

System Design & Analysis

LECTURE 07

Use case Relationship

- ▶ Some use cases have similar step in their behavior
- ▶ Others may have different modes or special cases
- ▶ Describing such use cases may cause to repeat them in the diagrams
- ▶ This will lead to large and complicated diagrams
- ▶ UML provides different notations for representing these behaviors

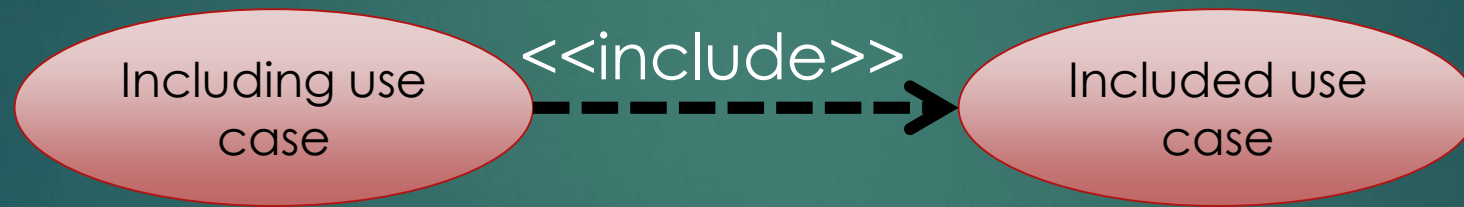
<<include>> Relationship

3

- ▶ A use case may reuse all the steps from another use case
 - ▶ It “includes” the steps from another use case
- ▶ This can be described through <<include>> relationship [also known as Gil maze]
- ▶ In diagram, shown as a dashed arrow between use case with <<include>> label
 - ▶ Tail end it towards the use case that reuses the steps
 - ▶ Arrow end points towards the use case that is reused
- ▶ Include use cases are mandatory and not optional

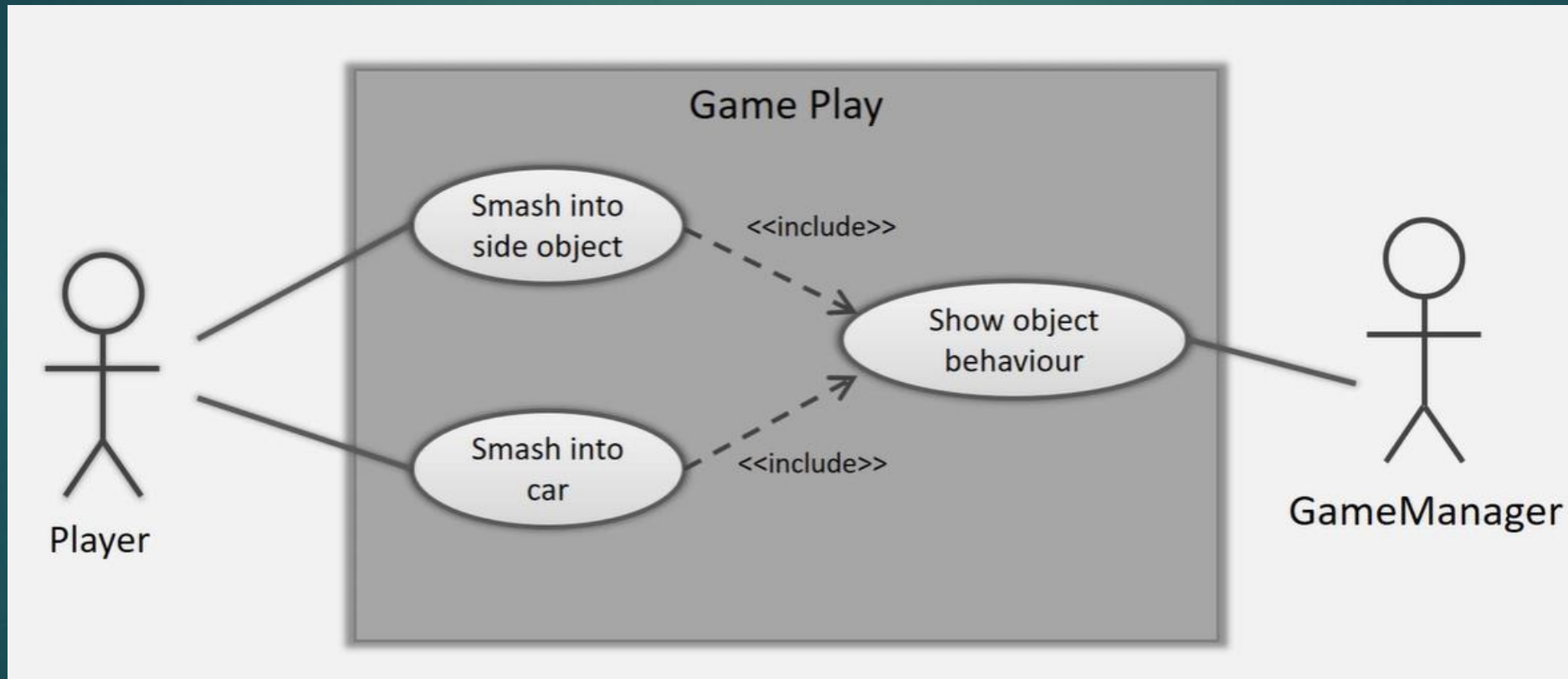
<<include>> Notation

4



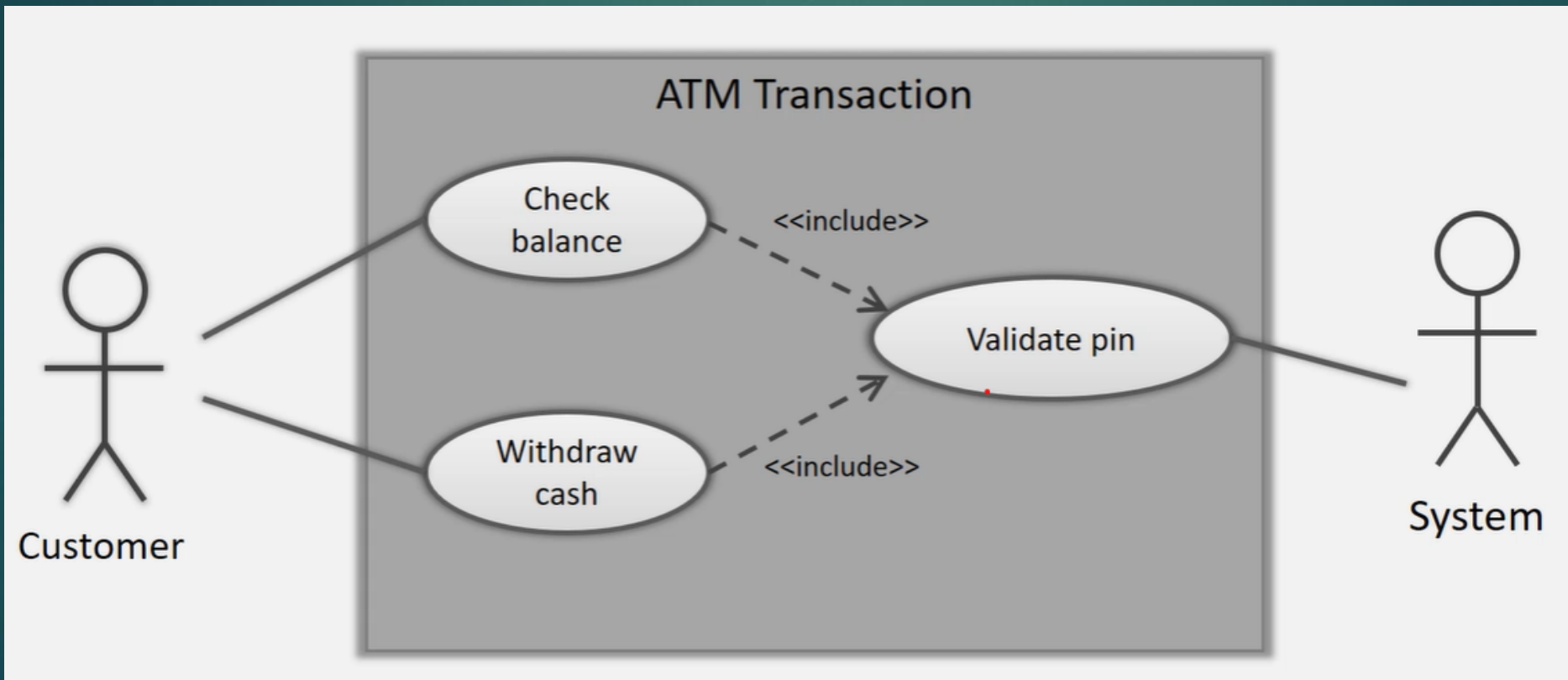
Example: <<include>>

5



Example: <<include>>

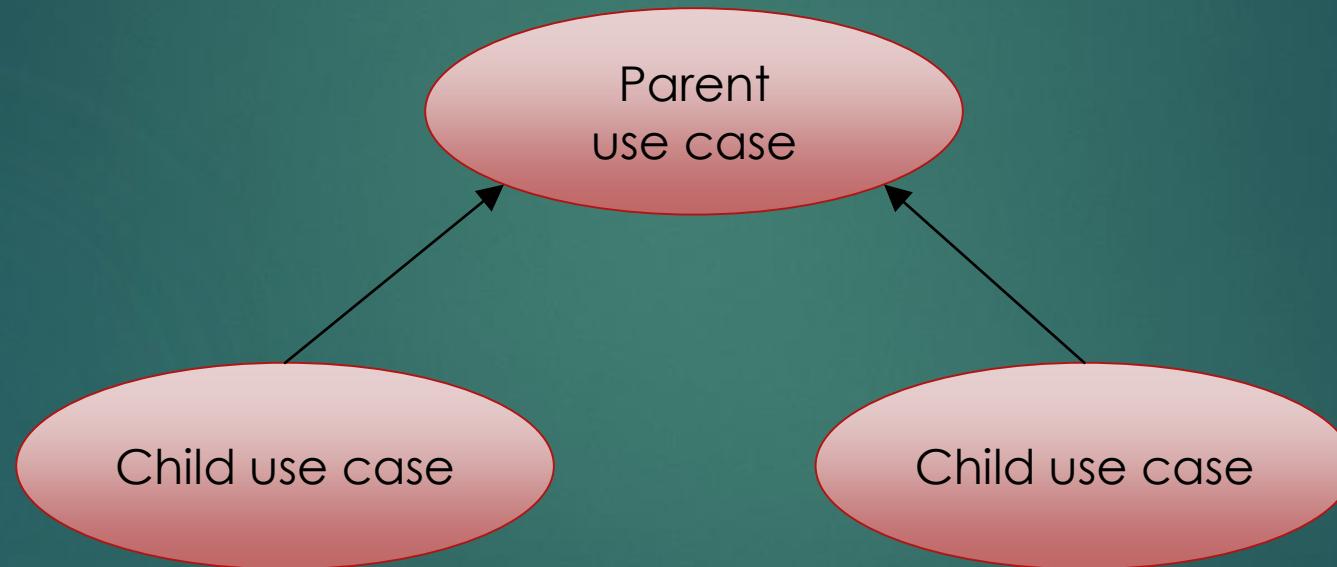
6



Generalization

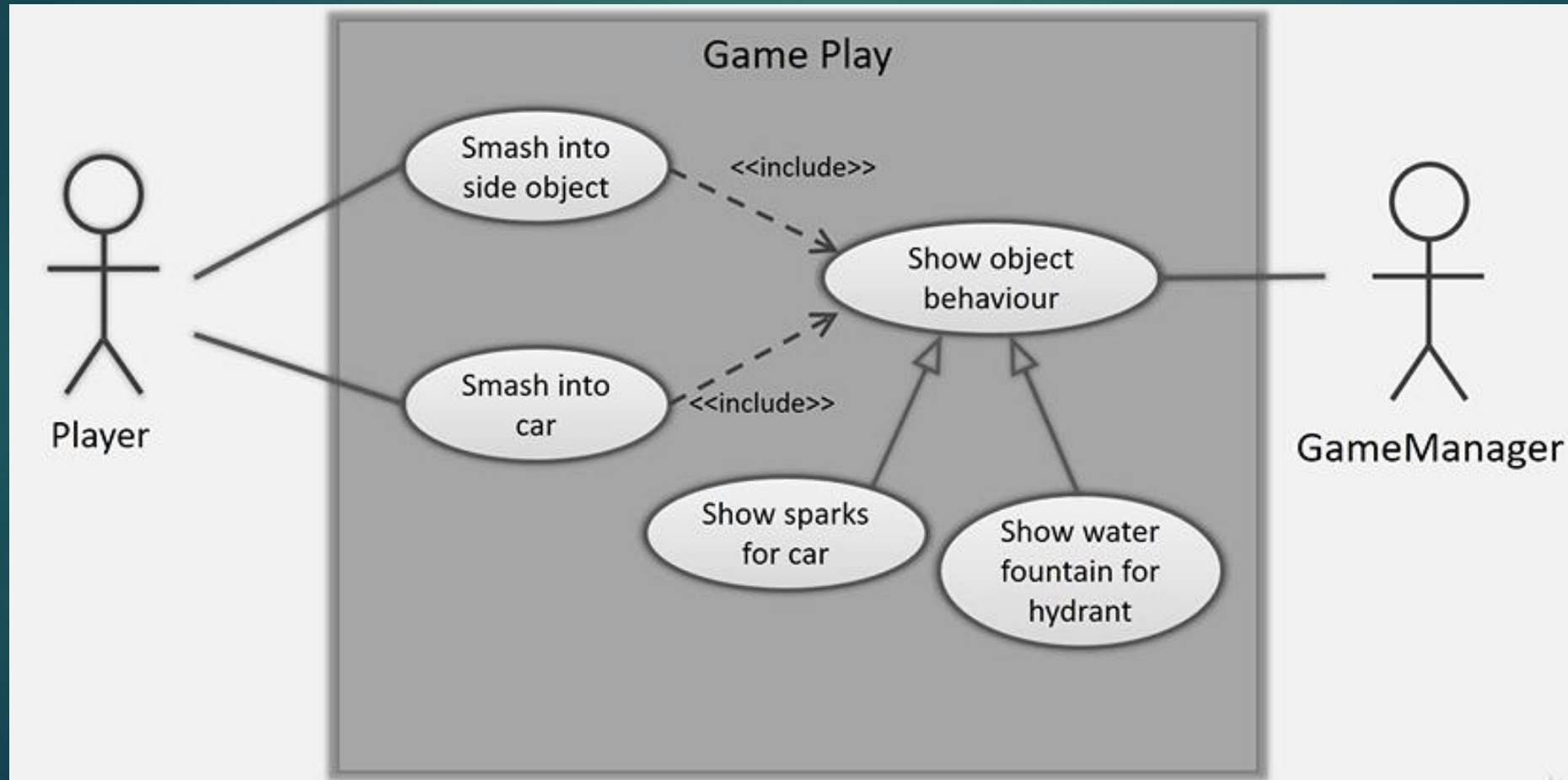
- ▶ This is similar to inheritance in object oriented programming.
- ▶ Used to show that one use case is a type of another, but with some changes
- ▶ Depicted through the generalization arrow
- ▶ Arrow head points to generalized use case
- ▶ Tail point to specialized use case

Generalization Notation



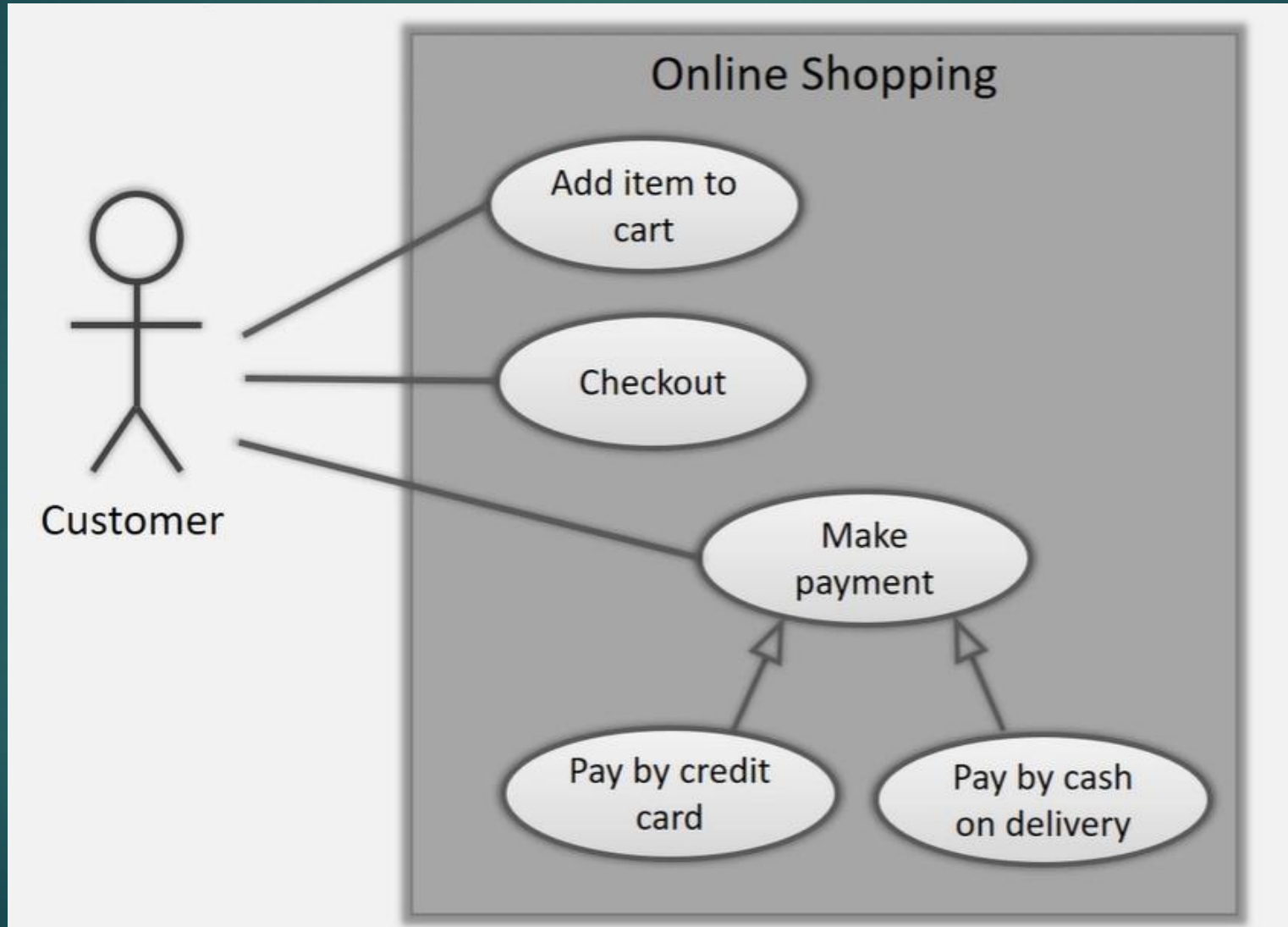
Example: Generalization

9



Example: Generalization

10



<<extend>> Relationship

11

- ▶ This is used to specify an optional behaviour
- ▶ This behavior appears as an extended use case
- ▶ Independent of the main use case, but owned by it
- ▶ Shown as a dashed arrow with <<extend>> label
- ▶ This behaviour is optional [whether it executes or not depends on some factor]

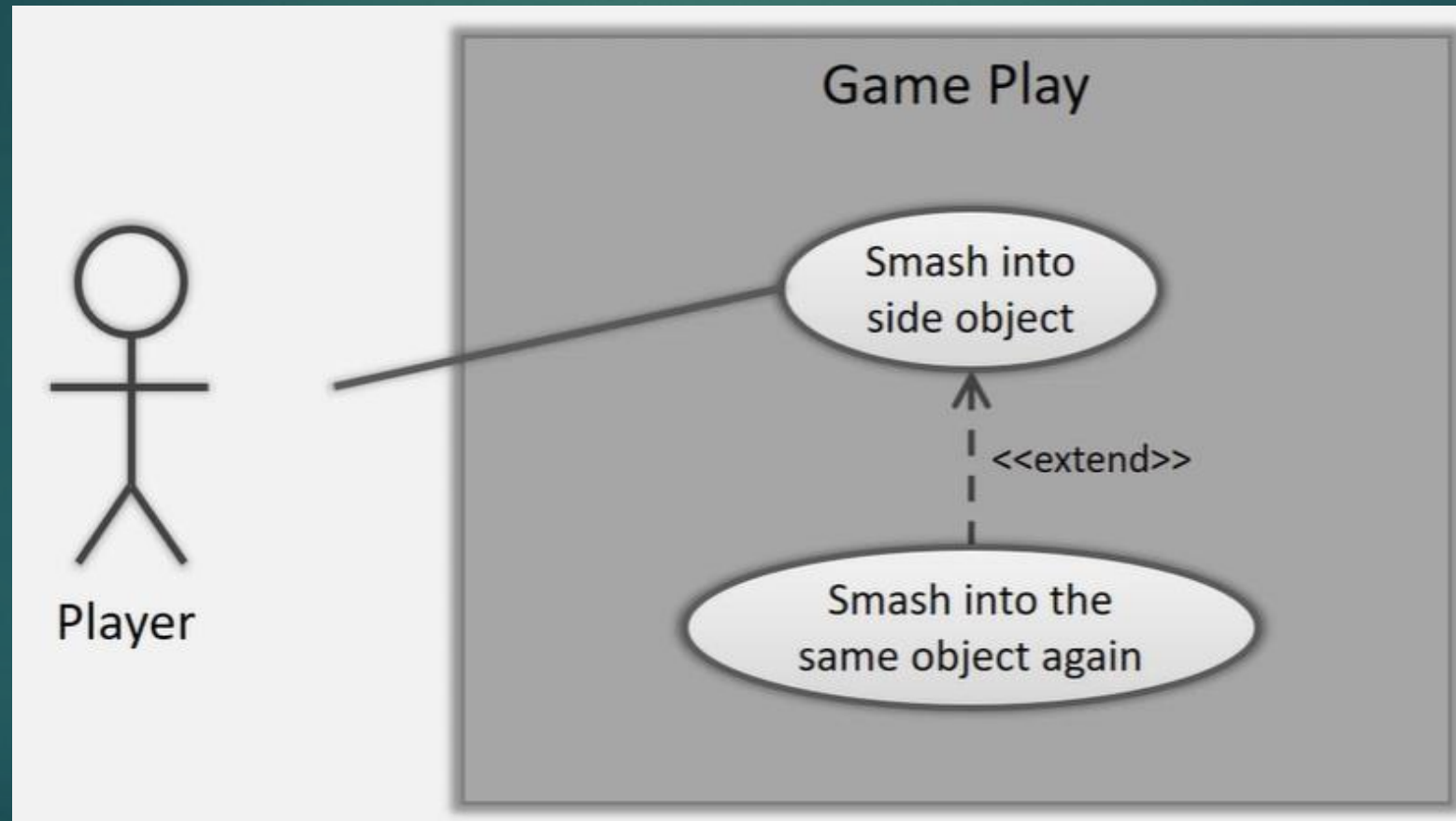
<<extend>> Notation

12



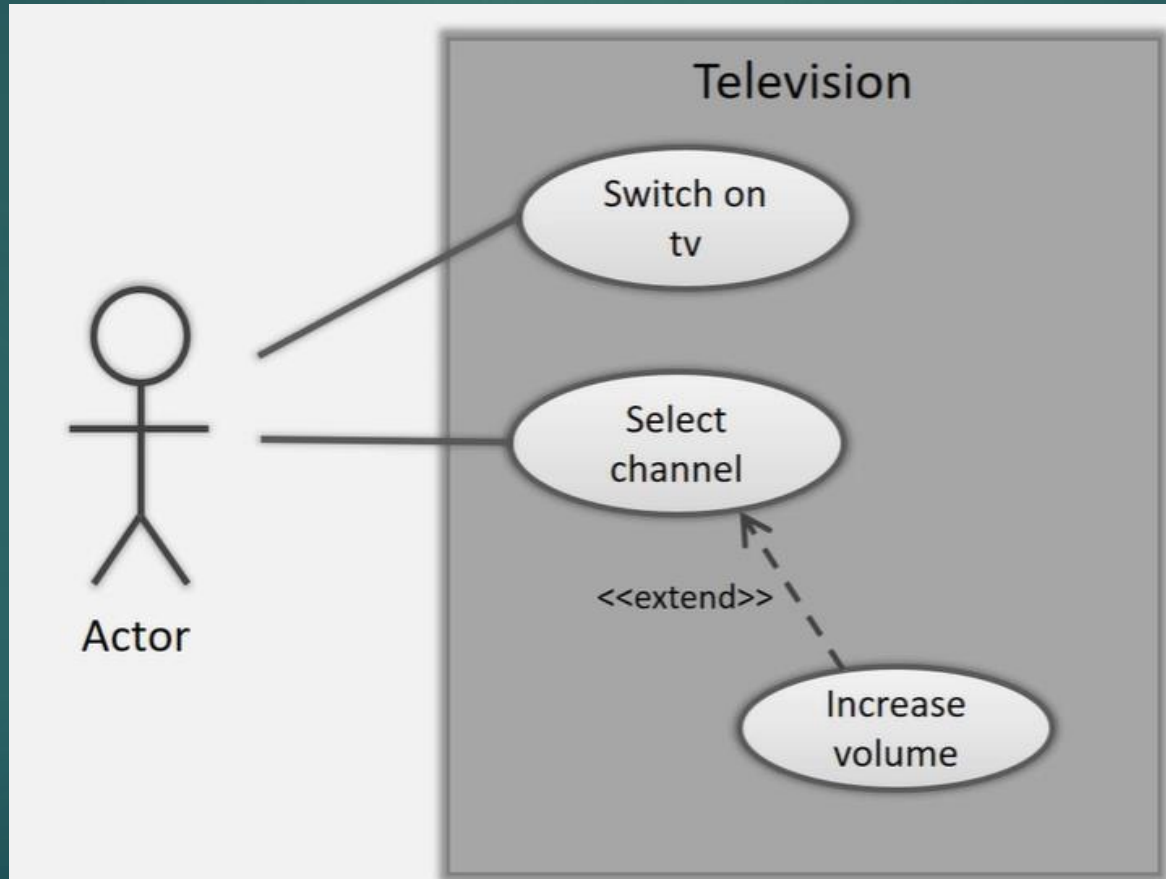
Example: <<extend>>

13



Example: <<extend>>

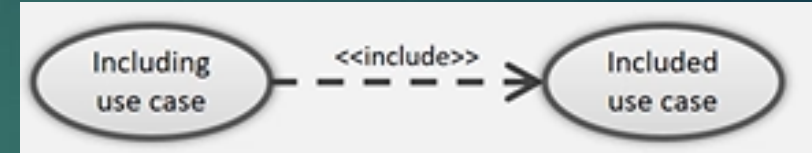
14



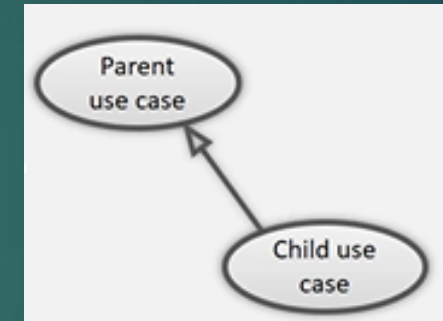
Summary

15

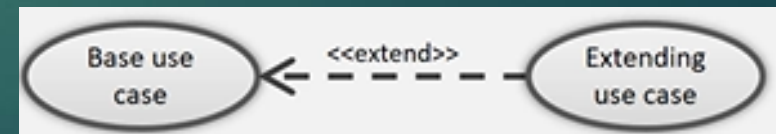
- ▶ <<include>>
 - ▶ Reuse steps from another use case mandatorily

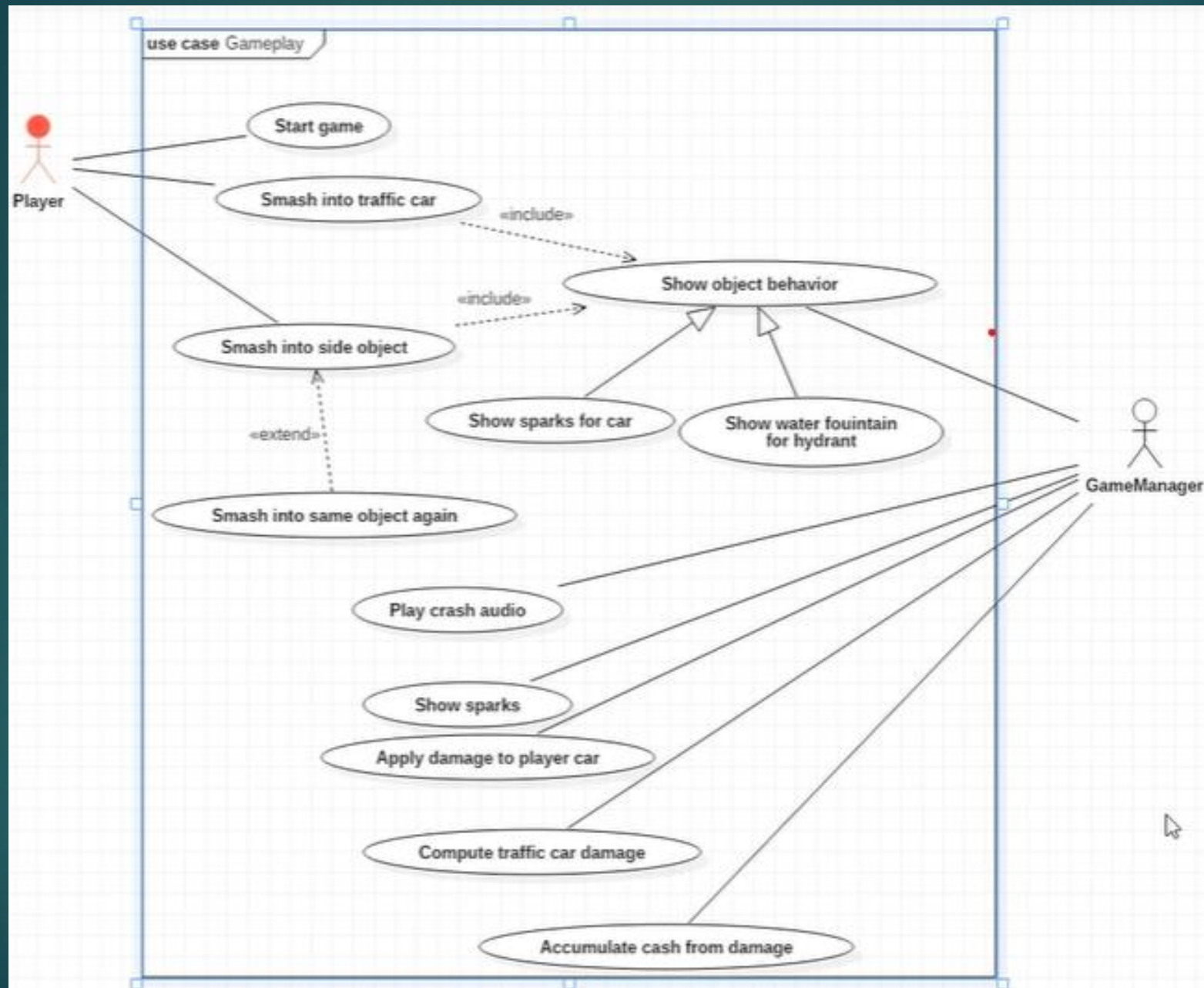


- ▶ Generalization
 - ▶ One use case is type of another with some change



- ▶ <<extend>>
 - ▶ Optional behavior of a use case





Guideline: Use case Description

17

- ▶ Start with goal
- ▶ Write the use case as narratives[e.g like a story]
- ▶ Use simple language
- ▶ Keep the technology specific out
- ▶ Don't include UI elements in use case
- ▶ Every use case should give one guarantee about the behavior of the system

Guideline: Use case Diagram

18

- ▶ Actors are the black box
- ▶ Actors don't interact with each other
- ▶ Primary actor on left side and secondary actor on right side
- ▶ Place use case in logical order
- ▶ Place included use case to right of the invoking use case
- ▶ Place inheriting use case below parent use case

Example

19

- ▶ Customer types the pin number in the textbox on the screen
- ▶ Encrypt the pin and perform an exact match with the encrypted pin stored in the web server
- ▶ System will execute the sql query “insert into table records (...)” after transaction
- ▶ Customer will click the “checkout” button that will send HTTPS request to the server to prepare the system for presenting the payment options
- ▶ Customer enter the pin
- ▶ Validate pin
- ▶ System will update the database4
- ▶ Customer will checkout the item