object oriented modeling technique

Benefits of Use Cases

- Captures operational requirements from user's perspective
- Gives a clear and consistent description of what the system should do
- A basis for performing system tests
- Provides the ability to trace functional requirements into actual classes and operations in the system

UML Use Case Diagrams

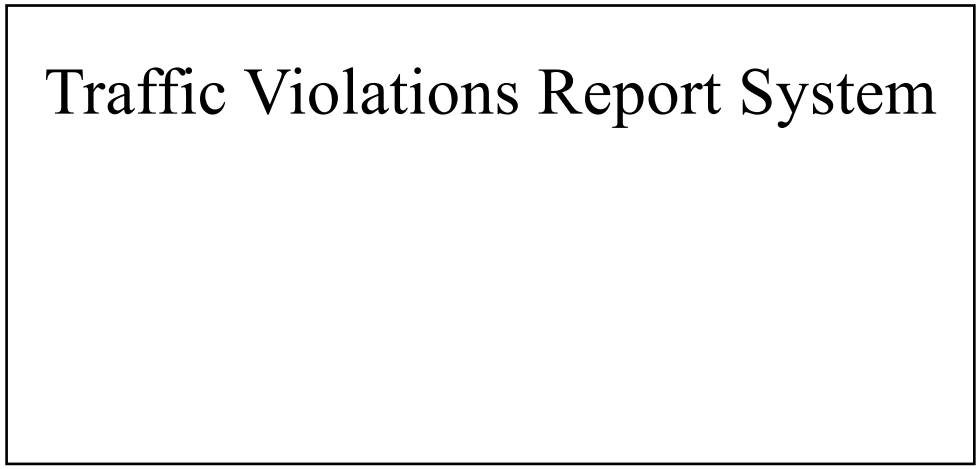
- A Use Case model is described in UML (Unified Modeling Language) as one or more Use Case Diagrams (UCDs)
- A UCD has 4 major elements:
  - The **system** described
  - The **actors** that the system interacts with
  - The **use-cases**, or services, that the system knows how to perform
  - The **relationships** between the above elements

System

- As part of use-case modeling, the **boundaries of the system** developed must be defined
- Defining the boundaries of the system is not trivial
  - Which tasks are automated and which are manual?
  - Which tasks are performed by other systems?
    - The entire solution that we supply should be included in the system boundaries
    - Incremental releases

System (cont.)

- A system in a UCD is represented as a box
- The name of the system appears above or inside the box



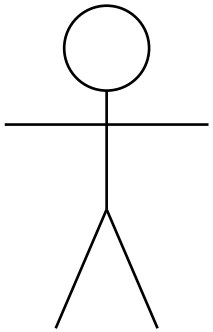
Actor

- Someone or something that interacts with the system (exchanges information with the system)
- An actor represents a role played with respect to the system, not an individual user of the system
- Example:
  - Policeman – Enters data
  - Supervisor – Allowed to modify/erase data
  - Manager – Allowed to view statistics.
- A single user may play more than one role

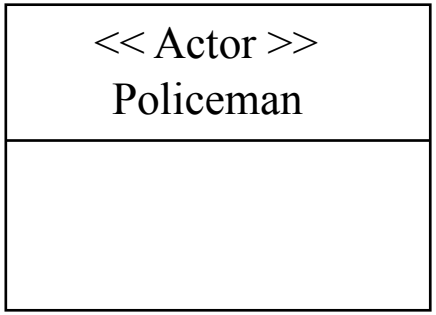
Actor (cont.)

- Actors have **goals**:
  - Add a Traffic Violation
  - Lookup a Traffic Violation
- Actors don't need to be human
  - May be an external system that interfaces with the developed system
- An actor has a name that reflects its role

Actor Icons

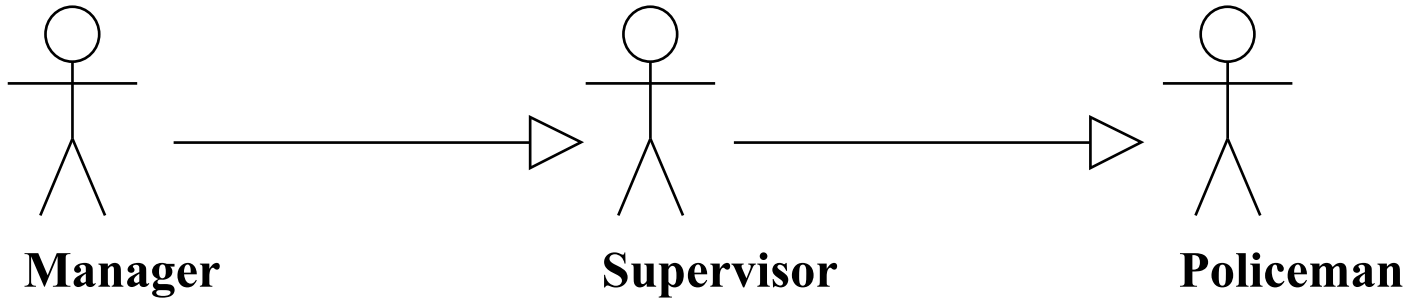


Policeman



Relationships between Actors

- When several actors as part of their roles, also play a more generalized role, it is described as **generalization**
- The behavior of the general role is described in an actor super-class
- The specialized actors inherit the behavior of the super-class and extend it in some way
- Relationships between actors are not always necessary



Use Case

- Represent a complete behavior as perceived by an actor
  - A use case satisfies an actor's goal
- Always initiated by an actor
- A use case is complete
  - Don't divide a use case into smaller use cases that implement each other (functional decomposition)

Use Case Description

- The scenarios of a use case are normally described textually
  - A simple and consistent specification about how the actors and the system interact
  - Use case description template
- Describe at the level of user intentions and system responses
  - Free of technology and mechanism details, especially those related to user interface

UC Description Template

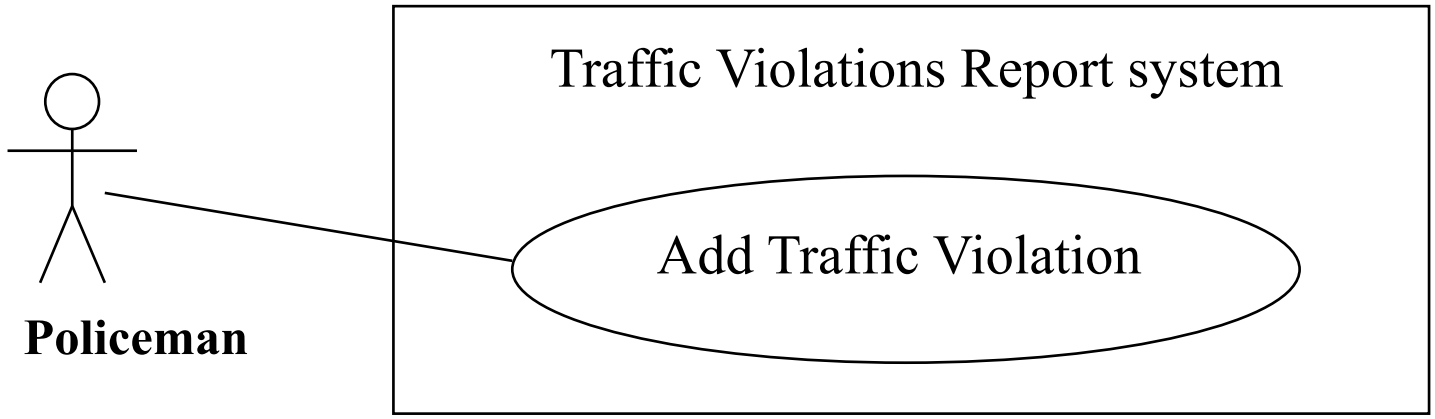
- **Name**
  - Name of use case, usually close to the user's goal
  - Forward traceability (unique)
- **Actors**
- **Goal description**
- **Reference to requirements**
  - Backward traceability
- **Pre-conditions**
  - The necessary conditions before the use case can be performed
  - Could be other Use Cases as well
- **Description**
  - A description of the basic or normal course that should be taken by the system if the system should perform as intended

UC Description Template (cont.)

- **Post-conditions**
  - The state of the system after the use case is performed
  - The value delivered to the actor
  - Distinguishes between variations and exceptions
- **Variations**
  - Expected condition causing the branch
  - Description of the alternative course or name of the **extending** Use Case
- **Exceptions**
  - Unexpected condition causing the branch (conflicts with post-condition)
  - Description of the alternative course

Use Case (cont.)

- Use Case Icon
  - An ellipsis containing the name of the Use Case
  - Placed inside the boundaries of the modeled system
  - Connected to at least one actor with a communication association
    - Except for specialized / extending use cases.

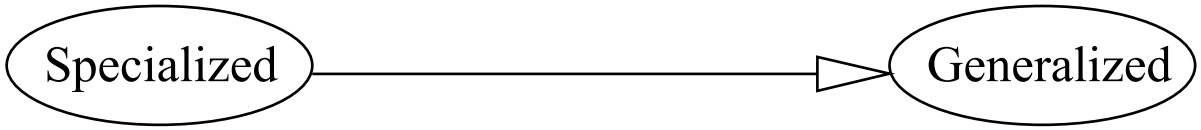


Use Case Relationships

- **Generalization**: A generalized Use Case describes the common of other specialized Use Cases.
- **Inclusion:** A Use Case is a part of another Use Case.
- **Extension:** A Use Case may extend another Use Case.

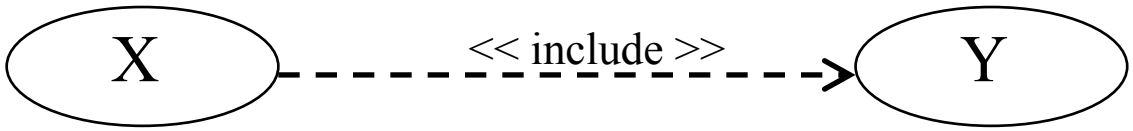
Generalization Relationships

- Used when a number of Use Cases all have some subtasks in common, but each one has something different about it
- The generalized and specialized use cases share the same goal
- A specialized Use Case may capture an alternative scenario of the generalized Use Case
- The Specialized use case may interact with new actors.
- The Specialized use case may add pre-conditions and post-conditions (AND semantics).



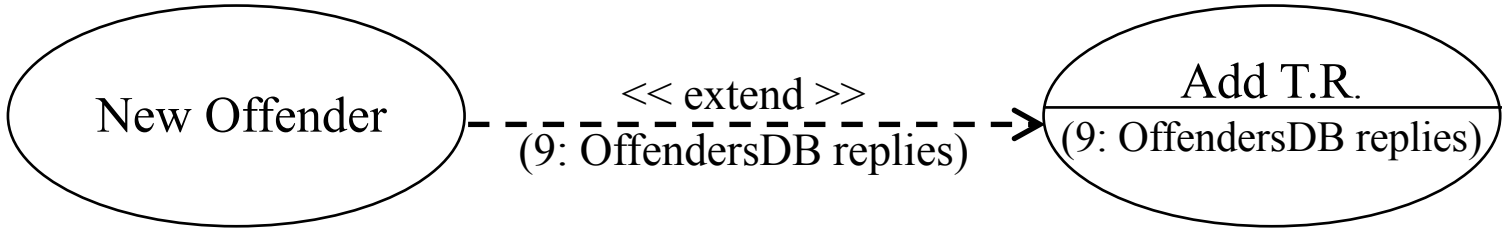
Include Relationship

- In older versions: “uses”
- When a number of Use Cases have common behavior, which can be modeled in a single use case
- X << includes >> Y indicates that the process of doing X always involves doing Y at least once
- The included Use Case must be complete
- X must satisfy the pre-conditions of Y before including it
- Not necessarily preserves the pre or post conditions.



Extend Relationship

- Serves as extension point to another Use Case
- The extended Use Case must explicitly declare its extension points
- The extension conditions of the extended Use Case are part of the pre-conditions (AND semantics)

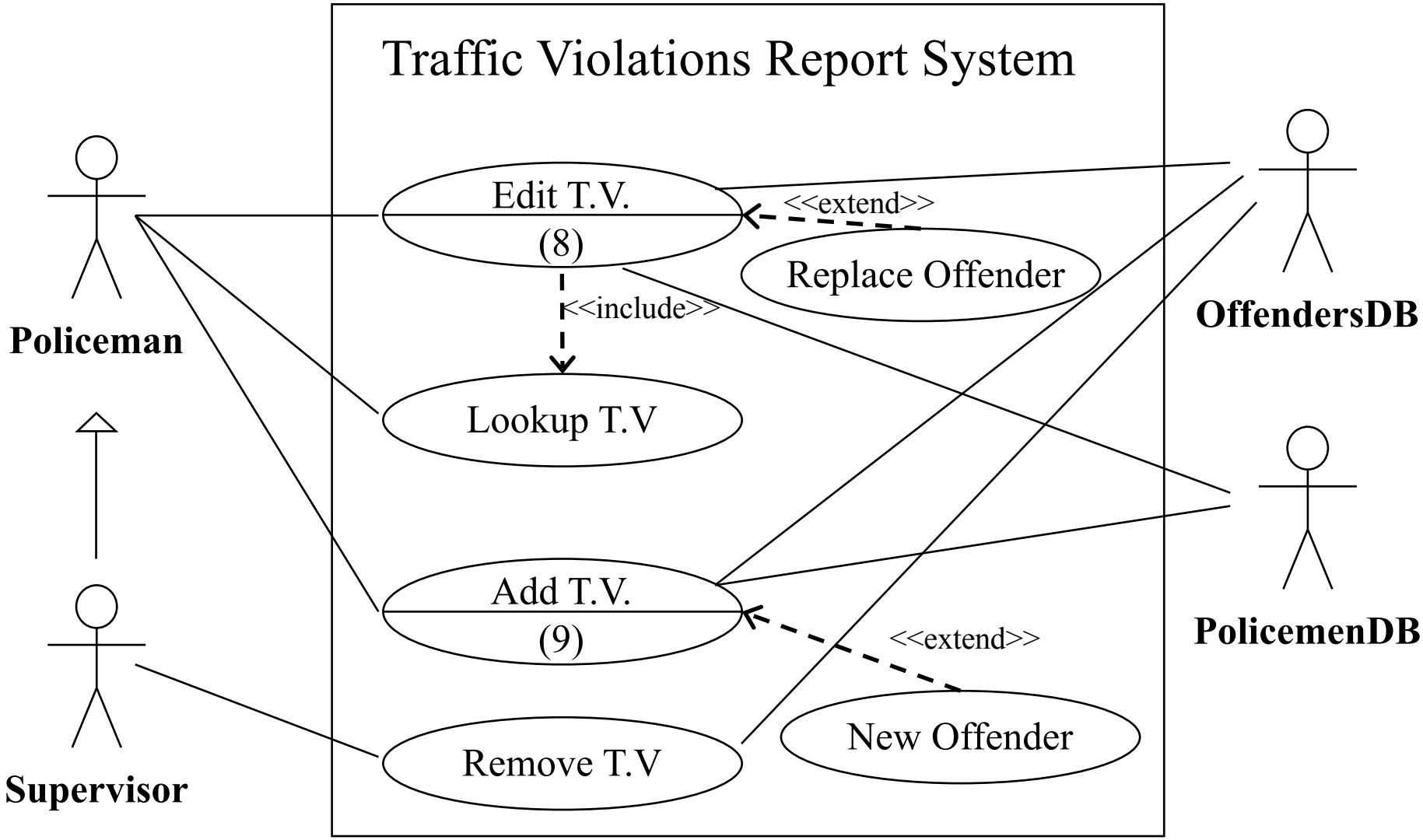


Recommended Workflow

1. Identify actors (and their relationships if necessary)
2. For each actor identified and until no new UC is discovered do
   a. Find all the goals of the actor
   b. Decide on the main course of success for each goal
   c. Create a Use Case for each of the goals
      - New actors/goals may be discovered
   d. Validate/correct existing Use Cases
3. Draw the Use Case diagram
   – Simplify model by repeating the process incase the produced diagram is too complex

Example:
TVRS Use Cases

TVRS Use Case Model



TVRS - Remove TV

- Name: Remove Traffic Violation
- Actors: Supervisor, OffendersDB.

External System

- Goal: Remove an existing Traffic Violation
- References to requirements: 1.2.3, 1.3.2.4, …
- Pre-conditions:
  - Normal Course of "Lookup Traffic Violation" UC is completed, and the details of an existing Traffic Violation are displayed
- Description:
  1. Supervisor calls for deletion of the chosen Traffic Violation
  2. TVRS prompts Supervisor for confirmation

TVRS - Remove TV

3. Supervisor confirms
4. TVRS requests OffendersDB to delete the Traffic Violation from the offender's record
5. OffendersDB approves that the Traffic Violation has been deleted
6. TVRS allows Supervisor to look up a new Traffic Violation as described in the "Lookup Traffic Violation" UC

– Post-conditions:
  - Removed Traffic Violation is no longer stored in the TVRS.
  - Traffic Violation is removed from the offender's record in the OffendersDB
  - "Lookup Traffic Violation" form is displayed

TVRS - Remove TV

Goal is not fulfilled

– Exceptions:
  - 3a: Supervisor cancels:

    3a1: TVRS Continues to item 6 without removing the Traffic Violation
  - 5a: Traffic Violation is not removed from the OffendersDB

    5a1: TVRS displays an error message describing the failure

    5a2: TVRS continues to item 6 without clearing chosen Traffic Violation details, and without deleting the Traffic Violation

TVRS - Add TV

(With planted mistakes)

- Name: Add Traffic Violation
- Actors: Policeman, PolicemenDB, OffendersDB, ~~Traffic Violation.~~ *TVRS*
- Goal: Add a new Traffic Violation to ~~OffendersDB.~~
- References to requirements: …
- Pre-conditions:
  - ~~Pliceman tries to add Traffic Violation.~~
  - *The Traffic Violation Management window is displayed*
- Description:
  1. ~~Policeman presses "Add" button~~
  1. *Policeman calls for addition of a new Traffic Violation*
  2. TVRS displays an empty Traffic Violation Details form
  3. Policeman enters violation details and calls for saving the new Traffic Violation

TVRS - Add TV

(With planted mistakes)

4. TVRS prompts Policeman for confirmation.
5. Policeman confirms

TVRS asks PolicemenDB

6. ~~PolocemenDB is asked~~ whether or not the policeman is known

7. PolicemenDB replies that the policeman is known
8. TVRS asks the OffendersDB whether or not the offender is known
9. *[Extenstion Point]* OffendersDB replies that the offender is known

…

Always?

TVRS - Add TV

(With planted mistakes)

- Post-conditions:
  - New Traffic Violation is stored in the TVRS
  - TVRS displays an empty Traffic Violation Details form
- Variations:
  - 5a: Policeman cancels

    ~~5a1: TVRS shows error message and closes Traffic Violation Management window.~~

    5a1: TVRS continues to item 2 without clearing the traffic violation details entered by Policeman
  - 9a: OffendersDB replies that the offender is not known.
    - Described in Use Case "New Offender"
  - 7a: Policeman is not stored in the PolicemenDB

    7a1: TVRS displays an error message

    7a2: TVRS continues to item 2 without clearing Traffic Violation details entered by Policeman
  - ...

Goal may be fulfilled

TVRS - Add TV

(With planted mistakes)

– Exceptions:

- 3a: Policeman cancels addition of the new Traffic Violation

  3a1: ~~TVRS continues to item 2 without clearing the traffic violation details entered by Policeman~~

  3a1: TVRS displays the "Traffic Violation Management" window without adding the Traffic Violation

- …

Use Case terminated

TVRS – New Offender

- Name: New Offender [extends “Add Traffic Violaton”]
- Actors:
- Goal:
- References to requirements: …
- Pre-conditions:
  - Offender is not stored in the OffendersDB

TVRS – New Offender

- Description:
  - 9a: OffendersDB replies that the offender is not known. [Add Traffic Violation]
  - 9b: TVRS displays an empty “Offender Details form”
  - 9c: Policeman enters offender details and calls for saving the new details
  - 9d: TVRS prompts Policeman for confirmation
  - 9e: Policeman confirms
  - 9f: TVRS requests OffendersDB to store the new offender
  - 9g: OffendersDB replies that offender was stored successfully
- Post-conditions:
  - New Offender is stored in the offenders DB
- ...

Rational Rose™