Information Systems Analysis & Modeling

Lecture 6: More details on Use case

Mona Taghavi

LaSalle College
Montréal

Modeling within phases of System Development

- Requirement Analysis
  - The functionality users require from the system
  - **Use-case model**
- OO Analysis
  - Discovering classes and relationships
  - Class diagram
- OO Design
  - Result of Analysis expanded into technical solution
  - Sequence diagram, state diagram, etc.
  - Results in detailed specs for the coding phase
- Implementation (Programming/coding)
  - Models are converted into code

Use-cases

- A Use-case in UML is defined as a set of sequences of actions a system performs that yield an observable result of value to a particular actor.
- Actions can involve communicating with number of actors as well as performing calculations and work inside the system.
- A Use-case
  - is always initiated by an actor.
  - provides a value to an actor.
  - must always be connected to at least one actor.
  - must be a complete description.

Use Cases as Requirements

- The set of use case descriptions specifies the complete functional requirements of a system
- Things to remember
  - Use cases are requirements;
  - They are not all of the requirements
  - Not good for specifying user interfaces, data formats, business rules, non-functional requirements
  - They are not easy to write!

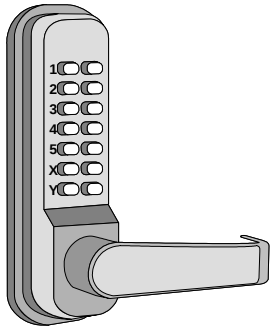
❖ Analogy: A good story is easy to read, but writing a good story is hard!

Which of these requirements should be represented directly in a use case?

- ❑ Special deals may not run longer than 6 months.
- ❑ Customers only become preferred after 1 year.
- ❑ A customer has one and only one sales contact.
- ❑ Database response time is less than 2 seconds.
- ❑ Web site uptime requirement is 99.8%.
- ❑ Number of simultaneous users will be 200 max.

Deriving Use Cases from System Requirements

REQ1: Keep door locked and auto-lock
REQ2: Lock when “LOCK” pressed
REQ3: Unlock when valid key provided
REQ4: Allow mistakes but prevent dictionary attacks
REQ5: Maintain a history log
REQ6: Adding/removing users at runtime
REQ7: Configuring the device activation preferences
REQ8: Inspecting the access history
REQ9: Filing inquiries



Actor	Actor's Goal	Use Case Name
Tenant	Unlock and enter home.	Unlock (UC-1)
Tenant	Lock the door.	Lock (UC-2)
Landlord	Create a new user account and allow access to home.	AddUser (UC-3)
Landlord	Retire an existing user account and disable access.	RetireUser (UC-4)
Tenant	Review the history of home accesses.	ViewHistory (UC-5)
Tenant	Configure the operational preferences for household devices.	SetDevicePrefs (UC-6)
Visitor	Visit a resident’s home.	VisitHome (UC-7)

Actors

- An actor is something or someone that interacts with the system.
- Actor communicates with the system by sending and receiving messages.
- Message sent by an actor may result in more messages to actors and to Use-cases.
- Actor is a role not an individual instance.

Finding Actors

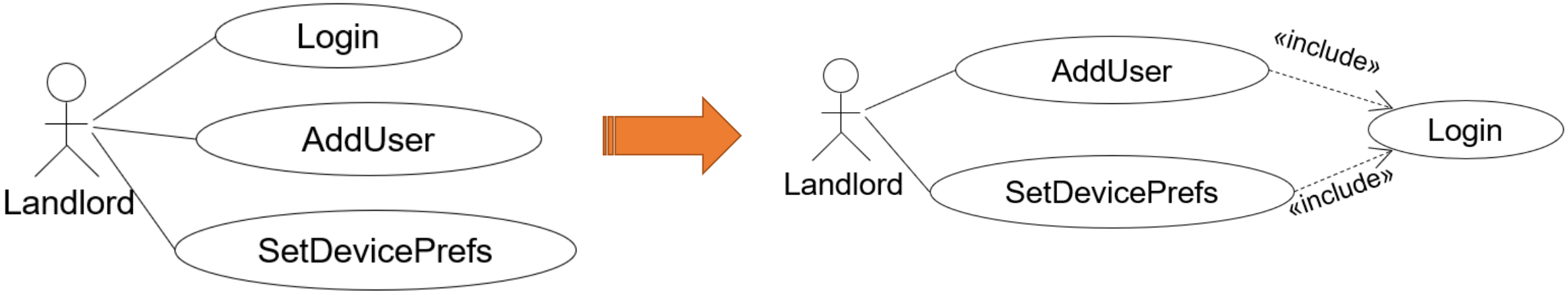
- The actors of a system can be identified by answering a number of questions:
  - Who will use the functionality of the system?
  - Who will maintain the system?
  - What devices does the system need to handle?
  - What other system does this system need to interact?
  - Who or what has interest in the results of this system?

Finding Use-cases

- For each actor ask these questions:
  - Which functions does the actor require from the system?
  - What does the actor need to do?
  - Could the actor's work be simplified or made efficient by new functions in the system?
  - What events are needed in the system?
  - What are the problems with the existing systems?
  - What are the inputs and outputs of the system?

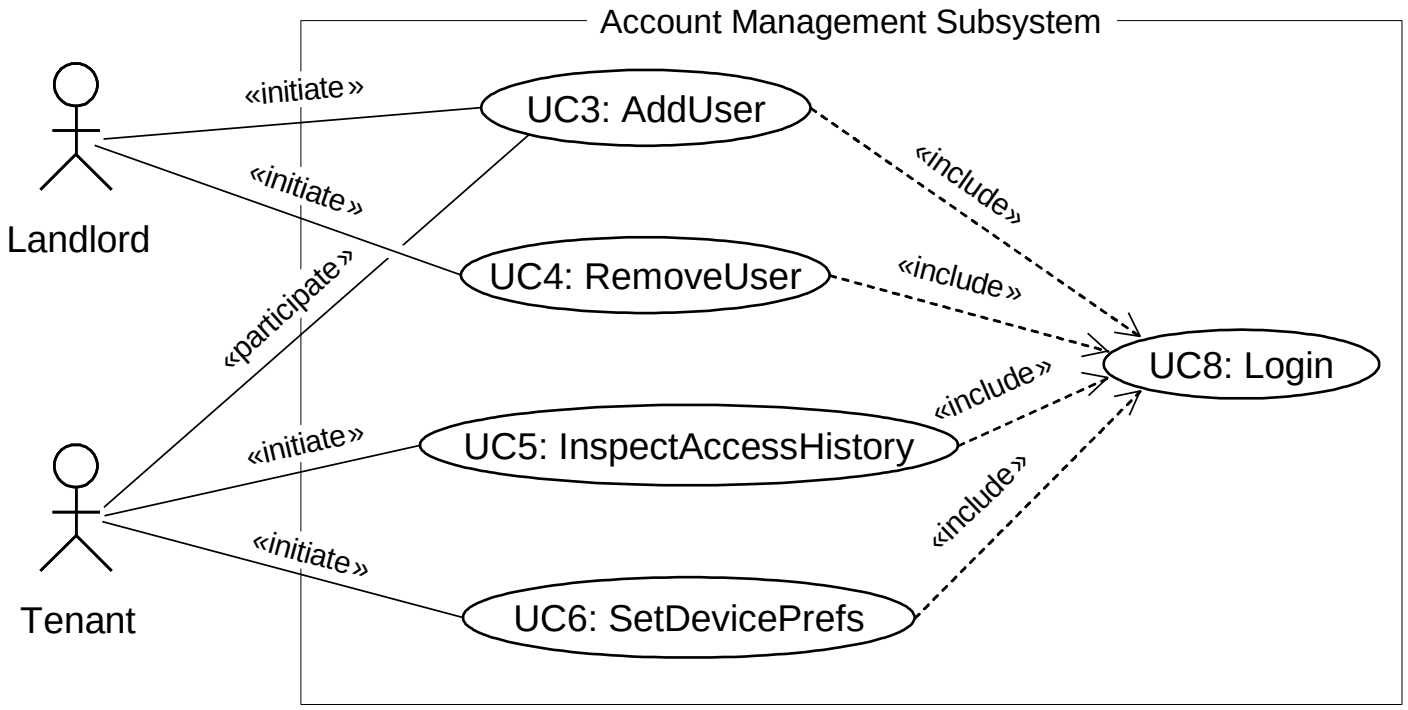
Relationships (Include)

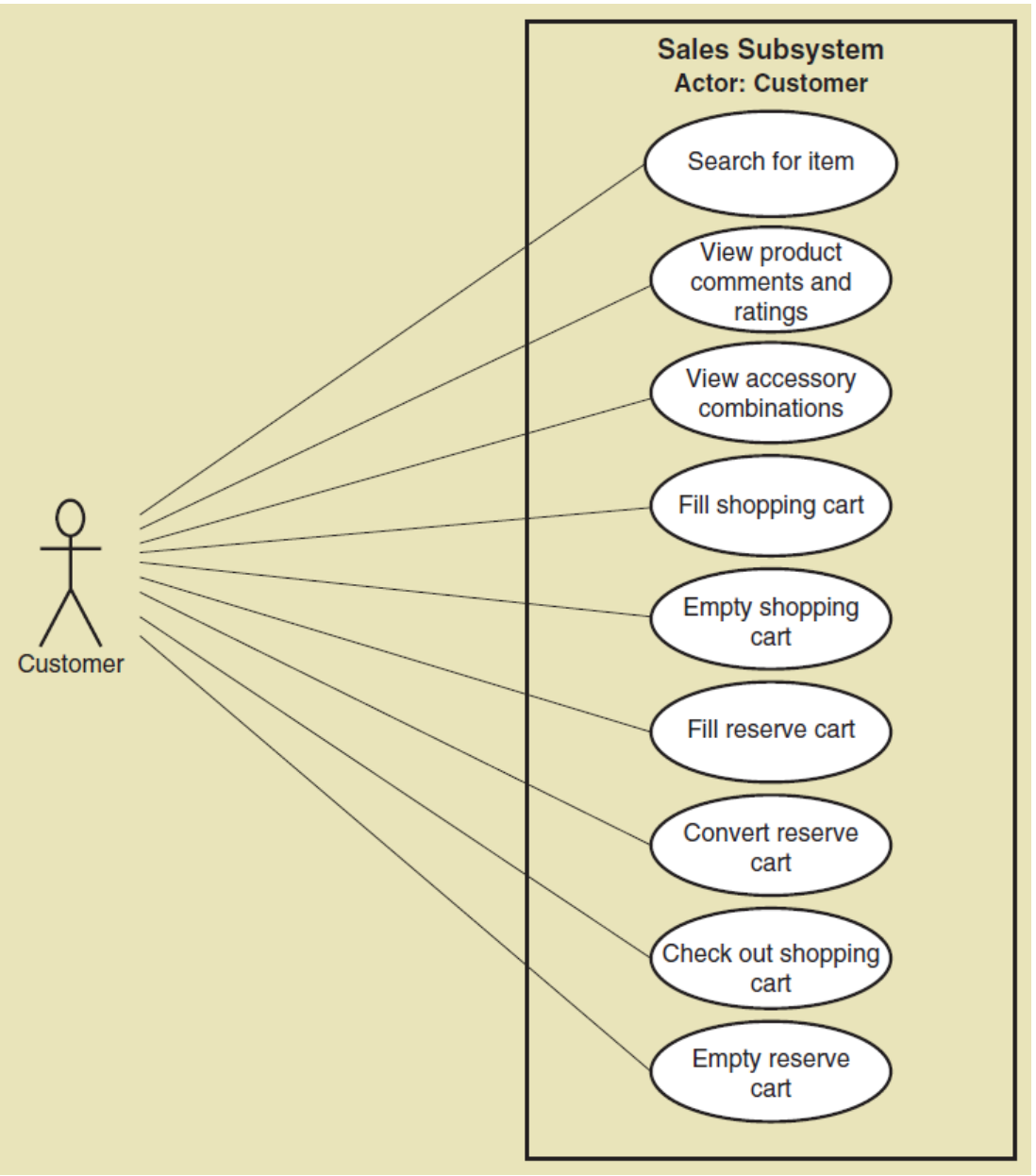
- A use case can include another use case within it. In other words, one use case might use the services of another use case.
- The association is stereotyped «include»

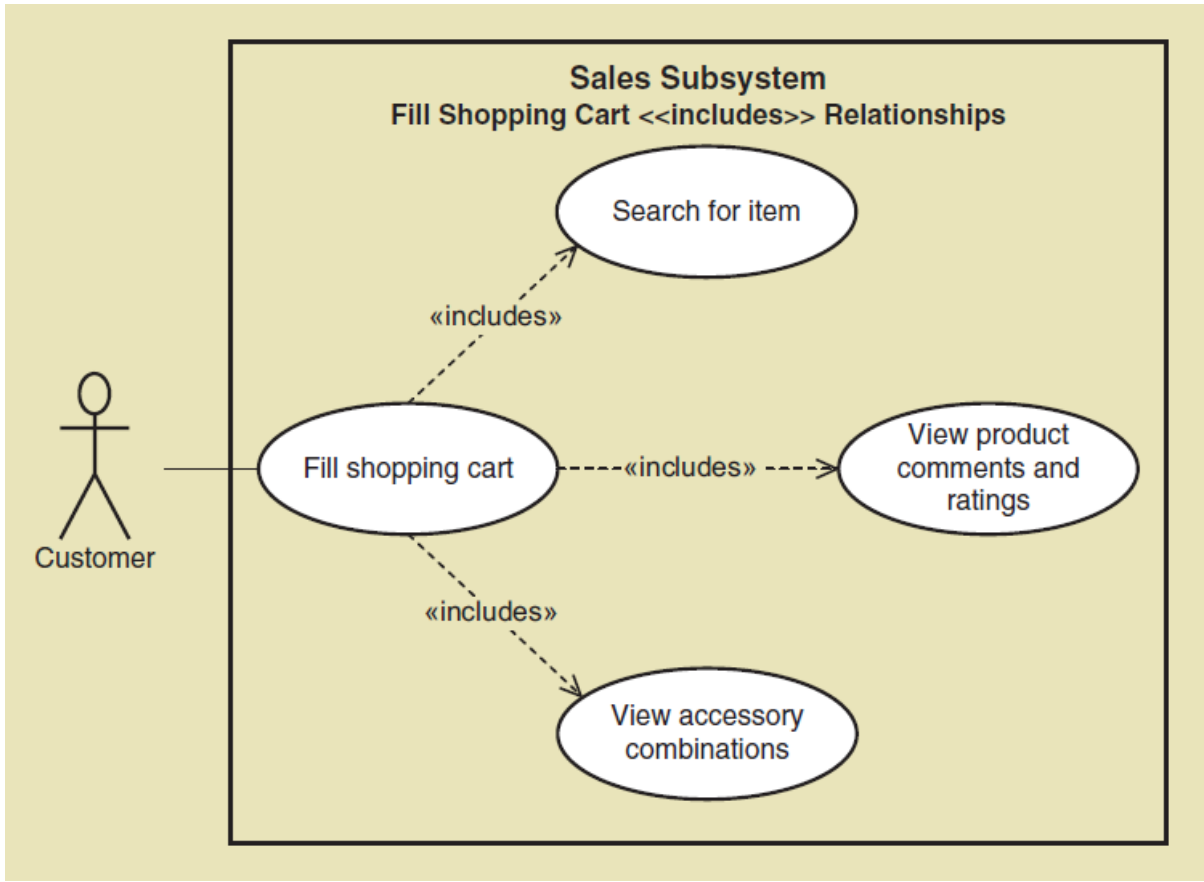


Use Case Diagram: Account Management

UC1: Unlock
UC2: Lock
UC3: AddUser
UC4: RemoveUser
UC5: InspectAccessHistory
UC6: SetDevicePrefs
UC7: AuthenticateUser
UC8: Login

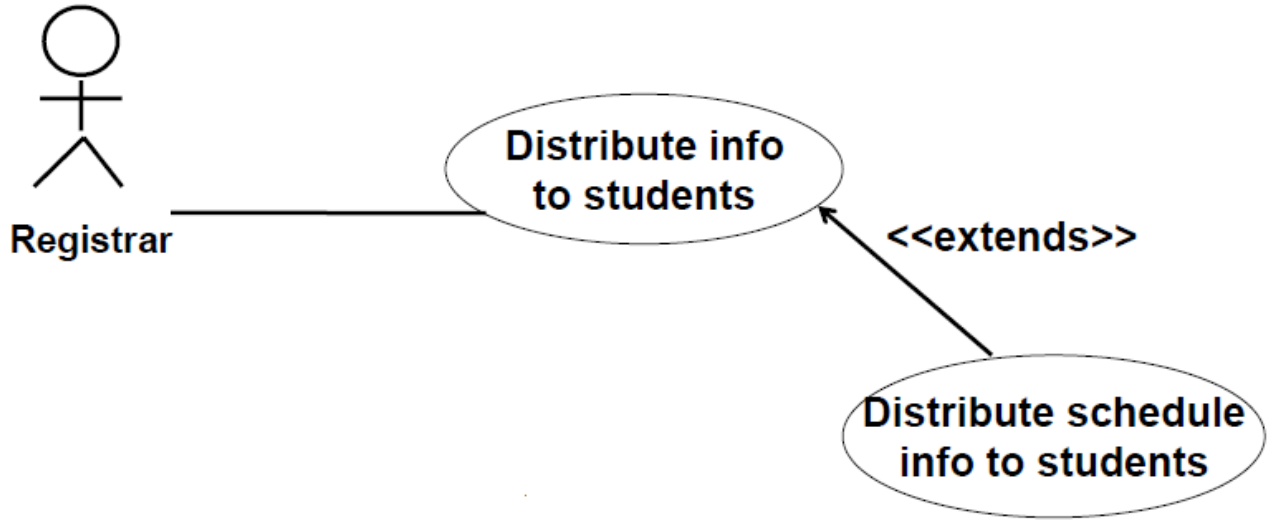



Sales Subsystem
Actor: Customer
Search for item
View product comments and ratings
View accessory combinations
Fill shopping cart
Empty shopping cart
Fill reserve cart
Convert reserve cart
Check out shopping cart
Empty reserve cart
Customer


Sales Subsystem
Fill Shopping Cart <<includes>> Relationships
Search for item
«includes»
Customer
Fill shopping cart
«includes»
View product comments and ratings
«includes»
View accessory combinations

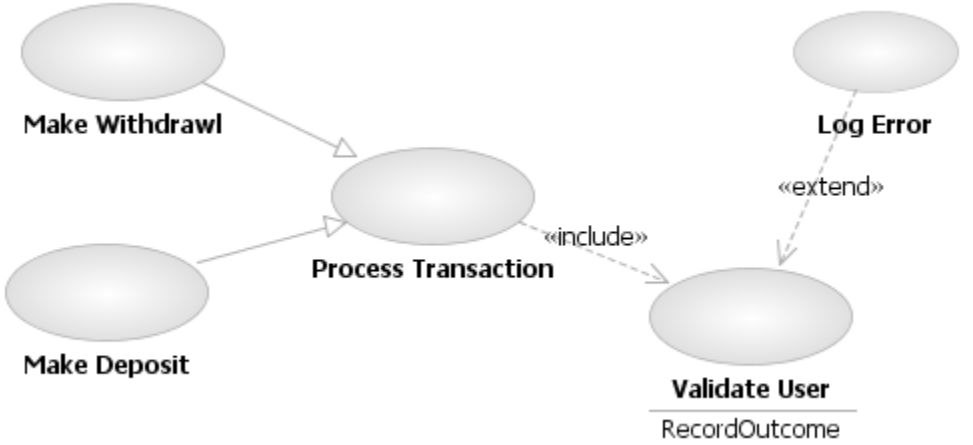
Relationships (Extend)

- When one use case adds behavior to a base case
  - used to model a part of a use case that the user may see as optional system behavior;
  - also models a separate sub-case which is executed conditionally.
- The association is stereotyped «extend»



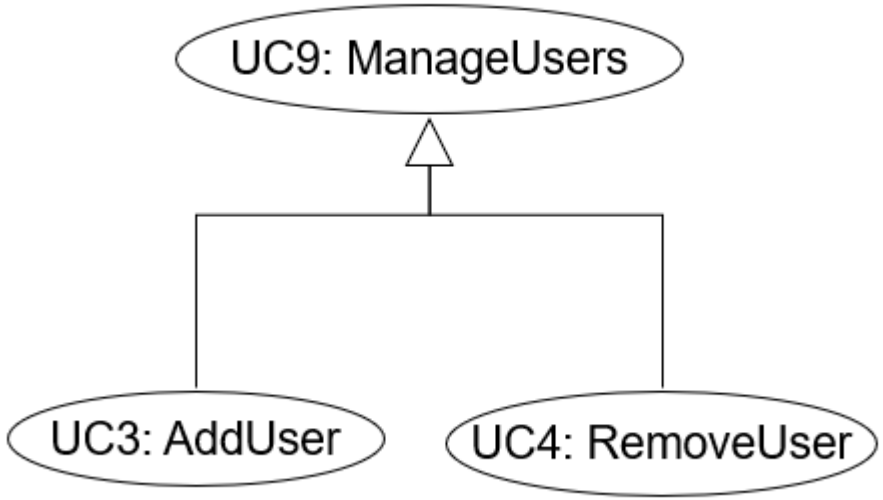
- Extensions could be triggered by an error or failure in the process.

- Think about how every step of the use case could fail.
- Give a plausible response to each extension from the system.
- Response should either jump to another step of the case or end it.



Generalization

- The UML also allows for inheritance relationships on actors and use cases



Common Rules

- Number Limit: The diagram should have between 3 to 10 base use-cases. No more than 15 use cases (base + included + extending).

  ---- If the dependency between two parts of a use-case is weak, they should be divided.

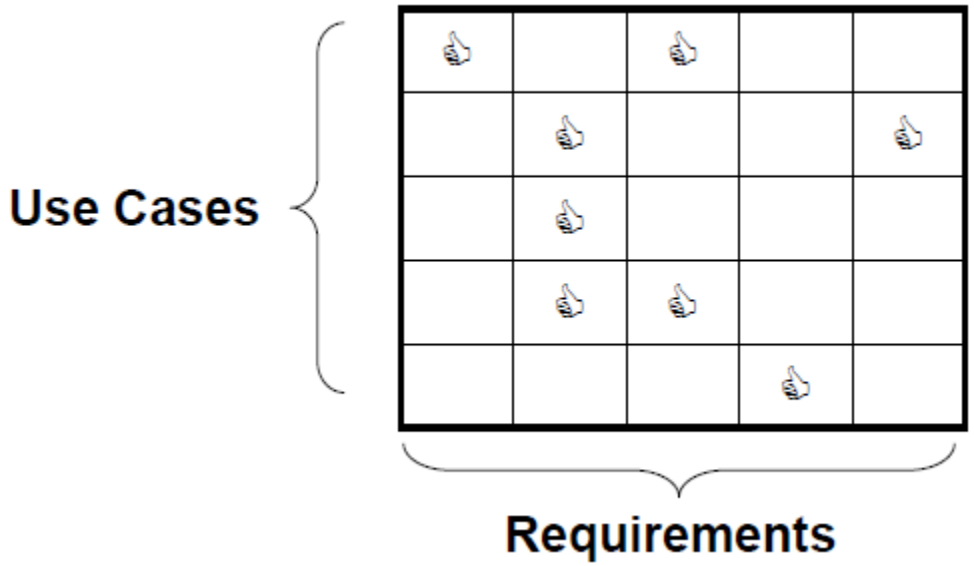
- Abstraction: All use-cases should be in similar abstraction levels.

- Size: Use cases should be described in half a page or more.

  - split using include/exclude

When we are done

- When every actor is specified.
- When every functional requirement has a use-case which satisfies it.
- A tractability matrix can help us determine it:



Traceability Matrix

Mapping: System requirements to Use cases

REQ1: Keep door locked and auto-lock
REQ2: Lock when “LOCK” pressed
REQ3: Unlock when valid key provided
REQ4: Allow mistakes but prevent dictionary attacks
REQ5: Maintain a history log
REQ6: Adding/removing users at runtime
REQ7: Configuring the device activation preferences
REQ8: Inspecting the access history
REQ9: Filing inquiries

UC1: Unlock
UC2: Lock
UC3: AddUser
UC4: RemoveUser
UC5: InspectAccessHistory
UC6: SetDevicePrefs
UC7: AuthenticateUser
UC8: Login

Req’t	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8
REQ1	X	X						
REQ2		X						
REQ3	X						X	
REQ4	X						X	
REQ5	X	X						
REQ6			X	X				X
REQ7						X		X
REQ8					X			X
REQ9					X			X

Purpose: To check that none of the use cases is introduced without a reason